Diskrete Mathematik für Informatik (SS 2025)

Martin Held

FB Informatik
Universität Salzburg
A-5020 Salzburg, Austria
held@cs.sbg.ac.at

12. Juni 2025

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Personalia

LVA-Leiter (VO+PS): Martin Held.

Email-Adresse: held@cs.sbg.ac.at.

Basis-URL: https://www.cosy.sbg.ac.at/~held.

Büro: Universität Salzburg, FB Informatik, Zi. 1.20,
Jakob-Haringer Str. 2, 5020 Salzburg-Itzling.

Telefonnummer (Büro): (0662) 8044-6304.

Telefonnummer (Sekr.): (0662) 8044-6300.

Personalia

LVA-Leiter (PS): Markus Flatz.

Email-Adresse: mflatz@cs.sbg.ac.at.

Telefonnummer (Sekr.): (0662) 8044-6300.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Personalia

LVA-Leiter (PS): Mara Grilnberger.

Email-Adresse: `mara.grilnberger@plus.ac.at`.

Büro: Universität Salzburg, FB Informatik, Zi. 2.34, Jakob-Haringer Str. 2, 5020 Salzburg-Itzling.

Telefonnummer (Sekr.): (0662) 8044-6300.

Formalia

LVA-URL (VO+PS): https://www.cosy.sbg.ac.at/~held/teaching/diskrete_mathematik/dm.html.

Allg. Information: Basis-URL/for_students.html.

PLUSonline: Bitte melden Sie sich unbedingt im PLUSonline zu VO/PS an!

Abhaltezeit der VO: Donnerstag 7^{45}–11^{00}, mit etwa 20–25 Minuten Pause.

Abhalteort der VO: T01, FB Informatik, Jakob-Haringer Str. 2.

Abhaltezeit des PS: Freitag 11^{40}–13^{40}.

Abhalteort des PS: T01+T02+T03, Jakob-Haringer Str. 2.

Tutorium: Andreas Auer und Jatin Kumar:
Montag 16^{00}–18^{00} (T06),
Mittwoch 12^{30}–14^{30} (T02);
FB Informatik, Jakob-Haringer Str. 2.

Achtung — das Proseminar ist prüfungsimmanent!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Electronic Slides and Online Material

In addition to these slides, you are encouraged to consult the WWW home page of this lecture:

https://www.cosy.sbg.ac.at/~held/teaching/diskrete_mathematik/dm.html.

In particular, this WWW page contains up-to-date information on the course, plus links to online notes, slides and (possibly) sample code.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

A Few Words of Warning

I hope that these slides will serve as a practice-minded introduction to various aspects of discrete mathematics which are of importance for computer science. I would like to warn you explicitly not to regard these slides as the sole source of information on the topics of my course. It may and will happen that I'll use the lecture for talking about subtle details that need not be covered in these slides! In particular, the slides won't contain all sample calculations, proofs of theorems, demonstrations of algorithms, or solutions to problems posed during my lecture. That is, by making these slides available to you I do not intend to encourage you to attend the lecture on an irregular basis.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Acknowledgments

These slides are a revised and extended version of a draft prepared by Kamran Safdar. Included is material written by Christian Alt, Caroline Atzl, Michael Burian, Peter Gintner, Bernhard Guillon, Yvonne Höller, Stefan Huber, Sandra Huemer, Christian Lercher, Sebastian Stenger, Alexander Zrinyi. I also benefited from comments and suggestions made by Stefan Huber and Peter Palfrader.

This revision and extension was carried out by myself, and I am responsible for all errors.

Salzburg, February 2025 Martin Held

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Legal Fine Print and Disclaimer

Recommended Textbooks I

- S. Maurer, A. Ralston.
  Discrete Algorithmic Mathematics
  A.K. Peters, 3rd edition, Jan 2005; ISBN 978-1-56881-166-6
- K.H. Rosen.
  Discrete Mathematics and Its Applications
  McGraw-Hill, 8th edition, 2019; ISBN 9781259676512
- B. Kolman, R.C. Busby, S.C. Ross.
  Discrete Mathematical Structures
  Pearson India, 6th edition, 2017; ISBN 978-0134696447.
- K.A. Ross, C.R.B. Wright.
  Discrete Mathematics
  Pearson Prentice Hall, 5th edition, Aug 2002; ISBN 9780130652478
- C. Stein, R.L.S. Drysdale, K. Bogart.
  Discrete Mathematics for Computer Science
  Addison-Wesley, March 2010; ISBN 978-0132122719.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Recommended Textbooks II

- J. O'Donnell, C. Hall, R. Page.
  Discrete Mathematics Using a Computer
  Springer, 2nd edition, 2006; ISBN 978-1-84628-241-6
- N.L. Biggs.
  Discrete Mathematics
  Oxford University Press, 2nd edition, Feb 2003, reprinted (with corrections) 2008; ISBN 978-0-19-850717-8
- M. Smid.
  Discrete Structures for Computer Science: Counting, Recursion, and Probability
  http://cglab.ca/~michiel/DiscreteStructures, 2019
- E. Lehman, F.T. Leighton, A.R. Meyer.
  Mathematics for Computer Science
  https://courses.csail.mit.edu/6.042, 2018
- M.M. Fleck.
  Building Blocks for Theoretical Computer Science
  http://mfleck.cs.illinois.edu/building-blocks/, 2017

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Table of Content

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

1 Introduction

- What is Discrete Mathematics?
- Motivation

What is Discrete Mathematics?

- No universally accepted definition of the scope of DM exists . . .
- Typically, objects studied in DM can only assume discrete, separate values rather than values out of a continuum; sets of such objects are countable.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

What is Discrete Mathematics?

- No universally accepted definition of the scope of DM exists . . .
- Typically, objects studied in DM can only assume discrete, separate values rather than values out of a continuum; sets of such objects are countable.
- Depending on what is covered in other courses a variety of topics tends to be studied within a course on DM:
  - Logic and Boolean algebra,
  - Mathematical language,
  - Set theory,
  - Functions and relations;

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

What is Discrete Mathematics?

- No universally accepted definition of the scope of DM exists . . .
- Typically, objects studied in DM can only assume discrete, separate values rather than values out of a continuum; sets of such objects are countable.
- Depending on what is covered in other courses a variety of topics tends to be studied within a course on DM:
  - Logic and Boolean algebra,
  - Mathematical language,
  - Set theory,
  - Functions and relations;
  - Computability theory,
  - Formal languages,
  - Automata theory;

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

What is Discrete Mathematics?

- No universally accepted definition of the scope of DM exists . . .
- Typically, objects studied in DM can only assume discrete, separate values rather than values out of a continuum; sets of such objects are countable.
- Depending on what is covered in other courses a variety of topics tends to be studied within a course on DM:
  - Logic and Boolean algebra,
  - Mathematical language,
  - Set theory,
  - Functions and relations;
  - Computability theory,
  - Formal languages,
  - Automata theory;
  - Algebraic structures,
  - Number theory,
  - Proofs and mathematical reasoning,
  - Counting and elementary combinatorics,
  - Graph theory,
  - Complexity theory,
  - Encoding and cryptography;

What is Discrete Mathematics?

- No universally accepted definition of the scope of DM exists . . .
- Typically, objects studied in DM can only assume discrete, separate values rather than values out of a continuum; sets of such objects are countable.
- Depending on what is covered in other courses a variety of topics tends to be studied within a course on DM:
  - Logic and Boolean algebra,
  - Mathematical language,
  - Set theory,
  - Functions and relations;
  - Computability theory,
  - Formal languages,
  - Automata theory;
  - Algebraic structures,
  - Number theory,
  - Proofs and mathematical reasoning,
  - Counting and elementary combinatorics,
  - Graph theory,
  - Complexity theory,
  - Encoding and cryptography;
  - Elementary probability theory.

Applications of Discrete Mathematics

- DM forms the mathematical language of computer science. It is at the very heart of several other parts of computer science.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Applications of Discrete Mathematics

- DM forms the mathematical language of computer science. It is at the very heart of several other parts of computer science.
- Applications of DM include — but are not limited to —
  - Algorithms and data structures,
  - Automated programming,
  - Automated theorem proving,
  - Combinatorial geometry,
  - Computational geometry,
  - Cryptography and cryptanalysis,
  - Discrete simulation,
  - Game theory,
  - Operations research and combinatorial optimization,
  - Theory of computing,
  - Queuing theory.

Applications of Discrete Mathematics

- DM forms the mathematical language of computer science. It is at the very heart of several other parts of computer science.
- Applications of DM include — but are not limited to —
  - Algorithms and data structures,
  - Automated programming,
  - Automated theorem proving,
  - Combinatorial geometry,
  - Computational geometry,
  - Cryptography and cryptanalysis,
  - Discrete simulation,
  - Game theory,
  - Operations research and combinatorial optimization,
  - Theory of computing,
  - Queuing theory.
- We start with a set of sample problems; solutions for all problems will be worked out or, at least, sketched during this course.

1 Introduction

- What is Discrete Mathematics?
- Motivation

Sample Problem: Summation Formula

- Suppose that an algorithm needs $1 + 2 + 3 + \cdots + (n - 1) + n$ many computational steps (of unit cost) to handle an input of size n.
- Question: Can we express this sum by means of a closed formula?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Summation Formula

- Suppose that an algorithm needs $1 + 2 + 3 + \cdots + (n-1) + n$ many computational steps (of unit cost) to handle an input of size n.
- Question: Can we express this sum by means of a closed formula?
- Basic math:

$$
\begin{aligned}
1 &= 1\\
1+2 &= 3\\
1+2+3 &= 6\\
1+2+3+4 &= 10\\
1+2+3+4+5 &= 15\\
1+2+3+4+5+6 &= 21\\
1+2+3+4+5+6+7 &= 28
\end{aligned}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Summation Formula

- Suppose that an algorithm needs $1 + 2 + 3 + \cdots + (n-1) + n$ many computational steps (of unit cost) to handle an input of size n.
- Question: Can we express this sum by means of a closed formula?
- Basic math:

$$
\begin{array}{lcrcl}
1 & = & 1 & = & {}^{1\cdot 2}\!/_2 \\
1+2 & = & 3 & = & {}^{2\cdot 3}\!/_2 \\
1+2+3 & = & 6 & = & {}^{3\cdot 4}\!/_2 \\
1+2+3+4 & = & 10 & = & {}^{4\cdot 5}\!/_2 \\
1+2+3+4+5 & = & 15 & = & {}^{5\cdot 6}\!/_2 \\
1+2+3+4+5+6 & = & 21 & = & {}^{6\cdot 7}\!/_2 \\
1+2+3+4+5+6+7 & = & 28 & = & {}^{7\cdot 8}\!/_2
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Summation Formula

- Suppose that an algorithm needs $1 + 2 + 3 + \cdots + (n-1) + n$ many computational steps (of unit cost) to handle an input of size n.
- Question: Can we express this sum by means of a closed formula?
- Basic math:

$$\begin{array}{lclcl} 1 & = & 1 & = & {}^{1\cdot 2}/_{2} \\ 1+2 & = & 3 & = & {}^{2\cdot 3}/_{2} \\ 1+2+3 & = & 6 & = & {}^{3\cdot 4}/_{2} \\ 1+2+3+4 & = & 10 & = & {}^{4\cdot 5}/_{2} \\ 1+2+3+4+5 & = & 15 & = & {}^{5\cdot 6}/_{2} \\ 1+2+3+4+5+6 & = & 21 & = & {}^{6\cdot 7}/_{2} \\ 1+2+3+4+5+6+7 & = & 28 & = & {}^{7\cdot 8}/_{2} \end{array}$$

- An inspection of the numbers on the right-hand side *might* let us suspect that

$$1 + 2 + 3 + \cdots + (n-1) + n = \frac{n(n+1)}{2}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Summation Formula

- Suppose that an algorithm needs $1 + 2 + 3 + \cdots + (n-1) + n$ many computational steps (of unit cost) to handle an input of size n.
- Question: Can we express this sum by means of a closed formula?
- Basic math:

$$\begin{array}{lcrcl}
1 & = & 1 & = & {}^{1\cdot 2}/_2 \\
1+2 & = & 3 & = & {}^{2\cdot 3}/_2 \\
1+2+3 & = & 6 & = & {}^{3\cdot 4}/_2 \\
1+2+3+4 & = & 10 & = & {}^{4\cdot 5}/_2 \\
1+2+3+4+5 & = & 15 & = & {}^{5\cdot 6}/_2 \\
1+2+3+4+5+6 & = & 21 & = & {}^{6\cdot 7}/_2 \\
1+2+3+4+5+6+7 & = & 28 & = & {}^{7\cdot 8}/_2
\end{array}$$

- An inspection of the numbers on the right-hand side *might* let us suspect that

$$1 + 2 + 3 + \cdots + (n-1) + n = \frac{n(n+1)}{2}.$$

- But is this indeed correct? And, by the way, what do the dots in this equation really mean??

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Summation Formula

- An answer can be established by means of number theory (natural numbers, induction). And we get indeed

$$1 + 2 + 3 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$$

for all "natural numbers" n.

Sample Problem: Summation Formula

- An answer can be established by means of number theory (natural numbers, induction). And we get indeed

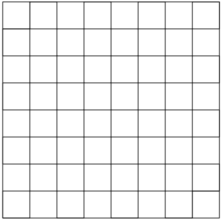
$$1 + 2 + 3 + \cdots + (n-1) + n = \frac{n(n+1)}{2}$$

  for all "natural numbers" n.

- Caution: Even after calculating this sum for all values of n between 1 and 500 one can not legitimately claim to know the sum for, say, $n := 1000$.
- Note: It would constitute a horrendous waste of CPU time to let a computer compute $1 + 2 + 3 + \cdots + (n-1) + n$ by successively adding numbers if we could simply obtain the result by evaluating $\frac{n(n+1)}{2}$.

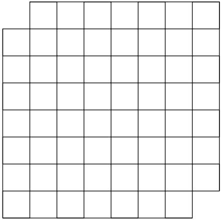
UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Consider an 8×8 chessboard



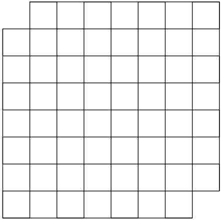
Sample Problem: Chessboard Tilings

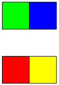
- Consider an 8×8 chessboard with the upper-left and lower-right cells removed,



Sample Problem: Chessboard Tilings

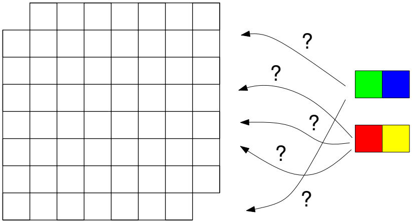
- Consider an 8×8 chessboard with the upper-left and lower-right cells removed, and assume that we are given red/yellow and green/blue domino blocks whose sizes match the size of two adjacent squares of the chessboard.



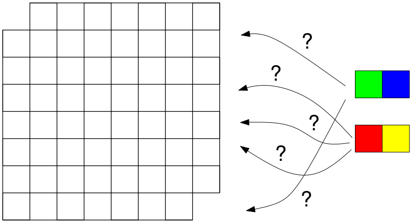


Sample Problem: Chessboard Tilings

- Consider an 8×8 chessboard with the upper-left and lower-right cells removed, and assume that we are given red/yellow and green/blue domino blocks whose sizes match the size of two adjacent squares of the chessboard.
- Question: Can this chessboard be covered completely by 31 domino blocks of arbitrary color combinations?



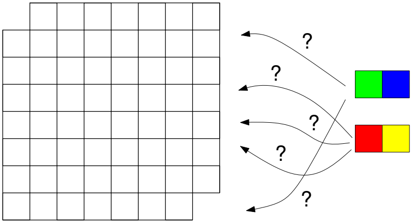
- Consider an 8×8 chessboard with the upper-left and lower-right cells removed, and assume that we are given red/yellow and green/blue domino blocks whose sizes match the size of two adjacent squares of the chessboard.
- Question: Can this chessboard be covered completely by 31 domino blocks of arbitrary color combinations?



- We consult counting principles and obtain the answer: No!

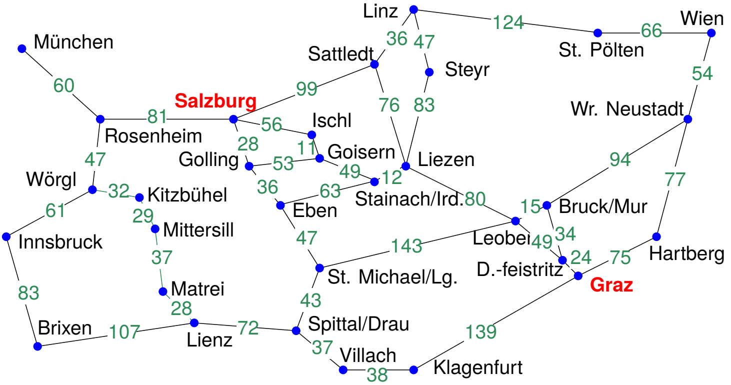
Sample Problem: Chessboard Tilings

- Consider an 8×8 chessboard with the upper-left and lower-right cells removed, and assume that we are given red/yellow and green/blue domino blocks whose sizes match the size of two adjacent squares of the chessboard.
- Question: Can this chessboard be covered completely by 31 domino blocks of arbitrary color combinations?



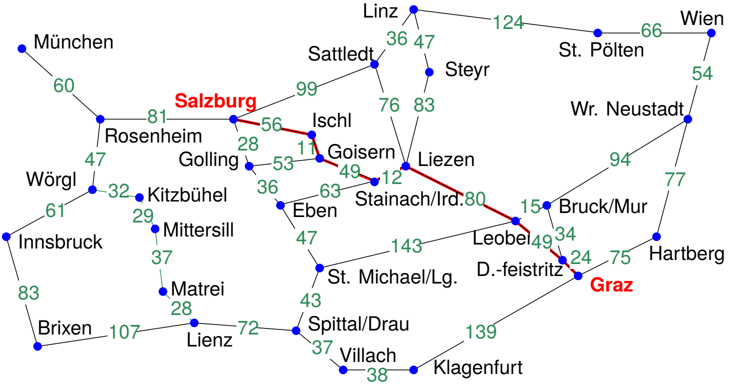
- We consult counting principles and obtain the answer: No!
- Caution: Simply trying out *all* possible placements of domino blocks hardly is an option for an 8×8 chessboard — and definitely no option for an $n \times n$ board!

- Question: What is the shortest route for driving from Salzburg to Graz?

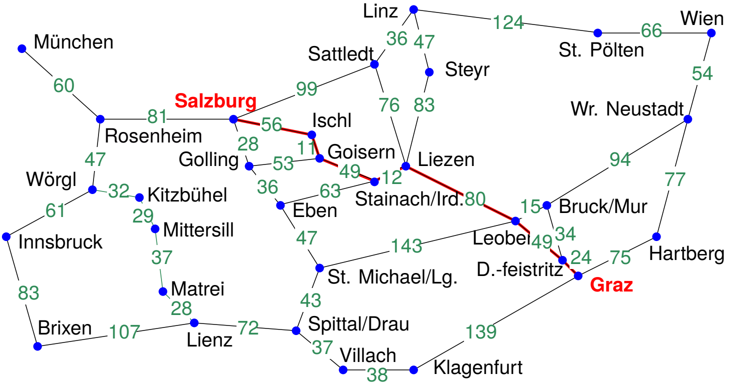


Sample Problem: Route Calculation

- Question: What is the shortest route for driving from Salzburg to Graz?
- Answer provided by computing a shortest path in a weighted graph: Salzburg → Bad Ischl → Bad Goisern → Stainach/Irdning → Liezen → Leoben → Graz.



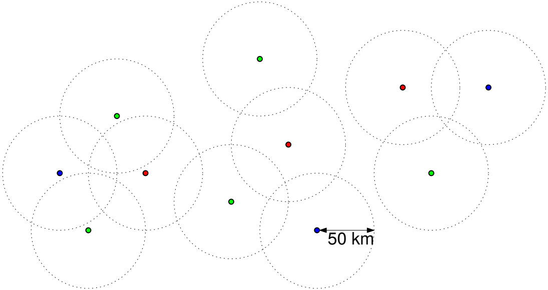
- Question: What is the shortest route for driving from Salzburg to Graz?
- Answer provided by computing a shortest path in a weighted graph: Salzburg → Bad Ischl → Bad Goisern → Stainach/Irdning → Liezen → Leoben → Graz.



- Note: Simply trying all possible routes gets tedious! (How would you even guarantee that all possible routes have indeed been checked?)

Sample Problem: Channel Assignment

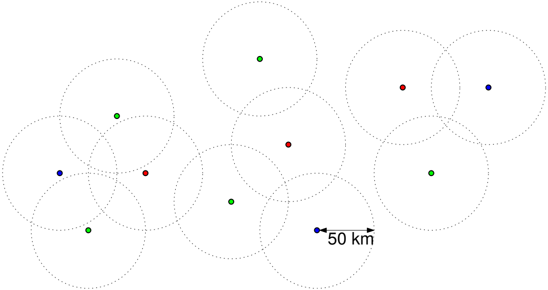
- Suppose that frequencies out of a set of m frequencies are to be assigned to n broadcast stations within Austria. We are told that the area serviced by a station lies within a disk with radius 50 kilometers. Obviously, no two different stations whose broadcast areas overlap may use the same frequency.



UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

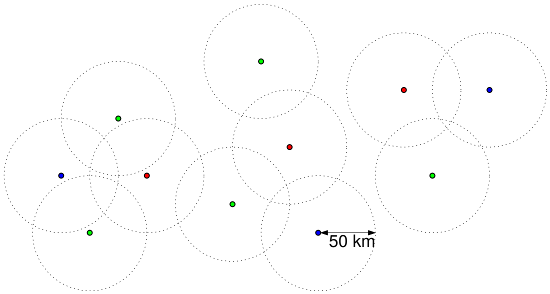
Sample Problem: Channel Assignment

- Suppose that frequencies out of a set of m frequencies are to be assigned to n broadcast stations within Austria. We are told that the area serviced by a station lies within a disk with radius 50 kilometers. Obviously, no two different stations whose broadcast areas overlap may use the same frequency.
- Question: Do we have enough frequencies? What is the minimum number of frequencies needed?



Sample Problem: Channel Assignment

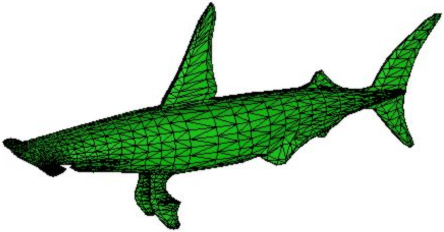
- Suppose that frequencies out of a set of m frequencies are to be assigned to n broadcast stations within Austria. We are told that the area serviced by a station lies within a disk with radius 50 kilometers. Obviously, no two different stations whose broadcast areas overlap may use the same frequency.
- Question: Do we have enough frequencies? What is the minimum number of frequencies needed?



- The solution can be obtained by using techniques of computational geometry combined with graph coloring.

Sample Problem: Memory Required for Storing a Polyhedron

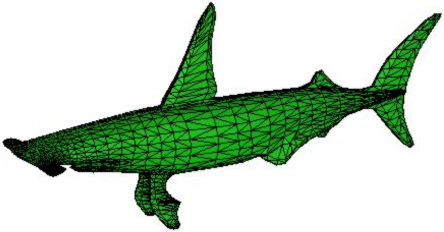
- Suppose that a polyhedral model has n vertices. How many edges and faces can it have at most? What is the storage complexity relative to n?



UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Memory Required for Storing a Polyhedron

- Suppose that a polyhedral model has n vertices. How many edges and faces can it have at most? What is the storage complexity relative to n?



- Answer provided by graph theory: A polyhedron with n vertices has at most $3n - 6$ edges and $2n - 4$ faces.

Sample Problem: Complexity of an Algorithm

- Suppose that an algorithm is given n numbers as input and that it solves a problem by proceeding as follows: During one round of computation, it performs n computational steps. We know that during each round it discards at least 25% of the numbers. The algorithm executes one round after the other until only one number is left.

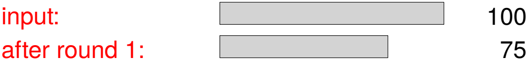
UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Complexity of an Algorithm

- Suppose that an algorithm is given n numbers as input and that it solves a problem by proceeding as follows: During one round of computation, it performs n computational steps. We know that during each round it discards at least 25% of the numbers. The algorithm executes one round after the other until only one number is left.

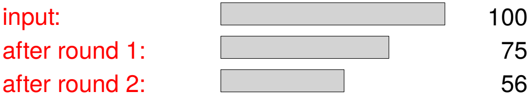
Sample Problem: Complexity of an Algorithm

- Suppose that an algorithm is given *n* numbers as input and that it solves a problem by proceeding as follows: During one round of computation, it performs *n* computational steps. We know that during each round it discards at least 25% of the numbers. The algorithm executes one round after the other until only one number is left.

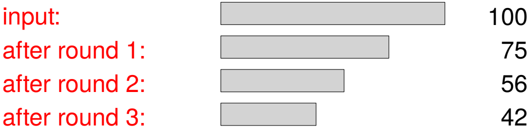


UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Suppose that an algorithm is given n numbers as input and that it solves a problem by proceeding as follows: During one round of computation, it performs n computational steps. We know that during each round it discards at least 25% of the numbers. The algorithm executes one round after the other until only one number is left.

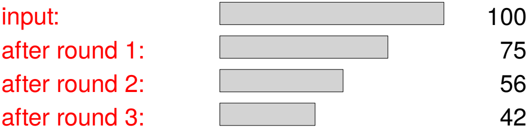


- Suppose that an algorithm is given n numbers as input and that it solves a problem by proceeding as follows: During one round of computation, it performs n computational steps. We know that during each round it discards at least 25% of the numbers. The algorithm executes one round after the other until only one number is left.



Sample Problem: Complexity of an Algorithm

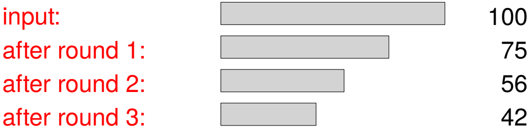
- Suppose that an algorithm is given *n* numbers as input and that it solves a problem by proceeding as follows: During one round of computation, it performs *n* computational steps. We know that during each round it discards at least 25% of the numbers. The algorithm executes one round after the other until only one number is left.



- Question: How many rounds does the algorithm run in the worst case (depending on the input size *n*)? How many computational steps are carried out in the worst case?

Sample Problem: Complexity of an Algorithm

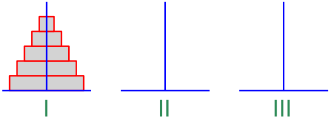
- Suppose that an algorithm is given n numbers as input and that it solves a problem by proceeding as follows: During one round of computation, it performs n computational steps. We know that during each round it discards at least 25% of the numbers. The algorithm executes one round after the other until only one number is left.



- Question: How many rounds does the algorithm run in the worst case (depending on the input size n)? How many computational steps are carried out in the worst case?
- Answer provided by the theory of recurrence relations: The number of computational steps is linear in n, and the number of rounds is logarithmic in n.
- In asymptotic notation: $O(n)$ and $O(\log n)$.

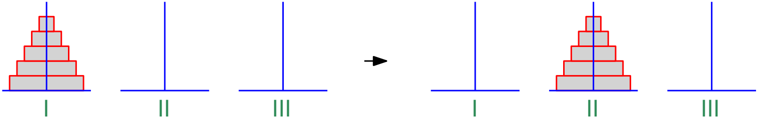
Sample Problem: Optimality of an Algorithm

- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top,



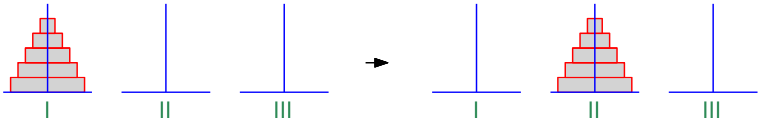
Sample Problem: Optimality of an Algorithm

- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.



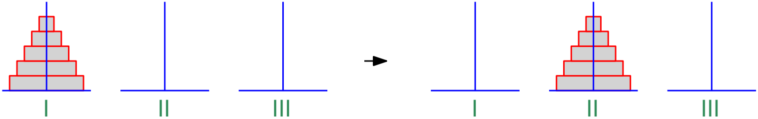
Sample Problem: Optimality of an Algorithm

- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.



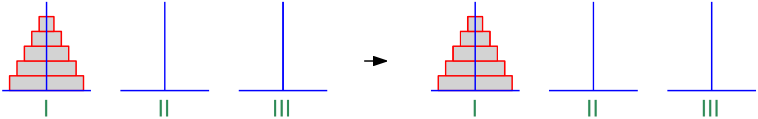
Sample Problem: Optimality of an Algorithm

- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.
- Goal: Find an algorithm that uses the minimum number of moves.



Sample Problem: Optimality of an Algorithm

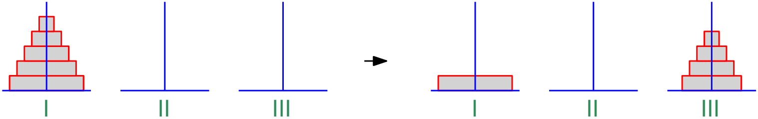
- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.
- Goal: Find an algorithm that uses the minimum number of moves.



- One can prove: A (straightforward) recursive algorithm needs $2^n - 1$ moves.

Sample Problem: Optimality of an Algorithm

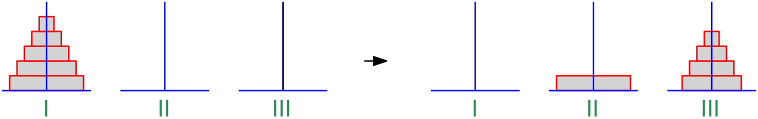
- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.
- Goal: Find an algorithm that uses the minimum number of moves.



- One can prove: A (straightforward) recursive algorithm needs $2^n - 1$ moves.

Sample Problem: Optimality of an Algorithm

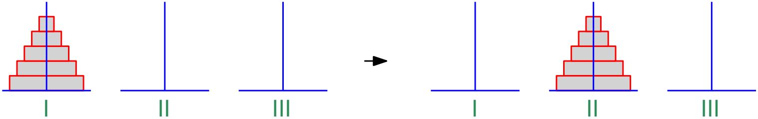
- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.
- Goal: Find an algorithm that uses the minimum number of moves.



- One can prove: A (straightforward) recursive algorithm needs $2^n - 1$ moves.

Sample Problem: Optimality of an Algorithm

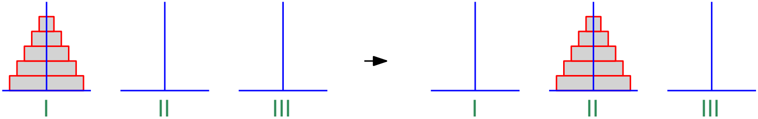
- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.
- Goal: Find an algorithm that uses the minimum number of moves.



- One can prove: A (straightforward) recursive algorithm needs $2^n - 1$ moves.

Sample Problem: Optimality of an Algorithm

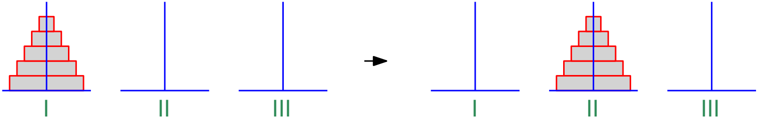
- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.
- Goal: Find an algorithm that uses the minimum number of moves.



- One can prove: A (straightforward) recursive algorithm needs $2^n - 1$ moves.
- One can also prove: Every(!) algorithm that solves ToH needs at least $2^n - 1$ moves.

Sample Problem: Optimality of an Algorithm

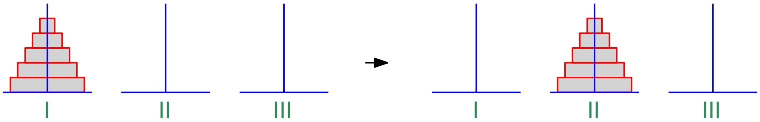
- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.
- Goal: Find an algorithm that uses the minimum number of moves.



- One can prove: A (straightforward) recursive algorithm needs $2^n - 1$ moves.
- One can also prove: Every(!) algorithm that solves ToH needs at least $2^n - 1$ moves.
- Thus, the solution achieved by the recursive algorithm is optimal as far as the number of moves is concerned.

Sample Problem: Optimality of an Algorithm

- Tower-of-Hanoi Problem (ToH): Given three pegs (labeled I,II,III) and a stack of n disks arranged on Peg I from largest at the bottom to smallest at the top, we are to move all disks to Peg II such that only one disk is moved at a time and such that no larger disk ever is placed on a smaller disk.
- Attributed to Édouard Lucas (1883). Supposedly based on an Indian legend about Brahmin priests moving 64 disks in the Great Temple of Benares; once they are finished, life on Earth will end.
- Goal: Find an algorithm that uses the minimum number of moves.



- One can prove: A (straightforward) recursive algorithm needs $2^n - 1$ moves.
- One can also prove: Every(!) algorithm that solves ToH needs at least $2^n - 1$ moves.
- Thus, the solution achieved by the recursive algorithm is optimal as far as the number of moves is concerned.
- [Buneman&Levy (1980)]: There exists a simple iterative solution that avoids an exponential-sized stack!

Sample Problem: The Power of Exponential Growth

- According to legend, the power of exponential growth was already known by the Brahmin Sissa ibn Dahir (ca. 300-400 AD): As a reward for the invention of the game of chess (or its Indian predecessor Chaturanga) he asked his king to place one grain of rice in the first square of a chessboard, two in the second, four in the third, and so on, doubling the amount of rice up to the 64-th square.

Sample Problem: The Power of Exponential Growth

- According to legend, the power of exponential growth was already known by the Brahmin Sissa ibn Dahir (ca. 300-400 AD): As a reward for the invention of the game of chess (or its Indian predecessor Chaturanga) he asked his king to place one grain of rice in the first square of a chessboard, two in the second, four in the third, and so on, doubling the amount of rice up to the 64-th square.
- So, how many grains of rice did Sissa ask for?

- According to legend, the power of exponential growth was already known by the Brahmin Sissa ibn Dahir (ca. 300-400 AD): As a reward for the invention of the game of chess (or its Indian predecessor Chaturanga) he asked his king to place one grain of rice in the first square of a chessboard, two in the second, four in the third, and so on, doubling the amount of rice up to the 64-th square.
- So, how many grains of rice did Sissa ask for?
- Let $R(64)$ denote the number of rice grains for 64 squares. We get

$$R(64) = 1 + 2 + 4 + \ldots + 2^{63}$$

Sample Problem: The Power of Exponential Growth

- According to legend, the power of exponential growth was already known by the Brahmin Sissa ibn Dahir (ca. 300-400 AD): As a reward for the invention of the game of chess (or its Indian predecessor Chaturanga) he asked his king to place one grain of rice in the first square of a chessboard, two in the second, four in the third, and so on, doubling the amount of rice up to the 64-th square.
- So, how many grains of rice did Sissa ask for?
- Let $R(64)$ denote the number of rice grains for 64 squares. We get

$$R(64) = 1 + 2 + 4 + \ldots + 2^{63}$$

and, in general, using the capital-sigma notation and geometric series,

$$R(n) = 1 + 2 + 4 + \ldots + 2^{n-1} = \sum_{i=1}^{n} 2^{i-1} = \sum_{i=0}^{n-1} 2^{i} = \frac{2^n - 1}{2 - 1} = 2^n - 1.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: The Power of Exponential Growth

- According to legend, the power of exponential growth was already known by the Brahmin Sissa ibn Dahir (ca. 300-400 AD): As a reward for the invention of the game of chess (or its Indian predecessor Chaturanga) he asked his king to place one grain of rice in the first square of a chessboard, two in the second, four in the third, and so on, doubling the amount of rice up to the 64-th square.
- So, how many grains of rice did Sissa ask for?
- Let $R(64)$ denote the number of rice grains for 64 squares. We get

$$R(64) = 1 + 2 + 4 + \ldots + 2^{63}$$

and, in general, using the capital-sigma notation and geometric series,

$$R(n) = 1 + 2 + 4 + \ldots + 2^{n-1} = \sum_{i=1}^{n} 2^{i-1} = \sum_{i=0}^{n-1} 2^{i} = \frac{2^n - 1}{2 - 1} = 2^n - 1.$$

- Hence, Sissa asked for

$$2^{64} - 1 = 18\,446\,744\,073\,709\,551\,615$$

grains of rice. This is about 1000 times the current global yearly production!

Sample Problem: The Power of Exponential Growth

- According to legend, the power of exponential growth was already known by the Brahmin Sissa ibn Dahir (ca. 300-400 AD): As a reward for the invention of the game of chess (or its Indian predecessor Chaturanga) he asked his king to place one grain of rice in the first square of a chessboard, two in the second, four in the third, and so on, doubling the amount of rice up to the 64-th square.
- So, how many grains of rice did Sissa ask for?
- Let $R(64)$ denote the number of rice grains for 64 squares. We get

  $$R(64) = 1 + 2 + 4 + \ldots + 2^{63}$$

  and, in general, using the capital-sigma notation and geometric series,

  $$R(n) = 1 + 2 + 4 + \ldots + 2^{n-1} = \sum_{i=1}^{n} 2^{i-1} = \sum_{i=0}^{n-1} 2^{i} = \frac{2^n - 1}{2 - 1} = 2^n - 1.$$

- Hence, Sissa asked for

  $$2^{64} - 1 = 18\,446\,744\,073\,709\,551\,615$$

  grains of rice. This is about 1000 times the current global yearly production!
- [Sagan 1997]: “Exponentials can’t go on forever, because they will gobble up everything”.
- The “*second half of the chessboard*” is a phrase, coined by Kurzweil in 1999, to refer to the point where exponential growth begins to have a significant impact.

UNIVERSITÄT SALZBURG Computational Geometry and Applications Lab

Sample Problem: Key Distribution and Message Encryption

- Suppose that two persons named Alice and Bob want to exchange a secret information, e.g., a key that can be used for decrypting their encrypted messages.

Sample Problem: Key Distribution and Message Encryption

- Suppose that two persons named Alice and Bob want to exchange a secret information, e.g., a key that can be used for decrypting their encrypted messages.
- Likely, they will not consider it to be safe to exchange the key as plain text via, say, email.
- What is a secure mechanism for them to exchange a key??

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Key Distribution and Message Encryption

- Suppose that two persons named Alice and Bob want to exchange a secret information, e.g., a key that can be used for decrypting their encrypted messages.
- Likely, they will not consider it to be safe to exchange the key as plain text via, say, email.
- What is a secure mechanism for them to exchange a key??
  - Meet in person at a secret place and share the key?!
  - Share in parts?!

Sample Problem: Key Distribution and Message Encryption

- Suppose that two persons named Alice and Bob want to exchange a secret information, e.g., a key that can be used for decrypting their encrypted messages.
- Likely, they will not consider it to be safe to exchange the key as plain text via, say, email.
- What is a secure mechanism for them to exchange a key??
  - Meet in person at a secret place and share the key?!
  - Share in parts?!
- Answer provided by cryptography: The Diffie-Hellman Algorithm provides a simple way to exchange a key via public communication channels.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Problem: Key Distribution and Message Encryption

- Suppose that two persons named Alice and Bob want to exchange a secret information, e.g., a key that can be used for decrypting their encrypted messages.
- Likely, they will not consider it to be safe to exchange the key as plain text via, say, email.
- What is a secure mechanism for them to exchange a key??
  - Meet in person at a secret place and share the key?!
  - Share in parts?!

- Answer provided by cryptography: The Diffie-Hellman Algorithm provides a simple way to exchange a key via public communication channels.
- By the way, how could Alice and Bob encrypt or decrypt messages once they have exchanged their key?

Sample Problem: Key Distribution and Message Encryption

- Suppose that two persons named Alice and Bob want to exchange a secret information, e.g., a key that can be used for decrypting their encrypted messages.
- Likely, they will not consider it to be safe to exchange the key as plain text via, say, email.
- What is a secure mechanism for them to exchange a key??
  - Meet in person at a secret place and share the key?!
  - Share in parts?!
- Answer provided by cryptography: The Diffie-Hellman Algorithm provides a simple way to exchange a key via public communication channels.
- By the way, how could Alice and Bob encrypt or decrypt messages once they have exchanged their key?
- Answer: This is yet another application of number theory!

2 Propositional and Predicate Logic

- Propositional Logic
- Predicate Logic
- Special Quantifiers

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

2 Propositional and Predicate Logic

- Propositional Logic
- Predicate Logic
- Special Quantifiers

Propositional Logic

- Goal: specification of a language for formally expressing theorems and proofs.
- Aka: propositional calculus, logic of statements, statement logic;
- Dt.: Aussagenlogik.

Definition 1 (Proposition, Dt.: Aussage)

A *proposition* is a statement that is either true or false.

- Propositions can be *atomic*,
  like "The sun is shining",
  or *compound*,
  like "The sun is shining and the temperature is high".
- In the latter case, the proposition is a composition of atomic or compound propositions by means of logical junctors. (Junctors are also known as connectives or operators.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 2 (Propositional formula, Dt.: aussagenlogische Formel)

A propositional formula is constructed inductively from a set of

- propositional variables (typically p, q, r or $p_1, p_2, \ldots$);
- junctors: $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$;
- parentheses: $($, $)$;
- constants (truth values): $\perp, \top$ (or F, T);

based on the following rules:

- A propositional variable is a propositional formula.
- The constants $\perp$ and $\top$ are propositional formulas.
- If ϕ_1 and ϕ_2 are propositional formulas then so are the following:

$$(\neg\phi_1), (\phi_1 \wedge \phi_2), (\phi_1 \vee \phi_2), (\phi_1 \Rightarrow \phi_2), (\phi_1 \Leftrightarrow \phi_2).$$

Precedence Rules

- Precedence rules (Dt.: Vorrangregeln) are used frequently to avoid the burden of too many parentheses. From highest to lowest precedence, the following order is common.

$$\neg, \quad \wedge, \quad \vee, \quad \begin{array}{c}\Rightarrow \\ \Leftrightarrow\end{array}$$

- Unfortunately, different precedence rules tend to be used by different authors.
- Thus, make it clear which order you use, or in case of doubt, insert parentheses!
- It is common to represent the truth values of a proposition under all possible assignments to its variables by means of a *truth table*.
- In addition to the standard junctors we also define two other operators, NAND, denoted by $\uparrow$ (or sometimes by $|$), and NOR, denoted by $\downarrow$.

Truth Tables

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$	$p \uparrow q$	$p \downarrow q$
T	T	F	T	T	T	T	F	F
T	F	F	F	T	F	F	T	F
F	T	T	F	T	T	F	T	F
F	F	T	F	F	T	T	T	T

- Common names for the junctors in natural language:
  - $\neg p$: NOT, negation;
  - $p \wedge q$: AND, conjunction;
  - $p \vee q$: OR, disjunction;
  - $p \Rightarrow q$: IMPLIES, conditional, if p then q, q if p, p sufficient for q, q necessary for p;
  - $p \Leftrightarrow q$: IFF, equivalence, biconditional, p if and only if q, p necessary and sufficient for q.
- Note: The truth table (Dt.: Wahrheitstabelle) of a formula with n variables has 2^n rows.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Tautologies, Contradictions

Definition 3 (Tautology, Dt.: Tautologie)

A propositional formula is a *tautology* if it is true under all truth assignments to its variables.

Definition 4 (Contradiction, Dt.: Widerspruch)

A propositional formula is a *contradiction* if it is false under all truth assignments to its variables.

- Standard examples: $(p \vee \neg p)$ and $(p \wedge \neg p)$.
- Easy to prove: The negation of a tautology yields a contradiction, and vice versa.

Logical Equivalence

Definition 5 (Logical equivalence, Dt.: logische Äquivalenz)

Two propositional formulas are *logically equivalent* if they have the same truth table. Logical equivalence of formulas ϕ_1, ϕ_2 is commonly denoted by $\phi_1 \equiv \phi_2$.

Theorem 6

Two propositional formulas ϕ_1, ϕ_2 are logically equivalent iff $\phi_1 \Leftrightarrow \phi_2$ is a tautology.

Definition 7 (Complete set of junctors, Dt.: vollständige Junktorenmenge)

A set S of junctors is said to be *complete* (or truth-functionally adequate/complete) if, for any given propositional formula, a logically equivalent one can be written using only junctors of S.

- Note: The sets $\{\uparrow\}$ and $\{\downarrow\}$ both are complete sets of junctors.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Laws for Logical Equivalence

Theorem 8

Let ϕ_1, ϕ_2 be propositional formulas. Then the following equivalences hold:

Identity:	$\phi_1 \wedge T \equiv \phi_1$	$\phi_1 \vee F \equiv \phi_1$
Domination:	$\phi_1 \vee T \equiv T$	$\phi_1 \wedge F \equiv F$
Idempotence:	$\phi_1 \vee \phi_1 \equiv \phi_1$	$\phi_1 \wedge \phi_1 \equiv \phi_1$
Double negation:	$\neg\neg\phi_1 \equiv \phi_1$	
Commutativity:	$\phi_1 \wedge \phi_2 \equiv \phi_2 \wedge \phi_1$	$\phi_1 \vee \phi_2 \equiv \phi_2 \vee \phi_1$
	$\phi_1 \Leftrightarrow \phi_2 \equiv \phi_2 \Leftrightarrow \phi_1$	
Distributivity:	$(\phi_1 \vee \phi_2) \wedge \phi_3 \equiv (\phi_1 \wedge \phi_3) \vee (\phi_2 \wedge \phi_3)$	
	$(\phi_1 \wedge \phi_2) \vee \phi_3 \equiv (\phi_1 \vee \phi_3) \wedge (\phi_2 \vee \phi_3)$	
Associativity:	$(\phi_1 \vee \phi_2) \vee \phi_3 \equiv \phi_1 \vee (\phi_2 \vee \phi_3)$	
	$(\phi_1 \wedge \phi_2) \wedge \phi_3 \equiv \phi_1 \wedge (\phi_2 \wedge \phi_3)$	
De Morgan's laws:	$\neg(\phi_1 \wedge \phi_2) \equiv \neg\phi_1 \vee \neg\phi_2$	
	$\neg(\phi_1 \vee \phi_2) \equiv \neg\phi_1 \wedge \neg\phi_2$	
Trivial tautology:	$\phi_1 \vee \neg\phi_1 \equiv T$	
Trivial contradiction:	$\phi_1 \wedge \neg\phi_1 \equiv F$	
Contraposition:	$\neg\phi_1 \Leftrightarrow \neg\phi_2 \equiv \phi_1 \Leftrightarrow \phi_2$	$\neg\phi_2 \Rightarrow \neg\phi_1 \equiv \phi_1 \Rightarrow \phi_2$
Implication as Disj.:	$\phi_1 \Rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2$	

Definition 9 (Logical implication, Dt.: logische Implikation)

A formula ϕ_1 *logically implies* ϕ_2, denoted by $\phi_1 \models \phi_2$, if $\phi_1 \Rightarrow \phi_2$ is a tautology.

Definition 10 (Proof, Dt.: Beweis)

A *proof* of ψ based on premises $\phi_1, \ldots, \phi_n$ is a finite sequence of propositions that ends in ψ such that each proposition is either a premise or a logical implication of the previous proposition.

- Note: Logical implication rather than logical equivalence!
- Thus,
  - note that it need not be possible to revert a proof!
  - pay close attention to which steps are actual equivalences if you intend to argue both ways!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Rules of Inference

- Aka: proof rules (Dt.: Schlußregeln).
- In addition to the following inference rules for propositional formulas ϕ_1, ϕ_2, all the equivalence rules apply: Each equivalence can be written as two inference rules since they are valid in both directions.

$$\frac{\phi_1 \wedge \phi_2}{\phi_1} \qquad \frac{\phi_1}{\phi_1 \vee \phi_2} \qquad \frac{\phi_1 \Rightarrow \phi_2}{\neg\phi_2 \Rightarrow \neg\phi_1} \quad \text{(Contraposition)}$$

$$\frac{\phi_1 \quad \phi_1 \Rightarrow \phi_2}{\phi_2} \quad \text{(Modus Ponens)} \qquad \frac{\neg\phi_1 \quad \phi_1 \vee \phi_2}{\phi_2} \quad \text{(Modus Tollendo Ponens)}$$

$$\frac{\phi_1 \Rightarrow \phi_2 \quad \neg\phi_1 \Rightarrow \phi_2}{\phi_2} \quad \text{(Rule of Cases)} \qquad \frac{\phi_1 \Rightarrow \phi_2 \quad \phi_2 \Rightarrow \phi_3}{\phi_1 \Rightarrow \phi_3} \quad \text{(Chain Rule)}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Satisfiability

Definition 11 (Satisfiability, Dt.: Erfüllbarkeit)

A formula ϕ is *satisfiable* if there exists at least one truth assignment to the variables of ϕ that makes ϕ true.

Definition 12 (Satisfiability equivalent)

Two formulas are *satisfiability equivalent* if both formulas are either satisfiable or not satisfiable.

Conjunctive Normal Form

- In mathematics, normal forms are canonical representations of objects such that all equivalent objects have the same representation.

Definition 13 (Literal, Dt.: Literal)

A *literal* is a propositional variable or the negation of a propositional variable. A *clause* is a disjunction of literals.

- E.g., if p, q are variables then p and $\neg q$ are literals, and $(p \vee \neg q)$ is a clause.

Definition 14 (Conjunctive normal form, Dt.: konjunktive Normalform)

A propositional formula is in (general) *conjunctive normal form* (CNF) if it is a conjunction of clauses.

- E.g., $\neg p_1 \wedge (p_2 \vee p_5 \vee \neg p_6) \wedge (\neg p_3 \vee p_4 \vee \neg p_6)$ is a CNF formula.

Definition 15 (k-CNF)

A CNF formula is a k-CNF formula if every clause contains at most k literals.

Conjunctive Normal Form

- Note: Some textbooks demand *exactly k literals* rather than *at most k literals*.
- Note: It is common to demand that no variable may appear more than once in a clause.
- Note: For $k \geqslant 3$, a general CNF formula can easily be converted in polynomial time (in the number of literals) into a k-CNF formula with exactly k literals per clause such that no variable appears more than once in a clause and such that the two formulas are satisfiability equivalent.

2 Propositional and Predicate Logic

- Propositional Logic
- Predicate Logic
- Special Quantifiers

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 16 (n-ary Relation, Dt.: n-stellige Relation)

Let $A_1, A_2, \ldots, A_n$ be sets, for some $n \in \mathbb{N}$. An *n-ary relation* $\mathcal{R}$ on $A_1, A_2, \ldots, A_n$ is a subset of their Cartesian product, i.e., $\mathcal{R} \subseteq A_1 \times A_2 \times \cdots \times A_n$.

Definition 17 (n-ary Function, Dt.: n-stellige Funktion)

Let $A_1, A_2, \ldots, A_n, B$ be sets, for some $n \in \mathbb{N}$. An *n-ary function* $\mathcal{F}$ from $A_1 \times A_2 \times \cdots \times A_n$ to B is an $(n+1)$-ary relation on $A_1, A_2, \ldots, A_n, B$ such that for any $(a_1, a_2, \ldots, a_n) \in A_1 \times A_2 \times \cdots \times A_n$ there exists a unique $b \in B$ such that $(a_1, a_2, \ldots, a_n, b) \in \mathcal{F}$.

- It is common to write $y = \mathcal{F}(a_1, \ldots, a_n)$ for "pick y such that $(a_1, \ldots, a_n, y) \in \mathcal{F}$".
- The set $A_1 \times A_2 \times \cdots \times A_n$ is called the *domain* and the set B is called the codomain of $\mathcal{F}$.
- An n-ary relation/function over a set A is a relation/function where $A_1 = A_2 = \ldots = A_n = A$, i.e., $A_1 \times A_2 \times \cdots \times A_n = A^n$. It is also called an n-place relation/function.
- A 1-ary relation/function is called *unary*, and a 2-ary relation/function is called *binary*.

Definition 18 (Predicate, Dt.: Prädikat)

For an n-ary relation $\mathcal{R}$ over A, an n-ary *predicate* over A is the n-ary function $f_{\mathcal{R}} : A^n \to \{T, F\}$, where

$$f_{\mathcal{R}}(a_1, \ldots, a_n) := \begin{cases} T & \text{if } (a_1, \ldots, a_n) \in \mathcal{R}, \\ F & \text{otherwise.} \end{cases}$$

- Thus, a predicate is a Boolean function.
- Note: This is a slight abuse of notation since the symbols ":" and "→" in "$f : M \to N$" actually form already a 3-ary predicate!
- An 1-ary predicate is called *unary*, and a 2-ary predicate is called *binary*.
- A sample unary predicate on $\mathbb{R}$ is

$$\text{“}x \text{ is non-negative”} := \begin{cases} T & \text{if } x \geqslant 0, \\ F & \text{otherwise.} \end{cases}$$

- Dt.: Prädikatenlogik.

Definition 19 (Predicate vocabulary, Dt.: Symbolmenge)

A *predicate vocabulary* consists of

- a set $\mathcal{C}$ of constant symbols,
- a set $\mathcal{F}$ of function symbols,
- a set $\mathcal{V}$ of variables, typically $\{x_1, x_2, \ldots\}$ or $\{a, b, \ldots\}$,
- a set $\mathcal{P}$ of predicate symbols, including the 0-ary predicate symbols (truth values) $\perp, \top$ or F, T,

together with

- logical junctors $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$,
- quantifiers $\exists, \forall$,
- parentheses.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 20 (Term)

A *term* over $(\mathcal{C}, \mathcal{V}, \mathcal{F})$ is defined inductively as follows:

- Every constant $c \in \mathcal{C}$ is a term.
- Every variable $x \in \mathcal{V}$ is a term.
- If $t_1, \ldots, t_n$ are terms and f is an n-ary function symbol then $f(t_1, \ldots, t_n)$ is a term.

- Note: Constants can be thought of as 0-ary function symbols. Thus, a set $\mathcal{C}$ of constants need not be considered when defining the language of predicate logic.

Definition 21 (Formulas)

The set of *formulas* over $(\mathcal{C}, \mathcal{V}, \mathcal{F}, \mathcal{P})$ is defined inductively as follows:

- $\perp$ and $\top$ are formulas.
- If $t_1, \ldots, t_n$ are terms and $P \in \mathcal{P}$ is an n-ary predicate, then $P(t_1, \ldots, t_n)$ is a (so-called *atomic*) formula.
- If ϕ and ψ are formulas then $(\neg\phi)$, $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \Rightarrow \psi)$ and $(\phi \Leftrightarrow \psi)$ are formulas.
- If ϕ is a formula then $(\forall x \;\; \phi)$ and $(\exists x \;\; \phi)$ are formulas. In both cases, the *scope* of the quantifier is given by the formula ϕ to which the quantifier is applied.

Definition 22 (Quantifier-free formula, Dt.: quantorenfreie Formel)

A *quantifier-free formula* is a formula which does not contain a quantifier.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Quantifiers

Definition 23 (Universe of discourse, Dt.: Wertebereich, Universum)

The *universe of discourse* specifies the set of values that the variable x may assume in $(\forall x \;\; \phi)$ and $(\exists x \;\; \phi)$.

Definition 24 (Universal quantifier, Dt.: Allquantor)

$(\forall x \;\; P(x))$ is the statement

"$P(x)$ is true for all x (in the universe of discourse)".

Definition 25 (Existential quantifier, Dt.: Existenzquantor)

$(\exists x \;\; P(x))$ is the statement

"there exists x (in the universe of discourse) such that $P(x)$ is true".

- The notation $(\exists! x \;\; P(x))$ is a convenience short-hand for

  "there exists exactly one x such that $P(x)$ is true",

  i.e., for denoting existence and uniqueness of a suitable x.

Precedence Rules for Quantified Formulas

- No universally accepted precedence rule exists.
- Thus, you have to make your specific order very clear.
- Even better, use parentheses or (significant!) spaces between coherent parts of the expression.

- First-order logic versus higher-order logic: In first-order predicate logic, predicate quantifiers or function quantifiers are not permitted, and variables are the only objects that may be quantified. Also, predicates are not allowed to have predicates as arguments.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Free Variables

Definition 26 (Free variables, Dt.: freie Variable)

The *free variables* of a formula ϕ or a term t, denoted by $FV(\phi)$ and $FV(t)$, are defined inductively as follows:

For a constant $c \in \mathcal{C}$: $FV(c) := \{\}$;

For a variable $x \in \mathcal{V}$: $FV(x) := \{x\}$;

For a term $f(t_1, \dots, t_n)$: $FV(f(t_1, \dots, t_n)) := FV(t_1) \cup \dots \cup FV(t_n)$;

For a formula $P(t_1, \dots, t_n)$: $FV(P(t_1, \dots, t_n)) := FV(t_1) \cup \dots \cup FV(t_n)$;

Also,

$$FV(\bot) := \{\},$$
$$FV(\top) := \{\};$$

For formulas ϕ and ψ:

$$FV((\neg\phi)) := FV(\phi),$$
$$FV((\phi \wedge \psi)) := FV(\phi) \cup FV(\psi),$$
$$FV((\phi \vee \psi)) := FV(\phi) \cup FV(\psi),$$
$$FV((\phi \Rightarrow \psi)) := FV(\phi) \cup FV(\psi),$$
$$FV((\phi \Leftrightarrow \psi)) := FV(\phi) \cup FV(\psi);$$

For a formula ϕ:

$$FV((\forall x \;\; \phi)) := FV(\phi) \setminus \{x\},$$
$$FV((\exists x \;\; \phi)) := FV(\phi) \setminus \{x\}.$$

Bound Variables

Definition 27 (Bound variables, Dt.: gebundene Variable)

The *bound variables* of a formula ϕ or a term t, denoted by $BV(\phi)$ and $BV(t)$, are defined inductively as follows:

$$
\begin{array}{lrcl}
\text{For a constant } c \in \mathcal{C}: & BV(c) & := & \{\}; \\
\text{For a variable } x \in \mathcal{V}: & BV(x) & := & \{\}; \\
\text{For a term } f(t_1, \ldots, t_n): & BV(f(t_1, \ldots, t_n)) & := & \{\}; \\
\text{For a formula } P(t_1, \ldots, t_n): & BV(P(t_1, \ldots, t_n)) & := & \{\}; \\
\text{Also,} & BV(\bot) & := & \{\}, \\
 & BV(\top) & := & \{\}; \\
\text{For formulas } \phi \text{ and } \psi: & BV((\neg\phi)) & := & BV(\phi), \\
 & BV((\phi \wedge \psi)) & := & BV(\phi) \cup BV(\psi), \\
 & BV((\phi \vee \psi)) & := & BV(\phi) \cup BV(\psi), \\
 & BV((\phi \Rightarrow \psi)) & := & BV(\phi) \cup BV(\psi), \\
 & BV((\phi \Leftrightarrow \psi)) & := & BV(\phi) \cup BV(\psi); \\
\text{For a formula } \phi: & BV((\forall x \;\; \phi)) & := & BV(\phi) \cup \{x\}, \\
 & BV((\exists x \;\; \phi)) & := & BV(\phi) \cup \{x\}.
\end{array}
$$

Free and Bound Variables

- Note: Technically speaking, one variable symbol may denote both a free and a bound variable of a formula!
- However, common sense dictates to use a different symbol if a different variable is meant, even if not required by the syntax of predicate logic:
  - Do not use the same symbol for bound and free variables! E.g.,

    $$(P(x) \Rightarrow (\forall x \;\; Q(x)))$$

    is syntactically correct but extremely difficult to parse for a human.
  - Also, do not re-use symbols of bound variables inside nested quantifiers! E.g.,

    $$(\forall x \;\; (P(x) \Rightarrow (\forall x \;\; Q(x))))$$

    is syntactically correct but horrible to parse.

Definition 28 (Sentence, Dt.: geschlossener Ausdruck)

A formula ϕ is a *sentence* if $FV(\phi) = \{\}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Substitutions

Definition 29 (Substitution, Dt.: Ersetzung)

For a formula ϕ, variable x and term t, we obtain the *substitution* of x by t, denoted as $\phi[t/x]$, by replacing each free occurrence of x in ϕ by t.

Definition 30 (Valid substitution, Dt.: gültige Ersetzung)

A substitution of t for x in a formula ϕ is *valid* if and only if no free variable of t ends up being bound in $\phi[t/x]$.

- Not a valid substitution of x: $\phi \equiv (\exists y \in \mathbb{N} \ \ y > 10 \ \ \wedge \ \ x < y)$ and $t := 2y + 5$.
- Again, it is very poor practice to substitute x by t if t contains any variable that also is a bound variable of ϕ!
  $\phi \equiv (\forall z \in \mathbb{N} \ \ z^2 > 0) \ \ \vee \ \ (\exists y \in \mathbb{N} \ \ y > 10 \ \ \wedge \ \ x < y)$ and $t := 2z + 5$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Rules

Theorem 31

Let x be a variable, and ϕ and ψ be formulas which normally contain x as a free variable. Then the following equivalences hold:

De Morgan's laws:
$$(\neg(\forall x \;\; \phi)) \equiv (\exists x \;\; (\neg\phi))$$
$$(\neg(\exists x \;\; \phi)) \equiv (\forall x \;\; (\neg\phi))$$

Trivial conjunction:
$$(\forall x \;\; (\phi \wedge \psi)) \equiv ((\forall x \;\; \phi) \wedge (\forall x \;\; \psi))$$

Only if $x \notin FV(\psi)$:
$$(\forall x \;\; (\phi \wedge \psi)) \equiv ((\forall x \;\; \phi) \wedge \psi)$$
$$(\forall x \;\; (\phi \vee \psi)) \equiv ((\forall x \;\; \phi) \vee \psi)$$

Rules of Inference

- Let x, y be variables and ϕ, ψ be propositional formulas. The following inference rules allow to deduce new formulas.

$$\frac{((\forall x \ \ \phi) \vee (\forall x \ \ \psi))}{(\forall x \ \ (\phi \vee \psi))} \qquad \frac{(\exists x \ \ (\phi \wedge \psi))}{(\exists x \ \ \phi) \wedge (\exists x \ \ \psi)} \qquad \frac{(\exists x \ \ (\forall y \ \ \phi))}{(\forall y \ \ (\exists x \ \ \phi))}$$

- Note that the other direction does not hold for any of these inference rules!
- In addition to these three inference rules all the equivalence rules apply: Each equivalence can be written as two inference rules since they are valid in both directions.

2 Propositional and Predicate Logic

- Propositional Logic
- Predicate Logic
- Special Quantifiers

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Special Quantifiers

- What is the syntactical meaning of

$$\sum_{i=m}^{n} f(i) \quad ?$$

- Apparently, this is the common short-hand notation for

$$\sum_{i=m}^{n} f(i) = \sum_{m \leqslant i \leqslant n} f(i) = \sum_{P(i,m,n)} f(i) = f(m) + f(m+1) + \cdots + f(n-1) + f(n),$$

where $f(i)$ is a term with the free variable i and $(m \leqslant i \leqslant n)$ is a formula with free variables i, m, n, and $P(i, m, n) :\Leftrightarrow [(i \geqslant m) \wedge (i \leqslant n)]$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Special Quantifiers

- Thus, the $\sum$-quantifier takes a predicate, $P(i, m, n)$, and and a term, $f(i)$, and converts it to the new term

$$(f(m) + f(m+1) + f(m+2) + \cdots + f(n-1) + f(n)),$$

  By convention, the variable i is bound inside of $\sum_{i=m}^{n} f(i)$, while m and n remain free.

- Similarly,

$$\prod_{i=m}^{n} f(i) := f(m) \cdot f(m+1) \cdot f(m+2) \cdot \ldots \cdot f(n-1) \cdot f(n).$$

- Again, by convention, if $n < m$ then

$$\sum_{i=m}^{n} f(i) := 0 \quad \text{and} \quad \prod_{i=m}^{n} f(i) := 1.$$

- Union ($\cup$) and intersection ($\cap$) of several sets are further examples of special quantifiers: $\cup_{i=1}^{n} A_i$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Special Quantifiers: Sets

- Standard notation for a set with a finite number of elements: $\{\ \ ,\ \ ,\ldots,\ \ \}$; e.g., $\{1, 2, 3, 4\}$.
- Obvious disadvantage: explicit enumeration of all elements of a set allows to specify only finite sets!
- Infinite sets require us to give a statement A to specify a *characteristic property* of the set:

  $$S := \{x : A\} \qquad \text{or} \qquad S := \{f(x) : A\},$$

  where S shall contain those elements x, or those terms $f(x)$, for some universe of discourse, for which the statement A holds.
- Typically, x will be a free variable of A.
- Thus, the three symbols "{" and ":" and "}" together act as a quantifier that binds x.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Convenient Short-Hand Notations

- The following short-hand notations are convenient for using the predicate $x \in X$ in conjunction with sets or quantifiers:

$$\{x \in X : A(x)\} \quad \text{is a short-hand notation for} \quad \{x : x \in X \land A(x)\}$$

$$(\forall x \in X \ \ A(x)) \quad \text{is a short-hand notation for} \quad (\forall x \ \ (x \in X \Rightarrow A(x)))$$

$$(\exists x \in X \ \ A(x)) \quad \text{is a short-hand notation for} \quad (\exists x \ \ (x \in X \land A(x)))$$

- If x is a typed variable — e.g., a real number — and P is a "simple" unary predicate — e.g., $P(x) :\Leftrightarrow (x > 3)$ — then the following notations are also used commonly:

$$(\forall P(x) \ \ A(x)) \quad \text{is a short-hand notation for} \quad (\forall x \ \ (P(x) \Rightarrow A(x)))$$

$$(\exists P(x) \ \ A(x)) \quad \text{is a short-hand notation for} \quad (\exists x \ \ (P(x) \land A(x)))$$

- Another wide-spread notation is to drop the parentheses:

$$\forall x \ \ P(x) \quad \text{instead of} \quad (\forall x \ \ P(x))$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

3 **Definitions and Theorem Proving**

- Need for Rigorous Analysis
- Definitions
- Syntactical Proof Techniques
- Types of Proofs

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

3 Definitions and Theorem Proving

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Need for Rigorous Analysis

- Suppose that we are to pick a bunch of integers between 1 and n such that no two of them differ by exactly 3 or exactly 5. Let's call these numbers "compatible".

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Need for Rigorous Analysis

- Suppose that we are to pick a bunch of integers between 1 and n such that no two of them differ by exactly 3 or exactly 5. Let's call these numbers "compatible".
- How many compatible numbers can you pick for $n := 20$?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Need for Rigorous Analysis

- Suppose that we are to pick a bunch of integers between 1 and n such that no two of them differ by exactly 3 or exactly 5. Let's call these numbers "compatible".
- How many compatible numbers can you pick for $n := 20$?
- Intuition: Start at 1 and scan the integers from 1 to 20, successively picking those integers which are compatible with all integers picked previously:

1 2 3 ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ 9 10 11 ~~12~~ ~~13~~ ~~14~~ ~~15~~ ~~16~~ 17 18 19 ~~20~~

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Need for Rigorous Analysis

- Suppose that we are to pick a bunch of integers between 1 and n such that no two of them differ by exactly 3 or exactly 5. Let's call these numbers "compatible".
- How many compatible numbers can you pick for $n := 20$?
- Intuition: Start at 1 and scan the integers from 1 to 20, successively picking those integers which are compatible with all integers picked previously:

1 2 3 ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ 9 10 11 ~~12~~ ~~13~~ ~~14~~ ~~15~~ ~~16~~ 17 18 19 ~~20~~

- We get 9 compatible integers. Our selection scheme makes it plausible that this is indeed the maximum number of compatible integers within $\{1, 2, 3, \ldots, 19, 20\}$. Right?

Need for Rigorous Analysis

- Suppose that we are to pick a bunch of integers between 1 and n such that no two of them differ by exactly 3 or exactly 5. Let's call these numbers "compatible".
- How many compatible numbers can you pick for $n := 20$?
- Intuition: Start at 1 and scan the integers from 1 to 20, successively picking those integers which are compatible with all integers picked previously:

1 2 3 ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ 9 10 11 ~~12~~ ~~13~~ ~~14~~ ~~15~~ ~~16~~ 17 18 19 ~~20~~

- We get 9 compatible integers. Our selection scheme makes it plausible that this is indeed the maximum number of compatible integers within $\{1, 2, 3, \ldots, 19, 20\}$. Right?
- Well, what about the following 10 integers?

1 3 5 7 9 11 13 15 17 19

Need for Rigorous Analysis

- Suppose that we are to pick a bunch of integers between 1 and n such that no two of them differ by exactly 3 or exactly 5. Let's call these numbers "compatible".
- How many compatible numbers can you pick for $n := 20$?
- Intuition: Start at 1 and scan the integers from 1 to 20, successively picking those integers which are compatible with all integers picked previously:

1 2 3 ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ 9 10 11 ~~12~~ ~~13~~ ~~14~~ ~~15~~ ~~16~~ 17 18 19 ~~20~~

- We get 9 compatible integers. Our selection scheme makes it plausible that this is indeed the maximum number of compatible integers within $\{1, 2, 3, \ldots, 19, 20\}$. Right?
- Well, what about the following 10 integers?

1 3 5 7 9 11 13 15 17 19

- Oops! Why should we believe that we can't find 11 or more compatible integers within $\{1, 2, 3, \ldots, 19, 20\}$?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Need for Rigorous Analysis

- Suppose that we are to pick a bunch of integers between 1 and n such that no two of them differ by exactly 3 or exactly 5. Let's call these numbers "compatible".
- How many compatible numbers can you pick for $n := 20$?
- Intuition: Start at 1 and scan the integers from 1 to 20, successively picking those integers which are compatible with all integers picked previously:

1 2 3 ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ 9 10 11 ~~12~~ ~~13~~ ~~14~~ ~~15~~ ~~16~~ 17 18 19 ~~20~~

- We get 9 compatible integers. Our selection scheme makes it plausible that this is indeed the maximum number of compatible integers within $\{1, 2, 3, \ldots, 19, 20\}$. Right?
- Well, what about the following 10 integers?

1 3 5 7 9 11 13 15 17 19

- Oops! Why should we believe that we can't find 11 or more compatible integers within $\{1, 2, 3, \ldots, 19, 20\}$?
- The answer is provided by the pigeonhole principle (Thm. 147): Every subset of compatible integers of $\{1, 2, 3, \ldots, 19, 20\}$ can contain at most one of each of the following 10 pairs:

1	2	3	4	5	11	12	13	14	15
6	7	8	9	10	16	17	18	19	20

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Need for Rigorous Analysis

Lesson Learned

1. An intuitively appealing argument or approach is no substitute for a formal proof: Intuition might be wrong!

Need for Rigorous Analysis

Lesson Learned

1. An intuitively appealing argument or approach is no substitute for a formal proof: Intuition might be wrong!
2. Consent of the majority is no substitute for a formal proof either.

Lesson Learned

1. An intuitively appealing argument or approach is no substitute for a formal proof: Intuition might be wrong!
2. Consent of the majority is no substitute for a formal proof either.
3. The so-called *greedy approach* need not always lead to the best solution for an optimization problem.

Need for Rigorous Analysis

Lesson Learned

1. An intuitively appealing argument or approach is no substitute for a formal proof: Intuition might be wrong!
2. Consent of the majority is no substitute for a formal proof either.
3. The so-called *greedy approach* need not always lead to the best solution for an optimization problem.

Proofs Needed!

Even though proofs and a rigorous formal analysis might seem boring (difficult, mind-boggling, mind-numbing, unnecessary, . . .) there is just no way around them if we want to be sure that our findings are correct!

Need for Rigorous Analysis

Lesson Learned

1. An intuitively appealing argument or approach is no substitute for a formal proof: Intuition might be wrong!
2. Consent of the majority is no substitute for a formal proof either.
3. The so-called *greedy approach* need not always lead to the best solution for an optimization problem.

Proofs Needed!

Even though proofs and a rigorous formal analysis might seem boring (difficult, mind-boggling, mind-numbing, unnecessary, . . .) there is just no way around them if we want to be sure that our findings are correct!

- So, be prepared for at least some boring (difficult, mind-boggling, mind-numbing, unnecessary, . . .) proofs! ☹

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

3 Definitions and Theorem Proving

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

How to Deal with Formal Statements . . .

- Experience tells me that students find it difficult
  - to parse and understand formal statements,
  - to formulate meaningful definitions,
  - to write clean and mathematically correct proofs.

- Experience tells me that students find it difficult
  - to parse and understand formal statements,
  - to formulate meaningful definitions,
  - to write clean and mathematically correct proofs.
- Hence, prior to diving into other areas of Discrete Mathematics, we start with taking a practical look at the formal nuts and bolts of mathematical reasoning.

How to Deal with Formal Statements . . .

- Experience tells me that students find it difficult
  - to parse and understand formal statements,
  - to formulate meaningful definitions,
  - to write clean and mathematically correct proofs.
- Hence, prior to diving into other areas of Discrete Mathematics, we start with taking a practical look at the formal nuts and bolts of mathematical reasoning.
- In the following slides on definitions and theorem proving we pre-suppose an “intuitive” understanding of natural numbers, integers, reals, etc.; e.g., as taught in school.

How to Deal with Formal Statements . . .

- Experience tells me that students find it difficult
  - to parse and understand formal statements,
  - to formulate meaningful definitions,
  - to write clean and mathematically correct proofs.
- Hence, prior to diving into other areas of Discrete Mathematics, we start with taking a practical look at the formal nuts and bolts of mathematical reasoning.
- In the following slides on definitions and theorem proving we pre-suppose an "intuitive" understanding of natural numbers, integers, reals, etc.; e.g., as taught in school.
- We will later on put these number systems on slightly more formal grounds.

Definitions

- We distinguish between *explicit* and *recursive* definitions.
- An explicit definition relates an entity that is to be specified ("*definiendum*") to an already known entity ("*definiens*").

Definitions

- We distinguish between *explicit* and *recursive* definitions.
- An explicit definition relates an entity that is to be specified ("*definiendum*") to an already known entity ("*definiens*").
- Explicit definition of a function f with n arguments:

$$f(x_1, x_2, \ldots, x_n) := t,$$

where the term t (normally) contains $x_1, x_2, \ldots, x_n$ as free variables.

Definitions

- We distinguish between *explicit* and *recursive* definitions.
- An explicit definition relates an entity that is to be specified ("*definiendum*") to an already known entity ("*definiens*").
- Explicit definition of a function f with n arguments:

$$f(x_1, x_2, \ldots, x_n) := t,$$

where the term t (normally) contains $x_1, x_2, \ldots, x_n$ as free variables.

- E.g., $f(x, y) := \sqrt{x^2 + y^2}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definitions

- We distinguish between *explicit* and *recursive* definitions.
- An explicit definition relates an entity that is to be specified ("*definiendum*") to an already known entity ("*definiens*").
- Explicit definition of a function f with n arguments:

$$f(x_1, x_2, \ldots, x_n) := t,$$

where the term t (normally) contains $x_1, x_2, \ldots, x_n$ as free variables.

- E.g., $f(x, y) := \sqrt{x^2 + y^2}$.
- Explicit definition of a predicate P with n arguments:

$$P(x_1, x_2, \ldots, x_n) :\Leftrightarrow A,$$

where the statement A (normally) contains $x_1, x_2, \ldots, x_n$ as free variables.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definitions

- We distinguish between *explicit* and *recursive* definitions.
- An explicit definition relates an entity that is to be specified ("*definiendum*") to an already known entity ("*definiens*").
- Explicit definition of a function f with n arguments:

$$f(x_1, x_2, \ldots, x_n) := t,$$

  where the term t (normally) contains $x_1, x_2, \ldots, x_n$ as free variables.
- E.g., $f(x, y) := \sqrt{x^2 + y^2}$.
- Explicit definition of a predicate P with n arguments:

$$P(x_1, x_2, \ldots, x_n) :\Leftrightarrow A,$$

  where the statement A (normally) contains $x_1, x_2, \ldots, x_n$ as free variables.
- E.g., $P(x, y) :\Leftrightarrow (x < y)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definitions

- We distinguish between *explicit* and *recursive* definitions.
- An explicit definition relates an entity that is to be specified ("*definiendum*") to an already known entity ("*definiens*").
- Explicit definition of a function f with n arguments:

$$f(x_1, x_2, \ldots, x_n) := t,$$

where the term t (normally) contains $x_1, x_2, \ldots, x_n$ as free variables.
- E.g., $f(x, y) := \sqrt{x^2 + y^2}$.
- Explicit definition of a predicate P with n arguments:

$$P(x_1, x_2, \ldots, x_n) :\Leftrightarrow A,$$

where the statement A (normally) contains $x_1, x_2, \ldots, x_n$ as free variables.
- E.g., $P(x, y) :\Leftrightarrow (x < y)$.

Warning

The definiendum does not occur in the definiens of an explicit definition of a function f or predicate P! That is, the symbols f and P do not appear on the right-hand side.

Definitions: The Symbols “$:=$” and “$:\Leftrightarrow$”

- It is common to use the special symbols $:=$ and $:\Leftrightarrow$ for definitions, where the symbol “:” appears on the side of the definiendum.
- Thus, one can also write $=:$ or $\Leftrightarrow:$ to indicate that the definiendum is on the right-hand side.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definitions: The Symbols “$:=$” and “$:\Leftrightarrow$”

- It is common to use the special symbols $:=$ and $:\Leftrightarrow$ for definitions, where the symbol “:” appears on the side of the definiendum.
- Thus, one can also write $=:$ or $\Leftrightarrow:$ to indicate that the definiendum is on the right-hand side.
- Using $=:$ and $\Leftrightarrow:$ is very good practice since
  - it makes it immediately obvious to the reader that what follows constitutes a definition rather than some lemma or claim,
  - it shows beyond doubt what is the definiens and what is the definiendum, and
  - it forces the author to decide whether or not something is a consequence of prior knowledge or some newly introduced entity.

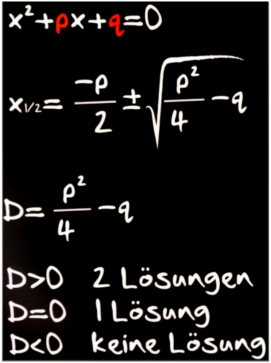
UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definitions: The Symbols “$:=$” and “$:\Leftrightarrow$”

- It is common to use the special symbols $:=$ and $:\Leftrightarrow$ for definitions, where the symbol “:” appears on the side of the definiendum.
- Thus, one can also write $=:$ or $\Leftrightarrow:$ to indicate that the definiendum is on the right-hand side.
- Using $=:$ and $\Leftrightarrow:$ is very good practice since
  - it makes it immediately obvious to the reader that what follows constitutes a definition rather than some lemma or claim,
  - it shows beyond doubt what is the definiens and what is the definiendum, and
  - it forces the author to decide whether or not something is a consequence of prior knowledge or some newly introduced entity.
- However, if “$:=$” or “$:\Leftrightarrow$” are used once in a text then they have to be used for absolutely all definitions in that text!!

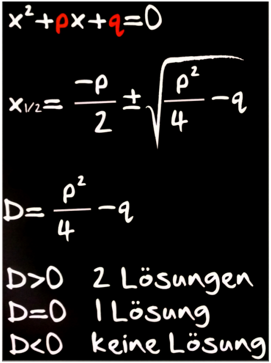
Definitions: The Symbols ":=" and ":⇔"

- Poster seen in a tutoring institute at Salzburg:



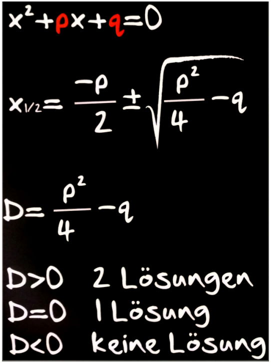
UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Poster seen in a tutoring institute at Salzburg:



  - Can $x_{1/2}$ be derived?
  - Can D be derived?

- Poster seen in a tutoring institute at Salzburg:

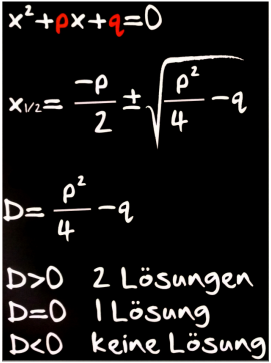


  - Can $x_{1/2}$ be derived?
  - Can D be derived?

- Better formalism:
  - If x_1, x_2 are the roots of the second-degree polynomial equation $x^2 + px + q = 0$, with $p, q \in \mathbb{R}$ and unknown $x \in \mathbb{R}$, then

$$x_{1/2} = -\frac{p}{2} \pm \sqrt{\frac{p^2}{4} - q}.$$

- Poster seen in a tutoring institute at Salzburg:



  - Can $x_{1/2}$ be derived?
  - Can D be derived?

- Better formalism:
  - If x_1, x_2 are the roots of the second-degree polynomial equation $x^2 + px + q = 0$, with $p, q \in \mathbb{R}$ and unknown $x \in \mathbb{R}$, then

$$x_{1/2} = -\frac{p}{2} \pm \sqrt{\frac{p^2}{4} - q}.$$

  - With $D := p^2 - 4q$ we get

$$D \left\{ \begin{array}{c} > \\ = \\ < \end{array} \right\} 0 : \left\{ \begin{array}{l} \text{2 distinct real roots,} \\ \text{1 real root,} \\ \text{0 real roots.} \end{array} \right.$$

Recursive Definitions

- Aka: *Inductive* definition.
- How can we state

  x is ancestor of y if x is parent of y, or if x is parent of parent of y, or if x is parent of parent of parent of y, or if ...

  in a form that does not need to resort to an ellipsis "..." ?

Recursive Definitions

- Aka: *Inductive* definition.
- How can we state

  x is ancestor of y if x is parent of y, or if x is parent of parent of y, or if x is parent of parent of parent of y, or if . . .

  in a form that does not need to resort to an ellipsis "..." ?
- Recursive definitions (typically) consist of two parts:
  - a *basis* in which the definiendum does not occur in the definiens, and
  - an *inductive step* in which the definiendum does occur.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Recursive Definitions

- Aka: *Inductive* definition.
- How can we state

  x is ancestor of y if x is parent of y, or if x is parent of parent of y, or if x is parent of parent of parent of y, or if . . .

  in a form that does not need to resort to an ellipsis ". . ." ?
- Recursive definitions (typically) consist of two parts:
  - a *basis* in which the definiendum does not occur in the definiens, and
  - an *inductive step* in which the definiendum does occur.
- E.g.,

  x is an ancestor of y if x is parent of y or x is ancestor of parent of y.

Recursive Definitions

- Aka: *Inductive* definition.
- How can we state

  x is ancestor of y if x is parent of y, or if x is parent of parent of y, or if x is parent of parent of parent of y, or if . . .

  in a form that does not need to resort to an ellipsis "..." ?
- Recursive definitions (typically) consist of two parts:
  - a *basis* in which the definiendum does not occur in the definiens, and
  - an *inductive step* in which the definiendum does occur.
- E.g.,

  x is an ancestor of y if x is parent of y or x is ancestor of parent of y.

Warning

To avoid infinite circles, the definiendum must not occur in the basis!

Recursive Definitions: Sum and Product

Definition 32 (Sum and product)

Consider k real numbers $a_1, a_2, \ldots, a_k \in \mathbb{R}$, together with some $m, n \in \mathbb{N}$ such that $1 \leqslant m, n \leqslant k$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Recursive Definitions: Sum and Product

Definition 32 (Sum and product)

Consider k real numbers $a_1, a_2, \ldots, a_k \in \mathbb{R}$, together with some $m, n \in \mathbb{N}$ such that $1 \leqslant m, n \leqslant k$. Then

$$\sum_{i=m}^{n} a_i \;:=\; \begin{cases} 0 & \text{if} \quad n < m, \\ a_m & \text{if} \quad n = m, \\ \left(\sum_{i=m}^{n-1} a_i\right) + a_n & \text{if} \quad n > m, \end{cases}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 32 (Sum and product)

Consider k real numbers $a_1, a_2, \ldots, a_k \in \mathbb{R}$, together with some $m, n \in \mathbb{N}$ such that $1 \leqslant m, n \leqslant k$. Then

$$\sum_{i=m}^{n} a_i \;:=\; \begin{cases} 0 & \text{if} \quad n < m, \\ a_m & \text{if} \quad n = m, \\ \left(\sum_{i=m}^{n-1} a_i\right) + a_n & \text{if} \quad n > m, \end{cases}$$

and

$$\prod_{i=m}^{n} a_i \;:=\; \begin{cases} 1 & \text{if} \quad n < m, \\ a_m & \text{if} \quad n = m, \\ \left(\prod_{i=m}^{n-1} a_i\right) \cdot a_n & \text{if} \quad n > m. \end{cases}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Recursive Definitions: Sum and Product

Definition 32 (Sum and product)

Consider k real numbers $a_1, a_2, \ldots, a_k \in \mathbb{R}$, together with some $m, n \in \mathbb{N}$ such that $1 \leqslant m, n \leqslant k$. Then

$$\sum_{i=m}^{n} a_i := \begin{cases} 0 & \text{if} \quad n < m, \\ a_m & \text{if} \quad n = m, \\ \left(\sum_{i=m}^{n-1} a_i\right) + a_n & \text{if} \quad n > m, \end{cases}$$

and

$$\prod_{i=m}^{n} a_i := \begin{cases} 1 & \text{if} \quad n < m, \\ a_m & \text{if} \quad n = m, \\ \left(\prod_{i=m}^{n-1} a_i\right) \cdot a_n & \text{if} \quad n > m. \end{cases}$$

- The definitions for $n < m$ are convenience settings that have turned out to be useful in practice.

Definition 33 (Factorial, Dt.: Fakultät, Faktorielle)

For $n \in \mathbb{N}_0$,

$$n! := \begin{cases} 1 & \text{if } n \leqslant 1, \\ n \cdot (n-1)! & \text{if } n > 1. \end{cases}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 33 (Factorial, Dt.: Fakultät, Faktorielle)

For $n \in \mathbb{N}_0$,

$$n! := \begin{cases} 1 & \text{if } n \leqslant 1, \\ n \cdot (n-1)! & \text{if } n > 1. \end{cases}$$

n	0	1	2	3	4	5	6	7	8	9	10
$n!$	1	1	2	6	24	120	720	5 040	40 320	362 880	3 628 800

Recursive Definitions: Factorial and Fibonacci

Definition 33 (Factorial, Dt.: Fakultät, Faktorielle)

For $n \in \mathbb{N}_0$,

$$n! := \begin{cases} 1 & \text{if } n \leqslant 1, \\ n \cdot (n-1)! & \text{if } n > 1. \end{cases}$$

n	0	1	2	3	4	5	6	7	8	9	10
$n!$	1	1	2	6	24	120	720	5 040	40 320	362 880	3 628 800

Definition 34 (Fibonacci numbers)

For $n \in \mathbb{N}_0$,

$$F_n := \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n \geqslant 2. \end{cases}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Recursive Definitions: Factorial and Fibonacci

Definition 33 (Factorial, Dt.: Fakultät, Faktorielle)

For $n \in \mathbb{N}_0$,

$$n! := \begin{cases} 1 & \text{if } n \leqslant 1, \\ n \cdot (n-1)! & \text{if } n > 1. \end{cases}$$

n	0	1	2	3	4	5	6	7	8	9	10
$n!$	1	1	2	6	24	120	720	5 040	40 320	362 880	3 628 800

Definition 34 (Fibonacci numbers)

For $n \in \mathbb{N}_0$,

$$F_n := \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n \geqslant 2. \end{cases}$$

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
F_n	0	1	1	2	3	5	8	13	21	34	55	89	144	233	377	610

Fibonacci Numbers

- The Fibonacci numbers are named after Leonardo da Pisa (1180?–1241?), aka “figlio di Bonaccio”.
- The Fibonacci numbers have been studied extensively; they exhibit lots of interesting mathematical properties. For instance,

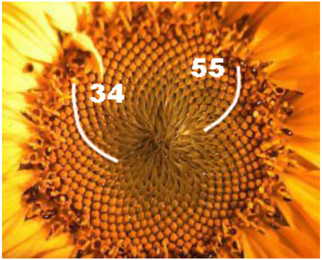
$$\lim_{n\to\infty} \frac{F_{n+1}}{F_n} = \phi, \quad \text{where } \phi := \frac{1+\sqrt{5}}{2} \text{ is known as } \textit{golden ratio}.$$

Fibonacci Numbers

- The Fibonacci numbers are named after Leonardo da Pisa (1180?–1241?), aka “figlio di Bonaccio”.
- The Fibonacci numbers have been studied extensively; they exhibit lots of interesting mathematical properties. For instance,

$$\lim_{n\to\infty} \frac{F_{n+1}}{F_n} = \phi, \quad \text{where } \phi := \frac{1+\sqrt{5}}{2} \text{ is known as } \textit{golden ratio}.$$

- The Fibonacci numbers are also found in nature: E.g., the numbers of CW/CCW spirals of sunflower heads are given by subsequent Fibonacci numbers.



[Image credit: Wikipedia.]

Words

- Consider an arbitrary (but fixed) finite set Σ. We call it the *alphabet*; the individual elements of Σ are called *symbols* or *characters*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Words

- Consider an arbitrary (but fixed) finite set Σ. We call it the *alphabet*; the individual elements of Σ are called *symbols* or *characters*.
- E.g., $\Sigma := \{a, b, c, \ldots, x, y, z\}$ or $\Sigma := \{0, 1\}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Consider an arbitrary (but fixed) finite set Σ. We call it the *alphabet*; the individual elements of Σ are called *symbols* or *characters*.
- E.g., $\Sigma := \{a, b, c, \ldots, x, y, z\}$ or $\Sigma := \{0, 1\}$.

Definition 35 (Word)

Let Σ be a finite set. The set Σ^* of *words* over Σ is defined follows:

1. Base clause: The empty word, denoted by the Greek letter ϵ, belongs to Σ^*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Consider an arbitrary (but fixed) finite set Σ. We call it the *alphabet*; the individual elements of Σ are called *symbols* or *characters*.
- E.g., $\Sigma := \{a, b, c, \ldots, x, y, z\}$ or $\Sigma := \{0, 1\}$.

Definition 35 (Word)

Let Σ be a finite set. The set Σ^* of *words* over Σ is defined follows:

1. Base clause: The empty word, denoted by the Greek letter ϵ, belongs to Σ^*.
2. Recursion clause: For all $a \in \Sigma$ and all $\sigma \in \Sigma^*$, the ordered pair (a, σ) belongs to Σ^*.

- Consider an arbitrary (but fixed) finite set Σ. We call it the *alphabet*; the individual elements of Σ are called *symbols* or *characters*.
- E.g., $\Sigma := \{a, b, c, \ldots, x, y, z\}$ or $\Sigma := \{0, 1\}$.

Definition 35 (Word)

Let Σ be a finite set. The set Σ^* of *words* over Σ is defined follows:

1. Base clause: The empty word, denoted by the Greek letter ϵ, belongs to Σ^*.
2. Recursion clause: For all $a \in \Sigma$ and all $\sigma \in \Sigma^*$, the ordered pair (a, σ) belongs to Σ^*.
3. Extremal clause: A word is in Σ^* if it is ϵ or if it can be constructed from ϵ via a finite number of applications of the recursion clause.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Words

- Consider an arbitrary (but fixed) finite set Σ. We call it the *alphabet*; the individual elements of Σ are called *symbols* or *characters*.
- E.g., $\Sigma := \{a, b, c, \ldots, x, y, z\}$ or $\Sigma := \{0, 1\}$.

Definition 35 (Word)

Let Σ be a finite set. The set Σ^* of *words* over Σ is defined follows:

1. Base clause: The empty word, denoted by the Greek letter ϵ, belongs to Σ^*.
2. Recursion clause: For all $a \in \Sigma$ and all $\sigma \in \Sigma^*$, the ordered pair (a, σ) belongs to Σ^*.
3. Extremal clause: A word is in Σ^* if it is ϵ or if it can be constructed from ϵ via a finite number of applications of the recursion clause.

- Aka *string* (Dt. Zeichenkette). The set Σ^* of all words over Σ is known as *Kleene closure* of Σ.

Words

- Consider an arbitrary (but fixed) finite set Σ. We call it the *alphabet*; the individual elements of Σ are called *symbols* or *characters*.
- E.g., $\Sigma := \{a, b, c, \ldots, x, y, z\}$ or $\Sigma := \{0, 1\}$.

Definition 35 (Word)

Let Σ be a finite set. The set Σ^* of *words* over Σ is defined follows:

1. Base clause: The empty word, denoted by the Greek letter ϵ, belongs to Σ^*.
2. Recursion clause: For all $a \in \Sigma$ and all $\sigma \in \Sigma^*$, the ordered pair (a, σ) belongs to Σ^*.
3. Extremal clause: A word is in Σ^* if it is ϵ or if it can be constructed from ϵ via a finite number of applications of the recursion clause.

- Aka *string* (Dt. Zeichenkette). The set Σ^* of all words over Σ is known as *Kleene closure* of Σ.
- Of course, in order to avoid confusion, ϵ is not allowed to be a character of Σ.

Words

- Consider an arbitrary (but fixed) finite set Σ. We call it the *alphabet*; the individual elements of Σ are called *symbols* or *characters*.
- E.g., $\Sigma := \{a, b, c, \ldots, x, y, z\}$ or $\Sigma := \{0, 1\}$.

Definition 35 (Word)

Let Σ be a finite set. The set Σ^* of *words* over Σ is defined follows:

1. Base clause: The empty word, denoted by the Greek letter ϵ, belongs to Σ^*.
2. Recursion clause: For all $a \in \Sigma$ and all $\sigma \in \Sigma^*$, the ordered pair (a, σ) belongs to Σ^*.
3. Extremal clause: A word is in Σ^* if it is ϵ or if it can be constructed from ϵ via a finite number of applications of the recursion clause.

- Aka *string* (Dt. Zeichenkette). The set Σ^* of all words over Σ is known as *Kleene closure* of Σ.
- Of course, in order to avoid confusion, ϵ is not allowed to be a character of Σ.
- It is important to note that every element of Σ^* is a finite sequence of zero or more characters (if we disregard the parentheses and commas) but that Σ^* itself is an infinite set containing words of every possible finite length.

Words: Length and Concatenation

Definition 36 (Length of a word)

Let Σ be a finite set. The *length* of a word σ over Σ is defined as follows:

$$|\sigma| := \begin{cases} 0 & \text{if } \sigma = \epsilon, \\ 1 + |\sigma'| & \text{if } \sigma = (a, \sigma') \text{ for some } a \in \Sigma \text{ and } \sigma' \in \Sigma^*. \end{cases}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 36 (Length of a word)

Let Σ be a finite set. The *length* of a word σ over Σ is defined as follows:

$$|\sigma| := \begin{cases} 0 & \text{if } \sigma = \epsilon, \\ 1 + |\sigma'| & \text{if } \sigma = (a, \sigma') \text{ for some } a \in \Sigma \text{ and } \sigma' \in \Sigma^*. \end{cases}$$

Definition 37 (Concatenation)

Let Σ be a finite set. *Concatenation* of two words σ_1, σ_2 over Σ, denoted by $\sigma_1 \bullet \sigma_2$, is defined as follows:

$$\sigma_1 \bullet \sigma_2 := \begin{cases} \sigma_2 & \text{if } \sigma_1 = \epsilon, \\ (a, \sigma_1' \bullet \sigma_2) & \text{if } \sigma_1 = (a, \sigma_1') \text{ for some } a \in \Sigma \text{ and } \sigma_1' \in \Sigma^*. \end{cases}$$

Definition 36 (Length of a word)

Let Σ be a finite set. The *length* of a word σ over Σ is defined as follows:

$$|\sigma| := \begin{cases} 0 & \text{if } \sigma = \epsilon, \\ 1 + |\sigma'| & \text{if } \sigma = (a, \sigma') \text{ for some } a \in \Sigma \text{ and } \sigma' \in \Sigma^*. \end{cases}$$

Definition 37 (Concatenation)

Let Σ be a finite set. *Concatenation* of two words σ_1, σ_2 over Σ, denoted by $\sigma_1 \bullet \sigma_2$, is defined as follows:

$$\sigma_1 \bullet \sigma_2 := \begin{cases} \sigma_2 & \text{if } \sigma_1 = \epsilon, \\ (a, \sigma_1' \bullet \sigma_2) & \text{if } \sigma_1 = (a, \sigma_1') \text{ for some } a \in \Sigma \text{ and } \sigma_1' \in \Sigma^*. \end{cases}$$

- In practice it is a convention to drop the ordered-pair notation and to write $a\sigma$ rather than (a, σ). E.g., *word* rather than $(w, (o, (r, (d, \epsilon))))$.
- Similarly, one writes *word* rather than $wo \bullet rd$. (This simplification is justified by the fact that the binary operator $\bullet$ is associative.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Caveats When Formulating Definitions

- Definitions like

$$P(x, y, z) :\Leftrightarrow (x < 2y) \qquad \text{or} \qquad P(x) :\Leftrightarrow (x < 2y)$$

can be seen as syntactically correct but they are semantically problematic!

Caveats When Formulating Definitions

- Definitions like

$$P(x, y, z) :\Leftrightarrow (x < 2y) \qquad \text{or} \qquad P(x) :\Leftrightarrow (x < 2y)$$

can be seen as syntactically correct but they are semantically problematic!

Rule of thumb

All arguments of the definiendum have to appear as free variables in the definiens, and vice versa!

Caveats When Formulating Definitions

- Definitions like

$$P(x, y, z) :\Leftrightarrow (x < 2y) \qquad \text{or} \qquad P(x) :\Leftrightarrow (x < 2y)$$

can be seen as syntactically correct but they are semantically problematic!

Rule of thumb

All arguments of the definiendum have to appear as free variables in the definiens, and vice versa!

Warning

An entity introduced in a definition has to be free of internal inconsistencies, and free of contradictions with prior facts.

Caveats When Formulating Definitions

- Definitions like

$$P(x, y, z) :\Leftrightarrow (x < 2y) \qquad \text{or} \qquad P(x) :\Leftrightarrow (x < 2y)$$

can be seen as syntactically correct but they are semantically problematic!

Rule of thumb

All arguments of the definiendum have to appear as free variables in the definiens, and vice versa!

Warning

An entity introduced in a definition has to be free of internal inconsistencies, and free of contradictions with prior facts.

- E.g., assume that for $\frac{m}{n}, \frac{p}{q} \in \mathbb{Q}$, with $m, p, n, q \in \mathbb{N}$, we define

$$\frac{m}{n} \sharp \frac{p}{q} := \frac{m+p}{n+q}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Caveats When Formulating Definitions

- Definitions like

$$P(x,y,z) :\Leftrightarrow (x < 2y) \qquad \text{or} \qquad P(x) :\Leftrightarrow (x < 2y)$$

can be seen as syntactically correct but they are semantically problematic!

Rule of thumb

All arguments of the definiendum have to appear as free variables in the definiens, and vice versa!

Warning

An entity introduced in a definition has to be free of internal inconsistencies, and free of contradictions with prior facts.

- E.g., assume that for $\frac{m}{n}, \frac{p}{q} \in \mathbb{Q}$, with $m, p, n, q \in \mathbb{N}$, we define

$$\frac{m}{n} \sharp \frac{p}{q} := \frac{m+p}{n+q}.$$

- Then $\frac{1}{1} \sharp \frac{2}{3} = \frac{3}{4}$, but $\frac{2}{2} \sharp \frac{2}{3} = \frac{4}{5}$.
- Since $\frac{1}{1} = \frac{2}{2}$, we conclude $\frac{4}{5} = \frac{3}{4}$, and, thus, $0 = 1$. Yikes!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

3 Definitions and Theorem Proving

- Need for Rigorous Analysis
- Definitions
- Syntactical Proof Techniques
  - Syntax and Proofs
  - Equivalence Transformations
- Types of Proofs

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 38 (Proof, Dt.: Beweis)

To *prove* a statement means to derive it from axioms (or postulates) and other previously established theorems by means of rules of logic.

Terminology

Definition 38 (Proof, Dt.: Beweis)

To *prove* a statement means to derive it from axioms (or postulates) and other previously established theorems by means of rules of logic.

- Note the difference between the English words “the proof” and “to prove”.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Terminology

Definition 38 (Proof, Dt.: Beweis)

To *prove* a statement means to derive it from axioms (or postulates) and other previously established theorems by means of rules of logic.

- Note the difference between the English words “the proof” and “to prove”.
- Common symbols to mark the end of a proof: $\square$, *qued* or *qed* (as an abbreviation for the Latin words “*quod erat demonstrandum*”, i.e., for ”what was to be shown”).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 38 (Proof, Dt.: Beweis)

To *prove* a statement means to derive it from axioms (or postulates) and other previously established theorems by means of rules of logic.

- Note the difference between the English words “the proof” and “to prove”.
- Common symbols to mark the end of a proof: $\square$, *qued* or *qed* (as an abbreviation for the Latin words “*quod erat demonstrandum*”, i.e., for ”what was to be shown”).

Definition 39 (Theorem, Dt.: Satz, Theorem)

A statement is a *theorem* if it has been proved. If the statement is of the form $H \Rightarrow C$ then we call H the *hypothesis* and C the *conclusion*.

Terminology

Definition 38 (Proof, Dt.: Beweis)

To *prove* a statement means to derive it from axioms (or postulates) and other previously established theorems by means of rules of logic.

- Note the difference between the English words “the proof” and “to prove”.
- Common symbols to mark the end of a proof: $\square$, *qued* or *qed* (as an abbreviation for the Latin words “*quod erat demonstrandum*”, i.e., for ”what was to be shown”).

Definition 39 (Theorem, Dt.: Satz, Theorem)

A statement is a *theorem* if it has been proved. If the statement is of the form $H \Rightarrow C$ then we call H the *hypothesis* and C the *conclusion*.

- Of course, a theorem may involve quantifiers. E.g., $\forall x \ \big(H(x) \Rightarrow C(x)\big)$.
- Depending on the importance of the result, terms like *lemma* (Dt.: Lemma, Hilfssatz) or *corollary* (Dt.: Korollar) are also used instead of “theorem”.

Definition 38 (Proof, Dt.: Beweis)

To *prove* a statement means to derive it from axioms (or postulates) and other previously established theorems by means of rules of logic.

- Note the difference between the English words "the proof" and "to prove".
- Common symbols to mark the end of a proof: $\square$, *qued* or *qed* (as an abbreviation for the Latin words "*quod erat demonstrandum*", i.e., for "what was to be shown").

Definition 39 (Theorem, Dt.: Satz, Theorem)

A statement is a *theorem* if it has been proved. If the statement is of the form $H \Rightarrow C$ then we call H the *hypothesis* and C the *conclusion*.

- Of course, a theorem may involve quantifiers. E.g., $\forall x \ \big(H(x) \Rightarrow C(x)\big)$.
- Depending on the importance of the result, terms like *lemma* (Dt.: Lemma, Hilfssatz) or *corollary* (Dt.: Korollar) are also used instead of "theorem".
- A *conjecture* is a statement which has not yet been proved or disproved.
- The status of a conjecture may remain unknown for decades or even centuries: Fermat's Last Theorem was stated by Pierre de Fermat in 1637 and proved by Andrew Wiles (with the help of Richard Taylor) in 1993–1995.

Syntactical Proof Techniques

- *Syntactical proof techniques* are proof techniques that are based on the analysis of the syntactical structure of a statement.
- Syntactical proof techniques allow us to reason about statements and to simplify statements with no or very little "understanding" of their mathematical meaning.
- In particular, syntactical proof techniques allow us to split complicated proofs into simpler proofs, without any need for an ingenious idea for how to carry out a specific proof.

Syntactical Proof Techniques

- *Syntactical proof techniques* are proof techniques that are based on the analysis of the syntactical structure of a statement.
- Syntactical proof techniques allow us to reason about statements and to simplify statements with no or very little “understanding” of their mathematical meaning.
- In particular, syntactical proof techniques allow us to split complicated proofs into simpler proofs, without any need for an ingenious idea for how to carry out a specific proof.
- On the next slides we will study the standard proof situation $H \Rightarrow C$, and formulate rules which depend on the syntax of H and/or C.

Syntactical Proof Techniques

- *Syntactical proof techniques* are proof techniques that are based on the analysis of the syntactical structure of a statement.
- Syntactical proof techniques allow us to reason about statements and to simplify statements with no or very little "understanding" of their mathematical meaning.
- In particular, syntactical proof techniques allow us to split complicated proofs into simpler proofs, without any need for an ingenious idea for how to carry out a specific proof.
- On the next slides we will study the standard proof situation $H \Rightarrow C$, and formulate rules which depend on the syntax of H and/or C.
- Recall the truth table for "$\Rightarrow$":

H	C	$H \Rightarrow C$	$\neg H \vee C$
0	0	1	1
0	1	1	1
1	0	0	0
1	1	1	1

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Syntactical Proof Techniques

- *Syntactical proof techniques* are proof techniques that are based on the analysis of the syntactical structure of a statement.
- Syntactical proof techniques allow us to reason about statements and to simplify statements with no or very little "understanding" of their mathematical meaning.
- In particular, syntactical proof techniques allow us to split complicated proofs into simpler proofs, without any need for an ingenious idea for how to carry out a specific proof.
- On the next slides we will study the standard proof situation $H \Rightarrow C$, and formulate rules which depend on the syntax of H and/or C.
- Recall the truth table for "$\Rightarrow$":

H	C	$H \Rightarrow C$	$\neg H \vee C$
0	0	1	1
0	1	1	1
1	0	0	0
1	1	1	1

$H \Rightarrow C$ . . .

. . . is true if either H is false (and C arbitrary) or if C is true for H being true.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(A \wedge B)$:
  - Prove A under the assumption H; and
  - Prove B under the assumption H.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(A \wedge B)$:
  - Prove A under the assumption H; and
  - Prove B under the assumption H.

- If conclusion C is of the form $(A \vee B)$:
  - Add $\neg A$ to the assumption H and prove B. That is, assume both H and $\neg A$ to be true and use this to prove B.
  - Alternatively, add $\neg B$ to the assumption H and prove A.

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(A \wedge B)$:
  - Prove A under the assumption H; and
  - Prove B under the assumption H.

- If conclusion C is of the form $(A \vee B)$:
  - Add $\neg A$ to the assumption H and prove B. That is, assume both H and $\neg A$ to be true and use this to prove B.
  - Alternatively, add $\neg B$ to the assumption H and prove A.

- If conclusion C is of the form $(A \Rightarrow B)$:
  - Add A to the assumption H and prove B.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(A \wedge B)$:
  - Prove A under the assumption H; and
  - Prove B under the assumption H.
- If conclusion C is of the form $(A \vee B)$:
  - Add $\neg A$ to the assumption H and prove B. That is, assume both H and $\neg A$ to be true and use this to prove B.
  - Alternatively, add $\neg B$ to the assumption H and prove A.
- If conclusion C is of the form $(A \Rightarrow B)$:
  - Add A to the assumption H and prove B.
- If conclusion C is of the form $(A \Leftrightarrow B)$:
  - Prove $A \Rightarrow B$ under the assumption H; and
  - Prove $B \Rightarrow A$ under the assumption H.

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(A \wedge B)$:
  - Prove A under the assumption H; and
  - Prove B under the assumption H.
- If conclusion C is of the form $(A \vee B)$:
  - Add $\neg A$ to the assumption H and prove B. That is, assume both H and $\neg A$ to be true and use this to prove B.
  - Alternatively, add $\neg B$ to the assumption H and prove A.
- If conclusion C is of the form $(A \Rightarrow B)$:
  - Add A to the assumption H and prove B.
- If conclusion C is of the form $(A \Leftrightarrow B)$:
  - Prove $A \Rightarrow B$ under the assumption H; and
  - Prove $B \Rightarrow A$ under the assumption H.

Warning

In all the rules on this slide, A and B must not be part of a quantified formula. (Otherwise, get rid of the quantifier first!)

- If conclusion C is of the form $(\forall x \;\; A)$:
  - Proof technique: Let x_0 be arbitrary but fixed (Dt.: “beliebig aber fix”). From now on, x_0 can be treated as a constant!
  - It remains to prove $A[x_0/x]$ under the assumption H.

- If conclusion C is of the form $(\forall x \;\; A)$:
  - Proof technique: Let x_0 be arbitrary but fixed (Dt.: “beliebig aber fix”). From now on, x_0 can be treated as a constant!
  - It remains to prove $A[x_0/x]$ under the assumption H.
  - Often one does not trouble to explicitly label the particular arbitrary-but-fixed choice of x as, say, x_0 but only states that x is now regarded to be fixed.

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(\forall x \;\; A)$:
  - Proof technique: Let x_0 be arbitrary but fixed (Dt.: “beliebig aber fix”). From now on, x_0 can be treated as a constant!
  - It remains to prove $A[x_0/x]$ under the assumption H.
  - Often one does not trouble to explicitly label the particular arbitrary-but-fixed choice of x as, say, x_0 but only states that x is now regarded to be fixed.

Warning

The crucial point is that x_0 has to be arbitrary, and the proof may not depend on the particular choice of x_0!

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(\forall x \ \ A)$:
  - Proof technique: Let x_0 be arbitrary but fixed (Dt.: "beliebig aber fix"). From now on, x_0 can be treated as a constant!
  - It remains to prove $A[x_0/x]$ under the assumption H.
  - Often one does not trouble to explicitly label the particular arbitrary-but-fixed choice of x as, say, x_0 but only states that x is now regarded to be fixed.

Warning

The crucial point is that x_0 has to be arbitrary, and the proof may not depend on the particular choice of x_0!

- The symbol x_0 may not occur anywhere in A, in the hypothesis H, or in some other part of the conclusion.

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(\forall x \;\; A)$:
  - Proof technique: Let x_0 be arbitrary but fixed (Dt.: "beliebig aber fix"). From now on, x_0 can be treated as a constant!
  - It remains to prove $A[x_0/x]$ under the assumption H.
  - Often one does not trouble to explicitly label the particular arbitrary-but-fixed choice of x as, say, x_0 but only states that x is now regarded to be fixed.

Warning

The crucial point is that x_0 has to be arbitrary, and the proof may not depend on the particular choice of x_0!

- The symbol x_0 may not occur anywhere in A, in the hypothesis H, or in some other part of the conclusion.
- We are not allowed to make any assumptions on x_0 except for those that hold for all x in the universe of discourse.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- If conclusion C is of the form $(\exists x \;\; A)$:
  - *Constructive Proof* (Dt.: konstruktiver Beweis):
    - It “suffices” to find a suitable x_0 such that $A[x_0/x]$ if H.
    - Such an x_0 is called the “solving term”.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- If conclusion C is of the form $(\exists x \;\; A)$:
  - *Constructive Proof* (Dt.: konstruktiver Beweis):
    - It "suffices" to find a suitable x_0 such that $A[x_0/x]$ if H.
    - Such an x_0 is called the "solving term".
  - *Existential Proof* (Dt.: Existenzbeweis):
    - Prove that some suitable x_0 exists.
    - No need to "construct" x_0 explicitly.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(\exists x \; A)$:
  - *Constructive Proof* (Dt.: konstruktiver Beweis):
    - It "suffices" to find a suitable x_0 such that $A[x_0/x]$ if H.
    - Such an x_0 is called the "solving term".
  - *Existential Proof* (Dt.: Existenzbeweis):
    - Prove that some suitable x_0 exists.
    - No need to "construct" x_0 explicitly.
- E.g., suppose that we want to prove the following claim: The polynomial $p(x) := x^3 - x^2 + x - 1$ has a real root over $\mathbb{R}$.

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(\exists x \; A)$:
  - *Constructive Proof* (Dt.: konstruktiver Beweis):
    - It "suffices" to find a suitable x_0 such that $A[x_0/x]$ if H.
    - Such an x_0 is called the "solving term".
  - *Existential Proof* (Dt.: Existenzbeweis):
    - Prove that some suitable x_0 exists.
    - No need to "construct" x_0 explicitly.
- E.g., suppose that we want to prove the following claim: The polynomial $p(x) := x^3 - x^2 + x - 1$ has a real root over $\mathbb{R}$.

  Proof (constructive) : Factoring $p(x)$ yields $p(x) = (x-1)(x^2+1)$. Thus, we learn that 1 is a real root. □

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(\exists x \;\; A)$:
  - *Constructive Proof* (Dt.: konstruktiver Beweis):
    - It "suffices" to find a suitable x_0 such that $A[x_0/x]$ if H.
    - Such an x_0 is called the "solving term".
  - *Existential Proof* (Dt.: Existenzbeweis):
    - Prove that some suitable x_0 exists.
    - No need to "construct" x_0 explicitly.
- E.g., suppose that we want to prove the following claim: The polynomial $p(x) := x^3 - x^2 + x - 1$ has a real root over $\mathbb{R}$.

  Proof (constructive) : Factoring $p(x)$ yields $p(x) = (x-1)(x^2+1)$. Thus, we learn that 1 is a real root. □

  Proof (existential) : We have $p(2) = 5 > 0$ and $p(0) = -1 < 0$. Since p is continuous on the closed interval $[0, 2]$, the Intermediate Value Theorem (Dt.: Zwischenwertsatz) tells us that there exists a real number x strictly between 0 and 2 such that $p(x) = 0$. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(\exists! x \;\; A)$:
  - Prove that such an x exists.
  - Prove its uniqueness.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Syntactical Proof Techniques for $H \Rightarrow C$

- If conclusion C is of the form $(\exists! x \;\; A)$:
  - Prove that such an x exists.
  - Prove its uniqueness.

- If hypothesis H is of the form $(\exists x \;\; A)$:
  - Let x_0 such that $A[x_0/x]$.
  - Add $A[x_0/x]$ to knowledge.
  - Again: x_0 must not occur anywhere else in H or C!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural-Language Synonyms of Formal Terms

- On many occasions a conjecture will not be stated in formal terms but by using a natural language.
- Then one has to *decode* the natural-language formulation and *translate* it into formal terms!

Natural-Language Synonyms of Formal Terms

- On many occasions a conjecture will not be stated in formal terms but by using a natural language.
- Then one has to *decode* the natural-language formulation and *translate* it into formal terms!
- Natural-language synonyms for $A \Rightarrow B$:
  - A implies B, A impliziert B,
  - If A then B,
  - B if A,
  - A only if B,
  - A is sufficient for B, A ist hinreichend für B,
  - B is necessary for A, B ist notwendig für A.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural-Language Synonyms of Formal Terms

- On many occasions a conjecture will not be stated in formal terms but by using a natural language.
- Then one has to *decode* the natural-language formulation and *translate* it into formal terms!
- Natural-language synonyms for $A \Rightarrow B$:
  - A implies B, A impliziert B,
  - If A then B,
  - B if A,
  - A only if B,
  - A is sufficient for B, A ist hinreichend für B,
  - B is necessary for A, B ist notwendig für A.
- Natural-language synonyms for $A \Leftrightarrow B$:
  - A equivalent to B, A äquivalent zu B,
  - A if and only if B, A genau dann wenn B,
  - A is necessary and sufficient for B, A ist notwendig und hinreichend für B.

Equivalence Transformations

- First attempt to prove $(\forall n \in \mathbb{N} \quad \frac{2n+1}{n+1} \geqslant \frac{3}{2})$:

$$\begin{aligned}
\frac{2n+1}{n+1} &\geqslant \frac{3}{2} \\
2(2n+1) &\geqslant 3(n+1) \\
4n+2 &\geqslant 3n+3 \\
n &\geqslant 1
\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations

- First attempt to prove $(\forall n \in \mathbb{N} \quad \frac{2n+1}{n+1} \geqslant \frac{3}{2})$:

$$\begin{aligned} \frac{2n+1}{n+1} &\geqslant \frac{3}{2} \\ 2(2n+1) &\geqslant 3(n+1) \\ 4n+2 &\geqslant 3n+3 \\ n &\geqslant 1 \end{aligned}$$

- Second refined attempt to prove $(\forall n \in \mathbb{N} \quad \frac{2n+1}{n+1} \geqslant \frac{3}{2})$:

$$\begin{aligned} & & \frac{2n+1}{n+1} &\geqslant \frac{3}{2} & & | \cdot 2(n+1) \\ \Longrightarrow & & 2(2n+1) &\geqslant 3(n+1) & & \\ \Longrightarrow & & 4n+2 &\geqslant 3n+3 & & | - (3n+2) \\ \Longrightarrow & & n &\geqslant 1 & & \end{aligned}$$

- First attempt to prove $(\forall n \in \mathbb{N} \quad \frac{2n+1}{n+1} \geqslant \frac{3}{2})$:

$$\begin{aligned} \frac{2n+1}{n+1} &\geqslant \frac{3}{2} \\ 2(2n+1) &\geqslant 3(n+1) \\ 4n+2 &\geqslant 3n+3 \\ n &\geqslant 1 \end{aligned}$$

- Second refined attempt to prove $(\forall n \in \mathbb{N} \quad \frac{2n+1}{n+1} \geqslant \frac{3}{2})$:

$$\begin{aligned} & \frac{2n+1}{n+1} \geqslant \frac{3}{2} && \mid \cdot 2(n+1) \\ \Longrightarrow \quad & 2(2n+1) \geqslant 3(n+1) && \\ \Longrightarrow \quad & 4n+2 \geqslant 3n+3 && \mid -(3n+2) \\ \Longrightarrow \quad & n \geqslant 1 && \end{aligned}$$

- Correct proof of $(\forall n \in \mathbb{N} \quad \frac{2n+1}{n+1} \geqslant \frac{3}{2})$: Let $n \in \mathbb{N}$ be arbitrary but fixed. Then:

$$\begin{aligned} & \frac{2n+1}{n+1} \geqslant \frac{3}{2} && \mid \cdot 2(n+1) \\ \Longleftrightarrow \quad & 2(2n+1) \geqslant 3(n+1) && \\ \Longleftrightarrow \quad & 4n+2 \geqslant 3n+3 && \mid -(3n+2) \\ \Longleftrightarrow \quad & n \geqslant 1 && \end{aligned}$$

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$a = b$$

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We “prove” that $1 = 2$:

$$
\begin{array}{lrll}
 & a & = b & \mid \cdot a \\
\Longleftrightarrow & a^2 & = ab &
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\iff & a^2 & = & ab & \mid -b^2 \\
\iff & a^2 - b^2 & = & ab - b^2 &
\end{array}
$$

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\iff & a^2 & = & ab & \mid -b^2 \\
\iff & a^2 - b^2 & = & ab - b^2 & \\
\iff & (a-b)\cdot(a+b) & = & b\cdot(a-b) &
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We “prove” that $1 = 2$:

$$
\begin{array}{llll}
 & a & = b & \mid \cdot a \\
\iff & a^2 & = ab & \mid - b^2 \\
\iff & a^2 - b^2 & = ab - b^2 & \\
\iff & (a-b)\cdot(a+b) & = b\cdot(a-b) & \mid \div (a-b) \\
\iff & (a+b) & = b &
\end{array}
$$

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\Longleftrightarrow & a^2 & = & ab & \mid -b^2 \\
\Longleftrightarrow & a^2 - b^2 & = & ab - b^2 & \\
\Longleftrightarrow & (a-b)\cdot(a+b) & = & b\cdot(a-b) & \mid \div (a-b) \\
\Longleftrightarrow & (a+b) & = & b & \mid a := b \\
\Longleftrightarrow & (b+b) & = & b &
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\Longleftrightarrow & a^2 & = & ab & \mid -b^2 \\
\Longleftrightarrow & a^2 - b^2 & = & ab - b^2 & \\
\Longleftrightarrow & (a-b)\cdot(a+b) & = & b \cdot (a-b) & \mid \div (a-b) \\
\Longleftrightarrow & (a+b) & = & b & \mid a := b \\
\Longleftrightarrow & (b+b) & = & b & \\
\Longleftrightarrow & 2b & = & b &
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & |\cdot a \\
\iff & a^2 & = & ab & |-b^2 \\
\iff & a^2 - b^2 & = & ab - b^2 & \\
\iff & (a-b)\cdot(a+b) & = & b\cdot(a-b) & |\div(a-b) \\
\iff & (a+b) & = & b & |\, a := b \\
\iff & (b+b) & = & b & \\
\iff & 2b & = & b & |\div b \\
\iff & 2 & = & 1 &
\end{array}
$$

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot\, a \\
\Longleftrightarrow & a^2 & = & ab & \mid -\, b^2 \\
\Longleftrightarrow & a^2 - b^2 & = & ab - b^2 & \\
\Longleftrightarrow & (a-b)\cdot(a+b) & = & b\cdot(a-b) & \mid \div\, (a-b) \\
\Longleftrightarrow & (a+b) & = & b & \mid a := b \\
\Longleftrightarrow & (b+b) & = & b & \\
\Longleftrightarrow & 2b & = & b & \mid \div\, b \\
\Longleftrightarrow & 2 & = & 1 &
\end{array}
$$

- And here comes a "proof" of $4 = 5$:

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\Longleftrightarrow & a^2 & = & ab & \mid -b^2 \\
\Longleftrightarrow & a^2 - b^2 & = & ab - b^2 & \\
\Longleftrightarrow & (a-b)\cdot(a+b) & = & b\cdot(a-b) & \mid \div (a-b) \\
\Longleftrightarrow & (a+b) & = & b & \mid a := b \\
\Longleftrightarrow & (b+b) & = & b & \\
\Longleftrightarrow & 2b & = & b & \mid \div b \\
\Longleftrightarrow & 2 & = & 1 &
\end{array}
$$

- And here comes a "proof" of $4 = 5$: Let $x := 4$ and $y := 5$. Then

$$x + y = 9$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\Longleftrightarrow & a^2 & = & ab & \mid - b^2 \\
\Longleftrightarrow & a^2 - b^2 & = & ab - b^2 & \\
\Longleftrightarrow & (a-b)\cdot(a+b) & = & b\cdot(a-b) & \mid \div (a-b) \\
\Longleftrightarrow & (a+b) & = & b & \mid a := b \\
\Longleftrightarrow & (b+b) & = & b & \\
\Longleftrightarrow & 2b & = & b & \mid \div b \\
\Longleftrightarrow & 2 & = & 1 &
\end{array}
$$

- And here comes a "proof" of $4 = 5$: Let $x := 4$ and $y := 5$. Then

$$
\begin{array}{rrcll}
 & x + y & = & 9 & \mid \cdot (x-y) \\
\Longleftrightarrow & x^2 - y^2 & = & 9x - 9y &
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{lrll}
 & a &= b & \mid \cdot a \\
\iff & a^2 &= ab & \mid -b^2 \\
\iff & a^2 - b^2 &= ab - b^2 & \\
\iff & (a-b)\cdot(a+b) &= b\cdot(a-b) & \mid \div (a-b) \\
\iff & (a+b) &= b & \mid a := b \\
\iff & (b+b) &= b & \\
\iff & 2b &= b & \mid \div b \\
\iff & 2 &= 1 &
\end{array}
$$

- And here comes a "proof" of $4 = 5$: Let $x := 4$ and $y := 5$. Then

$$
\begin{array}{lrll}
 & x + y &= 9 & \mid \cdot (x-y) \\
\iff & x^2 - y^2 &= 9x - 9y & \mid + \frac{81}{4} - 9x + y^2 \\
\iff & x^2 - 9x + \frac{81}{4} &= y^2 - 9y + \frac{81}{4} &
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\Longleftrightarrow & a^2 & = & ab & \mid -b^2 \\
\Longleftrightarrow & a^2 - b^2 & = & ab - b^2 & \\
\Longleftrightarrow & (a-b)\cdot(a+b) & = & b\cdot(a-b) & \mid \div (a-b) \\
\Longleftrightarrow & (a+b) & = & b & \mid a := b \\
\Longleftrightarrow & (b+b) & = & b & \\
\Longleftrightarrow & 2b & = & b & \mid \div b \\
\Longleftrightarrow & 2 & = & 1 &
\end{array}
$$

- And here comes a "proof" of $4 = 5$: Let $x := 4$ and $y := 5$. Then

$$
\begin{array}{rrcll}
 & x + y & = & 9 & \mid \cdot (x-y) \\
\Longleftrightarrow & x^2 - y^2 & = & 9x - 9y & \mid + \frac{81}{4} - 9x + y^2 \\
\Longleftrightarrow & x^2 - 9x + \frac{81}{4} & = & y^2 - 9y + \frac{81}{4} & \\
\Longleftrightarrow & (x - \frac{9}{2})^2 & = & (y - \frac{9}{2})^2 &
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We “prove” that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\Longleftrightarrow & a^2 & = & ab & \mid -b^2 \\
\Longleftrightarrow & a^2 - b^2 & = & ab - b^2 & \\
\Longleftrightarrow & (a-b)\cdot(a+b) & = & b\cdot(a-b) & \mid \div (a-b) \\
\Longleftrightarrow & (a+b) & = & b & \mid a := b \\
\Longleftrightarrow & (b+b) & = & b & \\
\Longleftrightarrow & 2b & = & b & \mid \div b \\
\Longleftrightarrow & 2 & = & 1 &
\end{array}
$$

- And here comes a “proof” of $4 = 5$: Let $x := 4$ and $y := 5$. Then

$$
\begin{array}{rrcll}
 & x + y & = & 9 & \mid \cdot (x-y) \\
\Longleftrightarrow & x^2 - y^2 & = & 9x - 9y & \mid + \frac{81}{4} - 9x + y^2 \\
\Longleftrightarrow & x^2 - 9x + \frac{81}{4} & = & y^2 - 9y + \frac{81}{4} & \\
\Longleftrightarrow & (x - \frac{9}{2})^2 & = & (y - \frac{9}{2})^2 & \mid \sqrt{} \\
\Longleftrightarrow & x - \frac{9}{2} & = & y - \frac{9}{2} &
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Caveats

- Let $a, b \in \mathbb{N}$ be equal natural numbers. We "prove" that $1 = 2$:

$$
\begin{array}{rrcll}
 & a & = & b & \mid \cdot a \\
\Longleftrightarrow & a^2 & = & ab & \mid - b^2 \\
\Longleftrightarrow & a^2 - b^2 & = & ab - b^2 & \\
\Longleftrightarrow & (a-b)\cdot(a+b) & = & b\cdot(a-b) & \mid \div (a-b) \\
\Longleftrightarrow & (a+b) & = & b & \mid a := b \\
\Longleftrightarrow & (b+b) & = & b & \\
\Longleftrightarrow & 2b & = & b & \mid \div b \\
\Longleftrightarrow & 2 & = & 1 &
\end{array}
$$

- And here comes a "proof" of $4 = 5$: Let $x := 4$ and $y := 5$. Then

$$
\begin{array}{rrcll}
 & x + y & = & 9 & \mid \cdot (x-y) \\
\Longleftrightarrow & x^2 - y^2 & = & 9x - 9y & \mid + \frac{81}{4} - 9x + y^2 \\
\Longleftrightarrow & x^2 - 9x + \frac{81}{4} & = & y^2 - 9y + \frac{81}{4} & \\
\Longleftrightarrow & (x - \frac{9}{2})^2 & = & (y - \frac{9}{2})^2 & \mid \sqrt{} \\
\Longleftrightarrow & x - \frac{9}{2} & = & y - \frac{9}{2} & \mid + \frac{9}{2} \\
\Longleftrightarrow & x & = & y & \\
\Longleftrightarrow & 4 & = & 5 &
\end{array}
$$

Warnings

- Squaring is not an equivalence transformation!

Warnings

- Squaring is not an equivalence transformation!
- If squaring is applied for solving an equation then all candidate solutions found need to be tested with the original equation.

Equivalence Transformations: Summary

Warnings

- Squaring is not an equivalence transformation!
- If squaring is applied for solving an equation then all candidate solutions found need to be tested with the original equation.
- Taking a square root is only permissible if both signs are considered. That is, $\sqrt{x^2}$ yields $\pm x$.

Warnings

- Squaring is not an equivalence transformation!
- If squaring is applied for solving an equation then all candidate solutions found need to be tested with the original equation.
- Taking a square root is only permissible if both signs are considered. That is, $\sqrt{x^2}$ yields $\pm x$.
- A division by x is only permissible if $x \neq 0$ can be assured.

Equivalence Transformations: Summary

Warnings

- Squaring is not an equivalence transformation!
- If squaring is applied for solving an equation then all candidate solutions found need to be tested with the original equation.
- Taking a square root is only permissible if both signs are considered. That is, $\sqrt{x^2}$ yields $\pm x$.
- A division by x is only permissible if $x \neq 0$ can be assured.
- Multiplication by a negative number is not an equivalence transformation for inequalities.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Summary

Warnings

- Squaring is not an equivalence transformation!
- If squaring is applied for solving an equation then all candidate solutions found need to be tested with the original equation.
- Taking a square root is only permissible if both signs are considered. That is, $\sqrt{x^2}$ yields $\pm x$.
- A division by x is only permissible if $x \neq 0$ can be assured.
- Multiplication by a negative number is not an equivalence transformation for inequalities.

Advice

- In general, a relation $a \circ b$ may only be replaced by a new relation $a' \circ b'$ if one can argue that $(a \circ b) \Leftrightarrow (a' \circ b')$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Equivalence Transformations: Summary

Warnings

- Squaring is not an equivalence transformation!
- If squaring is applied for solving an equation then all candidate solutions found need to be tested with the original equation.
- Taking a square root is only permissible if both signs are considered. That is, $\sqrt{x^2}$ yields $\pm x$.
- A division by x is only permissible if $x \neq 0$ can be assured.
- Multiplication by a negative number is not an equivalence transformation for inequalities.

Advice

- In general, a relation $a \circ b$ may only be replaced by a new relation $a' \circ b'$ if one can argue that $(a \circ b) \Leftrightarrow (a' \circ b')$.
- It is advisable to prove $a \circ b$, where $\circ \in \{=, <, >, \leqslant, \geqslant\}$, by constructing a chain $a_0 \circ a_1 \circ a_2 \circ \ldots \circ a_n$, with $a_0 = a$ and $a_n = b$, for some $n \in \mathbb{N}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

3 Definitions and Theorem Proving

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

W.l.o.g.

- "W.l.o.g., A" means "Without loss of generality, we assume A".
- Dt.: O.B.d.A. ("Ohne Beschränkung der Allgemeinheit").

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

W.l.o.g.

- "W.l.o.g., A" means "Without loss of generality, we assume A".
- Dt.: O.B.d.A. ("Ohne Beschränkung der Allgemeinheit").
- This means that we could also carry on without the particular assumption A, and would either
  - have to consider cases that are handled very similarly, or
  - could easily convert the general case to this special case.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

W.l.o.g.

- “W.l.o.g., A” means “Without loss of generality, we assume A”.
- Dt.: O.B.d.A. (“Ohne Beschränkung der Allgemeinheit”).
- This means that we could also carry on without the particular assumption A, and would either
  - have to consider cases that are handled very similarly, or
  - could easily convert the general case to this special case.
- That is, a “w.l.o.g.” assumption allows us to save space/paper by avoiding to replicate portions of a proof that differ only in trivial aspects.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

W.l.o.g.

- “W.l.o.g., A” means “Without loss of generality, we assume A”.
- Dt.: O.B.d.A. (“Ohne Beschränkung der Allgemeinheit”).
- This means that we could also carry on without the particular assumption A, and would either
  - have to consider cases that are handled very similarly, or
  - could easily convert the general case to this special case.
- That is, a “w.l.o.g.” assumption allows us to save space/paper by avoiding to replicate portions of a proof that differ only in trivial aspects.

Warning

Do not use “w.l.o.g.” unless *you could* indeed *explain* explicitly and in full detail how to carry on without that assumption!

Types of Proofs: Direct Enumeration

- *Direct Enumeration*
  - E.g.: The conjecture

    $2p + 1$ *is prime for all* $p \in \{2, 3, 5\}$

    can be proved by considering all finitely many possible values for p.

Types of Proofs: Direct Enumeration

- *Direct Enumeration*
  - E.g.: The conjecture

    $$2p + 1 \text{ is prime for all } p \in \{2, 3, 5\}$$

    can be proved by considering all finitely many possible values for p.
  - Note: Direct enumeration only works if the set given is finite!

Types of Proofs: Case Analysis

- Aka *Proof by Exhaustion*. Dt.: Fallunterscheidung.
- In order to prove $H \Rightarrow C$, it suffices to prove

$$A_1 \vee A_2 \vee \ldots \vee A_k$$

for some statements $A_1, A_2, \ldots, A_k$,

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Case Analysis

- Aka *Proof by Exhaustion*. Dt.: Fallunterscheidung.
- In order to prove $H \Rightarrow C$, it suffices to prove

$$A_1 \vee A_2 \vee \ldots \vee A_k$$

for some statements $A_1, A_2, \ldots, A_k$, and to prove

$$\begin{aligned}
(H \wedge A_1) &\Rightarrow C, \\
(H \wedge A_2) &\Rightarrow C, \\
&\vdots \\
(H \wedge A_k) &\Rightarrow C.
\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Case Analysis

- Aka *Proof by Exhaustion*. Dt.: Fallunterscheidung.
- In order to prove $H \Rightarrow C$, it suffices to prove

$$A_1 \vee A_2 \vee \ldots \vee A_k$$

for some statements $A_1, A_2, \ldots, A_k$, and to prove

$$\begin{aligned} (H \wedge A_1) &\Rightarrow C, \\ (H \wedge A_2) &\Rightarrow C, \\ &\vdots \\ (H \wedge A_k) &\Rightarrow C. \end{aligned}$$

Warning

It is essential to guarantee that $A_1 \vee A_2 \vee \ldots \vee A_k$ holds, i.e., that no case is missing!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Case Analysis

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Case Analysis

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Case Analysis

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n - 1)(n^2 + n + 1)(n + 1)(n^2 - n + 1).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Case Analysis

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n - 1)(n^2 + n + 1)(n + 1)(n^2 - n + 1).$$

Let $n := 7q + r$ with $q, r \in \mathbb{N}_0$ and $0 \leqslant r \leqslant 6$. We consider seven cases, depending on whether $r = 0, 1, 2, 3, 4, 5$ or 6.

Types of Proofs: Sample Case Analysis

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n-1)(n^2 + n + 1)(n+1)(n^2 - n + 1).$$

Let $n := 7q + r$ with $q, r \in \mathbb{N}_0$ and $0 \leqslant r \leqslant 6$. We consider seven cases, depending on whether $r = 0, 1, 2, 3, 4, 5$ or 6.

Case $n = 7q$: Then the factor n of $n^7 - n$ is divisible by 7.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n - 1)(n^2 + n + 1)(n + 1)(n^2 - n + 1).$$

Let $n := 7q + r$ with $q, r \in \mathbb{N}_0$ and $0 \leqslant r \leqslant 6$. We consider seven cases, depending on whether $r = 0, 1, 2, 3, 4, 5$ or 6.

Case $n = 7q$: Then the factor n of $n^7 - n$ is divisible by 7.

Case $n = 7q + 1$: Then the factor $n - 1 = 7q$ of $n^7 - n$ is divisible by 7.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n-1)(n^2 + n + 1)(n+1)(n^2 - n + 1).$$

Let $n := 7q + r$ with $q, r \in \mathbb{N}_0$ and $0 \leqslant r \leqslant 6$. We consider seven cases, depending on whether $r = 0, 1, 2, 3, 4, 5$ or 6.

Case $n = 7q$: Then the factor n of $n^7 - n$ is divisible by 7.

Case $n = 7q + 1$: Then the factor $n - 1 = 7q$ of $n^7 - n$ is divisible by 7.

Case $n = 7q + 2$: Then $n^2 + n + 1 = (7q + 2)^2 + (7q + 2) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Case Analysis

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n - 1)(n^2 + n + 1)(n + 1)(n^2 - n + 1).$$

Let $n := 7q + r$ with $q, r \in \mathbb{N}_0$ and $0 \leqslant r \leqslant 6$. We consider seven cases, depending on whether $r = 0, 1, 2, 3, 4, 5$ or 6.

Case $n = 7q$: Then the factor n of $n^7 - n$ is divisible by 7.

Case $n = 7q + 1$: Then the factor $n - 1 = 7q$ of $n^7 - n$ is divisible by 7.

Case $n = 7q + 2$: Then $n^2 + n + 1 = (7q + 2)^2 + (7q + 2) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

Case $n = 7q + 3$: Then $n^2 - n + 1 = (7q + 3)^2 - (7q + 3) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n-1)(n^2+n+1)(n+1)(n^2-n+1).$$

Let $n := 7q + r$ with $q, r \in \mathbb{N}_0$ and $0 \leqslant r \leqslant 6$. We consider seven cases, depending on whether $r = 0, 1, 2, 3, 4, 5$ or 6.

Case $n = 7q$: Then the factor n of $n^7 - n$ is divisible by 7.

Case $n = 7q + 1$: Then the factor $n - 1 = 7q$ of $n^7 - n$ is divisible by 7.

Case $n = 7q + 2$: Then $n^2 + n + 1 = (7q+2)^2 + (7q+2) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

Case $n = 7q + 3$: Then $n^2 - n + 1 = (7q+3)^2 - (7q+3) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

Case $n = 7q + 4$: Then $n^2 + n + 1 = (7q+4)^2 + (7q+4) + 1 = 49q^2 + 63q + 21$ is divisible by 7.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n - 1)(n^2 + n + 1)(n + 1)(n^2 - n + 1).$$

Let $n := 7q + r$ with $q, r \in \mathbb{N}_0$ and $0 \leqslant r \leqslant 6$. We consider seven cases, depending on whether $r = 0, 1, 2, 3, 4, 5$ or 6.

Case $n = 7q$: Then the factor n of $n^7 - n$ is divisible by 7.

Case $n = 7q + 1$: Then the factor $n - 1 = 7q$ of $n^7 - n$ is divisible by 7.

Case $n = 7q + 2$: Then $n^2 + n + 1 = (7q + 2)^2 + (7q + 2) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

Case $n = 7q + 3$: Then $n^2 - n + 1 = (7q + 3)^2 - (7q + 3) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

Case $n = 7q + 4$: Then $n^2 + n + 1 = (7q + 4)^2 + (7q + 4) + 1 = 49q^2 + 63q + 21$ is divisible by 7.

Case $n = 7q + 5$: Then $n^2 - n + 1 = (7q + 5)^2 - (7q + 5) + 1 = 49q^2 + 63q + 21$ is divisible by 7.

Types of Proofs: Sample Case Analysis

- Suppose that we want to prove the following claim: For all $n \in \mathbb{N}_0$ the number 7 divides $n^7 - n$ without remainder. E.g., for $n := 3$, we get $3^7 - 3 = 2184 = 7 \cdot 312$.

Proof: Factoring $n^7 - n$ yields

$$n^7 - n = n(n^6 - 1) = n(n^3 - 1)(n^3 + 1) = n(n-1)(n^2 + n + 1)(n+1)(n^2 - n + 1).$$

Let $n := 7q + r$ with $q, r \in \mathbb{N}_0$ and $0 \leqslant r \leqslant 6$. We consider seven cases, depending on whether $r = 0, 1, 2, 3, 4, 5$ or 6.

Case $n = 7q$: Then the factor n of $n^7 - n$ is divisible by 7.

Case $n = 7q + 1$: Then the factor $n - 1 = 7q$ of $n^7 - n$ is divisible by 7.

Case $n = 7q + 2$: Then $n^2 + n + 1 = (7q + 2)^2 + (7q + 2) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

Case $n = 7q + 3$: Then $n^2 - n + 1 = (7q + 3)^2 - (7q + 3) + 1 = 49q^2 + 35q + 7$ is divisible by 7.

Case $n = 7q + 4$: Then $n^2 + n + 1 = (7q + 4)^2 + (7q + 4) + 1 = 49q^2 + 63q + 21$ is divisible by 7.

Case $n = 7q + 5$: Then $n^2 - n + 1 = (7q + 5)^2 - (7q + 5) + 1 = 49q^2 + 63q + 21$ is divisible by 7.

Case $n = 7q + 6$: Then $n + 1 = 7q + 7$ is divisible by 7. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Direct Proof

- Dt.: direkter Beweis.
- We want to prove $H \Rightarrow C$:
  - We build a chain of reasoning that starts at H and ends in C.
  - This approach is the classical example of deductive reasoning, where a logically valid sequence of steps establishes the truth of C under the assumption of H.

Types of Proofs: Direct Proof

- Dt.: direkter Beweis.
- We want to prove $H \Rightarrow C$:
  - We build a chain of reasoning that starts at H and ends in C.
  - This approach is the classical example of deductive reasoning, where a logically valid sequence of steps establishes the truth of C under the assumption of H.
- Suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \;\; (x < y) \Rightarrow (x^2 < y^2))$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Direct Proof

- Dt.: direkter Beweis.
- We want to prove $H \Rightarrow C$:
  - We build a chain of reasoning that starts at H and ends in C.
  - This approach is the classical example of deductive reasoning, where a logically valid sequence of steps establishes the truth of C under the assumption of H.
- Suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \ \ (x < y) \Rightarrow (x^2 < y^2))$.

  Proof : (Direct Proof)
  Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed, with $x_0 < y_0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Dt.: direkter Beweis.
- We want to prove $H \Rightarrow C$:
  - We build a chain of reasoning that starts at H and ends in C.
  - This approach is the classical example of deductive reasoning, where a logically valid sequence of steps establishes the truth of C under the assumption of H.
- Suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \;\; (x < y) \Rightarrow (x^2 < y^2))$.

  Proof : (Direct Proof)
  Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed, with $x_0 < y_0$.
  We have $x_0 < y_0$, and therefore $x_0^2 = x_0 \cdot x_0 < y_0 \cdot x_0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Direct Proof

- Dt.: direkter Beweis.
- We want to prove $H \Rightarrow C$:
  - We build a chain of reasoning that starts at H and ends in C.
  - This approach is the classical example of deductive reasoning, where a logically valid sequence of steps establishes the truth of C under the assumption of H.
- Suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \ \ (x < y) \Rightarrow (x^2 < y^2))$.

Proof: (Direct Proof)
Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed, with $x_0 < y_0$.
We have $x_0 < y_0$, and therefore $x_0^2 = x_0 \cdot x_0 < y_0 \cdot x_0$. Since $x_0 < y_0$ we know $y_0 \cdot x_0 < y_0^2$, and obtain $x_0^2 < y_0 \cdot x_0 < y_0^2$, which finally establishes $x_0^2 < y_0^2$:

$$x_0^2 = x_0 \cdot x_0 < y_0 \cdot x_0 < y_0 \cdot y_0 = y_0^2$$

□

Types of Proofs: Proof by Contrapositive

- Dt.: Umkehrschluss, Kontraposition.
- We want to prove $H \Rightarrow C$:
  - In order to prove $H \Rightarrow C$ we build a (direct) proof for $(\neg C \Rightarrow \neg H)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Proof by Contrapositive

- Dt.: Umkehrschluss, Kontraposition.
- We want to prove $H \Rightarrow C$:
  - In order to prove $H \Rightarrow C$ we build a (direct) proof for $(\neg C \Rightarrow \neg H)$.
- Again, suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \ \ (x < y) \Rightarrow (x^2 < y^2))$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Proof by Contrapositive

- Dt.: Umkehrschluss, Kontraposition.
- We want to prove $H \Rightarrow C$:
  - In order to prove $H \Rightarrow C$ we build a (direct) proof for $(\neg C \Rightarrow \neg H)$.
- Again, suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \ \ (x < y) \Rightarrow (x^2 < y^2))$.

  Proof : Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Proof by Contrapositive

- Dt.: Umkehrschluss, Kontraposition.
- We want to prove $H \Rightarrow C$:
  - In order to prove $H \Rightarrow C$ we build a (direct) proof for $(\neg C \Rightarrow \neg H)$.
- Again, suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \;\; (x < y) \Rightarrow (x^2 < y^2))$.

  Proof: Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed. We prove $(x_0^2 \geqslant y_0^2) \Rightarrow (x_0 \geqslant y_0)$ similar to the direct proof before.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Proof by Contrapositive

- Dt.: Umkehrschluss, Kontraposition.
- We want to prove $H \Rightarrow C$:
  - In order to prove $H \Rightarrow C$ we build a (direct) proof for $(\neg C \Rightarrow \neg H)$.
- Again, suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \;\; (x < y) \Rightarrow (x^2 < y^2))$.

  Proof : Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed. We prove $(x_0^2 \geqslant y_0^2) \Rightarrow (x_0 \geqslant y_0)$ similar to the direct proof before. Since $(x_0^2 \geqslant y_0^2) \Leftrightarrow (x_0^2 - y_0^2 \geqslant 0)$,

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Proof by Contrapositive

- Dt.: Umkehrschluss, Kontraposition.
- We want to prove $H \Rightarrow C$:
  - In order to prove $H \Rightarrow C$ we build a (direct) proof for $(\neg C \Rightarrow \neg H)$.
- Again, suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \ \ (x < y) \Rightarrow (x^2 < y^2))$.

  Proof: Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed. We prove $(x_0^2 \geqslant y_0^2) \Rightarrow (x_0 \geqslant y_0)$ similar to the direct proof before. Since $(x_0^2 \geqslant y_0^2) \Leftrightarrow (x_0^2 - y_0^2 \geqslant 0)$, we get

  $$0 \leqslant x_0^2 - y_0^2 = (x_0 - y_0)(x_0 + y_0),$$

  which implies $y_0 \leqslant x_0$ since we may divide by the positive number $x_0 + y_0$. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Proof by Contrapositive

- Dt.: Umkehrschluss, Kontraposition.
- We want to prove $H \Rightarrow C$:
  - In order to prove $H \Rightarrow C$ we build a (direct) proof for $(\neg C \Rightarrow \neg H)$.
- Again, suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \ \ (x < y) \Rightarrow (x^2 < y^2))$.

  Proof: Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed. We prove $(x_0^2 \geqslant y_0^2) \Rightarrow (x_0 \geqslant y_0)$ similar to the direct proof before. Since $(x_0^2 \geqslant y_0^2) \Leftrightarrow (x_0^2 - y_0^2 \geqslant 0)$, we get

  $$0 \leqslant x_0^2 - y_0^2 = (x_0 - y_0)(x_0 + y_0),$$

  which implies $y_0 \leqslant x_0$ since we may divide by the positive number $x_0 + y_0$. □
- Suppose we want to prove $H \Rightarrow (\exists x \ \ A)$.

  Proof: Prove $(\forall x \ \ (\neg A)) \Rightarrow \neg H$. □

- Dt.: Umkehrschluss, Kontraposition.
- We want to prove $H \Rightarrow C$:
  - In order to prove $H \Rightarrow C$ we build a (direct) proof for $(\neg C \Rightarrow \neg H)$.
- Again, suppose we want to prove $(\forall x, y \in \mathbb{R}^+ \ \ (x < y) \Rightarrow (x^2 < y^2))$.

  Proof: Let $x_0, y_0 \in \mathbb{R}^+$ be arbitrary but fixed. We prove $(x_0^2 \geqslant y_0^2) \Rightarrow (x_0 \geqslant y_0)$ similar to the direct proof before. Since $(x_0^2 \geqslant y_0^2) \Leftrightarrow (x_0^2 - y_0^2 \geqslant 0)$, we get

  $$0 \leqslant x_0^2 - y_0^2 = (x_0 - y_0)(x_0 + y_0),$$

  which implies $y_0 \leqslant x_0$ since we may divide by the positive number $x_0 + y_0$. □
- Suppose we want to prove $H \Rightarrow (\exists x \ \ A)$.

  Proof: Prove $(\forall x \ \ (\neg A)) \Rightarrow \neg H$. □

Warning

Make sure that the statements are negated correctly!

Types of Proofs: Proof by Contradiction

- Dt.: Widerspruchsbeweis.
- We want to prove $H \Rightarrow C$:
  - We assume $(H \wedge \neg C)$ as new hypothesis and prove $\neg H$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Proof by Contradiction

- Dt.: Widerspruchsbeweis.
- We want to prove $H \Rightarrow C$:
  - We assume $(H \wedge \neg C)$ as new hypothesis and prove $\neg H$.
  - This approach is correct since $(H \Rightarrow C) \equiv ((H \wedge \neg C) \Rightarrow \neg H)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Proof by Contradiction

- Dt.: Widerspruchsbeweis.
- We want to prove $H \Rightarrow C$:
  - We assume $(H \wedge \neg C)$ as new hypothesis and prove $\neg H$.
  - This approach is correct since $(H \Rightarrow C) \equiv ((H \wedge \neg C) \Rightarrow \neg H)$.
- Warning: As when proving the contrapositive it is essential to check twice that the statements are indeed negated correctly!

Types of Proofs: Indirect Proof

- Aka *Reductio ad absurdum*.
- Dt.: indirekter Beweis.
- We want to prove $H \Rightarrow C$.
  - Consider a statement R that is known to be true, like $0 \neq 1$.

Types of Proofs: Indirect Proof

- Aka *Reductio ad absurdum*.
- Dt.: indirekter Beweis.
- We want to prove $H \Rightarrow C$.
  - Consider a statement R that is known to be true, like $0 \neq 1$.
  - Now assume $(H \wedge \neg C)$ and deduce $\neg R$, i.e., $0 = 1$.
  - This is absurd, and we conclude that $\neg C$ is false.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Indirect Proof

- Aka *Reductio ad absurdum*.
- Dt.: indirekter Beweis.
- We want to prove $H \Rightarrow C$.
  - Consider a statement R that is known to be true, like $0 \neq 1$.
  - Now assume $(H \wedge \neg C)$ and deduce $\neg R$, i.e., $0 = 1$.
  - This is absurd, and we conclude that $\neg C$ is false.
  - Formally, $(H \wedge \neg C \wedge R) \Rightarrow \neg R$.
  - This is of the form $(A \Rightarrow B)$, and we have $(A \Rightarrow B) \equiv T$, where $B \equiv F$. Thus, $A \equiv F$.

Types of Proofs: Indirect Proof

- Aka *Reductio ad absurdum*.
- Dt.: indirekter Beweis.
- We want to prove $H \Rightarrow C$.
    - Consider a statement R that is known to be true, like $0 \neq 1$.
    - Now assume $(H \wedge \neg C)$ and deduce $\neg R$, i.e., $0 = 1$.
    - This is absurd, and we conclude that $\neg C$ is false.
    - Formally, $(H \wedge \neg C \wedge R) \Rightarrow \neg R$.
    - This is of the form $(A \Rightarrow B)$, and we have $(A \Rightarrow B) \equiv T$, where $B \equiv F$. Thus, $A \equiv F$.

Note

Since an indirect proof is similar to a proof by contradiction, many textbooks treat it as one proof technique, or use the terms “reductio ad absurdum”, “indirect proof”, and “proof by contradiction” as synonyms.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Indirect Proof

- Suppose that we want to prove that the polynomial equation $x^3 + x + 1 = 0$ has no rational solution.

- Suppose that we want to prove that the polynomial equation $x^3 + x + 1 = 0$ has no rational solution.

Proof : Assume to the contrary that there exists a rational number $\frac{p}{q}$ which is a root of that polynomial.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Indirect Proof

- Suppose that we want to prove that the polynomial equation $x^3 + x + 1 = 0$ has no rational solution.

Proof: Assume to the contrary that there exists a rational number $\frac{p}{q}$ which is a root of that polynomial. W.l.o.g., we may assume $\frac{p}{q}$ to be irreducible. (A rational number $\frac{p}{q}$ is irreducible if there exists no integer other than ± 1 that divides both p and q.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Indirect Proof

- Suppose that we want to prove that the polynomial equation $x^3 + x + 1 = 0$ has no rational solution.

Proof: Assume to the contrary that there exists a rational number $\frac{p}{q}$ which is a root of that polynomial. W.l.o.g., we may assume $\frac{p}{q}$ to be irreducible. (A rational number $\frac{p}{q}$ is irreducible if there exists no integer other than ± 1 that divides both p and q.) We get

$$0 = \frac{p^3}{q^3} + \frac{p}{q} + 1 \qquad \text{and, thus,} \quad 0 = p^3 + pq^2 + q^3.$$

As statement R we take "0 is even".

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Indirect Proof

- Suppose that we want to prove that the polynomial equation $x^3 + x + 1 = 0$ has no rational solution.

Proof: Assume to the contrary that there exists a rational number $\frac{p}{q}$ which is a root of that polynomial. W.l.o.g., we may assume $\frac{p}{q}$ to be irreducible. (A rational number $\frac{p}{q}$ is irreducible if there exists no integer other than ± 1 that divides both p and q.) We get

$$0 = \frac{p^3}{q^3} + \frac{p}{q} + 1 \qquad \text{and, thus,} \quad 0 = p^3 + pq^2 + q^3.$$

As statement R we take “0 is even”.
We do a case analysis, depending on whether p, q are even or odd:
Case p, q odd: Then $p^3 + pq^2 + q^3$ is odd, but 0 is even, yielding a contradiction to R.

Types of Proofs: Sample Indirect Proof

- Suppose that we want to prove that the polynomial equation $x^3 + x + 1 = 0$ has no rational solution.

Proof: Assume to the contrary that there exists a rational number $\frac{p}{q}$ which is a root of that polynomial. W.l.o.g., we may assume $\frac{p}{q}$ to be irreducible. (A rational number $\frac{p}{q}$ is irreducible if there exists no integer other than ± 1 that divides both p and q.) We get

$$0 = \frac{p^3}{q^3} + \frac{p}{q} + 1 \qquad \text{and, thus,} \quad 0 = p^3 + pq^2 + q^3.$$

As statement R we take “0 is even”.
We do a case analysis, depending on whether p, q are even or odd:
Case p, q odd: Then $p^3 + pq^2 + q^3$ is odd, but 0 is even, yielding a contradiction to R.
Case p odd, q even: Then again $p^3 + pq^2 + q^3$ is odd; contradiction.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Proofs: Sample Indirect Proof

- Suppose that we want to prove that the polynomial equation $x^3 + x + 1 = 0$ has no rational solution.

Proof: Assume to the contrary that there exists a rational number $\frac{p}{q}$ which is a root of that polynomial. W.l.o.g., we may assume $\frac{p}{q}$ to be irreducible. (A rational number $\frac{p}{q}$ is irreducible if there exists no integer other than ± 1 that divides both p and q.) We get

$$0 = \frac{p^3}{q^3} + \frac{p}{q} + 1 \qquad \text{and, thus,} \quad 0 = p^3 + pq^2 + q^3.$$

As statement R we take “0 is even”.
We do a case analysis, depending on whether p, q are even or odd:

Case p, q odd: Then $p^3 + pq^2 + q^3$ is odd, but 0 is even, yielding a contradiction to R.
Case p odd, q even: Then again $p^3 + pq^2 + q^3$ is odd; contradiction.
Case p even, q odd: Then again $p^3 + pq^2 + q^3$ is odd; contradiction.

Types of Proofs: Sample Indirect Proof

- Suppose that we want to prove that the polynomial equation $x^3 + x + 1 = 0$ has no rational solution.

Proof: Assume to the contrary that there exists a rational number $\frac{p}{q}$ which is a root of that polynomial. W.l.o.g., we may assume $\frac{p}{q}$ to be irreducible. (A rational number $\frac{p}{q}$ is irreducible if there exists no integer other than ± 1 that divides both p and q.) We get

$$0 = \frac{p^3}{q^3} + \frac{p}{q} + 1 \qquad \text{and, thus,} \quad 0 = p^3 + pq^2 + q^3.$$

As statement R we take "0 is even".
We do a case analysis, depending on whether p, q are even or odd:

Case p, q odd: Then $p^3 + pq^2 + q^3$ is odd, but 0 is even, yielding a contradiction to R.
Case p odd, q even: Then again $p^3 + pq^2 + q^3$ is odd; contradiction.
Case p even, q odd: Then again $p^3 + pq^2 + q^3$ is odd; contradiction.
Case p, q even: This is not possible since we assumed (rightfully) that $\frac{p}{q}$ is irreducible.

□

Disproving Conjectures

- Sometimes conjectures are false ...
- If the conjecture is of the form $(\forall x \;\; A)$:
  - Then we can disprove this conjecture by showing $(\exists x \;\; \neg A)$.
  - The latter is proved if we can come up with a *counterexample* (Dt.: Gegenbeispiel) to the original claim.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Disproving Conjectures

- Sometimes conjectures are false ...
- If the conjecture is of the form $(\forall x \;\; A)$:
  - Then we can disprove this conjecture by showing $(\exists x \;\; \neg A)$.
  - The latter is proved if we can come up with a *counterexample* (Dt.: Gegenbeispiel) to the original claim.
  - E.g., the claim $\forall p \in \mathbb{P} \;\; (2p+1) \in \mathbb{P}$ is shown to be false by testing $p := 7$. (Note, though, that it is true for $p := 2, 3, 5, 11, \ldots$)

Disproving Conjectures

- Sometimes conjectures are false ...
- If the conjecture is of the form $(\forall x \;\; A)$:
  - Then we can disprove this conjecture by showing $(\exists x \;\; \neg A)$.
  - The latter is proved if we can come up with a *counterexample* (Dt.: Gegenbeispiel) to the original claim.
  - E.g., the claim $\forall p \in \mathbb{P} \;\; (2p+1) \in \mathbb{P}$ is shown to be false by testing $p := 7$. (Note, though, that it is true for $p := 2, 3, 5, 11, \ldots$)
  - Similarly, numbers of the form $2^{(2^n)} + 1$, for $n \in \mathbb{N}$, were once assumed to be primes. Indeed, this is correct for $n := 1, 2, 3, 4$ but $n := 5$ yields a counterexample:

$$2^{(2^5)} + 1 = 4\,294\,967\,297 = 641 \cdot 6\,700\,417.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Disproving Conjectures

- Sometimes conjectures are false . . .
- If the conjecture is of the form $(\forall x \;\; A)$:
  - Then we can disprove this conjecture by showing $(\exists x \;\; \neg A)$.
  - The latter is proved if we can come up with a *counterexample* (Dt.: Gegenbeispiel) to the original claim.
  - E.g., the claim $\forall p \in \mathbb{P} \;\; (2p+1) \in \mathbb{P}$ is shown to be false by testing $p := 7$. (Note, though, that it is true for $p := 2, 3, 5, 11, \ldots$)
  - Similarly, numbers of the form $2^{(2^n)} + 1$, for $n \in \mathbb{N}$, were once assumed to be primes. Indeed, this is correct for $n := 1, 2, 3, 4$ but $n := 5$ yields a counterexample:

$$2^{(2^5)} + 1 = 4\,294\,967\,297 = 641 \cdot 6\,700\,417.$$

- If, however, the conjecture is of the form $(\exists x \;\; A)$:
  - Then a counterexample does not suffice!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Disproving Conjectures

- Sometimes conjectures are false ...
- If the conjecture is of the form $(\forall x \;\; A)$:
  - Then we can disprove this conjecture by showing $(\exists x \;\; \neg A)$.
  - The latter is proved if we can come up with a *counterexample* (Dt.: Gegenbeispiel) to the original claim.
  - E.g., the claim $\forall p \in \mathbb{P} \;\; (2p+1) \in \mathbb{P}$ is shown to be false by testing $p := 7$. (Note, though, that it is true for $p := 2, 3, 5, 11, \ldots$)
  - Similarly, numbers of the form $2^{(2^n)} + 1$, for $n \in \mathbb{N}$, were once assumed to be primes. Indeed, this is correct for $n := 1, 2, 3, 4$ but $n := 5$ yields a counterexample:

$$2^{(2^5)} + 1 = 4\,294\,967\,297 = 641 \cdot 6\,700\,417.$$

- If, however, the conjecture is of the form $(\exists x \;\; A)$:
  - Then a counterexample does not suffice!
  - Rather, to disprove this conjecture, we'd have to prove formally $(\forall x \;\; \neg A)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.
- Formally: $\forall x \in \mathbb{R} \quad \left(\frac{x}{x^2+1} > 2\right) \implies \left(x < \frac{1}{2}\right)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.
- Formally: $\forall x \in \mathbb{R} \quad \left(\frac{x}{x^2+1} > 2\right) \implies \left(x < \frac{1}{2}\right)$.

Proof: Let $x \in \mathbb{R}$ arbitrary but fixed and suppose that $\frac{x}{x^2+1} > 2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.
- Formally: $\forall x \in \mathbb{R} \quad \left(\frac{x}{x^2+1} > 2\right) \Longrightarrow \left(x < \frac{1}{2}\right)$.

Proof : Let $x \in \mathbb{R}$ arbitrary but fixed and suppose that $\frac{x}{x^2+1} > 2$. This implies $x > 0$ and we get

$$\frac{1}{x} = \frac{x}{x^2} > \frac{x}{x^2+1} > 2,$$

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.
- Formally: $\forall x \in \mathbb{R} \quad \left(\frac{x}{x^2+1} > 2\right) \implies \left(x < \frac{1}{2}\right)$.

Proof : Let $x \in \mathbb{R}$ arbitrary but fixed and suppose that $\frac{x}{x^2+1} > 2$. This implies $x > 0$ and we get

$$\frac{1}{x} = \frac{x}{x^2} > \frac{x}{x^2+1} > 2, \qquad \text{thus } \frac{1}{x} > 2 \text{ and, therefore, } x < \frac{1}{2}.$$

□

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.
- Formally: $\forall x \in \mathbb{R} \quad \left(\frac{x}{x^2+1} > 2\right) \implies \left(x < \frac{1}{2}\right)$.

Proof: Let $x \in \mathbb{R}$ arbitrary but fixed and suppose that $\frac{x}{x^2+1} > 2$. This implies $x > 0$ and we get

$$\frac{1}{x} = \frac{x}{x^2} > \frac{x}{x^2+1} > 2, \qquad \text{thus } \frac{1}{x} > 2 \text{ and, therefore, } x < \frac{1}{2}.$$

□

- Note, though, that this lemma is of little use for mathematics: The hypothesis is never true!

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.
- Formally: $\forall x \in \mathbb{R} \quad \left(\frac{x}{x^2+1} > 2\right) \implies \left(x < \frac{1}{2}\right)$.

Proof: Let $x \in \mathbb{R}$ arbitrary but fixed and suppose that $\frac{x}{x^2+1} > 2$. This implies $x > 0$ and we get

$$\frac{1}{x} = \frac{x}{x^2} > \frac{x}{x^2+1} > 2, \qquad \text{thus } \frac{1}{x} > 2 \text{ and, therefore, } x < \frac{1}{2}.$$

□

- Note, though, that this lemma is of little use for mathematics: The hypothesis is never true! We have

$$\frac{x}{x^2+1} > 2 \iff 0 > 2x^2 - x + 2 \iff 2x^2 - x + 2 < 0.$$

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.
- Formally: $\forall x \in \mathbb{R} \quad \left(\frac{x}{x^2+1} > 2\right) \implies \left(x < \frac{1}{2}\right)$.

Proof : Let $x \in \mathbb{R}$ arbitrary but fixed and suppose that $\frac{x}{x^2+1} > 2$. This implies $x > 0$ and we get

$$\frac{1}{x} = \frac{x}{x^2} > \frac{x}{x^2+1} > 2, \qquad \text{thus } \frac{1}{x} > 2 \text{ and, therefore, } x < \frac{1}{2}.$$

□

- Note, though, that this lemma is of little use for mathematics: The hypothesis is never true! We have

$$\frac{x}{x^2+1} > 2 \iff 0 > 2x^2 - x + 2 \iff 2x^2 - x + 2 < 0.$$

However, by a simple case analysis,

$$\text{if } x \leqslant 1 \text{ then } 2x^2 - x + 2 = 2x^2 + 1 + (1 - x) \geqslant 2x^2 + 1 > 0,$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Caveat: Ex Falso Quodlibet!

- Consider the following lemma: Let $x \in \mathbb{R}$. If $\frac{x}{x^2+1} > 2$ then $x < \frac{1}{2}$.
- Formally: $\forall x \in \mathbb{R} \quad \left(\frac{x}{x^2+1} > 2\right) \implies \left(x < \frac{1}{2}\right)$.

Proof: Let $x \in \mathbb{R}$ arbitrary but fixed and suppose that $\frac{x}{x^2+1} > 2$. This implies $x > 0$ and we get

$$\frac{1}{x} = \frac{x}{x^2} > \frac{x}{x^2+1} > 2, \qquad \text{thus } \frac{1}{x} > 2 \text{ and, therefore, } x < \frac{1}{2}.$$

□

- Note, though, that this lemma is of little use for mathematics: The hypothesis is never true! We have

$$\frac{x}{x^2+1} > 2 \iff 0 > 2x^2 - x + 2 \iff 2x^2 - x + 2 < 0.$$

However, by a simple case analysis,

$$\text{if } x \leqslant 1 \text{ then } 2x^2 - x + 2 = 2x^2 + 1 + (1 - x) \geqslant 2x^2 + 1 > 0,$$

$$\text{if } x > 1 \text{ then } 2x^2 - x + 2 = x^2 + 2 + x(x - 1) \geqslant x^2 + 2 > 0.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

4 Numbers and Basics of Number Theory

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Algebraic Structures

- An algebraic structure consists of a non-empty set together with one or more operations on it which satisfy certain identities ("*axioms*").
- The axioms tell us the properties of the operations.
- Informally, an algebraic structure is a non-empty set upon which "arithmetic-like" operations have been defined.

- An algebraic structure consists of a non-empty set together with one or more operations on it which satisfy certain identities ("*axioms*").
- The axioms tell us the properties of the operations.
- Informally, an algebraic structure is a non-empty set upon which "arithmetic-like" operations have been defined.
- Well-known example: $\mathbb{R}$ with the standard addition "+".
- E.g., we have $(\sqrt{\pi}+1)-\sqrt{\pi}=1$ because

$$\begin{aligned}(\sqrt{\pi}+1)-\sqrt{\pi} &= (\sqrt{\pi}+1)+(-\sqrt{\pi}) = \sqrt{\pi}+1+(-\sqrt{\pi}) \\ &= \sqrt{\pi}+(-\sqrt{\pi})+1 = (\sqrt{\pi}+(-\sqrt{\pi}))+1 = 0+1 = 1.\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Algebraic Structures

- An algebraic structure consists of a non-empty set together with one or more operations on it which satisfy certain identities ("*axioms*").
- The axioms tell us the properties of the operations.
- Informally, an algebraic structure is a non-empty set upon which "arithmetic-like" operations have been defined.
- Well-known example: $\mathbb{R}$ with the standard addition "+".
- E.g., we have $(\sqrt{\pi}+1)-\sqrt{\pi}=1$ because

$$\begin{aligned}(\sqrt{\pi}+1)-\sqrt{\pi} &= (\sqrt{\pi}+1)+(-\sqrt{\pi}) = \sqrt{\pi}+1+(-\sqrt{\pi})\\ &= \sqrt{\pi}+(-\sqrt{\pi})+1 = (\sqrt{\pi}+(-\sqrt{\pi}))+1 = 0+1 = 1.\end{aligned}$$

In order to obtain this result we used commutativity, associativity and knowledge about inverse and neutral elements . . .

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- An algebraic structure consists of a non-empty set together with one or more operations on it which satisfy certain identities ("*axioms*").
- The axioms tell us the properties of the operations.
- Informally, an algebraic structure is a non-empty set upon which "arithmetic-like" operations have been defined.
- Well-known example: $\mathbb{R}$ with the standard addition "+".
- E.g., we have $(\sqrt{\pi}+1)-\sqrt{\pi}=1$ because

$$
\begin{aligned}
(\sqrt{\pi}+1)-\sqrt{\pi} &= (\sqrt{\pi}+1)+(-\sqrt{\pi}) = \sqrt{\pi}+1+(-\sqrt{\pi}) \\
&= \sqrt{\pi}+(-\sqrt{\pi})+1 = (\sqrt{\pi}+(-\sqrt{\pi}))+1 = 0+1 = 1.
\end{aligned}
$$

  In order to obtain this result we used commutativity, associativity and knowledge about inverse and neutral elements ...
- Algebraic structures get their names based on the type of operations and axioms supported.
- Well-known structures include group, ring, field, and vector space. (Many more algebraic structures are studied in abstract algebra, though!)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 40 (n-ary Operation, Dt.: n-stellig Verknüpfung)

Let n be a fixed non-negative integer and $X_1, X_2, \ldots, X_n$ be non-empty sets. An *n-ary operation* from $X_1, X_2, \ldots, X_n$ to another set Y is a function $\omega\colon X_1 \times X_2 \times \cdots \times X_n \to Y$.

Operation

Definition 40 (n-ary Operation, Dt.: n-stellig Verknüpfung)

Let n be a fixed non-negative integer and $X_1, X_2, \ldots, X_n$ be non-empty sets. An *n-ary operation* from $X_1, X_2, \ldots, X_n$ to another set Y is a function $\omega: X_1 \times X_2 \times \cdots \times X_n \to Y$. The set $X_1 \times X_2 \times \cdots \times X_n$ is called the *domain* (Dt.: Definitionsmenge) of the operation, the set Y is called the *codomain* (Dt.: Zielmenge) of the operation, and the number n of operands is called the *arity* (Dt.: Stelligkeit) of the operation.

Definition 40 (n-ary Operation, Dt.: n-stellig Verknüpfung)

Let n be a fixed non-negative integer and $X_1, X_2, \ldots, X_n$ be non-empty sets. An *n-ary operation* from $X_1, X_2, \ldots, X_n$ to another set Y is a function $\omega\colon X_1 \times X_2 \times \cdots \times X_n \to Y$. The set $X_1 \times X_2 \times \cdots \times X_n$ is called the *domain* (Dt.: Definitionsmenge) of the operation, the set Y is called the *codomain* (Dt.: Zielmenge) of the operation, and the number n of operands is called the *arity* (Dt.: Stelligkeit) of the operation.

An *n-ary operation on a set X* is a function $\omega\colon X^n \to X$, i.e., an n-ary operation where $X_1 = X_2 = \ldots = X_n = Y =: X$.

Operation

Definition 40 (*n*-ary Operation, Dt.: *n*-stellig Verknüpfung)

Let n be a fixed non-negative integer and $X_1, X_2, \ldots, X_n$ be non-empty sets. An *n-ary operation* from $X_1, X_2, \ldots, X_n$ to another set Y is a function $\omega\colon X_1 \times X_2 \times \cdots \times X_n \to Y$.
The set $X_1 \times X_2 \times \cdots \times X_n$ is called the *domain* (Dt.: Definitionsmenge) of the operation, the set Y is called the *codomain* (Dt.: Zielmenge) of the operation, and the number n of operands is called the *arity* (Dt.: Stelligkeit) of the operation.
An *n-ary operation on a set X* is a function $\omega\colon X^n \to X$, i.e., an n-ary operation where $X_1 = X_2 = \ldots = X_n = Y =: X$.

- An operation on a set X is also called an *internal operation* (Dt.: innere Verknüpfung).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Operation

Definition 40 (n-ary Operation, Dt.: n-stellig Verknüpfung)

Let n be a fixed non-negative integer and $X_1, X_2, \ldots, X_n$ be non-empty sets. An *n-ary operation* from $X_1, X_2, \ldots, X_n$ to another set Y is a function $\omega\colon X_1 \times X_2 \times \cdots \times X_n \to Y$. The set $X_1 \times X_2 \times \cdots \times X_n$ is called the *domain* (Dt.: Definitionsmenge) of the operation, the set Y is called the *codomain* (Dt.: Zielmenge) of the operation, and the number n of operands is called the *arity* (Dt.: Stelligkeit) of the operation.
An *n-ary operation on a set X* is a function $\omega\colon X^n \to X$, i.e., an n-ary operation where $X_1 = X_2 = \ldots = X_n = Y =: X$.

- An operation on a set X is also called an *internal operation* (Dt.: innere Verknüpfung).
- The set $\omega(X_1 \times X_2 \times \cdots \times X_n) \subseteq Y$ is called the *image* or *range* of ω; Dt.: Wertebereich.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Operation

Definition 40 (n-ary Operation, Dt.: n-stellig Verknüpfung)

Let n be a fixed non-negative integer and $X_1, X_2, \ldots, X_n$ be non-empty sets. An *n-ary operation* from $X_1, X_2, \ldots, X_n$ to another set Y is a function $\omega\colon X_1 \times X_2 \times \cdots \times X_n \to Y$. The set $X_1 \times X_2 \times \cdots \times X_n$ is called the *domain* (Dt.: Definitionsmenge) of the operation, the set Y is called the *codomain* (Dt.: Zielmenge) of the operation, and the number n of operands is called the *arity* (Dt.: Stelligkeit) of the operation.
An *n-ary operation on a set X* is a function $\omega\colon X^n \to X$, i.e., an n-ary operation where $X_1 = X_2 = \ldots = X_n = Y =: X$.

- An operation on a set X is also called an *internal operation* (Dt.: innere Verknüpfung).
- The set $\omega(X_1 \times X_2 \times \cdots \times X_n) \subseteq Y$ is called the *image* or *range* of ω; Dt.: Wertebereich.
- *Unary operation*: Arity one. E.g., inverting the sign of a number.
- *Binary operation*: Arity two. E.g., addition of numbers.

Operation

Definition 40 (n-ary Operation, Dt.: n-stellig Verknüpfung)

Let n be a fixed non-negative integer and $X_1, X_2, \ldots, X_n$ be non-empty sets. An *n-ary operation* from $X_1, X_2, \ldots, X_n$ to another set Y is a function $\omega\colon X_1 \times X_2 \times \cdots \times X_n \to Y$. The set $X_1 \times X_2 \times \cdots \times X_n$ is called the *domain* (Dt.: Definitionsmenge) of the operation, the set Y is called the *codomain* (Dt.: Zielmenge) of the operation, and the number n of operands is called the *arity* (Dt.: Stelligkeit) of the operation.
An *n-ary operation on a set X* is a function $\omega\colon X^n \to X$, i.e., an n-ary operation where $X_1 = X_2 = \ldots = X_n = Y =: X$.

- An operation on a set X is also called an *internal operation* (Dt.: innere Verknüpfung).
- The set $\omega(X_1 \times X_2 \times \cdots \times X_n) \subseteq Y$ is called the *image* or *range* of ω; Dt.: Wertebereich.
- *Unary operation*: Arity one. E.g., inverting the sign of a number.
- *Binary operation*: Arity two. E.g., addition of numbers.
- An operation of arity zero is simply an element of the codomain Y, i.e., a constant.

Operation

Definition 40 (n-ary Operation, Dt.: n-stellig Verknüpfung)

Let n be a fixed non-negative integer and $X_1, X_2, \ldots, X_n$ be non-empty sets. An *n-ary operation* from $X_1, X_2, \ldots, X_n$ to another set Y is a function $\omega\colon X_1 \times X_2 \times \cdots \times X_n \to Y$. The set $X_1 \times X_2 \times \cdots \times X_n$ is called the *domain* (Dt.: Definitionsmenge) of the operation, the set Y is called the *codomain* (Dt.: Zielmenge) of the operation, and the number n of operands is called the *arity* (Dt.: Stelligkeit) of the operation.
An *n-ary operation on a set X* is a function $\omega\colon X^n \to X$, i.e., an n-ary operation where $X_1 = X_2 = \ldots = X_n = Y =: X$.

- An operation on a set X is also called an *internal operation* (Dt.: innere Verknüpfung).
- The set $\omega(X_1 \times X_2 \times \cdots \times X_n) \subseteq Y$ is called the *image* or *range* of ω; Dt.: Wertebereich.
- *Unary operation*: Arity one. E.g., inverting the sign of a number.
- *Binary operation*: Arity two. E.g., addition of numbers.
- An operation of arity zero is simply an element of the codomain Y, i.e., a constant.
- Note: The standard division $\div$ is a binary operation neither on the natural numbers nor on the rational numbers.

Operation: Prefix, Infix and Postfix Notation

- So, a binary operation on a set X is a function

$$\omega\colon X \times X \to X \quad \text{with } (x_1, x_2) \mapsto \omega(x_1, x_2) \text{ for } x_1, x_2 \in X.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Operation: Prefix, Infix and Postfix Notation

- So, a binary operation on a set X is a function

$$\omega\colon X \times X \to X \quad \text{with } (x_1, x_2) \mapsto \omega(x_1, x_2) \text{ for } x_1, x_2 \in X.$$

- For binary operations it is customary to use symbols like $\star, \circ, +, \cdot, \div$ rather than letters like ω.
- Furthermore, for binary operations it is common to use the *infix notation*

$$x_1 \star x_2 \qquad \text{or} \qquad x_1 + x_2$$

  rather than the *prefix notation*

$$\star(x_1, x_2) \qquad \text{or} \qquad +(x_1, x_2).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Operation: Prefix, Infix and Postfix Notation

- So, a binary operation on a set X is a function
$$\omega\colon X \times X \to X \quad \text{with } (x_1, x_2) \mapsto \omega(x_1, x_2) \text{ for } x_1, x_2 \in X.$$
- For binary operations it is customary to use symbols like $\star, \circ, +, \cdot, \div$ rather than letters like ω.
- Furthermore, for binary operations it is common to use the *infix notation*
$$x_1 \star x_2 \qquad \text{or} \qquad x_1 + x_2$$
rather than the *prefix notation*
$$\star(x_1, x_2) \qquad \text{or} \qquad +(x_1, x_2).$$
However, prefix notation (aka Polish notation or Łukasiewicz notation) is used by some programming languages, e.g., Lisp.
- Postfix notation, aka reverse Polish notation (RPN), e.g.,
$$(x_1, x_2)\star \qquad \text{or} \qquad (x_1, x_2)+,$$
has been used by some desktop and hand-held calculators (e.g., several Hewlett-Packard products), and is used by stack-oriented programming languages such as Forth, PostScript and RPL.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Operation: Prefix, Infix and Postfix Notation

- So, a binary operation on a set X is a function

$$\omega \colon X \times X \to X \quad \text{with } (x_1, x_2) \mapsto \omega(x_1, x_2) \text{ for } x_1, x_2 \in X.$$

- For binary operations it is customary to use symbols like $\star, \circ, +, \cdot, \div$ rather than letters like ω.
- Furthermore, for binary operations it is common to use the *infix notation*

$$x_1 \star x_2 \qquad \text{or} \qquad x_1 + x_2$$

  rather than the *prefix notation*

$$\star(x_1, x_2) \qquad \text{or} \qquad +(x_1, x_2).$$

  However, prefix notation (aka Polish notation or Łukasiewicz notation) is used by some programming languages, e.g., Lisp.
- Postfix notation, aka reverse Polish notation (RPN), e.g.,

$$(x_1, x_2)\star \qquad \text{or} \qquad (x_1, x_2)+,$$

  has been used by some desktop and hand-held calculators (e.g., several Hewlett-Packard products), and is used by stack-oriented programming languages such as Forth, PostScript and RPL.
- The symbol $-$ tends to be used both for an unary and a binary operation.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Composition of Operations

Definition 41 (Composition, Dt.: Hintereinanderausführung)

Consider two operations $f: A \to B$ and $g: B \to C$. The composition (Dt.: Komposition, Hintereinanderausführung) $g \circ f$ of f and g is defined as

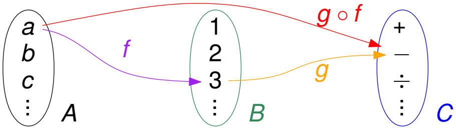
$$(g \circ f)(x) := g(f(x)) \quad \text{for all } x \in A.$$

Definition 41 (Composition, Dt.: Hintereinanderausführung)

Consider two operations $f\colon A \to B$ and $g\colon B \to C$. The composition (Dt.: Komposition, Hintereinanderausführung) $g \circ f$ of f and g is defined as

$$(g \circ f)(x) := g(f(x)) \quad \text{for all } x \in A.$$

- That is, the standard interpretation of $g \circ f$ is "carry out f followed by g".



Definition 41 (Composition, Dt.: Hintereinanderausführung)

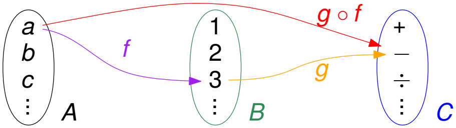
Consider two operations $f\colon A \to B$ and $g\colon B \to C$. The composition (Dt.: Komposition, Hintereinanderausführung) $g \circ f$ of f and g is defined as

$$(g \circ f)(x) := g(f(x)) \quad \text{for all } x \in A.$$

- That is, the standard interpretation of $g \circ f$ is "carry out f followed by g".

Warning

Not all authors stick to the convention $(g \circ f)(x) := g(f(x)) \ldots$



Definition 41 (Composition, Dt.: Hintereinanderausführung)

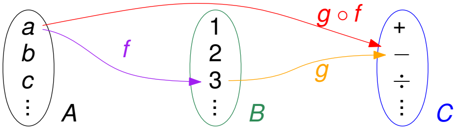
Consider two operations $f\colon A \to B$ and $g\colon B \to C$. The composition (Dt.: Komposition, Hintereinanderausführung) $g \circ f$ of f and g is defined as

$$(g \circ f)(x) := g(f(x)) \quad \text{for all } x \in A.$$

- That is, the standard interpretation of $g \circ f$ is “carry out f followed by g”.
- If $A = B = C := X$ then $\circ$ is a binary operation on operations from X to X.

Warning

Not all authors stick to the convention $(g \circ f)(x) := g(f(x))$ ...



Composition of Operations

Definition 41 (Composition, Dt.: Hintereinanderausführung)

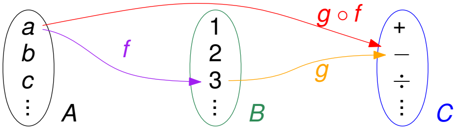
Consider two operations $f: A \to B$ and $g: B \to C$. The composition (Dt.: Komposition, Hintereinanderausführung) $g \circ f$ of f and g is defined as

$$(g \circ f)(x) := g(f(x)) \quad \text{for all } x \in A.$$

- That is, the standard interpretation of $g \circ f$ is "carry out f followed by g".
- If $A = B = C := X$ then $\circ$ is a binary operation on operations from X to X.
- We will use the symbol $\circ$ exclusively for denoting compositions of operations.

Warning

Not all authors stick to the convention $(g \circ f)(x) := g(f(x)) \ldots$



Properties of Operations: Associativity and Commutativity

Definition 42 (Associativity, Dt.: Assoziativität)

A binary operation $\star$ on a (non-empty) set G is *associative* if

$$\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Properties of Operations: Associativity and Commutativity

Definition 42 (Associativity, Dt.: Assoziativität)

A binary operation $\star$ on a (non-empty) set G is *associative* if

$$\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c).$$

- Associativity means that the order in which consecutive operations are applied does not change the result.
- That is, the result does not change depending on whether the parentheses are associated with the first pair or the second pair of operands when the operation is applied to three operands.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Properties of Operations: Associativity and Commutativity

Definition 42 (Associativity, Dt.: Assoziativität)

A binary operation $\star$ on a (non-empty) set G is *associative* if

$$\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c).$$

- Associativity means that the order in which consecutive operations are applied does not change the result.
- That is, the result does not change depending on whether the parentheses are associated with the first pair or the second pair of operands when the operation is applied to three operands.

Definition 43 (Commutativity, Dt.: Kommutativität)

A binary operation $\star$ on a (non-empty) set G is *commutative* if

$$\forall a, b \in G \quad a \star b = b \star a.$$

Definition 42 (Associativity, Dt.: Assoziativität)

A binary operation $\star$ on a (non-empty) set G is *associative* if

$$\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c).$$

- Associativity means that the order in which consecutive operations are applied does not change the result.
- That is, the result does not change depending on whether the parentheses are associated with the first pair or the second pair of operands when the operation is applied to three operands.

Definition 43 (Commutativity, Dt.: Kommutativität)

A binary operation $\star$ on a (non-empty) set G is *commutative* if

$$\forall a, b \in G \quad a \star b = b \star a.$$

- Commutativity means that the order of the operands does not change its result.

Definition 44 (Distributivity, Dt.: Distributivität)

A binary operation $\cdot$ on a (non-empty) set G is *distributive over* a binary operation $+$ on G if

$$\forall a, b, c \in G \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c),$$
$$\forall a, b, c \in G \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c).$$

Properties of Operations: Distributivity

Definition 44 (Distributivity, Dt.: Distributivität)

A binary operation $\cdot$ on a (non-empty) set G is *distributive over* a binary operation $+$ on G if

$$\forall a, b, c \in G \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c),$$
$$\forall a, b, c \in G \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c).$$

- With the standard meaning of $\cdot$ and $+$ over $\mathbb{R}$, multiplication distributes over addition, that is, when multiplying a sum by a factor we can distribute the factor over the summands.
- Note that addition does not distribute over multiplication (over $\mathbb{R}$).

Properties of Operations: Distributivity

Definition 44 (Distributivity, Dt.: Distributivität)

A binary operation $\cdot$ on a (non-empty) set G is *distributive over* a binary operation $+$ on G if

$$\forall a, b, c \in G \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c),$$
$$\forall a, b, c \in G \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c).$$

- With the standard meaning of $\cdot$ and $+$ over $\mathbb{R}$, multiplication distributes over addition, that is, when multiplying a sum by a factor we can distribute the factor over the summands.
- Note that addition does not distribute over multiplication (over $\mathbb{R}$).
- Some textbooks prefer to split up the conditions of Def. 44 and say that $\cdot$ is *left-distributive* if

$$\forall a, b, c \in G \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c),$$

and *right-distributive* if

$$\forall a, b, c \in G \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c).$$

Definition 45 (Neutral element, Dt.: neutrales Element)

The element $n \in G$ is a *neutral element* (aka zero element, identity element) of a binary operation $\star$ on a (non-empty) set G if

$$\forall a \in G \quad a \star n = a = n \star a.$$

Definition 45 (Neutral element, Dt.: neutrales Element)

The element $n \in G$ is a *neutral element* (aka zero element, identity element) of a binary operation $\star$ on a (non-empty) set G if

$$\forall a \in G \quad a \star n = a = n \star a.$$

- Hence, a neutral element of $\star$ on G is an element in G that does not change the value of other elements when combined with them under the operation $\star$.

Definition 45 (Neutral element, Dt.: neutrales Element)

The element $n \in G$ is a *neutral element* (aka zero element, identity element) of a binary operation $\star$ on a (non-empty) set G if

$$\forall a \in G \quad a \star n = a = n \star a.$$

- Hence, a neutral element of $\star$ on G is an element in G that does not change the value of other elements when combined with them under the operation $\star$.
- While addition over $\mathbb{R}$ has zero as neutral element, subtraction does not have a neutral element: We get $a - 0 = a$ but, in general, $0 - a \neq a$.

Properties of Operations: Neutral Element and Inverse Element

Definition 45 (Neutral element, Dt.: neutrales Element)

The element $n \in G$ is a *neutral element* (aka zero element, identity element) of a binary operation $\star$ on a (non-empty) set G if

$$\forall a \in G \quad a \star n = a = n \star a.$$

- Hence, a neutral element of $\star$ on G is an element in G that does not change the value of other elements when combined with them under the operation $\star$.
- While addition over $\mathbb{R}$ has zero as neutral element, subtraction does not have a neutral element: We get $a - 0 = a$ but, in general, $0 - a \neq a$.

Definition 46 (Inverse element, Dt.: inverses Element)

The element $b \in G$ is an *inverse element* of the element $a \in G$ for the binary operation $\star$ on a (non-empty) set G if

$$a \star b = n = b \star a,$$

where n denotes the neutral element of $\star$ on G.

Properties of Operations: Uniqueness of Neutral Element

Lemma 47

A binary operation $\star$ on a (non-empty) set G has at most one neutral element.

Lemma 47

A binary operation $\star$ on a (non-empty) set G has at most one neutral element.

Proof : Assume that $n_1, n_2 \in G$ are neutral elements of $\star$ on G. By Def. 45,

$$\forall a \in G \quad a \star n_1 = a = n_1 \star a \qquad \text{and} \qquad \forall a \in G \quad a \star n_2 = a = n_2 \star a.$$

Lemma 47

A binary operation $\star$ on a (non-empty) set G has at most one neutral element.

Proof : Assume that $n_1, n_2 \in G$ are neutral elements of $\star$ on G. By Def. 45,

$$\forall a \in G \quad a \star n_1 = a = n_1 \star a \qquad \text{and} \qquad \forall a \in G \quad a \star n_2 = a = n_2 \star a.$$

These identities hold for all $a \in G$. Hence, in particular, they have to hold if $a := n_1$ and $a := n_2$:

$$n_2 \star n_1 = n_2 = n_1 \star n_2 \qquad \text{and} \qquad n_1 \star n_2 = n_1 = n_2 \star n_1.$$

Lemma 47

A binary operation $\star$ on a (non-empty) set G has at most one neutral element.

Proof: Assume that $n_1, n_2 \in G$ are neutral elements of $\star$ on G. By Def. 45,

$$\forall a \in G \quad a \star n_1 = a = n_1 \star a \qquad \text{and} \qquad \forall a \in G \quad a \star n_2 = a = n_2 \star a.$$

These identities hold for all $a \in G$. Hence, in particular, they have to hold if $a := n_1$ and $a := n_2$:

$$n_2 \star n_1 = n_2 = n_1 \star n_2 \qquad \text{and} \qquad n_1 \star n_2 = n_1 = n_2 \star n_1.$$

We get

$$n_2 = n_1 \star n_2 = n_1.$$

□

Properties of Operations: Uniqueness of Neutral Element

Lemma 47

A binary operation $\star$ on a (non-empty) set G has at most one neutral element.

Proof : Assume that $n_1, n_2 \in G$ are neutral elements of $\star$ on G. By Def. 45,

$$\forall a \in G \quad a \star n_1 = a = n_1 \star a \qquad \text{and} \qquad \forall a \in G \quad a \star n_2 = a = n_2 \star a.$$

These identities hold for all $a \in G$. Hence, in particular, they have to hold if $a := n_1$ and $a := n_2$:

$$n_2 \star n_1 = n_2 = n_1 \star n_2 \qquad \text{and} \qquad n_1 \star n_2 = n_1 = n_2 \star n_1.$$

We get

$$n_2 = n_1 \star n_2 = n_1.$$

□

Corollary 48

If a binary operation $\star$ on a (non-empty) set G has a neutral element then it is unique.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Properties of Operations: Uniqueness of Neutral Element

Lemma 47

A binary operation $\star$ on a (non-empty) set G has at most one neutral element.

Proof: Assume that $n_1, n_2 \in G$ are neutral elements of $\star$ on G. By Def. 45,

$$\forall a \in G \quad a \star n_1 = a = n_1 \star a \qquad \text{and} \qquad \forall a \in G \quad a \star n_2 = a = n_2 \star a.$$

These identities hold for all $a \in G$. Hence, in particular, they have to hold if $a := n_1$ and $a := n_2$:

$$n_2 \star n_1 = n_2 = n_1 \star n_2 \qquad \text{and} \qquad n_1 \star n_2 = n_1 = n_2 \star n_1.$$

We get

$$n_2 = n_1 \star n_2 = n_1.$$

□

Corollary 48

If a binary operation $\star$ on a (non-empty) set G has a neutral element then it is unique.

- The neutral element is often denoted by 0 if $+$ is used to denote the operation, and by 1 if $\cdot$ denotes the operation.

Properties of Operations: Uniqueness of Inverse Element

Lemma 49

An element $a \in G$ has at most one inverse element $b \in G$ for an associative binary operation $\star$ on G.

Properties of Operations: Uniqueness of Inverse Element

Lemma 49

An element $a \in G$ has at most one inverse element $b \in G$ for an associative binary operation $\star$ on G.

Proof: Assume that $b_1, b_2 \in G$ are inverse elements for $a \in G$ relative to an associative binary operation $\star$ on G. Let $n \in G$ be the neutral element. By Def. 46,

$$a \star b_1 = n = b_1 \star a \qquad \text{and} \qquad a \star b_2 = n = b_2 \star a.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Properties of Operations: Uniqueness of Inverse Element

Lemma 49

An element $a \in G$ has at most one inverse element $b \in G$ for an associative binary operation $\star$ on G.

Proof: Assume that $b_1, b_2 \in G$ are inverse elements for $a \in G$ relative to an associative binary operation $\star$ on G. Let $n \in G$ be the neutral element. By Def. 46,

$$a \star b_1 = n = b_1 \star a \qquad \text{and} \qquad a \star b_2 = n = b_2 \star a.$$

Hence,

$$b_1 = b_1 \star n = b_1 \star (a \star b_2) = (b_1 \star a) \star b_2 = n \star b_2 = b_2.$$

□

Properties of Operations: Uniqueness of Inverse Element

Lemma 49

An element $a \in G$ has at most one inverse element $b \in G$ for an associative binary operation $\star$ on G.

Proof: Assume that $b_1, b_2 \in G$ are inverse elements for $a \in G$ relative to an associative binary operation $\star$ on G. Let $n \in G$ be the neutral element. By Def. 46,

$$a \star b_1 = n = b_1 \star a \qquad \text{and} \qquad a \star b_2 = n = b_2 \star a.$$

Hence,

$$b_1 = b_1 \star n = b_1 \star (a \star b_2) = (b_1 \star a) \star b_2 = n \star b_2 = b_2.$$

□

Corollary 50

If an element $a \in G$ has an inverse element relative to an associative binary operation $\star$ on G then it is unique.

Lemma 49

An element $a \in G$ has at most one inverse element $b \in G$ for an associative binary operation $\star$ on G.

Proof: Assume that $b_1, b_2 \in G$ are inverse elements for $a \in G$ relative to an associative binary operation $\star$ on G. Let $n \in G$ be the neutral element. By Def. 46,

$$a \star b_1 = n = b_1 \star a \qquad \text{and} \qquad a \star b_2 = n = b_2 \star a.$$

Hence,

$$b_1 = b_1 \star n = b_1 \star (a \star b_2) = (b_1 \star a) \star b_2 = n \star b_2 = b_2.$$

□

Corollary 50

If an element $a \in G$ has an inverse element relative to an associative binary operation $\star$ on G then it is unique.

- Again, one may consider a left-inverse element and a right-inverse element.

Properties of Operations: Uniqueness of Inverse Element

Lemma 49

An element $a \in G$ has at most one inverse element $b \in G$ for an associative binary operation $\star$ on G.

Proof: Assume that $b_1, b_2 \in G$ are inverse elements for $a \in G$ relative to an associative binary operation $\star$ on G. Let $n \in G$ be the neutral element. By Def. 46,

$$a \star b_1 = n = b_1 \star a \qquad \text{and} \qquad a \star b_2 = n = b_2 \star a.$$

Hence,

$$b_1 = b_1 \star n = b_1 \star (a \star b_2) = (b_1 \star a) \star b_2 = n \star b_2 = b_2.$$

□

Corollary 50

If an element $a \in G$ has an inverse element relative to an associative binary operation $\star$ on G then it is unique.

- Again, one may consider a left-inverse element and a right-inverse element.
- The inverse element of a is often denoted by a^{-1} if $\cdot$ or $\circ$ is used to denote the operation, and by $-a$ if $+$ denotes the operation.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 51 (Group, Dt.: Gruppe)

A set G together with a binary operation $\star$ on G defines a *group* if the following properties hold:

Definition 51 (Group, Dt.: Gruppe)

A set G together with a binary operation $\star$ on G defines a *group* if the following properties hold:

1. Associativity: $\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c)$.

Definition 51 (Group, Dt.: Gruppe)

A set G together with a binary operation $\star$ on G defines a *group* if the following properties hold:

1. Associativity: $\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c)$.
2. Neutral element: There exists an element $n \in G$ such that $\forall a \in G \quad n \star a = a = a \star n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 51 (Group, Dt.: Gruppe)

A set G together with a binary operation $\star$ on G defines a *group* if the following properties hold:

1. Associativity: $\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c)$.
2. Neutral element: There exists an element $n \in G$ such that $\forall a \in G \quad n \star a = a = a \star n$.
3. Inverse element: For all $a \in G$ there exists an inverse $b \in G$, satisfying $a \star b = n = b \star a$.

Group

Definition 51 (Group, Dt.: Gruppe)

A set G together with a binary operation $\star$ on G defines a *group* if the following properties hold:

1. Associativity: $\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c)$.
2. Neutral element: There exists an element $n \in G$ such that $\forall a \in G \quad n \star a = a = a \star n$.
3. Inverse element: For all $a \in G$ there exists an inverse $b \in G$, satisfying $a \star b = n = b \star a$.

- Since $\star$ is a binary operation on G, we know that G is closed under the application of $\star$. That is, if $a, b \in G$ then $a \star b \in G$.

Group

Definition 51 (Group, Dt.: Gruppe)

A set G together with a binary operation $\star$ on G defines a *group* if the following properties hold:

1. Associativity: $\forall a, b, c \in G \quad (a \star b) \star c = a \star (b \star c)$.
2. Neutral element: There exists an element $n \in G$ such that $\forall a \in G \quad n \star a = a = a \star n$.
3. Inverse element: For all $a \in G$ there exists an inverse $b \in G$, satisfying $a \star b = n = b \star a$.

- Since $\star$ is a binary operation on G, we know that G is closed under the application of $\star$. That is, if $a, b \in G$ then $a \star b \in G$.
- Note that $a \star b = b \star a$ is not required for all $a, b \in G$. That is, commutativity need not hold!

Definition 52 (Abelian Group, Dt.: Abelsche Gruppe)

A set G together with a binary operation $\star$ on G defines an *Abelian group* (aka *commutative group*) if the following properties hold:

Abelian Group

Definition 52 (Abelian Group, Dt.: Abelsche Gruppe)

A set G together with a binary operation $\star$ on G defines an *Abelian group* (aka *commutative group*) if the following properties hold:

1. $(G, \star)$ is a group.

Abelian Group

Definition 52 (Abelian Group, Dt.: Abelsche Gruppe)

A set G together with a binary operation $\star$ on G defines an *Abelian group* (aka *commutative group*) if the following properties hold:

1. $(G, \star)$ is a group.
2. Commutativity: $\forall a, b \in G \quad a \star b = b \star a$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Abelian Group

Definition 52 (Abelian Group, Dt.: Abelsche Gruppe)

A set G together with a binary operation $\star$ on G defines an *Abelian group* (aka *commutative group*) if the following properties hold:

1. $(G, \star)$ is a group.
2. Commutativity: $\forall a, b \in G \quad a \star b = b \star a$.

- Sample (Abelian) groups: the integers $\mathbb{Z}$ under addition, non-zero rational numbers $\mathbb{Q} \backslash \{0\}$ under multiplication.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Abelian Group

Definition 52 (Abelian Group, Dt.: Abelsche Gruppe)

A set G together with a binary operation $\star$ on G defines an *Abelian group* (aka *commutative group*) if the following properties hold:

1. $(G, \star)$ is a group.
2. Commutativity: $\forall a, b \in G \quad a \star b = b \star a$.

- Sample (Abelian) groups: the integers $\mathbb{Z}$ under addition, non-zero rational numbers $\mathbb{Q} \backslash \{0\}$ under multiplication.
- Not a group: The integers under multiplication.

Finite Group

- A group $(G, \star)$ is finite if G has a finite number of elements.
- The number of elements of a finite group is called the *order* of the group.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Finite Group

- A group $(G, \star)$ is finite if G has a finite number of elements.
- The number of elements of a finite group is called the *order* of the group.
- A finite group is completely described by its *multiplication table* (aka *Cayley table*). Dt.: Verknüpfungstabelle.
- By convention, in a multiplication table the result for $a \star b$ is found by intersecting row a with column b.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Finite Group

- A group $(G, \star)$ is finite if G has a finite number of elements.
- The number of elements of a finite group is called the *order* of the group.
- A finite group is completely described by its *multiplication table* (aka *Cayley table*). Dt.: Verknüpfungstabelle.
- By convention, in a multiplication table the result for $a \star b$ is found by intersecting row a with column b.
- Multiplication tables for groups of orders two and three:

$\star$	n	a
n	n	a
a	a	n

$\star$	n	a	b
n	n	a	b
a	a	b	n
b	b	n	a

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Finite Group

- A group $(G, \star)$ is finite if G has a finite number of elements.
- The number of elements of a finite group is called the *order* of the group.
- A finite group is completely described by its *multiplication table* (aka *Cayley table*). Dt.: Verknüpfungstabelle.
- By convention, in a multiplication table the result for $a \star b$ is found by intersecting row a with column b.
- Multiplication tables for groups of orders two and three:

$\star$	n	a
n	n	a
a	a	n

$\star$	n	a	b
n	n	a	b
a	a	b	n
b	b	n	a

- Up to renaming the elements of the groups, these are the only possible multiplication tables for groups of orders two and three.

Finite Group

- A group $(G, \star)$ is finite if G has a finite number of elements.
- The number of elements of a finite group is called the *order* of the group.
- A finite group is completely described by its *multiplication table* (aka *Cayley table*). Dt.: Verknüpfungstabelle.
- By convention, in a multiplication table the result for $a \star b$ is found by intersecting row a with column b.
- Multiplication tables for groups of orders two and three:

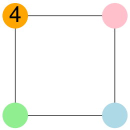
$\star$	n	a
n	n	a
a	a	n

$\star$	n	a	b
n	n	a	b
a	a	b	n
b	b	n	a

- Up to renaming the elements of the groups, these are the only possible multiplication tables for groups of orders two and three.
- Again up to renaming, there are only two possible multiplication tables for groups with four elements, i.e., only two different groups.

Finite Group: Dihedral Group D_4

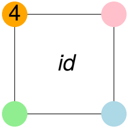
- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:



UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

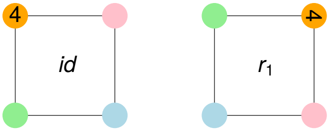
Finite Group: Dihedral Group D_4

- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - *id*,



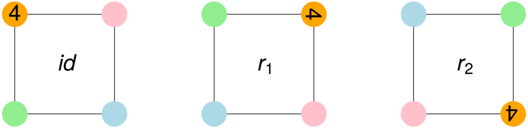
Finite Group: Dihedral Group D_4

- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o),



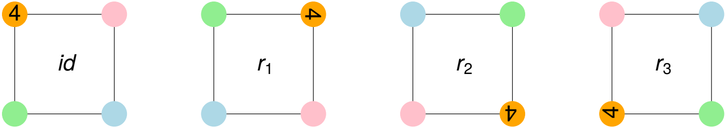
Finite Group: Dihedral Group D_4

- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o), r_2 (CW rotation by 180^o),



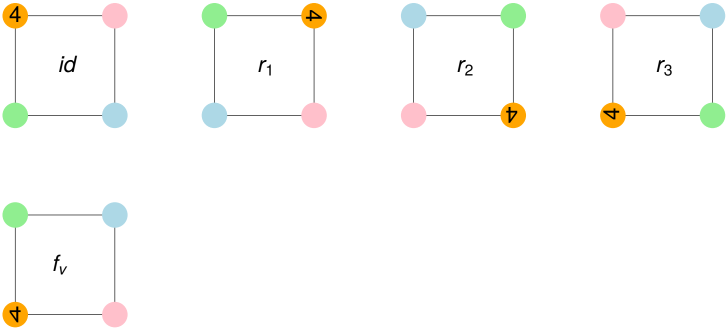
Finite Group: Dihedral Group D_4

- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o), r_2 (CW rotation by 180^o), r_3 (CW rotation by 270^o);



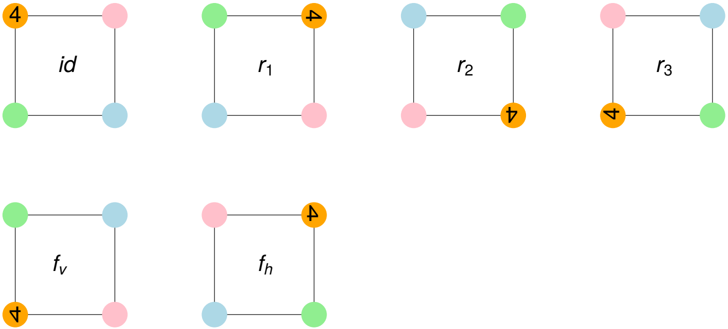
Finite Group: Dihedral Group D_4

- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o), r_2 (CW rotation by 180^o), r_3 (CW rotation by 270^o);
  - f_v (vertical flip),

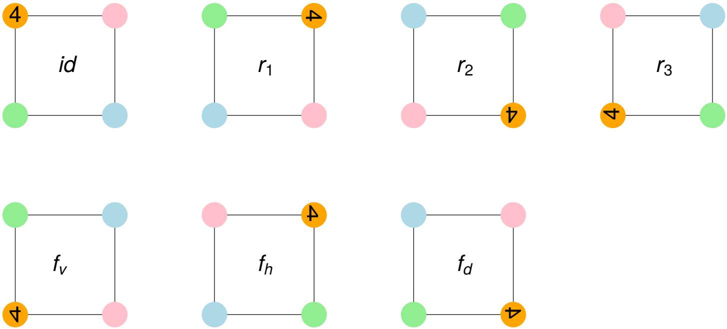


Finite Group: Dihedral Group D_4

- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o), r_2 (CW rotation by 180^o), r_3 (CW rotation by 270^o);
  - f_v (vertical flip), f_h (horizontal flip),

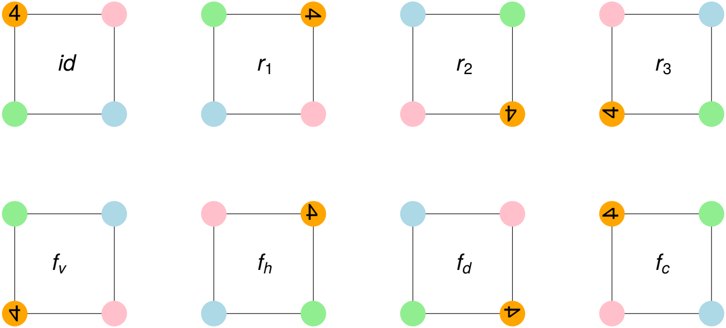


- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o), r_2 (CW rotation by 180^o), r_3 (CW rotation by 270^o);
  - f_v (vertical flip), f_h (horizontal flip), f_d (diagonal flip),



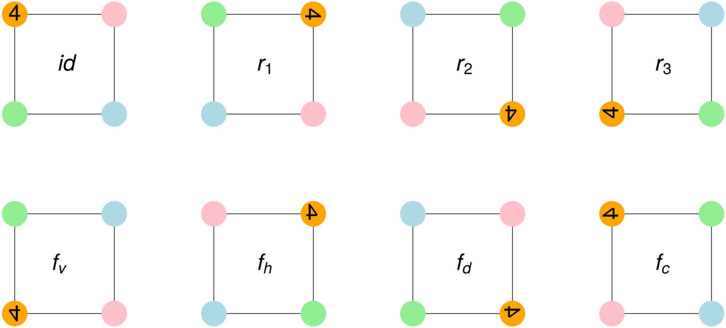
Finite Group: Dihedral Group D_4

- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o), r_2 (CW rotation by 180^o), r_3 (CW rotation by 270^o);
  - f_v (vertical flip), f_h (horizontal flip), f_d (diagonal flip), f_c (counter-diagonal flip).



Finite Group: Dihedral Group D_4

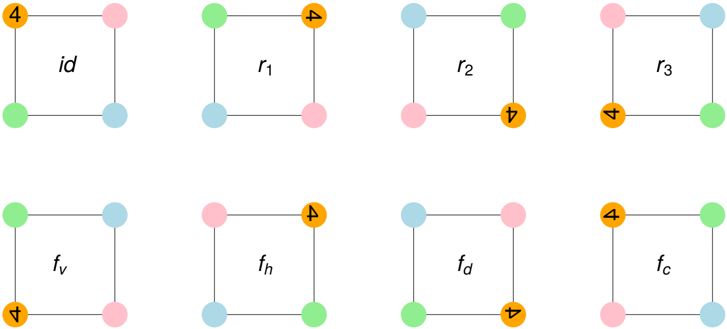
- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o), r_2 (CW rotation by 180^o), r_3 (CW rotation by 270^o);
  - f_v (vertical flip), f_h (horizontal flip), f_d (diagonal flip), f_c (counter-diagonal flip).



- Does D_4 have eight elements? Or did we miss any element?

Finite Group: Dihedral Group D_4

- The *dihedral group* (Dt.: Diedergruppe) D_4 is formed by the clockwise rotations and reflections of a square which map the square onto itself:
  - id, r_1 (CW rotation by 90^o), r_2 (CW rotation by 180^o), r_3 (CW rotation by 270^o);
  - f_v (vertical flip), f_h (horizontal flip), f_d (diagonal flip), f_c (counter-diagonal flip).



- Does D_4 have eight elements? Or did we miss any element?
- No, we didn't!

Finite Group: Dihedral Group D_4

- We denote the composition of functions by $\circ$.
- Multiplication table of D_4:

$\circ$	id	r_1	r_2	r_3	f_v	f_h	f_d	f_c
id	id	r_1	r_2	r_3	f_v	f_h	f_d	f_c
r_1	r_1	r_2	r_3	id	f_c	f_d	f_v	f_h
r_2	r_2	r_3	id	r_1	f_h	f_v	f_c	f_d
r_3	r_3	id	r_1	r_2	f_d	f_c	f_h	f_v
f_v	f_v	f_d	f_h	f_c	id	r_2	r_1	r_3
f_h	f_h	f_c	f_v	f_d	r_2	id	r_3	r_1
f_d	f_d	f_h	f_c	f_v	r_3	r_1	id	r_2
f_c	f_c	f_v	f_d	f_h	r_1	r_3	r_2	id

- E.g., $f_d \circ f_v$, which means flip vertically and then flip diagonally, corresponds to a (clockwise) rotation by 270^o, i.e., to r_3.

- We denote the composition of functions by $\circ$.
- Multiplication table of D_4:

$\circ$	id	r_1	r_2	r_3	f_v	f_h	f_d	f_c
id	id	r_1	r_2	r_3	f_v	f_h	f_d	f_c
r_1	r_1	r_2	r_3	id	f_c	f_d	f_v	f_h
r_2	r_2	r_3	id	r_1	f_h	f_v	f_c	f_d
r_3	r_3	id	r_1	r_2	f_d	f_c	f_h	f_v
f_v	f_v	f_d	f_h	f_c	id	r_2	r_1	r_3
f_h	f_h	f_c	f_v	f_d	r_2	id	r_3	r_1
f_d	f_d	f_h	f_c	f_v	r_3	r_1	id	r_2
f_c	f_c	f_v	f_d	f_h	r_1	r_3	r_2	id

- E.g., $f_d \circ f_v$, which means flip vertically and then flip diagonally, corresponds to a (clockwise) rotation by 270^o, i.e., to r_3.
- Note: $f_d \circ f_v \neq f_v \circ f_d$. That is, D_4 is not commutative.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- We denote the composition of functions by $\circ$.
- Multiplication table of D_4:

$\circ$	id	r_1	r_2	r_3	f_v	f_h	f_d	f_c
id	id	r_1	r_2	r_3	f_v	f_h	f_d	f_c
r_1	r_1	r_2	r_3	id	f_c	f_d	f_v	f_h
r_2	r_2	r_3	id	r_1	f_h	f_v	f_c	f_d
r_3	r_3	id	r_1	r_2	f_d	f_c	f_h	f_v
f_v	f_v	f_d	f_h	f_c	id	r_2	r_1	r_3
f_h	f_h	f_c	f_v	f_d	r_2	id	r_3	r_1
f_d	f_d	f_h	f_c	f_v	r_3	r_1	id	r_2
f_c	f_c	f_v	f_d	f_h	r_1	r_3	r_2	id

- E.g., $f_d \circ f_v$, which means flip vertically and then flip diagonally, corresponds to a (clockwise) rotation by 270^o, i.e., to r_3.
- Note: $f_d \circ f_v \neq f_v \circ f_d$. That is, D_4 is not commutative.
- Note that each one of the transformations appears exactly once in each row and each column of the table: *Latin square*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Geometric Crystal Classes

- D_4 is one of the so-called *crystallographic point groups*, which describe sets of symmetry operations relative to a fixed point. Aka *geometric crystal class*.
- Each operation leaves the structure of the crystal unchanged. That is, the same types of atoms appear in similar positions as before the transformation induced by the operation.

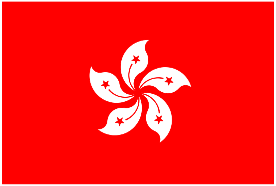
Real-World Application: Geometric Crystal Classes

- D_4 is one of the so-called *crystallographic point groups*, which describe sets of symmetry operations relative to a fixed point. Aka *geometric crystal class*.
- Each operation leaves the structure of the crystal unchanged. That is, the same types of atoms appear in similar positions as before the transformation induced by the operation.
- Crystallographic point groups and their cousins, three-dimensional space groups, are studied and used by scientists such as crystallographers, mineralogists, and physicists.
- See, e.g., the International Tables for Crystallography by Hahn, doi:10.1107/97809553602060000100.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Geometric Crystal Classes

- D_4 is one of the so-called *crystallographic point groups*, which describe sets of symmetry operations relative to a fixed point. Aka *geometric crystal class*.
- Each operation leaves the structure of the crystal unchanged. That is, the same types of atoms appear in similar positions as before the transformation induced by the operation.
- Crystallographic point groups and their cousins, three-dimensional space groups, are studied and used by scientists such as crystallographers, mineralogists, and physicists.
- See, e.g., the International Tables for Crystallography by Hahn, doi:10.1107/97809553602060000100.



The Bauhinia flower has C_5 symmetry, and each star has D_5 symmetry.

This (color-inverted) snowflake has D_6 symmetry.

Definition 53 (Ring, Dt.: Ring mit Eins)

A set R which possesses an “addition” $+ : R \times R \to R$ and a “multiplication” $\cdot : R \times R \to R$ defines a *(unit) ring* if the following conditions hold:

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 53 (Ring, Dt.: Ring mit Eins)

A set R which possesses an “addition” $+ : R \times R \to R$ and a “multiplication” $\cdot : R \times R \to R$ defines a *(unit) ring* if the following conditions hold:

1. $(R, +)$ is an Abelian group with neutral element $0 \in R$ (“zero” element).

Ring

Definition 53 (Ring, Dt.: Ring mit Eins)

A set R which possesses an "addition" $+ : R \times R \to R$ and a "multiplication" $\cdot : R \times R \to R$ defines a *(unit) ring* if the following conditions hold:

1. $(R, +)$ is an Abelian group with neutral element $0 \in R$ ("zero" element).
2. Associativity: $\forall a, b, c \in R \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 53 (Ring, Dt.: Ring mit Eins)

A set R which possesses an "addition" $+ : R \times R \to R$ and a "multiplication" $\cdot : R \times R \to R$ defines a *(unit) ring* if the following conditions hold:

1. $(R, +)$ is an Abelian group with neutral element $0 \in R$ ("zero" element).
2. Associativity: $\forall a, b, c \in R \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Distributivity:
   $\forall a, b, c \in R \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c)$;
   $\forall a, b, c \in R \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Ring

Definition 53 (Ring, Dt.: Ring mit Eins)

A set R which possesses an "addition" $+ : R \times R \to R$ and a "multiplication" $\cdot : R \times R \to R$ defines a *(unit) ring* if the following conditions hold:

1. $(R, +)$ is an Abelian group with neutral element $0 \in R$ ("zero" element).
2. Associativity: $\forall a, b, c \in R \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Distributivity:
   $\forall a, b, c \in R \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c)$;
   $\forall a, b, c \in R \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c)$.
4. Neutral element: There exists an element $1 \in R$ ("one" element) such that $\forall a \in R \quad 1 \cdot a = a = a \cdot 1$.

Definition 53 (Ring, Dt.: Ring mit Eins)

A set R which possesses an “addition” $+ : R \times R \to R$ and a “multiplication” $\cdot : R \times R \to R$ defines a *(unit) ring* if the following conditions hold:

1. $(R, +)$ is an Abelian group with neutral element $0 \in R$ (“zero” element).
2. Associativity: $\forall a, b, c \in R \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Distributivity:
   $\forall a, b, c \in R \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c)$;
   $\forall a, b, c \in R \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c)$.
4. Neutral element: There exists an element $1 \in R$ (“one” element) such that $\forall a \in R \quad 1 \cdot a = a = a \cdot 1$.

- Note: The elements of a ring need not be numbers even though it is customary to use the terminology of arithmetic applied to numbers.

Definition 53 (Ring, Dt.: Ring mit Eins)

A set R which possesses an "addition" $+ : R \times R \to R$ and a "multiplication" $\cdot : R \times R \to R$ defines a *(unit) ring* if the following conditions hold:

1. $(R, +)$ is an Abelian group with neutral element $0 \in R$ ("zero" element).
2. Associativity: $\forall a, b, c \in R \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Distributivity:
   $\forall a, b, c \in R \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c)$;
   $\forall a, b, c \in R \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c)$.
4. Neutral element: There exists an element $1 \in R$ ("one" element) such that $\forall a \in R \quad 1 \cdot a = a = a \cdot 1$.

- Note: The elements of a ring need not be numbers even though it is customary to use the terminology of arithmetic applied to numbers.
- Note that $a \cdot b = b \cdot a$ for all $a, b \in R$ is not required. If commutativity holds then $(R, +, \cdot)$ forms a *commutative ring*.

Definition 53 (Ring, Dt.: Ring mit Eins)

A set R which possesses an “addition” $+ : R \times R \to R$ and a “multiplication” $\cdot : R \times R \to R$ defines a *(unit) ring* if the following conditions hold:

1. $(R, +)$ is an Abelian group with neutral element $0 \in R$ (“zero” element).
2. Associativity: $\forall a, b, c \in R \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Distributivity:
   $\forall a, b, c \in R \quad a \cdot (b + c) = (a \cdot b) + (a \cdot c)$;
   $\forall a, b, c \in R \quad (a + b) \cdot c = (a \cdot c) + (b \cdot c)$.
4. Neutral element: There exists an element $1 \in R$ (“one” element) such that $\forall a \in R \quad 1 \cdot a = a = a \cdot 1$.

- Note: The elements of a ring need not be numbers even though it is customary to use the terminology of arithmetic applied to numbers.
- Note that $a \cdot b = b \cdot a$ for all $a, b \in R$ is not required. If commutativity holds then $(R, +, \cdot)$ forms a *commutative ring*.
- Sample ring: The set of all continuous real-valued functions defined over an interval $[\alpha, \beta] \subset \mathbb{R}$, with addition and multiplication of functions as operations, forms a ring.

Field

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

Field

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

1. Associativity: $\forall a, b, c \in F \quad (a + b) + c = a + (b + c)$.
2. Associativity: $\forall a, b, c \in F \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

1. Associativity: $\forall a, b, c \in F \quad (a + b) + c = a + (b + c)$.
2. Associativity: $\forall a, b, c \in F \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Commutativity: $\forall a, b \in F \quad a + b = b + a$.
4. Commutativity: $\forall a, b \in F \quad a \cdot b = b \cdot a$.

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

1. Associativity: $\forall a, b, c \in F \quad (a + b) + c = a + (b + c)$.
2. Associativity: $\forall a, b, c \in F \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Commutativity: $\forall a, b \in F \quad a + b = b + a$.
4. Commutativity: $\forall a, b \in F \quad a \cdot b = b \cdot a$.
5. Distributivity: $\forall a, b, c \in F \quad a \cdot (b + c) = a \cdot b + a \cdot c$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

1. Associativity: $\forall a, b, c \in F \quad (a + b) + c = a + (b + c)$.
2. Associativity: $\forall a, b, c \in F \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Commutativity: $\forall a, b \in F \quad a + b = b + a$.
4. Commutativity: $\forall a, b \in F \quad a \cdot b = b \cdot a$.
5. Distributivity: $\forall a, b, c \in F \quad a \cdot (b + c) = a \cdot b + a \cdot c$.
6. Neutral element: There exists an element $0 \in F$ such that $\forall a \in F \quad 0 + a = a$.
7. Neutral element: There exists an element $1 \in F$ such that $\forall a \in F \quad 1 \cdot a = a$.

Field

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

1. Associativity: $\forall a, b, c \in F \quad (a + b) + c = a + (b + c)$.
2. Associativity: $\forall a, b, c \in F \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Commutativity: $\forall a, b \in F \quad a + b = b + a$.
4. Commutativity: $\forall a, b \in F \quad a \cdot b = b \cdot a$.
5. Distributivity: $\forall a, b, c \in F \quad a \cdot (b + c) = a \cdot b + a \cdot c$.
6. Neutral element: There exists an element $0 \in F$ such that $\forall a \in F \quad 0 + a = a$.
7. Neutral element: There exists an element $1 \in F$ such that $\forall a \in F \quad 1 \cdot a = a$.
8. For all $a \in F$ there exists an additive inverse $b \in F$, satisfying $a + b = 0$.
9. For all $a \in F \backslash \{0\}$ there exists a multiplicative inverse $b \in F$, satisfying $a \cdot b = 1$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

1. Associativity: $\forall a, b, c \in F \quad (a + b) + c = a + (b + c)$.
2. Associativity: $\forall a, b, c \in F \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Commutativity: $\forall a, b \in F \quad a + b = b + a$.
4. Commutativity: $\forall a, b \in F \quad a \cdot b = b \cdot a$.
5. Distributivity: $\forall a, b, c \in F \quad a \cdot (b + c) = a \cdot b + a \cdot c$.
6. Neutral element: There exists an element $0 \in F$ such that $\forall a \in F \quad 0 + a = a$.
7. Neutral element: There exists an element $1 \in F$ such that $\forall a \in F \quad 1 \cdot a = a$.
8. For all $a \in F$ there exists an additive inverse $b \in F$, satisfying $a + b = 0$.
9. For all $a \in F \setminus \{0\}$ there exists a multiplicative inverse $b \in F$, satisfying $a \cdot b = 1$.
10. $0 \neq 1$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

1. Associativity: $\forall a, b, c \in F \quad (a + b) + c = a + (b + c)$.
2. Associativity: $\forall a, b, c \in F \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Commutativity: $\forall a, b \in F \quad a + b = b + a$.
4. Commutativity: $\forall a, b \in F \quad a \cdot b = b \cdot a$.
5. Distributivity: $\forall a, b, c \in F \quad a \cdot (b + c) = a \cdot b + a \cdot c$.
6. Neutral element: There exists an element $0 \in F$ such that $\forall a \in F \quad 0 + a = a$.
7. Neutral element: There exists an element $1 \in F$ such that $\forall a \in F \quad 1 \cdot a = a$.
8. For all $a \in F$ there exists an additive inverse $b \in F$, satisfying $a + b = 0$.
9. For all $a \in F \setminus \{0\}$ there exists a multiplicative inverse $b \in F$, satisfying $a \cdot b = 1$.
10. $0 \neq 1$.

- Again, the elements of F need not be numbers.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field

Definition 54 (Field, Dt.: Körper)

A set F which possesses an "addition" $+ : F \times F \to F$ and a "multiplication" $\cdot : F \times F \to F$ defines a *field* if the following conditions hold:

1. Associativity: $\forall a, b, c \in F \quad (a + b) + c = a + (b + c)$.
2. Associativity: $\forall a, b, c \in F \quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$.
3. Commutativity: $\forall a, b \in F \quad a + b = b + a$.
4. Commutativity: $\forall a, b \in F \quad a \cdot b = b \cdot a$.
5. Distributivity: $\forall a, b, c \in F \quad a \cdot (b + c) = a \cdot b + a \cdot c$.
6. Neutral element: There exists an element $0 \in F$ such that $\forall a \in F \quad 0 + a = a$.
7. Neutral element: There exists an element $1 \in F$ such that $\forall a \in F \quad 1 \cdot a = a$.
8. For all $a \in F$ there exists an additive inverse $b \in F$, satisfying $a + b = 0$.
9. For all $a \in F \setminus \{0\}$ there exists a multiplicative inverse $b \in F$, satisfying $a \cdot b = 1$.
10. $0 \neq 1$.

- Again, the elements of F need not be numbers.
- Note: The multiplication sign is often dropped if the meaning is clear within a specific context: It is common to write ab rather than $a \cdot b$.

Field: Subtraction and Division

- In the sequel, we denote the additive neutral element of a field $(F, +, \cdot)$ by 0 and its multiplicative neutral element by 1. Furthermore, we denote the inverse elements of $b \in F$ by $-b$ and b^{-1}.

Field: Subtraction and Division

- In the sequel, we denote the additive neutral element of a field $(F, +, \cdot)$ by 0 and its multiplicative neutral element by 1. Furthermore, we denote the inverse elements of $b \in F$ by $-b$ and b^{-1}.

Definition 55

Let $(F, +, \cdot)$ be a field. We define the binary operation "subtraction" $-: F \times F \rightarrow F$:

$$\forall a, b \in F \quad a - b := a + (-b)$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Subtraction and Division

- In the sequel, we denote the additive neutral element of a field $(F, +, \cdot)$ by 0 and its multiplicative neutral element by 1. Furthermore, we denote the inverse elements of $b \in F$ by $-b$ and b^{-1}.

Definition 55

Let $(F, +, \cdot)$ be a field. We define the binary operation "subtraction" $-: F \times F \to F$:

$$\forall a, b \in F \quad a - b := a + (-b)$$

Definition 56

Let $(F, +, \cdot)$ be a field. We define the binary operation "division" $\div: F \times (F\backslash\{0\}) \to F$:

$$\forall a \in F, b \in F\backslash\{0\} \quad a \div b := a \cdot b^{-1}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Subtraction and Division

- In the sequel, we denote the additive neutral element of a field $(F, +, \cdot)$ by 0 and its multiplicative neutral element by 1. Furthermore, we denote the inverse elements of $b \in F$ by $-b$ and b^{-1}.

Definition 55

Let $(F, +, \cdot)$ be a field. We define the binary operation "subtraction" $- : F \times F \to F$:

$$\forall a, b \in F \quad a - b := a + (-b)$$

Definition 56

Let $(F, +, \cdot)$ be a field. We define the binary operation "division" $\div : F \times (F \backslash \{0\}) \to F$:

$$\forall a \in F, b \in F \backslash \{0\} \quad a \div b := a \cdot b^{-1}$$

Lemma 57

Let $(F, +, \cdot)$ be a field.

$$\forall a \in F \quad a - a = 0 \qquad \text{and} \qquad \forall a \in F \backslash \{0\} \quad a \div a = 1.$$

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \quad \text{and} \quad 1^{-1} = 1 \quad \text{and} \quad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$. Similarly, $1 = 1 \cdot 1^{-1} = 1^{-1}$.

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$. Similarly, $1 = 1 \cdot 1^{-1} = 1^{-1}$.
Let $a \in F$ be arbitrary but fixed. Then

$$0 = 0 \cdot a + \big(-(0 \cdot a)\big)$$

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$. Similarly, $1 = 1 \cdot 1^{-1} = 1^{-1}$.
Let $a \in F$ be arbitrary but fixed. Then

$$0 = 0 \cdot a + \big(-(0 \cdot a)\big) = (0 + 0) \cdot a - 0 \cdot a$$

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$. Similarly, $1 = 1 \cdot 1^{-1} = 1^{-1}$.
Let $a \in F$ be arbitrary but fixed. Then

$$0 = 0 \cdot a + \big(-(0 \cdot a)\big) = (0+0) \cdot a - 0 \cdot a = (0 \cdot a + 0 \cdot a) - 0 \cdot a$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$. Similarly, $1 = 1 \cdot 1^{-1} = 1^{-1}$.
Let $a \in F$ be arbitrary but fixed. Then

$$\begin{aligned} 0 &= 0 \cdot a + \big(-(0 \cdot a)\big) = (0+0) \cdot a - 0 \cdot a = (0 \cdot a + 0 \cdot a) - 0 \cdot a \\ &= 0 \cdot a + (0 \cdot a - 0 \cdot a) \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$. Similarly, $1 = 1 \cdot 1^{-1} = 1^{-1}$.
Let $a \in F$ be arbitrary but fixed. Then

$$\begin{aligned} 0 &= 0 \cdot a + \big(-(0 \cdot a)\big) = (0+0) \cdot a - 0 \cdot a = (0 \cdot a + 0 \cdot a) - 0 \cdot a \\ &= 0 \cdot a + (0 \cdot a - 0 \cdot a) = 0 \cdot a + 0 \end{aligned}$$

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$. Similarly, $1 = 1 \cdot 1^{-1} = 1^{-1}$.
Let $a \in F$ be arbitrary but fixed. Then

$$\begin{aligned} 0 &= 0 \cdot a + \big(-(0 \cdot a)\big) = (0 + 0) \cdot a - 0 \cdot a = (0 \cdot a + 0 \cdot a) - 0 \cdot a \\ &= 0 \cdot a + (0 \cdot a - 0 \cdot a) = 0 \cdot a + 0 = 0 \cdot a. \end{aligned}$$

□

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Theorem 58

Let $(F, +, \cdot)$ be a field. Then

$$-0 = 0 \qquad \text{and} \qquad 1^{-1} = 1 \qquad \text{and} \qquad \forall a \in F \quad 0 \cdot a = 0.$$

Proof: We have $0 = 0 + (-0) = -0$. Similarly, $1 = 1 \cdot 1^{-1} = 1^{-1}$.
Let $a \in F$ be arbitrary but fixed. Then

$$\begin{aligned} 0 &= 0 \cdot a + \big(-(0 \cdot a)\big) = (0+0) \cdot a - 0 \cdot a = (0 \cdot a + 0 \cdot a) - 0 \cdot a \\ &= 0 \cdot a + (0 \cdot a - 0 \cdot a) = 0 \cdot a + 0 = 0 \cdot a. \end{aligned}$$

□

Theorem 59

Let $(F, +, \cdot)$ be a field. Then, for all $a, b \in F$,

$$(-1) \cdot a = -a \qquad \text{and} \qquad -(-a) = a \qquad \text{and}$$

$$(-a) \cdot b = -(a \cdot b) = a \cdot (-b) \qquad \text{and} \qquad (-a) \cdot (-b) = a \cdot b.$$

Field: Properties of the Operations

Theorem 60

Let $(F, +, \cdot)$ be a field. Then, for all $a, b \in F$,

$$a \cdot b = 0 \quad \Rightarrow \quad \big(a = 0 \;\text{ or }\; b = 0\big).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Theorem 60

Let $(F, +, \cdot)$ be a field. Then, for all $a, b \in F$,

$$a \cdot b = 0 \quad \Rightarrow \quad \big(a = 0 \text{ or } b = 0\big).$$

Proof : Let $a, b \in F$ be arbitrary but fixed with $a \cdot b = 0$ and $a \neq 0$. We get

$$0 = a^{-1} \cdot 0$$

Field: Properties of the Operations

Theorem 60

Let $(F, +, \cdot)$ be a field. Then, for all $a, b \in F$,

$$a \cdot b = 0 \quad \Rightarrow \quad \big(a = 0 \ \text{ or } \ b = 0\big).$$

Proof: Let $a, b \in F$ be arbitrary but fixed with $a \cdot b = 0$ and $a \neq 0$. We get

$$0 = a^{-1} \cdot 0 = a^{-1} \cdot (a \cdot b)$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Theorem 60

Let $(F, +, \cdot)$ be a field. Then, for all $a, b \in F$,

$$a \cdot b = 0 \quad \Rightarrow \quad \big(a = 0 \text{ or } b = 0\big).$$

Proof: Let $a, b \in F$ be arbitrary but fixed with $a \cdot b = 0$ and $a \neq 0$. We get

$$0 = a^{-1} \cdot 0 = a^{-1} \cdot (a \cdot b) = (a^{-1} \cdot a) \cdot b$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Theorem 60

Let $(F, +, \cdot)$ be a field. Then, for all $a, b \in F$,

$$a \cdot b = 0 \quad \Rightarrow \quad \big(a = 0 \text{ or } b = 0\big).$$

Proof: Let $a, b \in F$ be arbitrary but fixed with $a \cdot b = 0$ and $a \neq 0$. We get

$$0 = a^{-1} \cdot 0 = a^{-1} \cdot (a \cdot b) = (a^{-1} \cdot a) \cdot b = 1 \cdot b = b.$$

□

Theorem 60

Let $(F, +, \cdot)$ be a field. Then, for all $a, b \in F$,

$$a \cdot b = 0 \quad \Rightarrow \quad \big(a = 0 \text{ or } b = 0\big).$$

Proof: Let $a, b \in F$ be arbitrary but fixed with $a \cdot b = 0$ and $a \neq 0$. We get

$$0 = a^{-1} \cdot 0 = a^{-1} \cdot (a \cdot b) = (a^{-1} \cdot a) \cdot b = 1 \cdot b = b.$$

□

- Hence, a field does not have a *non-trivial zero divisor*, Dt.: nullteilerfrei.

Theorem 60

Let $(F,+,\cdot)$ be a field. Then, for all $a,b \in F$,

$$a \cdot b = 0 \quad \Rightarrow \quad \big(a = 0 \text{ or } b = 0\big).$$

Proof: Let $a,b \in F$ be arbitrary but fixed with $a \cdot b = 0$ and $a \neq 0$. We get

$$0 = a^{-1} \cdot 0 = a^{-1} \cdot (a \cdot b) = (a^{-1} \cdot a) \cdot b = 1 \cdot b = b.$$

□

- Hence, a field does not have a *non-trivial zero divisor*, Dt.: nullteilerfrei.

Theorem 61

Let $(F,+,\cdot)$ be a field. Then, for all $a,b \in F \backslash \{0\}$,

$$(a^{-1})^{-1} = a \qquad \text{and} \qquad (a \cdot b)^{-1} = a^{-1} \cdot b^{-1}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Definition 62 (Fraction, Dt.: Bruch)

For $a \in F, b \in F \setminus \{0\}$, the *fraction* $\frac{a}{b}$ is defined as

$$\frac{a}{b} := a \div b.$$

We call a the *enumerator* (Dt.: Zähler) and b the *denominator* (Dt.: Nenner).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Field: Properties of the Operations

Definition 62 (Fraction, Dt.: Bruch)

For $a \in F, b \in F \backslash \{0\}$, the *fraction* $\frac{a}{b}$ is defined as

$$\frac{a}{b} := a \div b.$$

We call a the *enumerator* (Dt.: Zähler) and b the *denominator* (Dt.: Nenner).

Theorem 63

Let $(F, +, \cdot)$ be a field. Then, for all $a, x \in F$ and all $b, y \in F \backslash \{0\}$,

$$\frac{a}{b} = \frac{x}{y} \quad \Leftrightarrow \quad a \cdot y = b \cdot x.$$

Field: Properties of the Operations

Definition 62 (Fraction, Dt.: Bruch)

For $a \in F, b \in F \backslash \{0\}$, the *fraction* $\frac{a}{b}$ is defined as

$$\frac{a}{b} := a \div b.$$

We call a the *enumerator* (Dt.: Zähler) and b the *denominator* (Dt.: Nenner).

Theorem 63

Let $(F, +, \cdot)$ be a field. Then, for all $a, x \in F$ and all $b, y \in F \backslash \{0\}$,

$$\frac{a}{b} = \frac{x}{y} \quad \Leftrightarrow \quad a \cdot y = b \cdot x.$$

Theorem 64

Let $(F, +, \cdot)$ be a field. Then, for all $a, b, x, y \in F$ for which no denominator equals 0,

$$\frac{a}{b} \pm \frac{x}{y} = \frac{a \cdot y \pm b \cdot x}{b \cdot y} \qquad \text{and} \qquad \frac{a}{b} \cdot \frac{x}{y} = \frac{a \cdot x}{b \cdot y} \qquad \text{and} \qquad \frac{a}{b} \div \frac{x}{y} = \frac{a \cdot y}{b \cdot x}.$$

Homomorphism and Isomorphism

- A *homomorphism* is a function that maps one algebraic structure to another algebraic structure of the same type such that it is compatible with all operations.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Homomorphism and Isomorphism

- A *homomorphism* is a function that maps one algebraic structure to another algebraic structure of the same type such that it is compatible with all operations.
- So, for two structures A, B and two binary operations $\star_A$ (on A) and $\star_B$ (on B), if $f: A \to B$ is a homomorphism then, for all $x, y \in A$,

$$f(x \star_A y) = f(x) \star_B f(y).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Homomorphism and Isomorphism

- A *homomorphism* is a function that maps one algebraic structure to another algebraic structure of the same type such that it is compatible with all operations.
- So, for two structures A, B and two binary operations $\star_A$ (on A) and $\star_B$ (on B), if $f : A \to B$ is a homomorphism then, for all $x, y \in A$,

$$f(x \star_A y) = f(x) \star_B f(y).$$

- A group homomorphism from A to B maps the neutral element of A to the neutral element of B, and maps the inverse of an element of A to the inverse of the image of this element.

Homomorphism and Isomorphism

- A *homomorphism* is a function that maps one algebraic structure to another algebraic structure of the same type such that it is compatible with all operations.
- So, for two structures A, B and two binary operations $\star_A$ (on A) and $\star_B$ (on B), if $f\colon A \to B$ is a homomorphism then, for all $x, y \in A$,

$$f(x \star_A y) = f(x) \star_B f(y).$$

- A group homomorphism from A to B maps the neutral element of A to the neutral element of B, and maps the inverse of an element of A to the inverse of the image of this element.
- E.g., $(\mathbb{R}, +)$ and $(\mathbb{R}^+, \cdot)$ form groups. A group homomorphism from $(\mathbb{R}, +)$ to $(\mathbb{R}^+, \cdot)$ is given by the exponential function $x \mapsto e^x$. (Recall that $e^{x+y} = e^x \cdot e^y$.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Homomorphism and Isomorphism

- A *homomorphism* is a function that maps one algebraic structure to another algebraic structure of the same type such that it is compatible with all operations.
- So, for two structures A, B and two binary operations $\star_A$ (on A) and $\star_B$ (on B), if $f: A \to B$ is a homomorphism then, for all $x, y \in A$,

$$f(x \star_A y) = f(x) \star_B f(y).$$

- A group homomorphism from A to B maps the neutral element of A to the neutral element of B, and maps the inverse of an element of A to the inverse of the image of this element.
- E.g., $(\mathbb{R}, +)$ and $(\mathbb{R}^+, \cdot)$ form groups. A group homomorphism from $(\mathbb{R}, +)$ to $(\mathbb{R}^+, \cdot)$ is given by the exponential function $x \mapsto e^x$. (Recall that $e^{x+y} = e^x \cdot e^y$.)
- A ring homomorphism from A to B is compatible with ring addition and multiplication, and maps the multiplicative neutral element of A to the multiplicative neutral element of B.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Homomorphism and Isomorphism

- A *homomorphism* is a function that maps one algebraic structure to another algebraic structure of the same type such that it is compatible with all operations.
- So, for two structures A, B and two binary operations $\star_A$ (on A) and $\star_B$ (on B), if $f: A \to B$ is a homomorphism then, for all $x, y \in A$,

$$f(x \star_A y) = f(x) \star_B f(y).$$

- A group homomorphism from A to B maps the neutral element of A to the neutral element of B, and maps the inverse of an element of A to the inverse of the image of this element.
- E.g., $(\mathbb{R}, +)$ and $(\mathbb{R}^+, \cdot)$ form groups. A group homomorphism from $(\mathbb{R}, +)$ to $(\mathbb{R}^+, \cdot)$ is given by the exponential function $x \mapsto e^x$. (Recall that $e^{x+y} = e^x \cdot e^y$.)
- A ring homomorphism from A to B is compatible with ring addition and multiplication, and maps the multiplicative neutral element of A to the multiplicative neutral element of B.
- An *isomorphism* is a bijective homomorphism.

4 Numbers and Basics of Number Theory

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

How Shall We Define Natural Numbers or Real Numbers?

- Three options:
  1. Ignore all formal details and presuppose an "intuitive" understanding of reals, integers, ...
  2. Introduce the natural numbers, $\mathbb{N}$, and then construct a hierarchy of number systems: $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R}$.
  3. Set up the reals, $\mathbb{R}$, axiomatically and then define proper subsets for $\mathbb{N}, \mathbb{Z}, \mathbb{Q}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

How Shall We Define Natural Numbers or Real Numbers?

- Three options:
  1. Ignore all formal details and presuppose an "intuitive" understanding of reals, integers, . . .
  2. Introduce the natural numbers, $\mathbb{N}$, and then construct a hierarchy of number systems: $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R}$.
  3. Set up the reals, $\mathbb{R}$, axiomatically and then define proper subsets for $\mathbb{N}, \mathbb{Z}, \mathbb{Q}$.
- What is the best approach for a course on (applied) discrete mathematics? Much scholarly debate — no consensus!
- We will start with introducing the natural numbers. However, since the gory details result in a lengthy discussion which provides little additional insight in $\mathbb{N}$ — and this is no course on number theory — we base our introduction of $\mathbb{N}$ on a simplified treatment of the so-called Peano axioms; see a book on number theory for a more formalized introduction of $\mathbb{N}$.

Natural Numbers: $\mathbb{N}$

- Intuitively, the natural numbers $\mathbb{N}$ are given by $\{1, 2, 3, 4, 5, \ldots\}$ or by $\{0, 1, 2, 3, 4, 5, \ldots\}$.
- Unfortunately, there is no general agreement on whether or not to include 0 ...

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural Numbers: $\mathbb{N}$

- Intuitively, the natural numbers $\mathbb{N}$ are given by $\{1, 2, 3, 4, 5, \ldots\}$ or by $\{0, 1, 2, 3, 4, 5, \ldots\}$.
- Unfortunately, there is no general agreement on whether or not to include 0 ...
- Paulo Ribenboim (1996): "Let P be a set of natural numbers; whenever convenient, it may be assumed that 0 is an element of P."

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural Numbers: $\mathbb{N}$

- Intuitively, the natural numbers $\mathbb{N}$ are given by $\{1, 2, 3, 4, 5, \ldots\}$ or by $\{0, 1, 2, 3, 4, 5, \ldots\}$.
- Unfortunately, there is no general agreement on whether or not to include 0 . . .
- Paulo Ribenboim (1996): "Let P be a set of natural numbers; whenever convenient, it may be assumed that 0 is an element of P."

Convention

In this course we adopt the following convention:

$$\mathbb{N} := \{1, 2, 3, 4, 5, \ldots\} \text{ and } \mathbb{N}_0 := \{0, 1, 2, 3, 4, 5, \ldots\}.$$

Natural Numbers: $\mathbb{N}$

- Intuitively, the natural numbers $\mathbb{N}$ are given by $\{1, 2, 3, 4, 5, \ldots\}$ or by $\{0, 1, 2, 3, 4, 5, \ldots\}$.
- Unfortunately, there is no general agreement on whether or not to include 0 . . .
- Paulo Ribenboim (1996): "Let P be a set of natural numbers; whenever convenient, it may be assumed that 0 is an element of P."

Convention

In this course we adopt the following convention:

$$\mathbb{N} := \{1, 2, 3, 4, 5, \ldots\} \text{ and } \mathbb{N}_0 := \{0, 1, 2, 3, 4, 5, \ldots\}.$$

- Caution: Read a text carefully to learn what an author means by ”natural number”. In particular, watch for clues such as terms like ”positive natural numbers” (which indicates that zero is included) or statements like ”n is a natural number, so it must be greater than zero” (which indicates that zero is not included).
- If one treats 0 as an element of $\mathbb{N}$ then $\{1, 2, 3, 4, 5, \ldots\}$ is often denoted by $\mathbb{N}^*$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

1. Reflexivity: $a \preceq a$.

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

1. Reflexivity: $a \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.

Partial Order

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

1. Reflexivity: $a \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \wedge \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

Partial Order

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

1. Reflexivity: $a \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \wedge \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

If $\preceq$ is a partial order on S then $(S, \preceq)$ is called a *partially ordered set*, aka a *poset*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Partial Order

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

1. Reflexivity: $a \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \wedge \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

If $\preceq$ is a partial order on S then $(S, \preceq)$ is called a *partially ordered set*, aka a *poset*.

Definition 66 (Strict partial order, Dt.: strikte Halbordnung)

A binary relation $\prec$ on a set S forms a *strict partial order* on S if the following two properties hold for all $a, b, c \in S$:

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

1. Reflexivity: $a \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \land \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \land \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

If $\preceq$ is a partial order on S then $(S, \preceq)$ is called a *partially ordered set*, aka a *poset*.

Definition 66 (Strict partial order, Dt.: strikte Halbordnung)

A binary relation $\prec$ on a set S forms a *strict partial order* on S if the following two properties hold for all $a, b, c \in S$:

1. Irreflexivity: $\neg(a \prec a)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Partial Order

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

1. Reflexivity: $a \preceq a$.
2. Anti-symmetry: $(a \preceq b \;\wedge\; b \preceq a) \;\Rightarrow\; a = b$.
3. Transitivity: $(a \preceq b \;\wedge\; b \preceq c) \;\Rightarrow\; a \preceq c$.

If $\preceq$ is a partial order on S then $(S, \preceq)$ is called a *partially ordered set*, aka a *poset*.

Definition 66 (Strict partial order, Dt.: strikte Halbordnung)

A binary relation $\prec$ on a set S forms a *strict partial order* on S if the following two properties hold for all $a, b, c \in S$:

1. Irreflexivity: $\neg(a \prec a)$.
2. Transitivity: $(a \prec b \;\wedge\; b \prec c) \;\Rightarrow\; a \prec c$.

Partial Order

Definition 65 (Partial order, Dt.: Halbordnung)

A *partial order* on a set S is a binary relation $\preceq$, i.e., a subset of $S \times S$, such that the following three properties hold for all $a, b, c \in S$:

1. Reflexivity: $a \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \wedge \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

If $\preceq$ is a partial order on S then $(S, \preceq)$ is called a *partially ordered set*, aka a *poset*.

Definition 66 (Strict partial order, Dt.: strikte Halbordnung)

A binary relation $\prec$ on a set S forms a *strict partial order* on S if the following two properties hold for all $a, b, c \in S$:

1. Irreflexivity: $\neg(a \prec a)$.
2. Transitivity: $(a \prec b \quad \wedge \quad b \prec c) \quad \Rightarrow \quad a \prec c$.

- A strict partial order is always *asymmetric*: If $a \prec b$ then $\neg(b \prec a)$.

  $(a \prec b \quad \wedge \quad b \prec a) \quad \stackrel{trans.}{\Rightarrow} \quad a \prec a,$ in contradiction to the irreflexivity: $\neg(a \prec a)$.

Theorem 67

There is a one-to-one correspondence between non-strict and strict partial orders. Let S be a set and $a, b \in S$.

1. If $\preceq$ is a non-strict partial order on S then the corresponding strict partial order "$\prec$" on S is the *reflexive reduction* given by
$$a \prec b \quad :\Leftrightarrow \quad a \preceq b \text{ and } a \neq b.$$
2. If, on the other hand, $\prec$ is a strict partial order on S then the corresponding non-strict partial order "$\preceq$" on S is the *reflexive closure* given by
$$a \preceq b \quad :\Leftrightarrow \quad a \prec b \text{ or } a = b.$$

Partial Order

Theorem 67

There is a one-to-one correspondence between non-strict and strict partial orders. Let S be a set and $a, b \in S$.

1. If $\preceq$ is a non-strict partial order on S then the corresponding strict partial order "$\prec$" on S is the *reflexive reduction* given by
$$a \prec b \quad :\Leftrightarrow \quad a \preceq b \text{ and } a \neq b.$$
2. If, on the other hand, $\prec$ is a strict partial order on S then the corresponding non-strict partial order "$\preceq$" on S is the *reflexive closure* given by
$$a \preceq b \quad :\Leftrightarrow \quad a \prec b \text{ or } a = b.$$

- As a notational convention, we omit the indication of an equality sign if we refer to a strict order, e.g., we write $\prec$ rather than $\preceq$ or $\subset$ rather than $\subseteq$.

Partial Order

- E.g., $(\mathbb{Z}, \trianglerighteq)$ with (the non-strict order) $\trianglerighteq$ as defined below forms a poset:

  if a and b are even: $a \trianglerighteq b \;:\Leftrightarrow\; a \geqslant b$

  if a and b are odd: $a \trianglerighteq b \;:\Leftrightarrow\; a \leqslant b$

Partial Order

- E.g., $(\mathbb{Z}, \trianglerighteq)$ with (the non-strict order) $\trianglerighteq$ as defined below forms a poset:

  if a and b are even: $\quad a \trianglerighteq b \quad :\Leftrightarrow \quad a \geqslant b$

  if a and b are odd: $\quad a \trianglerighteq b \quad :\Leftrightarrow \quad a \leqslant b$

  Note that we do not know $a \trianglerighteq b$ if one of a, b is even and the other one is odd. That is, if $(S, \preceq)$ is a poset then not all pairs of elements of S need to be comparable!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Partial Order

- E.g., $(\mathbb{Z}, \trianglerighteq)$ with (the non-strict order) $\trianglerighteq$ as defined below forms a poset:

  if a and b are even: $\quad a \trianglerighteq b \quad :\Leftrightarrow \quad a \geqslant b$

  if a and b are odd: $\quad a \trianglerighteq b \quad :\Leftrightarrow \quad a \leqslant b$

  Note that we do not know $a \trianglerighteq b$ if one of a, b is even and the other one is odd. That is, if $(S, \preceq)$ is a poset then not all pairs of elements of S need to be comparable!
- The subset relation, $\subset$, on the powerset $\mathcal{P}(X)$ of a set X is a strict partial order.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Partial Order

- E.g., $(\mathbb{Z}, \trianglerighteq)$ with (the non-strict order) $\trianglerighteq$ as defined below forms a poset:

  $$\text{if } a \text{ and } b \text{ are even:} \quad a \trianglerighteq b \quad :\Leftrightarrow \quad a \geqslant b$$

  $$\text{if } a \text{ and } b \text{ are odd:} \quad a \trianglerighteq b \quad :\Leftrightarrow \quad a \leqslant b$$

  Note that we do not know $a \trianglerighteq b$ if one of a, b is even and the other one is odd. That is, if $(S, \preceq)$ is a poset then not all pairs of elements of S need to be comparable!
- The subset relation, $\subset$, on the powerset $\mathcal{P}(X)$ of a set X is a strict partial order.

Definition 68 (Dual order, Dt.: duale Ordnung)

Let $(S, \preceq)$ resp. $(S, \prec)$ be a (strict) poset. The *dual order* (or *reverse order*) on S, $\succeq$ resp. $\succ$, is defined as follows for $a, b \in S$:

$$a \succeq b \quad :\Leftrightarrow \quad b \preceq a \qquad\qquad a \succ b \quad :\Leftrightarrow \quad b \prec a.$$

Definition 69 (Minimal element, Dt.: minimales Element)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *minimal element* of T if no $b \in T \backslash \{a\}$ exists such that $b \preceq a$.

Definition 69 (Minimal element, Dt.: minimales Element)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *minimal element* of T if no $b \in T \backslash \{a\}$ exists such that $b \preceq a$.

Definition 70 (Least element, Dt.: kleinstes Element, Minimum)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *least element* (or *minimum*) of T if $\quad \forall b \in T \backslash \{a\} \;\; a \preceq b$.

Definition 69 (Minimal element, Dt.: minimales Element)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *minimal element* of T if no $b \in T \backslash \{a\}$ exists such that $b \preceq a$.

Definition 70 (Least element, Dt.: kleinstes Element, Minimum)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *least element* (or *minimum*) of T if $\quad \forall b \in T \backslash \{a\} \;\; a \preceq b$.

Definition 71 (Maximal element, Dt.: maximales Element)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *maximal element* of T if no $b \in T \backslash \{a\}$ exists such that $a \preceq b$.

Extreme Elements

Definition 69 (Minimal element, Dt.: minimales Element)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *minimal element* of T if no $b \in T \setminus \{a\}$ exists such that $b \preceq a$.

Definition 70 (Least element, Dt.: kleinstes Element, Minimum)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *least element* (or *minimum*) of T if $\quad \forall b \in T \setminus \{a\} \;\; a \preceq b$.

Definition 71 (Maximal element, Dt.: maximales Element)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *maximal element* of T if no $b \in T \setminus \{a\}$ exists such that $a \preceq b$.

Definition 72 (Greatest element, Dt.: größtes Element, Maximum)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *greatest element* (or *maximum*) of T if $\quad \forall b \in T \setminus \{a\} \;\; b \preceq a$.

Extreme Elements

Definition 69 (Minimal element, Dt.: minimales Element)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *minimal element* of T if no $b \in T \backslash \{a\}$ exists such that $b \preceq a$.

Definition 70 (Least element, Dt.: kleinstes Element, Minimum)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *least element* (or *minimum*) of T if $\quad \forall b \in T \backslash \{a\} \;\; a \preceq b$.

Definition 71 (Maximal element, Dt.: maximales Element)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *maximal element* of T if no $b \in T \backslash \{a\}$ exists such that $a \preceq b$.

Definition 72 (Greatest element, Dt.: größtes Element, Maximum)

Let $(S, \preceq)$ be a poset and $T \subseteq S$. An element $a \in T$ is a *greatest element* (or *maximum*) of T if $\quad \forall b \in T \backslash \{a\} \;\; b \preceq a$.

- Note: If a minimum or maximum exists then the anti-symmetry ensures that it is unique. Minimal or maximal elements need not be unique, though.

Definition 73 (Total order, Dt.: totale Ordnung)

A binary relation $\preceq$ on a set S forms a *total order* (or *linear order*) on S if the following three statements hold for all $a, b, c \in S$:

Total Order

Definition 73 (Total order, Dt.: totale Ordnung)

A binary relation $\preceq$ on a set S forms a *total order* (or *linear order*) on S if the following three statements hold for all $a, b, c \in S$:

1. Totality: $a \preceq b \quad \vee \quad b \preceq a$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 73 (Total order, Dt.: totale Ordnung)

A binary relation $\preceq$ on a set S forms a *total order* (or *linear order*) on S if the following three statements hold for all $a, b, c \in S$:

1. Totality: $a \preceq b \quad \vee \quad b \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.

Total Order

Definition 73 (Total order, Dt.: totale Ordnung)

A binary relation $\preceq$ on a set S forms a *total order* (or *linear order*) on S if the following three statements hold for all $a, b, c \in S$:

1. Totality: $a \preceq b \quad \vee \quad b \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \wedge \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 73 (Total order, Dt.: totale Ordnung)

A binary relation $\preceq$ on a set S forms a *total order* (or *linear order*) on S if the following three statements hold for all $a, b, c \in S$:

1. Totality: $a \preceq b \quad \vee \quad b \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \wedge \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

If $\preceq$ is a total order on S then $(S, \preceq)$ is called a *totally ordered set*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Total Order

Definition 73 (Total order, Dt.: totale Ordnung)

A binary relation $\preceq$ on a set S forms a *total order* (or *linear order*) on S if the following three statements hold for all $a, b, c \in S$:

1. Totality: $a \preceq b \quad \vee \quad b \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \wedge \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

If $\preceq$ is a total order on S then $(S, \preceq)$ is called a *totally ordered set*.

- Note that (1) in Def. 73 implies reflexivity: $a \preceq a$ for all $a \in S$.
- That is, a total order on S is a (non-strict) partial order such that every pair of elements of S is comparable.

Total Order

Definition 73 (Total order, Dt.: totale Ordnung)

A binary relation $\preceq$ on a set S forms a *total order* (or *linear order*) on S if the following three statements hold for all $a, b, c \in S$:

1. Totality: $a \preceq b \quad \vee \quad b \preceq a$.
2. Anti-symmetry: $(a \preceq b \quad \wedge \quad b \preceq a) \quad \Rightarrow \quad a = b$.
3. Transitivity: $(a \preceq b \quad \wedge \quad b \preceq c) \quad \Rightarrow \quad a \preceq c$.

If $\preceq$ is a total order on S then $(S, \preceq)$ is called a *totally ordered set*.

- Note that (1) in Def. 73 implies reflexivity: $a \preceq a$ for all $a \in S$.
- That is, a total order on S is a (non-strict) partial order such that every pair of elements of S is comparable.

Definition 74 (Well-order, Dt.: Wohlordnung)

A total order $\preceq$ on a set S forms a *well-order* if every non-empty subset of S has a least element.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

N1 $1 \in \mathbb{N}$.

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \;\wedge\; n < n+1$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \;\wedge\; n < n+1$.
- **N3** $\forall n \in \mathbb{N}, n \neq 1 \;\; \exists m \in \mathbb{N} \quad n = m+1$.

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \ \wedge \ n < n+1$.
- **N3** $\forall n \in \mathbb{N}, n \neq 1 \ \exists m \in \mathbb{N} \quad n = m+1$.
- **N4** Every non-empty subset of $\mathbb{N}$ has a least element.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \;\wedge\; n < n+1$.
- **N3** $\forall n \in \mathbb{N}, n \neq 1 \;\; \exists m \in \mathbb{N} \quad n = m+1$.
- **N4** Every non-empty subset of $\mathbb{N}$ has a least element.

The number $n+1$ is called the *successor* of n, sometimes denoted by $\text{succ}(n)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \quad \wedge \quad n < n+1$.
- **N3** $\forall n \in \mathbb{N}, n \neq 1 \;\; \exists m \in \mathbb{N} \quad n = m+1$.
- **N4** Every non-empty subset of $\mathbb{N}$ has a least element.

The number $n+1$ is called the *successor* of n, sometimes denoted by $\text{succ}(n)$.

- **N1** together with **N2** establish the infinite sequence $1 < 2 < 3 < \ldots$

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \quad \wedge \quad n < n+1$.
- **N3** $\forall n \in \mathbb{N}, n \neq 1 \;\; \exists m \in \mathbb{N} \quad n = m+1$.
- **N4** Every non-empty subset of $\mathbb{N}$ has a least element.

The number $n+1$ is called the *successor* of n, sometimes denoted by $\mathrm{succ}(n)$.

- **N1** together with **N2** establish the infinite sequence $1 < 2 < 3 < \ldots$
- **N3** guarantees that every $n \in \mathbb{N}$ (except 1) is the successor of some number in $\mathbb{N}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \;\wedge\; n < n+1$.
- **N3** $\forall n \in \mathbb{N}, n \neq 1 \;\; \exists m \in \mathbb{N} \quad n = m+1$.
- **N4** Every non-empty subset of $\mathbb{N}$ has a least element.

The number $n+1$ is called the *successor* of n, sometimes denoted by $\mathrm{succ}(n)$.

- **N1** together with **N2** establish the infinite sequence $1 < 2 < 3 < \ldots$
- **N3** guarantees that every $n \in \mathbb{N}$ (except 1) is the successor of some number in $\mathbb{N}$.
- The so-called *well-ordering principle*, **N4**, weeds out numbers like $\frac{1}{2}$ or π.

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \quad \wedge \quad n < n+1$.
- **N3** $\forall n \in \mathbb{N}, n \neq 1 \ \exists m \in \mathbb{N} \quad n = m+1$.
- **N4** Every non-empty subset of $\mathbb{N}$ has a least element.

The number $n+1$ is called the *successor* of n, sometimes denoted by $\text{succ}(n)$.

- **N1** together with **N2** establish the infinite sequence $1 < 2 < 3 < \ldots$
- **N3** guarantees that every $n \in \mathbb{N}$ (except 1) is the successor of some number in $\mathbb{N}$.
- The so-called *well-ordering principle*, **N4**, weeds out numbers like $\frac{1}{2}$ or π.
- One can show that the standard algebraic rules are compatible with the conditions imposed on $\mathbb{N}$, and that algebra and order interact smoothly within $\mathbb{N}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Natural Numbers and Peano's Axioms

- The following definition of $\mathbb{N}$ is based on a simplified version of Peano's Axioms, as proposed by Giuseppe Peano (1858–1932) in 1889.

Definition 75 (Natural numbers, Dt.: natürliche Zahlen)

The set of all *natural numbers*, $\mathbb{N}$, together with an order relation $\leqslant$, is a totally ordered set defined as follows:

- **N1** $1 \in \mathbb{N}$.
- **N2** $\forall n \in \mathbb{N} \quad n+1 \in \mathbb{N} \quad \wedge \quad n < n+1$.
- **N3** $\forall n \in \mathbb{N}, n \neq 1 \;\; \exists m \in \mathbb{N} \quad n = m+1$.
- **N4** Every non-empty subset of $\mathbb{N}$ has a least element.

The number $n+1$ is called the *successor* of n, sometimes denoted by $\mathrm{succ}(n)$.

- **N1** together with **N2** establish the infinite sequence $1 < 2 < 3 < \ldots$
- **N3** guarantees that every $n \in \mathbb{N}$ (except 1) is the successor of some number in $\mathbb{N}$.
- The so-called *well-ordering principle*, **N4**, weeds out numbers like $\frac{1}{2}$ or π.
- One can show that the standard algebraic rules are compatible with the conditions imposed on $\mathbb{N}$, and that algebra and order interact smoothly within $\mathbb{N}$.
- One can also show that (up to a renaming of elements) there is only one set that fulfills all conditions of Def. 75. Hence, $\mathbb{N}$ is uniquely defined.

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

The Principle of Mathematical Induction

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

Theorem 77

If a set $K \subseteq \mathbb{N}$ is inductive then $K = \mathbb{N}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

Theorem 77

If a set $K \subseteq \mathbb{N}$ is inductive then $K = \mathbb{N}$.

Proof: Suppose that $K \neq \mathbb{N}$, i.e., $K \subset \mathbb{N}$.

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

Theorem 77

If a set $K \subseteq \mathbb{N}$ is inductive then $K = \mathbb{N}$.

Proof: Suppose that $K \neq \mathbb{N}$, i.e., $K \subset \mathbb{N}$. Hence, $K' := \mathbb{N} \backslash K$ is not empty.

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

Theorem 77

If a set $K \subseteq \mathbb{N}$ is inductive then $K = \mathbb{N}$.

Proof: Suppose that $K \neq \mathbb{N}$, i.e., $K \subset \mathbb{N}$. Hence, $K' := \mathbb{N} \backslash K$ is not empty. By the well-ordering principle, (N4), K' has a least element, n.

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

Theorem 77

If a set $K \subseteq \mathbb{N}$ is inductive then $K = \mathbb{N}$.

Proof: Suppose that $K \neq \mathbb{N}$, i.e., $K \subset \mathbb{N}$. Hence, $K' := \mathbb{N} \setminus K$ is not empty. By the well-ordering principle, (N4), K' has a least element, n. Since $n \neq 1$, (N3) guarantees that we can pick a number $k \in \mathbb{N}$ such that $k + 1 = n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

Theorem 77

If a set $K \subseteq \mathbb{N}$ is inductive then $K = \mathbb{N}$.

Proof: Suppose that $K \neq \mathbb{N}$, i.e., $K \subset \mathbb{N}$. Hence, $K' := \mathbb{N} \backslash K$ is not empty. By the well-ordering principle, (N4), K' has a least element, n. Since $n \neq 1$, (N3) guarantees that we can pick a number $k \in \mathbb{N}$ such that $k + 1 = n$. Thus, $k < n$.

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

Theorem 77

If a set $K \subseteq \mathbb{N}$ is inductive then $K = \mathbb{N}$.

Proof: Suppose that $K \neq \mathbb{N}$, i.e., $K \subset \mathbb{N}$. Hence, $K' := \mathbb{N} \backslash K$ is not empty. By the well-ordering principle, (N4), K' has a least element, n. Since $n \neq 1$, (N3) guarantees that we can pick a number $k \in \mathbb{N}$ such that $k + 1 = n$. Thus, $k < n$. As n is the least element of K', we have $k \in K$.

Definition 76 (Inductive)

A set $K \subseteq \mathbb{N}$ is *inductive* if

1. $1 \in K$,
2. $\forall k \in K \quad (k+1) \in K$.

Theorem 77

If a set $K \subseteq \mathbb{N}$ is inductive then $K = \mathbb{N}$.

Proof: Suppose that $K \neq \mathbb{N}$, i.e., $K \subset \mathbb{N}$. Hence, $K' := \mathbb{N} \backslash K$ is not empty. By the well-ordering principle, (N4), K' has a least element, n. Since $n \neq 1$, (N3) guarantees that we can pick a number $k \in \mathbb{N}$ such that $k + 1 = n$. Thus, $k < n$. As n is the least element of K', we have $k \in K$. Applying modus ponens to $k \in K$ and Condition 2 yields $k + 1 = n \in K$, i.e., a contradiction. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 78 (Weak Principle of Induction (W.P.I.))

Consider a predicate P over $\mathbb{N}$.
If

$$P(1)$$

and if

$$\forall k \in \mathbb{N} \;\; (P(k) \Rightarrow P(k+1))$$

then

$$\forall n \in \mathbb{N} \;\; P(n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 78 (Weak Principle of Induction (W.P.I.))

Consider a predicate P over $\mathbb{N}$.
If

$$P(1)$$

and if

$$\forall k \in \mathbb{N} \;\; (P(k) \Rightarrow P(k+1))$$

then

$$\forall n \in \mathbb{N} \;\; P(n).$$

Proof: Define $K := \{n \in \mathbb{N} : P(n)\}$. We have

1. $1 \in K$, and
2. $\forall k \in K \;\; (k+1) \in K$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 78 (Weak Principle of Induction (W.P.I.))

Consider a predicate P over $\mathbb{N}$.
If

$$P(1)$$

and if

$$\forall k \in \mathbb{N} \;\; (P(k) \Rightarrow P(k+1))$$

then

$$\forall n \in \mathbb{N} \;\; P(n).$$

Proof: Define $K := \{n \in \mathbb{N} : \; P(n)\}$. We have

1. $1 \in K$, and
2. $\forall k \in K \;\; (k+1) \in K$.

Thus, Thm. 77 is applicable and we conclude $K = \mathbb{N}$. That is, the predicate P holds for all natural numbers. □

Three Main Steps of a Proof by Induction

- Franciscus Maurolicus (1494–1575), an abbot of Messina, seems to have been first to use induction for proving a theorem. (He proved $\sum_{i=1}^{n}(2i-1) = n^2$.)
- Today's view of induction is based on the work of Giuseppe Peano (1858–1932).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Franciscus Maurolicus (1494–1575), an abbot of Messina, seems to have been first to use induction for proving a theorem. (He proved $\sum_{i=1}^{n}(2i-1) = n^2$.)
- Today's view of induction is based on the work of Giuseppe Peano (1858–1932).

Induction — we proceed as follows:

Suppose that we want to prove

$$\forall n \in \mathbb{N} \;\; P(n),$$

for some predicate P over $\mathbb{N}$ by using induction:

Three Main Steps of a Proof by Induction

- Franciscus Maurolicus (1494–1575), an abbot of Messina, seems to have been first to use induction for proving a theorem. (He proved $\sum_{i=1}^{n}(2i-1)=n^2$.)
- Today's view of induction is based on the work of Giuseppe Peano (1858–1932).

Induction — we proceed as follows:

Suppose that we want to prove

$$\forall n \in \mathbb{N} \;\; P(n),$$

for some predicate P over $\mathbb{N}$ by using induction:

1. *Induction basis* ("IB"): A basis step is done, i.e., $P(1)$ is proved to be true.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Three Main Steps of a Proof by Induction

- Franciscus Maurolicus (1494–1575), an abbot of Messina, seems to have been first to use induction for proving a theorem. (He proved $\sum_{i=1}^{n}(2i-1) = n^2$.)
- Today's view of induction is based on the work of Giuseppe Peano (1858–1932).

Induction — we proceed as follows:

Suppose that we want to prove

$$\forall n \in \mathbb{N} \;\; P(n),$$

for some predicate P over $\mathbb{N}$ by using induction:

1. *Induction basis* ("IB"): A basis step is done, i.e., $P(1)$ is proved to be true.
2. *Induction hypothesis* ("IH"): We assume $P(k)$ to be true for an arbitrary but fixed $k \in \mathbb{N}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Three Main Steps of a Proof by Induction

- Franciscus Maurolicus (1494–1575), an abbot of Messina, seems to have been first to use induction for proving a theorem. (He proved $\sum_{i=1}^{n}(2i-1) = n^2$.)
- Today's view of induction is based on the work of Giuseppe Peano (1858–1932).

Induction — we proceed as follows:

Suppose that we want to prove

$$\forall n \in \mathbb{N} \;\; P(n),$$

for some predicate P over $\mathbb{N}$ by using induction:

1. *Induction basis* ("IB"): A basis step is done, i.e., $P(1)$ is proved to be true.
2. *Induction hypothesis* ("IH"): We assume $P(k)$ to be true for an arbitrary but fixed $k \in \mathbb{N}$.
3. *Inductive step* ("IS"): We prove $P(k+1)$ based on the knowledge that $P(k)$ is true.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Gauß' Problem Revisited: Sample Inductive Proof

- We claim that $\sum_{i=1}^{n} i = \frac{n \cdot (n+1)}{2}$ holds for all $n \in \mathbb{N}$.

Gauß' Problem Revisited: Sample Inductive Proof

- We claim that $\sum_{i=1}^{n} i = \frac{n\cdot(n+1)}{2}$ holds for all $n \in \mathbb{N}$.

Proof : We use induction to prove our claim as follows:

- We define a suitable predicate P:

$$\forall n \in \mathbb{N} \quad \left(P(n) \; :\Leftrightarrow \; \sum_{i=1}^{n} i = \frac{n \cdot (n+1)}{2} \right).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Gauß' Problem Revisited: Sample Inductive Proof

- We claim that $\sum_{i=1}^{n} i = \frac{n\cdot(n+1)}{2}$ holds for all $n \in \mathbb{N}$.

Proof : We use induction to prove our claim as follows:

- We define a suitable predicate P:

$$\forall n \in \mathbb{N} \quad \left(P(n) \;:\Leftrightarrow\; \sum_{i=1}^{n} i = \frac{n\cdot(n+1)}{2} \right).$$

- *Induction basis (IB):* We establish the truth of $P(1)$:

$$\sum_{i=1}^{1} i = 1 = \frac{1\cdot 2}{2}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Gauß' Problem Revisited: Sample Inductive Proof

- We claim that $\sum_{i=1}^{n} i = \frac{n \cdot (n+1)}{2}$ holds for all $n \in \mathbb{N}$.

Proof: We use induction to prove our claim as follows:

- We define a suitable predicate P:

$$\forall n \in \mathbb{N} \quad \left(P(n) \quad :\Leftrightarrow \sum_{i=1}^{n} i = \frac{n \cdot (n+1)}{2} \right).$$

- *Induction basis (IB):* We establish the truth of $P(1)$:

$$\sum_{i=1}^{1} i = 1 = \frac{1 \cdot 2}{2}.$$

- *Induction hypothesis (IH)*: Assume $P(k)$ true for an arbitrary but fixed $k \in \mathbb{N}$. That is, we assume (for this $k \in \mathbb{N}$)

$$\sum_{i=1}^{k} i = \frac{k \cdot (k+1)}{2}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proof (cont'd) :

- *Inductive step (IS)*: We have to prove $P(k+1)$ based on the induction hypothesis. That is, we have to prove

$$\sum_{i=1}^{k+1} i = \frac{(k+1)\cdot(k+2)}{2}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Gauß' Problem Revisited: Sample Inductive Proof

Proof (cont'd) :

- *Inductive step (IS)*: We have to prove $P(k+1)$ based on the induction hypothesis. That is, we have to prove

$$\sum_{i=1}^{k+1} i = \frac{(k+1)\cdot(k+2)}{2}.$$

We get

$$\sum_{i=1}^{k+1} i = \left(\sum_{i=1}^{k} i\right) \;+\; (k+1)$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proof (cont'd) :

- *Inductive step (IS)*: We have to prove $P(k+1)$ based on the induction hypothesis. That is, we have to prove

$$\sum_{i=1}^{k+1} i = \frac{(k+1)\cdot(k+2)}{2}.$$

We get

$$\sum_{i=1}^{k+1} i = \left(\sum_{i=1}^{k} i\right) + (k+1)$$
$$\overset{I.H.}{=} \frac{k\cdot(k+1)}{2} + (k+1)$$

Gauß' Problem Revisited: Sample Inductive Proof

Proof (cont'd) :

- *Inductive step (IS)*: We have to prove $P(k+1)$ based on the induction hypothesis. That is, we have to prove

$$\sum_{i=1}^{k+1} i = \frac{(k+1)\cdot(k+2)}{2}.$$

We get

$$\begin{aligned}\sum_{i=1}^{k+1} i &= \left(\sum_{i=1}^{k} i\right) + (k+1) \\ &\overset{I.H.}{=} \frac{k\cdot(k+1)}{2} + (k+1) \\ &= \frac{k\cdot(k+1)+2(k+1)}{2} \\ &= \frac{(k+1)\cdot(k+2)}{2}.\end{aligned}$$

□

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 79 (Strong Principle of Induction (S.P.I.))

Consider a predicate P over $\mathbb{N}$.
If

$$P(1)$$

and if

$$\forall k \in \mathbb{N} \ \left[\left(P(1) \wedge P(2) \wedge \ldots \wedge P(k)\right) \ \Rightarrow \ P(k+1)\right]$$

then

$$\forall n \in \mathbb{N} \ \ P(n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 79 (Strong Principle of Induction (S.P.I.))

Consider a predicate P over $\mathbb{N}$.
If

$$P(1)$$

and if

$$\forall k \in \mathbb{N} \ \left[\left(P(1) \wedge P(2) \wedge \ldots \wedge P(k)\right) \Rightarrow P(k+1)\right]$$

then

$$\forall n \in \mathbb{N} \ \ P(n).$$

- Since

$$[P(k) \Rightarrow P(k+1)] \Rightarrow [(P(1) \wedge P(2) \wedge \ldots \wedge P(k)) \Rightarrow P(k+1)],$$

all theorems that can be proved by W.P.I. can also be proved by S.P.I.

Theorem 79 (Strong Principle of Induction (S.P.I.))

Consider a predicate P over $\mathbb{N}$.
If

$$P(1)$$

and if

$$\forall k \in \mathbb{N} \ \left[\left(P(1) \wedge P(2) \wedge \ldots \wedge P(k)\right) \Rightarrow P(k+1)\right]$$

then

$$\forall n \in \mathbb{N} \ \ P(n).$$

- Since

$$[P(k) \Rightarrow P(k+1)] \Rightarrow [(P(1) \wedge P(2) \wedge \ldots \wedge P(k)) \Rightarrow P(k+1)],$$

  all theorems that can be proved by W.P.I. can also be proved by S.P.I.
- But W.P.I. and S.P.I. are equivalent, at least from a theoretical point of view.

Theorem 80 (S.P.I. with Larger Base)

Consider a predicate P over $\mathbb{N}$, and let $m \in \mathbb{N}$.
If

$$P(m)$$

and if

$$\forall(k \in \mathbb{N}, k \geqslant m) \;\; \big[\big(P(m) \wedge P(m+1) \wedge \ldots \wedge P(k)\big) \;\Rightarrow\; P(k+1)\big]$$

then

$$\forall(n \in \mathbb{N}, n \geqslant m) \;\; P(n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 80 (S.P.I. with Larger Base)

Consider a predicate P over $\mathbb{N}$, and let $m \in \mathbb{N}$.
If

$$P(m)$$

and if

$$\forall(k \in \mathbb{N}, k \geqslant m) \ \left[\big(P(m) \wedge P(m+1) \wedge \ldots \wedge P(k)\big) \ \Rightarrow \ P(k+1)\right]$$

then

$$\forall(n \in \mathbb{N}, n \geqslant m) \ \ P(n).$$

Proof : We define a new predicate P' over $\mathbb{N}$ with

$$P'(n) \quad :\Longleftrightarrow \quad P(m-1+n) \qquad \text{for all } \ n \in \mathbb{N},$$

and apply the standard S.P.I. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Variations of the Induction Principle

Theorem 80 (S.P.I. with Larger Base)

Consider a predicate P over $\mathbb{N}$, and let $m \in \mathbb{N}$.
If

$$P(m)$$

and if

$$\forall (k \in \mathbb{N}, k \geqslant m) \ \big[\big(P(m) \wedge P(m+1) \wedge \ldots \wedge P(k)\big) \ \Rightarrow \ P(k+1)\big]$$

then

$$\forall (n \in \mathbb{N}, n \geqslant m) \ \ P(n).$$

Proof: We define a new predicate P' over $\mathbb{N}$ with

$$P'(n) \quad :\Longleftrightarrow \quad P(m-1+n) \qquad \text{for all } \ n \in \mathbb{N},$$

and apply the standard S.P.I. □

- We could also carry out induction for smaller base values. That is, induction works for claims over $\mathbb{N}_0$. (And even for negative base values!)

Mathematical Induction: Caveats

- We may not assume anything in the inductive step $n \to n+1$ besides that $P(n)$ holds and, of course, the standard properties of $\mathbb{N}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- We may not assume anything in the inductive step $n \to n+1$ besides that $P(n)$ holds and, of course, the standard properties of $\mathbb{N}$.
- The inductive step alone does not suffice! By carrying out only the inductive step one can "prove" that

$$\forall n \in \mathbb{N} \quad n = n + 5.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- We may not assume anything in the inductive step $n \to n+1$ besides that $P(n)$ holds and, of course, the standard properties of $\mathbb{N}$.
- The inductive step alone does not suffice! By carrying out only the inductive step one can "prove" that

$$\forall n \in \mathbb{N} \quad n = n + 5.$$

Let $k \in \mathbb{N}$ be arbitrary but fixed and assume as I.H. that $k = k + 5$:

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- We may not assume anything in the inductive step $n \to n+1$ besides that $P(n)$ holds and, of course, the standard properties of $\mathbb{N}$.
- The inductive step alone does not suffice! By carrying out only the inductive step one can “prove” that

$$\forall n \in \mathbb{N} \quad n = n + 5.$$

Let $k \in \mathbb{N}$ be arbitrary but fixed and assume as I.H. that $k = k + 5$:

$$k + 1 \overset{I.H.}{=} (k + 5) + 1 = (k + 1) + 5.$$

Mathematical Induction: Caveats

- We may not assume anything in the inductive step $n \to n+1$ besides that $P(n)$ holds and, of course, the standard properties of $\mathbb{N}$.
- The inductive step alone does not suffice! By carrying out only the inductive step one can "prove" that

$$\forall n \in \mathbb{N} \;\; n = n + 5.$$

Let $k \in \mathbb{N}$ be arbitrary but fixed and assume as I.H. that $k = k + 5$:

$$k + 1 \overset{I.H.}{=} (k + 5) + 1 = (k + 1) + 5.$$

- Thus, proving the base is mandatory!

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^\infty \left(\prod_{i=0}^{n} \frac{\sin\left(\frac{x}{2i+1}\right)}{\frac{x}{2i+1}} \right) dx.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^\infty \left(\prod_{i=0}^{n} \frac{\sin(\frac{x}{2i+1})}{\frac{x}{2i+1}} \right) dx.$$

- Calculus shows that

$$f(0) = f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{\pi}{2}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^\infty \left(\prod_{i=0}^{n} \frac{\sin(\frac{x}{2i+1})}{\frac{x}{2i+1}} \right) dx.$$

- Calculus shows that

$$f(0) = f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{\pi}{2}.$$

- So, what is $f(7)$? It ought to equal $\pi/2$, doesn't it?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^\infty \left(\prod_{i=0}^{n} \frac{\sin(\frac{x}{2i+1})}{\frac{x}{2i+1}} \right) dx.$$

- Calculus shows that

$$f(0) = f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{\pi}{2}.$$

- So, what is $f(7)$? It ought to equal $\pi/2$, doesn't it?
- Well ...

$$f(7) = \frac{467807924713440738696537864469\pi}{935615849440640907310521750000} \approx 0.99999999998529 \cdot \frac{\pi}{2}$$

Mathematical Induction: Caveats

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^{\infty} \left(\prod_{i=0}^{n} \frac{\sin(\frac{x}{2i+1})}{\frac{x}{2i+1}} \right) dx.$$

- Calculus shows that

$$f(0) = f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{\pi}{2}.$$

- So, what is $f(7)$? It ought to equal $\pi/2$, doesn't it?
- Well ...

$$f(7) = \frac{467807924713440738696537864469\pi}{935615849440640907310521750000} \approx 0.99999999998529 \cdot \frac{\pi}{2}$$

- For $n \in \mathbb{N}_0$, let $f(n) := n^2 - n + 41$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^\infty \left(\prod_{i=0}^{n} \frac{\sin(\frac{x}{2i+1})}{\frac{x}{2i+1}} \right) dx.$$

- Calculus shows that

$$f(0) = f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{\pi}{2}.$$

- So, what is $f(7)$? It ought to equal $\pi/2$, doesn't it?
- Well ...

$$f(7) = \frac{467807924713440738696537864469\pi}{935615849440640907310521750000} \approx 0.99999999998529 \cdot \frac{\pi}{2}$$

- For $n \in \mathbb{N}_0$, let $f(n) := n^2 - n + 41$.
- We learn that $f(n)$ is prime for all $0 \leqslant n \leqslant 40$.

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^\infty \left(\prod_{i=0}^{n} \frac{\sin\left(\frac{x}{2i+1}\right)}{\frac{x}{2i+1}} \right) dx.$$

- Calculus shows that

$$f(0) = f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{\pi}{2}.$$

- So, what is $f(7)$? It ought to equal $\pi/2$, doesn't it?
- Well ...

$$f(7) = \frac{467807924713440738696537864469\pi}{935615849440640907310521750000} \approx 0.99999999998529 \cdot \frac{\pi}{2}$$

- For $n \in \mathbb{N}_0$, let $f(n) := n^2 - n + 41$.
- We learn that $f(n)$ is prime for all $0 \leqslant n \leqslant 40$. So, is $f(n)$ always prime?

Mathematical Induction: Caveats

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^\infty \left(\prod_{i=0}^{n} \frac{\sin(\frac{x}{2i+1})}{\frac{x}{2i+1}} \right) dx.$$

- Calculus shows that

$$f(0) = f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{\pi}{2}.$$

- So, what is $f(7)$? It ought to equal $\pi/2$, doesn't it?
- Well ...

$$f(7) = \frac{467807924713440738696537864469\pi}{935615849440640907310521750000} \approx 0.99999999998529 \cdot \frac{\pi}{2}$$

- For $n \in \mathbb{N}_0$, let $f(n) := n^2 - n + 41$.
- We learn that $f(n)$ is prime for all $0 \leqslant n \leqslant 40$. So, is $f(n)$ always prime?
- No! For instance, $f(41)$ is not prime.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- Several base cases alone do not suffice! For $n \in \mathbb{N}_0$, let

$$f(n) := \int_0^\infty \left(\prod_{i=0}^{n} \frac{\sin(\frac{x}{2i+1})}{\frac{x}{2i+1}} \right) dx.$$

- Calculus shows that

$$f(0) = f(1) = f(2) = f(3) = f(4) = f(5) = f(6) = \frac{\pi}{2}.$$

- So, what is $f(7)$? It ought to equal $\pi/2$, doesn't it?
- Well ...

$$f(7) = \frac{467807924713440738696537864469\pi}{935615849440640907310521750000} \approx 0.99999999998529 \cdot \frac{\pi}{2}$$

- For $n \in \mathbb{N}_0$, let $f(n) := n^2 - n + 41$.
- We learn that $f(n)$ is prime for all $0 \leqslant n \leqslant 40$. So, is $f(n)$ always prime?
- No! For instance, $f(41)$ is not prime.
- Thus, proving the inductive step is truly mandatory!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- George Pólya (1887–1985): ”All cats have the same color”, or $\forall n \in \mathbb{N}$ (for all sets S of n cats (all cats of S have the same color)).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- George Pólya (1887–1985): ”All cats have the same color”, or $\forall n \in \mathbb{N}$ (for all sets S of n cats (all cats of S have the same color)).
- We use induction to prove this claim.
  - IB: Obviously true for $n := 1$. (No matter what we take as “color” of a cat . . .)

Mathematical Induction: Caveats

- George Pólya (1887–1985): "All cats have the same color", or $\forall n \in \mathbb{N}$ (for all sets S of n cats (all cats of S have the same color)).
- We use induction to prove this claim.
  - IB: Obviously true for $n := 1$. (No matter what we take as "color" of a cat ...)
  - IH: For all sets S of n cats, all cats of S have the same color, for $n \in \mathbb{N}$ arbitrary but fixed.

Mathematical Induction: Caveats

- George Pólya (1887–1985): ”All cats have the same color”, or $\forall n \in \mathbb{N}$ (for all sets S of n cats (all cats of S have the same color)).
- We use induction to prove this claim.
  - IB: Obviously true for $n := 1$. (No matter what we take as “color” of a cat . . .)
  - IH: For all sets S of n cats, all cats of S have the same color, for $n \in \mathbb{N}$ arbitrary but fixed.
  - IS: Consider a set S of $n+1$ cats, and let A and B be two subsets of S such that

$$|A| = |B| = n \quad \text{and} \quad A \cup B = S.$$

    Using the induction hypothesis and the transitivity of the equivalence, we conclude that all cats of the set S have the same color!

Mathematical Induction: Caveats

- George Pólya (1887–1985): ”All cats have the same color”, or $\forall n \in \mathbb{N}$ (for all sets S of n cats (all cats of S have the same color)).
- We use induction to prove this claim.
  - IB: Obviously true for $n := 1$. (No matter what we take as “color” of a cat . . .)
  - IH: For all sets S of n cats, all cats of S have the same color, for $n \in \mathbb{N}$ arbitrary but fixed.
  - IS: Consider a set S of $n+1$ cats, and let A and B be two subsets of S such that

    $$|A| = |B| = n \quad \text{and} \quad A \cup B = S.$$

    Using the induction hypothesis and the transitivity of the equivalence, we conclude that all cats of the set S have the same color!

As nature shows, this “proof” is seriously flawed . . .

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- We claim that $2 \cdot n = 0$ for all $n \in \mathbb{N}_0$.

Mathematical Induction: Caveats

- We claim that $2 \cdot n = 0$ for all $n \in \mathbb{N}_0$.
- We use induction to prove this claim:
  - IB: Obviously true for $n := 0$.

Mathematical Induction: Caveats

- We claim that $2 \cdot n = 0$ for all $n \in \mathbb{N}_0$.
- We use induction to prove this claim:
  - IB: Obviously true for $n := 0$.
  - IH: Suppose that the claim holds for all $k \in \mathbb{N}_0$ with $k \leqslant n$, for some arbitrary but fixed $n \in \mathbb{N}_0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- We claim that $2 \cdot n = 0$ for all $n \in \mathbb{N}_0$.
- We use induction to prove this claim:
  - IB: Obviously true for $n := 0$.
  - IH: Suppose that the claim holds for all $k \in \mathbb{N}_0$ with $k \leqslant n$, for some arbitrary but fixed $n \in \mathbb{N}_0$.
  - IS: We write $n + 1$ as $n + 1 = k_1 + k_2$, where $k_1, k_2 \in \mathbb{N}_0$ with $k_1, k_2 \leqslant n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- We claim that $2 \cdot n = 0$ for all $n \in \mathbb{N}_0$.
- We use induction to prove this claim:
  - IB: Obviously true for $n := 0$.
  - IH: Suppose that the claim holds for all $k \in \mathbb{N}_0$ with $k \leqslant n$, for some arbitrary but fixed $n \in \mathbb{N}_0$.
  - IS: We write $n + 1$ as $n + 1 = k_1 + k_2$, where $k_1, k_2 \in \mathbb{N}_0$ with $k_1, k_2 \leqslant n$. Then

$$2 \cdot (n + 1) = 2 \cdot (k_1 + k_2)$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- We claim that $2 \cdot n = 0$ for all $n \in \mathbb{N}_0$.
- We use induction to prove this claim:
  - IB: Obviously true for $n := 0$.
  - IH: Suppose that the claim holds for all $k \in \mathbb{N}_0$ with $k \leqslant n$, for some arbitrary but fixed $n \in \mathbb{N}_0$.
  - IS: We write $n + 1$ as $n + 1 = k_1 + k_2$, where $k_1, k_2 \in \mathbb{N}_0$ with $k_1, k_2 \leqslant n$. Then

$$2 \cdot (n + 1) = 2 \cdot (k_1 + k_2) = 2 \cdot k_1 + 2 \cdot k_2$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Mathematical Induction: Caveats

- We claim that $2 \cdot n = 0$ for all $n \in \mathbb{N}_0$.
- We use induction to prove this claim:
  - IB: Obviously true for $n := 0$.
  - IH: Suppose that the claim holds for all $k \in \mathbb{N}_0$ with $k \leqslant n$, for some arbitrary but fixed $n \in \mathbb{N}_0$.
  - IS: We write $n+1$ as $n+1 = k_1 + k_2$, where $k_1, k_2 \in \mathbb{N}_0$ with $k_1, k_2 \leqslant n$. Then

$$2 \cdot (n+1) = 2 \cdot (k_1 + k_2) = 2 \cdot k_1 + 2 \cdot k_2 \stackrel{I.H.}{=} 0 + 0 = 0,$$

thus finishing the inductive "proof" . . .

Real-World Application: Fair Resource Distribution

- Suppose that some limited and non-uniform resource has to be distributed fairly among n receivers, for some $n \in \mathbb{N}$ with $n > 1$.
- E.g., a cake (with fruits, whipped cream, chocolate crumbs, icing, etc.) might have to be distributed fairly among n kids. Aka: “Cake Cutting Problem”.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

- Suppose that some limited and non-uniform resource has to be distributed fairly among n receivers, for some $n \in \mathbb{N}$ with $n > 1$.
- E.g., a cake (with fruits, whipped cream, chocolate crumbs, icing, etc.) might have to be distributed fairly among n kids. Aka: "Cake Cutting Problem".
- To make the situation worse, each kid might value different portions of the cake differently. (Bob likes fruits, Alice hates them; Alice likes whipped cream, but Bob hates it.)
- The distribution should involve all kids such that each kid has to agree that it received a fair share of the cake by his/her preferences.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

- Suppose that some limited and non-uniform resource has to be distributed fairly among n receivers, for some $n \in \mathbb{N}$ with $n > 1$.
- E.g., a cake (with fruits, whipped cream, chocolate crumbs, icing, etc.) might have to be distributed fairly among n kids. Aka: “Cake Cutting Problem”.
- To make the situation worse, each kid might value different portions of the cake differently. (Bob likes fruits, Alice hates them; Alice likes whipped cream, but Bob hates it.)
- The distribution should involve all kids such that each kid has to agree that it received a fair share of the cake by his/her preferences.

Definition 81 (Fair distribution protocol)

A protocol for the distribution of a resource among n receivers is considered *fair* if each receiver gets at least $1/n$-th of the resource (by his/her preferences), no matter what the preferences of the other receivers are and what the other receivers get.

Real-World Application: Fair Resource Distribution

- Suppose that some limited and non-uniform resource has to be distributed fairly among *n* receivers, for some $n \in \mathbb{N}$ with $n > 1$.
- E.g., a cake (with fruits, whipped cream, chocolate crumbs, icing, etc.) might have to be distributed fairly among *n* kids. Aka: “Cake Cutting Problem”.
- To make the situation worse, each kid might value different portions of the cake differently. (Bob likes fruits, Alice hates them; Alice likes whipped cream, but Bob hates it.)
- The distribution should involve all kids such that each kid has to agree that it received a fair share of the cake by his/her preferences.

Definition 81 (Fair distribution protocol)

A protocol for the distribution of a resource among *n* receivers is considered *fair* if each receiver gets at least $1/n$-th of the resource (by his/her preferences), no matter what the preferences of the other receivers are and what the other receivers get.

- How can we come up with a fair distribution protocol? Is there a general algorithm for fair cake cutting in the presence of *n* kids??

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

If $n = 2$: Cut-and-choose distribution protocol.

1. Alice cuts the cake into two equal pieces (equal by her preferences).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

If $n = 2$: Cut-and-choose distribution protocol.

1. Alice cuts the cake into two equal pieces (equal by her preferences).
2. Bob chooses whichever piece seems larger (by his preferences).

Real-World Application: Fair Resource Distribution

If $n = 2$: Cut-and-choose distribution protocol.

1. Alice cuts the cake into two equal pieces (equal by her preferences).
2. Bob chooses whichever piece seems larger (by his preferences).
3. Alice takes the remaining piece.

Real-World Application: Fair Resource Distribution

If $n = 2$: Cut-and-choose distribution protocol.

1. Alice cuts the cake into two equal pieces (equal by her preferences).
2. Bob chooses whichever piece seems larger (by his preferences).
3. Alice takes the remaining piece.

If $n > 2$: Recursive application of the cut-and-choose distribution protocol.

1. The first $n-1$ kids cut the cake into $n-1$ pieces by applying the cut-and-choose distribution protocol recursively to $n-2$, $n-3$ etc. kids, thus each obtaining (hopefully) at least a fair $1/n-1$ portion of the cake.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

If $n = 2$: Cut-and-choose distribution protocol.

1. Alice cuts the cake into two equal pieces (equal by her preferences).
2. Bob chooses whichever piece seems larger (by his preferences).
3. Alice takes the remaining piece.

If $n > 2$: Recursive application of the cut-and-choose distribution protocol.

1. The first $n-1$ kids cut the cake into $n-1$ pieces by applying the cut-and-choose distribution protocol recursively to $n-2$, $n-3$ etc. kids, thus each obtaining (hopefully) at least a fair $1/n-1$ portion of the cake.
2. The n-th kid asks all other $n-1$ kids to cut his/her portion of the cake into n pieces such that the cutting is fair according to his/her preferences. (That is, according to each kid's preferences, each of the n pieces of his/her portion is equally desirable, for all of the first $n-1$ kids.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

If $n = 2$: Cut-and-choose distribution protocol.

1. Alice cuts the cake into two equal pieces (equal by her preferences).
2. Bob chooses whichever piece seems larger (by his preferences).
3. Alice takes the remaining piece.

If $n > 2$: Recursive application of the cut-and-choose distribution protocol.

1. The first $n-1$ kids cut the cake into $n-1$ pieces by applying the cut-and-choose distribution protocol recursively to $n-2$, $n-3$ etc. kids, thus each obtaining (hopefully) at least a fair $1/n-1$ portion of the cake.
2. The n-th kid asks all other $n-1$ kids to cut his/her portion of the cake into n pieces such that the cutting is fair according to his/her preferences. (That is, according to each kid's preferences, each of the n pieces of his/her portion is equally desirable, for all of the first $n-1$ kids.)
3. The n-th kid walks around and collects one piece — the most desirable piece according to his/her preferences! — from all the other $n-1$ kids.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

If $n = 2$: Cut-and-choose distribution protocol.

1. Alice cuts the cake into two equal pieces (equal by her preferences).
2. Bob chooses whichever piece seems larger (by his preferences).
3. Alice takes the remaining piece.

If $n > 2$: Recursive application of the cut-and-choose distribution protocol.

1. The first $n-1$ kids cut the cake into $n-1$ pieces by applying the cut-and-choose distribution protocol recursively to $n-2$, $n-3$ etc. kids, thus each obtaining (hopefully) at least a fair $1/_{n-1}$ portion of the cake.
2. The n-th kid asks all other $n-1$ kids to cut his/her portion of the cake into n pieces such that the cutting is fair according to his/her preferences. (That is, according to each kid's preferences, each of the n pieces of his/her portion is equally desirable, for all of the first $n-1$ kids.)
3. The n-th kid walks around and collects one piece — the most desirable piece according to his/her preferences! — from all the other $n-1$ kids.

Theorem 82

The recursive cut-and-choose distribution protocol is fair.

Real-World Application: Fair Resource Distribution

Proof of Thm. 82 by induction : Assume that the total cake is worth 1 for each kid.

I.B.: $n := 2$ Alice cut the cake into two pieces that are equally desirable (according to her preferences) and, thus, both worth $1/2$. Hence, she will get one half of the cake (by her preferences), no matter how Bob behaves.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proof of Thm. 82 by induction: Assume that the total cake is worth 1 for each kid.

I.B.: $n := 2$ Alice cut the cake into two pieces that are equally desirable (according to her preferences) and, thus, both worth $1/2$. Hence, she will get one half of the cake (by her preferences), no matter how Bob behaves.
Bob sees two pieces, one worth w_1 and the other one worth $1 - w_1$ (by his preferences). Trivially, either $w_1 \geqslant 1/2$ or $1 - w_1 \geqslant 1/2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

Proof of Thm. 82 by induction : Assume that the total cake is worth 1 for each kid.

I.B.: $n := 2$ Alice cut the cake into two pieces that are equally desirable (according to her preferences) and, thus, both worth $1/2$. Hence, she will get one half of the cake (by her preferences), no matter how Bob behaves.
Bob sees two pieces, one worth w_1 and the other one worth $1 - w_1$ (by his preferences). Trivially, either $w_1 \geqslant 1/2$ or $1 - w_1 \geqslant 1/2$.
Hence, Bob can choose at least one half of the cake (according to his preferences), and both kids have no reason to complain about an unfair cutting.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

Proof of Thm. 82 by induction: Assume that the total cake is worth 1 for each kid.

I.B.: $n := 2$ Alice cut the cake into two pieces that are equally desirable (according to her preferences) and, thus, both worth $1/2$. Hence, she will get one half of the cake (by her preferences), no matter how Bob behaves.
Bob sees two pieces, one worth w_1 and the other one worth $1 - w_1$ (by his preferences). Trivially, either $w_1 \geqslant 1/2$ or $1 - w_1 \geqslant 1/2$.
Hence, Bob can choose at least one half of the cake (according to his preferences), and both kids have no reason to complain about an unfair cutting.

I.H.: Assume that the recursive cut-and-choose cake cutting has been considered fair by the first $k - 1$ kids, for $k \geqslant 3$ arbitrary but fixed. Hence, each of the first $k - 1$ kids got a portion that is a least worth (according to the kid's preferences) $\frac{1}{k-1}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

Proof of Thm. 82 by induction : Assume that the total cake is worth 1 for each kid.

I.B.: $n := 2$ Alice cut the cake into two pieces that are equally desirable (according to her preferences) and, thus, both worth $1/2$. Hence, she will get one half of the cake (by her preferences), no matter how Bob behaves.
Bob sees two pieces, one worth w_1 and the other one worth $1 - w_1$ (by his preferences). Trivially, either $w_1 \geqslant 1/2$ or $1 - w_1 \geqslant 1/2$.
Hence, Bob can choose at least one half of the cake (according to his preferences), and both kids have no reason to complain about an unfair cutting.

I.H.: Assume that the recursive cut-and-choose cake cutting has been considered fair by the first $k - 1$ kids, for $k \geqslant 3$ arbitrary but fixed. Hence, each of the first $k - 1$ kids got a portion that is a least worth (according to the kid's preferences) $\frac{1}{k-1}$.

I.S.: After the cuts for the k-th kid were made, each kid has k pieces each worth $\frac{1}{(k-1)\cdot k}$.

Real-World Application: Fair Resource Distribution

Proof of Thm. 82 by induction : Assume that the total cake is worth 1 for each kid.

I.B.: $n := 2$ Alice cut the cake into two pieces that are equally desirable (according to her preferences) and, thus, both worth $1/2$. Hence, she will get one half of the cake (by her preferences), no matter how Bob behaves.
Bob sees two pieces, one worth w_1 and the other one worth $1 - w_1$ (by his preferences). Trivially, either $w_1 \geqslant 1/2$ or $1 - w_1 \geqslant 1/2$.
Hence, Bob can choose at least one half of the cake (according to his preferences), and both kids have no reason to complain about an unfair cutting.

I.H.: Assume that the recursive cut-and-choose cake cutting has been considered fair by the first $k - 1$ kids, for $k \geqslant 3$ arbitrary but fixed. Hence, each of the first $k - 1$ kids got a portion that is a least worth (according to the kid's preferences) $\frac{1}{k-1}$.

I.S.: After the cuts for the k-th kid were made, each kid has k pieces each worth $\frac{1}{(k-1)\cdot k}$. After the k-th kid took one piece from each of them, each of the first $k - 1$ kids is left with $k - 1$ pieces each worth $\frac{1}{(k-1)\cdot k}$, i.e., with a total worth of $\frac{1}{k}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fair Resource Distribution

Proof of Thm. 82 by induction : Assume that the total cake is worth 1 for each kid.

I.B.: $n := 2$ Alice cut the cake into two pieces that are equally desirable (according to her preferences) and, thus, both worth $1/2$. Hence, she will get one half of the cake (by her preferences), no matter how Bob behaves.
Bob sees two pieces, one worth w_1 and the other one worth $1 - w_1$ (by his preferences). Trivially, either $w_1 \geqslant 1/2$ or $1 - w_1 \geqslant 1/2$.
Hence, Bob can choose at least one half of the cake (according to his preferences), and both kids have no reason to complain about an unfair cutting.

I.H.: Assume that the recursive cut-and-choose cake cutting has been considered fair by the first $k-1$ kids, for $k \geqslant 3$ arbitrary but fixed. Hence, each of the first $k-1$ kids got a portion that is a least worth (according to the kid's preferences) $\frac{1}{k-1}$.

I.S.: After the cuts for the k-th kid were made, each kid has k pieces each worth $\frac{1}{(k-1)\cdot k}$. After the k-th kid took one piece from each of them, each of the first $k-1$ kids is left with $k-1$ pieces each worth $\frac{1}{(k-1)\cdot k}$, i.e., with a total worth of $\frac{1}{k}$.
Suppose that the k-th kid values the portion of the i-th kid with w_i, for $i \in \{1, 2, \ldots, k-1\}$. Of course, $w_1 + w_2 + \ldots + w_{k-1} = 1$.

Real-World Application: Fair Resource Distribution

Proof of Thm. 82 by induction: Assume that the total cake is worth 1 for each kid.

I.B.: $n := 2$ Alice cut the cake into two pieces that are equally desirable (according to her preferences) and, thus, both worth $1/2$. Hence, she will get one half of the cake (by her preferences), no matter how Bob behaves.
Bob sees two pieces, one worth w_1 and the other one worth $1 - w_1$ (by his preferences). Trivially, either $w_1 \geqslant 1/2$ or $1 - w_1 \geqslant 1/2$.
Hence, Bob can choose at least one half of the cake (according to his preferences), and both kids have no reason to complain about an unfair cutting.

I.H.: Assume that the recursive cut-and-choose cake cutting has been considered fair by the first $k-1$ kids, for $k \geqslant 3$ arbitrary but fixed. Hence, each of the first $k-1$ kids got a portion that is a least worth (according to the kid's preferences) $\frac{1}{k-1}$.

I.S.: After the cuts for the k-th kid were made, each kid has k pieces each worth $\frac{1}{(k-1)\cdot k}$. After the k-th kid took one piece from each of them, each of the first $k-1$ kids is left with $k-1$ pieces each worth $\frac{1}{(k-1)\cdot k}$, i.e., with a total worth of $\frac{1}{k}$.
Suppose that the k-th kid values the portion of the i-th kid with w_i, for $i \in \{1, 2, \ldots, k-1\}$. Of course, $w_1 + w_2 + \ldots + w_{k-1} = 1$. Since the k-th kid gets at least w_i/k from the i-th kid, the k-th kid gets in total at least

$$\frac{w_1}{k} + \frac{w_2}{k} + \ldots + \frac{w_{k-1}}{k} = \frac{1}{k}(w_1 + w_2 + \ldots + w_{k-1}) = \frac{1}{k}.$$

Cardinality

- Intuitively, the cardinality $|A|$ of a set A specifies the number of elements of A: If $A := \{1, 2, 3\}$ then $|A| = 3$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Cardinality

- Intuitively, the cardinality $|A|$ of a set A specifies the number of elements of A: If $A := \{1, 2, 3\}$ then $|A| = 3$.
- However, this notion of cardinality becomes tricky for "infinite" sets.
- E.g., $\mathbb{N} \backslash \{1\}$ should have one element less than $\mathbb{N}$, right?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Cardinality

- Intuitively, the cardinality $|A|$ of a set A specifies the number of elements of A: If $A := \{1, 2, 3\}$ then $|A| = 3$.
- However, this notion of cardinality becomes tricky for "infinite" sets.
- E.g., $\mathbb{N} \setminus \{1\}$ should have one element less than $\mathbb{N}$, right?

Definition 83 (Cardinality; Dt.: Mächtigkeit, Kardinalität)

The set A has n elements, aka *cardinality* n, for some $n \in \mathbb{N}$, if there exists a bijection from $\{1, 2, \ldots, n-1, n\}$ to A. The cardinality of A is denoted by $|A|$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Cardinality

- Intuitively, the cardinality $|A|$ of a set A specifies the number of elements of A: If $A := \{1, 2, 3\}$ then $|A| = 3$.
- However, this notion of cardinality becomes tricky for “infinite” sets.
- E.g., $\mathbb{N} \backslash \{1\}$ should have one element less than $\mathbb{N}$, right?

Definition 83 (Cardinality; Dt.: Mächtigkeit, Kardinalität)

The set A has n elements, aka *cardinality* n, for some $n \in \mathbb{N}$, if there exists a bijection from $\{1, 2, \ldots, n-1, n\}$ to A. The cardinality of A is denoted by $|A|$.
The sets A, B have the *same cardinality*, denoted by $|A| = |B|$, if there exists a bijection from A to B.

Cardinality

- Intuitively, the cardinality $|A|$ of a set A specifies the number of elements of A: If $A := \{1, 2, 3\}$ then $|A| = 3$.
- However, this notion of cardinality becomes tricky for “infinite” sets.
- E.g., $\mathbb{N} \backslash \{1\}$ should have one element less than $\mathbb{N}$, right?

Definition 83 (Cardinality; Dt.: Mächtigkeit, Kardinalität)

The set A has n elements, aka *cardinality* n, for some $n \in \mathbb{N}$, if there exists a bijection from $\{1, 2, \ldots, n-1, n\}$ to A. The cardinality of A is denoted by $|A|$.
The sets A, B have the *same cardinality*, denoted by $|A| = |B|$, if there exists a bijection from A to B.
The set A is of *strictly smaller cardinality* than B, denoted by $|A| < |B|$, if there exists an injective function but no bijective function from A to B.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Cardinality

- Intuitively, the cardinality $|A|$ of a set A specifies the number of elements of A: If $A := \{1, 2, 3\}$ then $|A| = 3$.
- However, this notion of cardinality becomes tricky for "infinite" sets.
- E.g., $\mathbb{N}\backslash\{1\}$ should have one element less than $\mathbb{N}$, right?

Definition 83 (Cardinality; Dt.: Mächtigkeit, Kardinalität)

The set A has n elements, aka *cardinality* n, for some $n \in \mathbb{N}$, if there exists a bijection from $\{1, 2, \ldots, n-1, n\}$ to A. The cardinality of A is denoted by $|A|$.
The sets A, B have the *same cardinality*, denoted by $|A| = |B|$, if there exists a bijection from A to B.
The set A is of *strictly smaller cardinality* than B, denoted by $|A| < |B|$, if there exists an injective function but no bijective function from A to B.

Definition 84 (Finite, countably infinite, uncountable, Dt: endlich, abzählbar unendlich, überabzählbar unendlich)

The set A is a *finite set* if $|A| < |\mathbb{N}|$.

Cardinality

- Intuitively, the cardinality $|A|$ of a set A specifies the number of elements of A: If $A := \{1, 2, 3\}$ then $|A| = 3$.
- However, this notion of cardinality becomes tricky for "infinite" sets.
- E.g., $\mathbb{N} \setminus \{1\}$ should have one element less than $\mathbb{N}$, right?

Definition 83 (Cardinality; Dt.: Mächtigkeit, Kardinalität)

The set A has n elements, aka *cardinality* n, for some $n \in \mathbb{N}$, if there exists a bijection from $\{1, 2, \ldots, n-1, n\}$ to A. The cardinality of A is denoted by $|A|$.
The sets A, B have the *same cardinality*, denoted by $|A| = |B|$, if there exists a bijection from A to B.
The set A is of *strictly smaller cardinality* than B, denoted by $|A| < |B|$, if there exists an injective function but no bijective function from A to B.

Definition 84 (Finite, countably infinite, uncountable, Dt: endlich, abzählbar unendlich, überabzählbar unendlich)

The set A is a *finite set* if $|A| < |\mathbb{N}|$.
The set A is a *countably infinite set* if $|A| = |\mathbb{N}|$.

Cardinality

- Intuitively, the cardinality $|A|$ of a set A specifies the number of elements of A: If $A := \{1, 2, 3\}$ then $|A| = 3$.
- However, this notion of cardinality becomes tricky for “infinite” sets.
- E.g., $\mathbb{N} \backslash \{1\}$ should have one element less than $\mathbb{N}$, right?

Definition 83 (Cardinality; Dt.: Mächtigkeit, Kardinalität)

The set A has n elements, aka *cardinality* n, for some $n \in \mathbb{N}$, if there exists a bijection from $\{1, 2, \ldots, n-1, n\}$ to A. The cardinality of A is denoted by $|A|$.
The sets A, B have the *same cardinality*, denoted by $|A| = |B|$, if there exists a bijection from A to B.
The set A is of *strictly smaller cardinality* than B, denoted by $|A| < |B|$, if there exists an injective function but no bijective function from A to B.

Definition 84 (Finite, countably infinite, uncountable, Dt: endlich, abzählbar unendlich, überabzählbar unendlich)

The set A is a *finite set* if $|A| < |\mathbb{N}|$.
The set A is a *countably infinite set* if $|A| = |\mathbb{N}|$.
The set A is an *uncountable set* if $|A| > |\mathbb{N}|$.

Theorem 85

A subset of a countably infinite set is a finite or a countably infinite set itself.

Cardinality

Theorem 85

A subset of a countably infinite set is a finite or a countably infinite set itself.

Theorem 86 (Cantor&Schröder&Bernstein)

Consider two sets A and B. If there exist injective functions $f: A \to B$ and $g: B \to A$ between the sets A and B, then there exists a bijective function between A and B.

Cardinality

Theorem 85

A subset of a countably infinite set is a finite or a countably infinite set itself.

Theorem 86 (Cantor&Schröder&Bernstein)

Consider two sets A and B. If there exist injective functions $f: A \to B$ and $g: B \to A$ between the sets A and B, then there exists a bijective function between A and B.

- Stated by Cantor in 1887 (without a proof) and with a proof (defacto relying on the Axiom of Choice) in 1895, proved by Dedekind (not relying on the Axiom of Choice) in 1887, incorrectly proved by Schröder in 1897, proved by Bernstein (not relying on the Axiom of Choice) in 1897.
- Theorem 86 makes it easier to prove that two sets are of the same cardinality even if it is difficult to construct a bijection explicitly.
- E.g., $|\mathbb{N}| = |\mathbb{N} \times \mathbb{N}|$.

Theorem 85

A subset of a countably infinite set is a finite or a countably infinite set itself.

Theorem 86 (Cantor&Schröder&Bernstein)

Consider two sets A and B. If there exist injective functions $f: A \to B$ and $g: B \to A$ between the sets A and B, then there exists a bijective function between A and B.

- Stated by Cantor in 1887 (without a proof) and with a proof (defacto relying on the Axiom of Choice) in 1895, proved by Dedekind (not relying on the Axiom of Choice) in 1887, incorrectly proved by Schröder in 1897, proved by Bernstein (not relying on the Axiom of Choice) in 1897.
- Theorem 86 makes it easier to prove that two sets are of the same cardinality even if it is difficult to construct a bijection explicitly.
- E.g., $|\mathbb{N}| = |\mathbb{N} \times \mathbb{N}|$.

Corollary 87

Consider three sets A, B and C. If $A \subseteq B \subseteq C$ and $|A| = |C|$ then $|A| = |B| = |C|$.

4 Numbers and Basics of Number Theory

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Integers: $\mathbb{Z}$

- Intuitive way to define the integers: $\mathbb{Z} := \mathbb{N}_0 \cup \{-n : n \in \mathbb{N}\}$.
- Thus, $\mathbb{Z} = \{0, \pm 1, \pm 2, \pm 3, \pm 4, \pm 5, \ldots\}$.
- The blackboard-bold letter $\mathbb{Z}$ stands for the German word “Zahlen”.

Integers: $\mathbb{Z}$

- Intuitive way to define the integers: $\mathbb{Z} := \mathbb{N}_0 \cup \{-n : n \in \mathbb{N}\}$.
- Thus, $\mathbb{Z} = \{0, \pm 1, \pm 2, \pm 3, \pm 4, \pm 5, \ldots\}$.
- The blackboard-bold letter $\mathbb{Z}$ stands for the German word “Zahlen”.

- But what are the properties of the elements $-n$??
- And how could we define $a + b$ and $a \cdot b$ for $a, b \in \mathbb{Z}$??

Integers: $\mathbb{Z}$

- Intuitive way to define the integers: $\mathbb{Z} := \mathbb{N}_0 \cup \{-n : n \in \mathbb{N}\}$.
- Thus, $\mathbb{Z} = \{0, \pm 1, \pm 2, \pm 3, \pm 4, \pm 5, \ldots\}$.
- The blackboard-bold letter $\mathbb{Z}$ stands for the German word "Zahlen".

- But what are the properties of the elements $-n$??
- And how could we define $a + b$ and $a \cdot b$ for $a, b \in \mathbb{Z}$??

- In order to put $\mathbb{Z}$ on a more solid basis, we "extend" $\mathbb{N}$ to obtain $\mathbb{Z}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Construction of $\mathbb{Z}$ Based on $\mathbb{N}$

- Let $\cong_Z$ be a relation over $\mathbb{N}_0$ such that

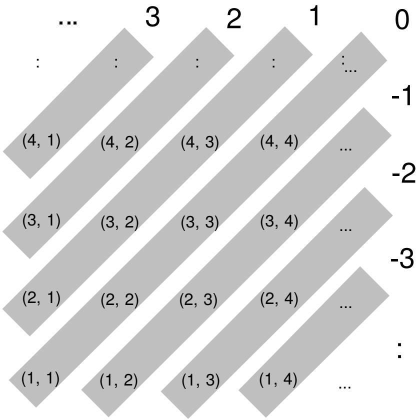
$$(a, b) \cong_Z (c, d) \quad :\Leftrightarrow \quad a + d = c + b.$$

Construction of $\mathbb{Z}$ Based on $\mathbb{N}$

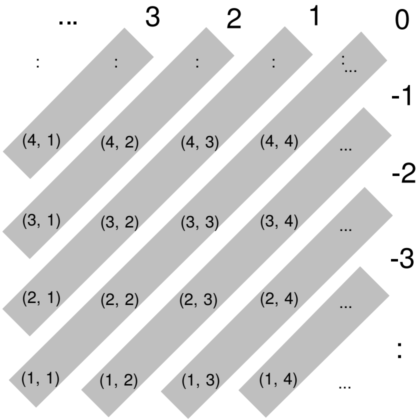
- Let $\cong_Z$ be a relation over $\mathbb{N}_0$ such that

$$(a, b) \cong_Z (c, d) \quad :\Leftrightarrow \quad a + d = c + b.$$

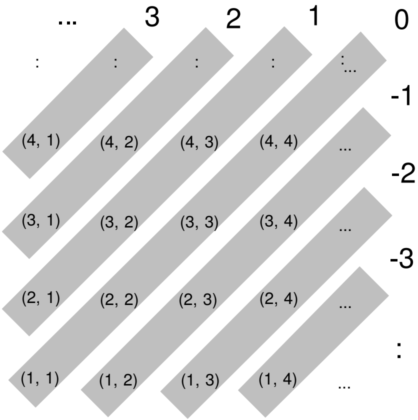
- Easy to show: $\cong_Z$ is an equivalence relation over $\mathbb{N}_0$, with the equivalence classes shown below.



- We interprete $[(a, b)]_{\cong_Z}$ as $a - b$.



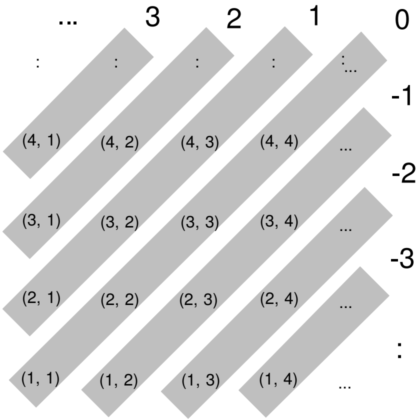
- We interprete $[(a, b)]_{\cong_Z}$ as $a - b$.
- For $n \in \mathbb{N}$, the equivalence classes $[(n, 0)]_{\cong_Z}$ form the natural numbers, while $[(0, n)]_{\cong_Z}$ form the negative integers.
- Zero is given by $[(0, 0)]_{\cong_Z}$.



Construction of $\mathbb{Z}$ Based on $\mathbb{N}$

Definition 88 (Integers)

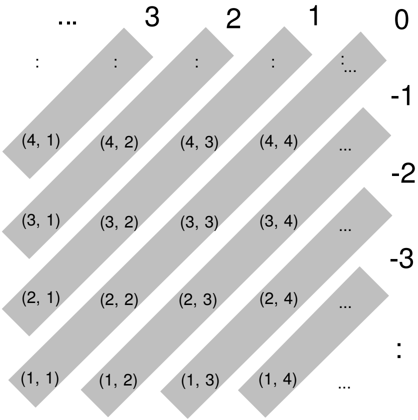
The *integers* $\mathbb{Z}$ are defined as $\mathbb{Z} := \{[(a,b)]_{\cong_Z} : \; a,b \in \mathbb{N}_0\}$.



Definition 88 (Integers)

The *integers* $\mathbb{Z}$ are defined as $\mathbb{Z} := \{[(a, b)]_{\cong_Z} : a, b \in \mathbb{N}_0\}$.

- Furthermore, $\mathbb{Z}^+ := \mathbb{N}$ and $\mathbb{Z}_0^+ := \mathbb{N}_0$.



- It remains to define addition, multiplication and order on $\mathbb{Z}$. For $a, b, c, d \in \mathbb{N}_0$ we define an addition $+_Z$, a multiplication $\cdot_Z$ and an order $\leqslant_Z$ as follows:

$$
\begin{aligned}
[(a,b)]_{\cong_Z} +_Z [(c,d)]_{\cong_Z} &:= [(a+c, b+d)]_{\cong_Z}\\
[(a,b)]_{\cong_Z} \cdot_Z [(c,d)]_{\cong_Z} &:= [(a\cdot c + b\cdot d, a\cdot d + b\cdot c)]_{\cong_Z}\\
[(a,b)]_{\cong_Z} \leqslant_Z [(c,d)]_{\cong_Z} &:\Leftrightarrow a+d \leqslant b+c
\end{aligned}
$$

Construction of $\mathbb{Z}$ Based on $\mathbb{N}$

- It remains to define addition, multiplication and order on $\mathbb{Z}$. For $a, b, c, d \in \mathbb{N}_0$ we define an addition $+_Z$, a multiplication $\cdot_Z$ and an order $\leqslant_Z$ as follows:

$$\begin{aligned}
[(a,b)]_{\cong_Z} +_Z [(c,d)]_{\cong_Z} &:= [(a+c, b+d)]_{\cong_Z} \\
[(a,b)]_{\cong_Z} \cdot_Z [(c,d)]_{\cong_Z} &:= [(a\cdot c + b\cdot d, a\cdot d + b\cdot c)]_{\cong_Z} \\
[(a,b)]_{\cong_Z} \leqslant_Z [(c,d)]_{\cong_Z} &:\Leftrightarrow a+d \leqslant b+c
\end{aligned}$$

- It is easy to show that
  - addition, multiplication and order are well-defined,
  - the standard rules of arithmetic hold, with $[(0,0)]_{\cong_Z}$ as zero element ("zero"),
  - $\leqslant_Z$ defines a total order on $\mathbb{N}_0 \times \mathbb{N}_0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Construction of $\mathbb{Z}$ Based on $\mathbb{N}$

- It remains to define addition, multiplication and order on $\mathbb{Z}$. For $a, b, c, d \in \mathbb{N}_0$ we define an addition $+_Z$, a multiplication $\cdot_Z$ and an order $\leqslant_Z$ as follows:

$$\begin{aligned} [(a,b)]_{\cong_Z} +_Z [(c,d)]_{\cong_Z} &:= [(a+c, b+d)]_{\cong_Z} \\ [(a,b)]_{\cong_Z} \cdot_Z [(c,d)]_{\cong_Z} &:= [(a\cdot c + b\cdot d, a\cdot d + b\cdot c)]_{\cong_Z} \\ [(a,b)]_{\cong_Z} \leqslant_Z [(c,d)]_{\cong_Z} &:\Leftrightarrow a+d \leqslant b+c \end{aligned}$$

- It is easy to show that
  - addition, multiplication and order are well-defined,
  - the standard rules of arithmetic hold, with $[(0,0)]_{\cong_Z}$ as zero element ("zero"),
  - $\leqslant_Z$ defines a total order on $\mathbb{N}_0 \times \mathbb{N}_0$.

Definition 89 (Positive/negative)

An integer is *positive* if it is greater than zero and *negative* if it is less than zero; zero is neither positive nor negative.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Construction of $\mathbb{Z}$ Based on $\mathbb{N}$

- It remains to define addition, multiplication and order on $\mathbb{Z}$. For $a, b, c, d \in \mathbb{N}_0$ we define an addition $+_Z$, a multiplication $\cdot_Z$ and an order $\leqslant_Z$ as follows:

$$
\begin{aligned}
[(a,b)]_{\cong_Z} +_Z [(c,d)]_{\cong_Z} &:= [(a+c, b+d)]_{\cong_Z} \\
[(a,b)]_{\cong_Z} \cdot_Z [(c,d)]_{\cong_Z} &:= [(a\cdot c + b\cdot d, a\cdot d + b\cdot c)]_{\cong_Z} \\
[(a,b)]_{\cong_Z} \leqslant_Z [(c,d)]_{\cong_Z} &:\Leftrightarrow a+d \leqslant b+c
\end{aligned}
$$

- It is easy to show that
  - addition, multiplication and order are well-defined,
  - the standard rules of arithmetic hold, with $[(0,0)]_{\cong_Z}$ as zero element (“zero”),
  - $\leqslant_Z$ defines a total order on $\mathbb{N}_0 \times \mathbb{N}_0$.

Definition 89 (Positive/negative)

An integer is *positive* if it is greater than zero and *negative* if it is less than zero; zero is neither positive nor negative.

Theorem 90

$\mathbb{Z}$ is a countably infinite set. That is, $|\mathbb{N}| = |\mathbb{Z}|$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Integral Powers

Definition 91 (Integral power, Dt.: ganzzahlige Potenz)

Consider $x \in F$ for a field $(F, +, \cdot)$, with additive neutral element e. For $n \in \mathbb{N}_0$, we define integral powers of x as follows:

$$x^n := \begin{cases} 1 & \text{if} \quad n = 0 \text{ and } x \neq e, \\ x & \text{if} \quad n = 1, \\ x^{n-1} \cdot x & \text{if} \quad n > 1. \end{cases}$$

Integral Powers

Definition 91 (Integral power, Dt.: ganzzahlige Potenz)

Consider $x \in F$ for a field $(F, +, \cdot)$, with additive neutral element e. For $n \in \mathbb{N}_0$, we define integral powers of x as follows:

$$x^n := \begin{cases} 1 & \text{if} \quad n = 0 \text{ and } x \neq e, \\ x & \text{if} \quad n = 1, \\ x^{n-1} \cdot x & \text{if} \quad n > 1. \end{cases}$$

Furthermore, for $n \in \mathbb{Z}$ with $n < 0$,

$$x^n := \frac{1}{x^{-n}} \quad \text{if } x \neq e.$$

Integral Powers

Definition 91 (Integral power, Dt.: ganzzahlige Potenz)

Consider $x \in F$ for a field $(F, +, \cdot)$, with additive neutral element e. For $n \in \mathbb{N}_0$, we define integral powers of x as follows:

$$x^n := \begin{cases} 1 & \text{if} \quad n = 0 \text{ and } x \neq e, \\ x & \text{if} \quad n = 1, \\ x^{n-1} \cdot x & \text{if} \quad n > 1. \end{cases}$$

Furthermore, for $n \in \mathbb{Z}$ with $n < 0$,

$$x^n := \frac{1}{x^{-n}} \quad \text{if } x \neq e.$$

- Normally, in $\mathbb{R}$ the term 0^0 remains undefined, and $x \neq 0$ is implicitly assumed for x^0. However, there are applications — e.g., in geometric modeling when defining Bernstein basis polynomials — for which it is convenient to regard 0^0 as 1.

Definition 91 (Integral power, Dt.: ganzzahlige Potenz)

Consider $x \in F$ for a field $(F, +, \cdot)$, with additive neutral element e. For $n \in \mathbb{N}_0$, we define integral powers of x as follows:

$$x^n := \begin{cases} 1 & \text{if} \quad n = 0 \text{ and } x \neq e, \\ x & \text{if} \quad n = 1, \\ x^{n-1} \cdot x & \text{if} \quad n > 1. \end{cases}$$

Furthermore, for $n \in \mathbb{Z}$ with $n < 0$,

$$x^n := \frac{1}{x^{-n}} \quad \text{if } x \neq e.$$

- Normally, in $\mathbb{R}$ the term 0^0 remains undefined, and $x \neq 0$ is implicitly assumed for x^0. However, there are applications — e.g., in geometric modeling when defining Bernstein basis polynomials — for which it is convenient to regard 0^0 as 1.

Lemma 92

Let $(F, +, \cdot)$ be a field. Then, for all $x, y \in F$ and all $m, n \in \mathbb{Z}$,

$$x^m \cdot x^n = x^{m+n} \qquad \text{and} \qquad x^n \cdot y^n = (x \cdot y)^n.$$

Definition 93 (Divisor, Dt.: Teiler, Faktor)

Let $a, b \in \mathbb{Z}$ with $a \neq 0$. Then a *divides* b, denoted by $a \mid b$, if there exists $c \in \mathbb{Z}$ such that $b = c \cdot a$.

$$a \mid b \quad :\Leftrightarrow \quad \exists c \in \mathbb{Z} \;\; b = c \cdot a.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 93 (Divisor, Dt.: Teiler, Faktor)

Let $a, b \in \mathbb{Z}$ with $a \neq 0$. Then *a divides b*, denoted by $a \mid b$, if there exists $c \in \mathbb{Z}$ such that $b = c \cdot a$.

$$a \mid b \quad :\Leftrightarrow \quad \exists c \in \mathbb{Z} \;\; b = c \cdot a.$$

In this case, we also say that b is a *multiple* of a, or a is a *divisor* or *factor* of b, or b is *divisible* by a. Otherwise we have $a \nmid b$. We have a *genuine divisor* if $a \mid b$ and $a \neq \pm 1$ and $a \neq \pm b$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 93 (Divisor, Dt.: Teiler, Faktor)

Let $a, b \in \mathbb{Z}$ with $a \neq 0$. Then *a divides b*, denoted by $a \mid b$, if there exists $c \in \mathbb{Z}$ such that $b = c \cdot a$.

$$a \mid b \quad :\Leftrightarrow \quad \exists c \in \mathbb{Z} \;\; b = c \cdot a.$$

In this case, we also say that b is a *multiple* of a, or a is a *divisor* or *factor* of b, or b is *divisible* by a. Otherwise we have $a \nmid b$. We have a *genuine divisor* if $a \mid b$ and $a \neq \pm 1$ and $a \neq \pm b$.

- Note that $a \mid 0$ for all $a \in \mathbb{Z} \setminus \{0\}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 93 (Divisor, Dt.: Teiler, Faktor)

Let $a, b \in \mathbb{Z}$ with $a \neq 0$. Then a *divides* b, denoted by $a \mid b$, if there exists $c \in \mathbb{Z}$ such that $b = c \cdot a$.

$$a \mid b \quad :\Leftrightarrow \quad \exists c \in \mathbb{Z} \;\; b = c \cdot a.$$

In this case, we also say that b is a *multiple* of a, or a is a *divisor* or *factor* of b, or b is *divisible* by a. Otherwise we have $a \nmid b$. We have a *genuine divisor* if $a \mid b$ and $a \neq \pm 1$ and $a \neq \pm b$.

- Note that $a \mid 0$ for all $a \in \mathbb{Z} \backslash \{0\}$.

Definition 94 (Even/odd, Dt.: gerade/ungerade)

A number $b \in \mathbb{Z}$ is said to be *even* if and only if $2 \mid b$; otherwise, b is *odd*.

Lemma 95

1. $\forall a \in \mathbb{Z}\backslash\{0\} \qquad a \mid a.$
2. $\forall a \in \mathbb{Z}\backslash\{0\}\ \forall b \in \mathbb{Z} \qquad a \mid b \ \Rightarrow\ \big(\forall c \in \mathbb{Z}\ \ a \mid (b \cdot c)\big).$
3. $\forall a, b \in \mathbb{Z}\backslash\{0\}\ \forall c \in \mathbb{Z} \qquad (a \mid b\ \wedge\ b \mid c)\ \Rightarrow\ a \mid c.$
4. $\forall a \in \mathbb{Z}\backslash\{0\}\ \forall b, c \in \mathbb{Z} \qquad (a \mid b\ \wedge\ a \mid c)\ \Rightarrow\ \big(\forall s, t \in \mathbb{Z}\ \ a \mid (b \cdot s + c \cdot t)\big).$
5. $\forall a, c \in \mathbb{Z}\backslash\{0\}\ \forall b \in \mathbb{Z} \qquad a \mid b\ \Leftrightarrow\ (a \cdot c) \mid (b \cdot c).$
6. $\forall a, b \in \mathbb{Z}\backslash\{0\} \qquad (a \mid b\ \wedge\ b \mid a)\ \Rightarrow\ (a = b\ \vee\ a = -b).$

Lemma 95

1. $\forall a \in \mathbb{Z}\backslash\{0\} \qquad a \mid a.$
2. $\forall a \in \mathbb{Z}\backslash\{0\}\ \forall b \in \mathbb{Z} \qquad a \mid b \ \Rightarrow\ \big(\forall c \in \mathbb{Z}\ \ a \mid (b \cdot c)\big).$
3. $\forall a, b \in \mathbb{Z}\backslash\{0\}\ \forall c \in \mathbb{Z} \qquad (a \mid b\ \wedge\ b \mid c)\ \Rightarrow\ a \mid c.$
4. $\forall a \in \mathbb{Z}\backslash\{0\}\ \forall b, c \in \mathbb{Z} \qquad (a \mid b\ \wedge\ a \mid c)\ \Rightarrow\ \big(\forall s, t \in \mathbb{Z}\ \ a \mid (b \cdot s + c \cdot t)\big).$
5. $\forall a, c \in \mathbb{Z}\backslash\{0\}\ \forall b \in \mathbb{Z} \qquad a \mid b\ \Leftrightarrow\ (a \cdot c) \mid (b \cdot c).$
6. $\forall a, b \in \mathbb{Z}\backslash\{0\} \qquad (a \mid b\ \wedge\ b \mid a)\ \Rightarrow\ (a = b\ \vee\ a = -b).$

Lemma 96

For all $a, b, c \in \mathbb{Z}$ and all $k \in \mathbb{Z}\backslash\{0\}$,

$$(a = b + c \quad \wedge \quad k \mid b) \qquad \Rightarrow \qquad (k \mid a \quad \Leftrightarrow \quad k \mid c).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Divisibility Rules

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

2. if its last digit is even, i.e., 0, 2, 4, 6 or 8;

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

2. if its last digit is even, i.e., 0, 2, 4, 6 or 8;
3. if the sum of its digits is divisible by three;

Divisibility Rules

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

2. if its last digit is even, i.e., 0, 2, 4, 6 or 8;
3. if the sum of its digits is divisible by three;
4. if its last two digits form a number that is divisible by four;

Divisibility Rules

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

2. if its last digit is even, i.e., 0, 2, 4, 6 or 8;
3. if the sum of its digits is divisible by three;
4. if its last two digits form a number that is divisible by four;
5. if its last digit is 0 or 5;

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

2. if its last digit is even, i.e., 0, 2, 4, 6 or 8;
3. if the sum of its digits is divisible by three;
4. if its last two digits form a number that is divisible by four;
5. if its last digit is 0 or 5;
6. if it is divisible by two and three;

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

- 2 if its last digit is even, i.e., 0, 2, 4, 6 or 8;
- 3 if the sum of its digits is divisible by three;
- 4 if its last two digits form a number that is divisible by four;
- 5 if its last digit is 0 or 5;
- 6 if it is divisible by two and three;
- 8 if the hundreds digit is even and the number formed by the last two digits is divisible by eight, or if the hundreds digit is odd and the number formed by the last two digits plus four is divisible by eight;

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

- **2** if its last digit is even, i.e., 0, 2, 4, 6 or 8;
- **3** if the sum of its digits is divisible by three;
- **4** if its last two digits form a number that is divisible by four;
- **5** if its last digit is 0 or 5;
- **6** if it is divisible by two and three;
- **8** if the hundreds digit is even and the number formed by the last two digits is divisible by eight, or if the hundreds digit is odd and the number formed by the last two digits plus four is divisible by eight;
- **9** if the sum of its digits is divisible by nine;

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

2. if its last digit is even, i.e., 0, 2, 4, 6 or 8;
3. if the sum of its digits is divisible by three;
4. if its last two digits form a number that is divisible by four;
5. if its last digit is 0 or 5;
6. if it is divisible by two and three;
8. if the hundreds digit is even and the number formed by the last two digits is divisible by eight, or if the hundreds digit is odd and the number formed by the last two digits plus four is divisible by eight;
9. if the sum of its digits is divisible by nine;
10. if its last digit is 0;

Divisibility Rules

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

- 2 if its last digit is even, i.e., 0, 2, 4, 6 or 8;
- 3 if the sum of its digits is divisible by three;
- 4 if its last two digits form a number that is divisible by four;
- 5 if its last digit is 0 or 5;
- 6 if it is divisible by two and three;
- 8 if the hundreds digit is even and the number formed by the last two digits is divisible by eight, or if the hundreds digit is odd and the number formed by the last two digits plus four is divisible by eight;
- 9 if the sum of its digits is divisible by nine;
- 10 if its last digit is 0;
- 11 if the alternating sum of its digits is divisible by eleven;

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

- **2** if its last digit is even, i.e., 0, 2, 4, 6 or 8;
- **3** if the sum of its digits is divisible by three;
- **4** if its last two digits form a number that is divisible by four;
- **5** if its last digit is 0 or 5;
- **6** if it is divisible by two and three;
- **8** if the hundreds digit is even and the number formed by the last two digits is divisible by eight, or if the hundreds digit is odd and the number formed by the last two digits plus four is divisible by eight;
- **9** if the sum of its digits is divisible by nine;
- **10** if its last digit is 0;
- **11** if the alternating sum of its digits is divisible by eleven;
- **12** if it is divisible by three and four.

Lemma 97

A number $a \in \mathbb{N}$ is divisible by

- **2** if its last digit is even, i.e., 0, 2, 4, 6 or 8;
- **3** if the sum of its digits is divisible by three;
- **4** if its last two digits form a number that is divisible by four;
- **5** if its last digit is 0 or 5;
- **6** if it is divisible by two and three;
- **8** if the hundreds digit is even and the number formed by the last two digits is divisible by eight, or if the hundreds digit is odd and the number formed by the last two digits plus four is divisible by eight;
- **9** if the sum of its digits is divisible by nine;
- **10** if its last digit is 0;
- **11** if the alternating sum of its digits is divisible by eleven;
- **12** if it is divisible by three and four.

- There also exist divisibility rules for seven but all of them are a bit ackward.

Divisibility Rules

Proof of Lem. 97 : We prove only the divisibility by three. Let $n \in \mathbb{N}$ and $a_0, a_1, \ldots, a_n \in \{0, 1, \ldots, 9\}$ such that

$$a = \sum_{i=0}^{n} a_i \cdot 10^i.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Divisibility Rules

Proof of Lem. 97 : We prove only the divisibility by three. Let $n \in \mathbb{N}$ and $a_0, a_1, \ldots, a_n \in \{0, 1, \ldots, 9\}$ such that

$$a = \sum_{i=0}^{n} a_i \cdot 10^i.$$

We get

$$a = \sum_{i=0}^{n} a_i \cdot 10^i = \sum_{i=0}^{n} a_i \cdot (10^i - 1) + \sum_{i=0}^{n} a_i.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Divisibility Rules

Proof of Lem. 97: We prove only the divisibility by three. Let $n \in \mathbb{N}$ and $a_0, a_1, \ldots, a_n \in \{0, 1, \ldots, 9\}$ such that

$$a = \sum_{i=0}^{n} a_i \cdot 10^i.$$

We get

$$a = \sum_{i=0}^{n} a_i \cdot 10^i = \sum_{i=0}^{n} a_i \cdot (10^i - 1) + \sum_{i=0}^{n} a_i.$$

Since

$$3 \mid \left(\sum_{i=0}^{n} a_i \cdot (10^i - 1) \right),$$

Lemma 96 implies that the number a is divisible by three if and only if

$$3 \mid \left(\sum_{i=0}^{n} a_i \right).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 98 (Prime, Dt.: Primzahl)

A natural number $p \in \mathbb{N}$ is a *prime number*, or is *prime*, if $p \geqslant 2$ and if p is divisible only by 1 and p itself. All other natural numbers $p \geqslant 2$ are called *composite*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Prime Numbers

Definition 98 (Prime, Dt.: Primzahl)

A natural number $p \in \mathbb{N}$ is a *prime number*, or is *prime*, if $p \geqslant 2$ and if p is divisible only by 1 and p itself. All other natural numbers $p \geqslant 2$ are called *composite*.

- The number 1 is no prime number!
- The only even prime number is 2.
- All primes greater than 2 are odd numbers.
- The set of all prime numbers is frequently (but not always) denoted by $\mathbb{P}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Prime Numbers

Definition 98 (Prime, Dt.: Primzahl)

A natural number $p \in \mathbb{N}$ is a *prime number*, or is *prime*, if $p \geqslant 2$ and if p is divisible only by 1 and p itself. All other natural numbers $p \geqslant 2$ are called *composite*.

- The number 1 is no prime number!
- The only even prime number is 2.
- All primes greater than 2 are odd numbers.
- The set of all prime numbers is frequently (but not always) denoted by $\mathbb{P}$.

Definition 99 (Prime factor, Dt.: Primfaktor)

A natural number $p \in \mathbb{N}$ is a *prime factor* of $n \in \mathbb{N}$ if p is prime and $p \mid n$. If p is a prime factor of n then its *multiplicity* (Dt.: Vielfachheit) is the largest exponent k for which $p^k \mid n$.

Lemma 100

Let $k \in \mathbb{N}$ and $a_1, a_2, \ldots, a_k \in \mathbb{Z}$ and $p \in \mathbb{P}$. Then

$$p \mid a_1 \cdot a_2 \cdot \ldots \cdot a_k \quad \Leftrightarrow \quad (\exists (1 \leqslant j \leqslant k) \quad p \mid a_j).$$

Lemma 100

Let $k \in \mathbb{N}$ and $a_1, a_2, \ldots, a_k \in \mathbb{Z}$ and $p \in \mathbb{P}$. Then

$$p \mid a_1 \cdot a_2 \cdot \ldots \cdot a_k \quad \Leftrightarrow \quad (\exists (1 \leqslant j \leqslant k) \quad p \mid a_j).$$

Corollary 101

If two products of primes are identical then the primes are identical up to the order in which they appear in the products.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 100

Let $k \in \mathbb{N}$ and $a_1, a_2, \ldots, a_k \in \mathbb{Z}$ and $p \in \mathbb{P}$. Then

$$p \mid a_1 \cdot a_2 \cdot \ldots \cdot a_k \quad \Leftrightarrow \quad (\exists (1 \leqslant j \leqslant k) \quad p \mid a_j).$$

Corollary 101

If two products of primes are identical then the primes are identical up to the order in which they appear in the products.

Theorem 102 (Fundamental Theorem of Arithmetic)

Every natural number $n > 1$ is representable uniquely in the form

$$n = p_1^{m_1} \cdot p_2^{m_2} \cdot \ldots \cdot p_k^{m_k},$$

where $p_1 < \ldots < p_k$ are primes and $m_j \in \mathbb{N}$ are multiplicities for every $j = 1, \ldots, k$.

Lemma 100

Let $k \in \mathbb{N}$ and $a_1, a_2, \ldots, a_k \in \mathbb{Z}$ and $p \in \mathbb{P}$. Then

$$p \mid a_1 \cdot a_2 \cdot \ldots \cdot a_k \quad \Leftrightarrow \quad (\exists (1 \leqslant j \leqslant k) \quad p \mid a_j).$$

Corollary 101

If two products of primes are identical then the primes are identical up to the order in which they appear in the products.

Theorem 102 (Fundamental Theorem of Arithmetic)

Every natural number $n > 1$ is representable uniquely in the form

$$n = p_1^{m_1} \cdot p_2^{m_2} \cdot \ldots \cdot p_k^{m_k},$$

where $p_1 < \ldots < p_k$ are primes and $m_j \in \mathbb{N}$ are multiplicities for every $j = 1, \ldots, k$.

Corollary 103

There are infinitely many prime numbers.

Definition 104 (Mersenne prime)

A *Mersenne number* is of the form $2^n - 1$ for $n \in \mathbb{N}$. A *Mersenne prime* is a Mersenne number which is prime.

Definition 104 (Mersenne prime)

A *Mersenne number* is of the form $2^n - 1$ for $n \in \mathbb{N}$. A *Mersenne prime* is a Mersenne number which is prime.

- `INT_MAX` (in C/C++) is the eight Mersenne prime: $2\,147\,483\,647 = 2^{31} - 1$.
- Mersenne primes are used by the *Mersenne twister*, a pseudo-random number generator developed in 1997 by Matsumoto and Nishimura.

Definition 104 (Mersenne prime)

A *Mersenne number* is of the form $2^n - 1$ for $n \in \mathbb{N}$. A *Mersenne prime* is a Mersenne number which is prime.

- `INT_MAX` (in C/C++) is the eight Mersenne prime: $2\,147\,483\,647 = 2^{31} - 1$.
- Mersenne primes are used by the *Mersenne twister*, a pseudo-random number generator developed in 1997 by Matsumoto and Nishimura.
- Several unsolved problems related to Mersenne numbers:
  - Since $2^{11} - 1 = 2047 = 23 \cdot 89$, not all Mersenne numbers are primes!
  - Are there infinitely many Mersenne primes? Only 52 Mersenne primes are known, with $2^{136\,279\,841} - 1$ being the largest known prime. (It was discovered by the "Great Internet Mersenne Prime Search", www.mersenne.org, in October 2024.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

More on Prime Numbers

Definition 104 (Mersenne prime)

A *Mersenne number* is of the form $2^n - 1$ for $n \in \mathbb{N}$. A *Mersenne prime* is a Mersenne number which is prime.

- `INT_MAX` (in C/C++) is the eight Mersenne prime: $2\,147\,483\,647 = 2^{31} - 1$.
- Mersenne primes are used by the *Mersenne twister*, a pseudo-random number generator developed in 1997 by Matsumoto and Nishimura.
- Several unsolved problems related to Mersenne numbers:
  - Since $2^{11} - 1 = 2047 = 23 \cdot 89$, not all Mersenne numbers are primes!
  - Are there infinitely many Mersenne primes? Only 52 Mersenne primes are known, with $2^{136\,279\,841} - 1$ being the largest known prime. (It was discovered by the "Great Internet Mersenne Prime Search", www.mersenne.org, in October 2024.)
  - What is a sufficient condition on n for $2^n - 1$ to be prime?

Definition 104 (Mersenne prime)

A *Mersenne number* is of the form $2^n - 1$ for $n \in \mathbb{N}$. A *Mersenne prime* is a Mersenne number which is prime.

- `INT_MAX` (in C/C++) is the eight Mersenne prime: $2\,147\,483\,647 = 2^{31} - 1$.
- Mersenne primes are used by the *Mersenne twister*, a pseudo-random number generator developed in 1997 by Matsumoto and Nishimura.
- Several unsolved problems related to Mersenne numbers:
  - Since $2^{11} - 1 = 2047 = 23 \cdot 89$, not all Mersenne numbers are primes!
  - Are there infinitely many Mersenne primes? Only 52 Mersenne primes are known, with $2^{136\,279\,841} - 1$ being the largest known prime. (It was discovered by the “Great Internet Mersenne Prime Search”, www.mersenne.org, in October 2024.)
  - What is a sufficient condition on n for $2^n - 1$ to be prime?

Lemma 105

If $2^n - 1$ is prime for some $n \in \mathbb{N}$ then n is prime.

Conjecture 106 (Goldbach 1742, “weak version” or “ternary conjecture”)

Every odd natural number greater than 5 can be written as the sum of three primes.

Chances to Become Famous: Conjectures About Primes

Conjecture 106 (Goldbach 1742, “weak version” or “ternary conjecture”)

Every odd natural number greater than 5 can be written as the sum of three primes.

Conjecture 107 (Goldbach-Euler 1742, “strong version”)

Every even natural number greater than 3 can be written as the sum of two primes.

Conjecture 106 (Goldbach 1742, "weak version" or "ternary conjecture")

Every odd natural number greater than 5 can be written as the sum of three primes.

Conjecture 107 (Goldbach-Euler 1742, "strong version")

Every even natural number greater than 3 can be written as the sum of two primes.

- Christian Goldbach (1690–1764), Leonhard Euler (1707–1783).
- The strong version of this conjecture implies the weak version: If $n \in \mathbb{N}$, with $n \geqslant 7$, is odd then $n' := n - 3$ is even with $n' > 3$. Hence, if n' can be written as the sum of two primes, then n can be written as the sum of three primes.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Chances to Become Famous: Conjectures About Primes

Conjecture 106 (Goldbach 1742, “weak version” or “ternary conjecture”)

Every odd natural number greater than 5 can be written as the sum of three primes.

Conjecture 107 (Goldbach-Euler 1742, “strong version”)

Every even natural number greater than 3 can be written as the sum of two primes.

- Christian Goldbach (1690–1764), Leonhard Euler (1707–1783).
- The strong version of this conjecture implies the weak version: If $n \in \mathbb{N}$, with $n \geqslant 7$, is odd then $n' := n - 3$ is even with $n' > 3$. Hence, if n' can be written as the sum of two primes, then n can be written as the sum of three primes.
- In 1937, Vinogradov proved the weak version for ”sufficiently large numbers”, and later on his student Borozdin proved $3^{3^{15}}$ to be sufficiently large.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Chances to Become Famous: Conjectures About Primes

Conjecture 106 (Goldbach 1742, "weak version" or "ternary conjecture")

Every odd natural number greater than 5 can be written as the sum of three primes.

Conjecture 107 (Goldbach-Euler 1742, "strong version")

Every even natural number greater than 3 can be written as the sum of two primes.

- Christian Goldbach (1690–1764), Leonhard Euler (1707–1783).
- The strong version of this conjecture implies the weak version: If $n \in \mathbb{N}$, with $n \geqslant 7$, is odd then $n' := n - 3$ is even with $n' > 3$. Hence, if n' can be written as the sum of two primes, then n can be written as the sum of three primes.
- In 1937, Vinogradov proved the weak version for "sufficiently large numbers", and later on his student Borozdin proved $3^{3^{15}}$ to be sufficiently large.
- By means of distributed computer search, as of Dec 2012, Tomás Oliveira e Silva verified the strong version of Goldbach's conjecture up to $4 \cdot 10^{18}$.
- Also by distributed computing, in 2013 Harald Helfgott and David Platt verified the weak Goldbach conjecture up to (roughly) $8 \cdot 10^{30}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Chances to Become Famous: Conjectures About Primes

Conjecture 106 (Goldbach 1742, "weak version" or "ternary conjecture")

Every odd natural number greater than 5 can be written as the sum of three primes.

Conjecture 107 (Goldbach-Euler 1742, "strong version")

Every even natural number greater than 3 can be written as the sum of two primes.

- Christian Goldbach (1690–1764), Leonhard Euler (1707–1783).
- The strong version of this conjecture implies the weak version: If $n \in \mathbb{N}$, with $n \geqslant 7$, is odd then $n' := n - 3$ is even with $n' > 3$. Hence, if n' can be written as the sum of two primes, then n can be written as the sum of three primes.
- In 1937, Vinogradov proved the weak version for "sufficiently large numbers", and later on his student Borozdin proved $3^{3^{15}}$ to be sufficiently large.
- By means of distributed computer search, as of Dec 2012, Tomás Oliveira e Silva verified the strong version of Goldbach's conjecture up to $4 \cdot 10^{18}$.
- Also by distributed computing, in 2013 Harald Helfgott and David Platt verified the weak Goldbach conjecture up to (roughly) $8 \cdot 10^{30}$.
- In 2013, Helfgott released a 240-page analysis that, if accepted as correct, yields a formal proof of the weak conjecture for all natural numbers greater than $\approx 10^{30}$.

Chances to Become Famous: Conjectures About Primes

Conjecture 108 (Polignac, 1849)

For every natural number k there exist infinitely many numbers p such that p and $p + 2k$ are consecutive primes.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Chances to Become Famous: Conjectures About Primes

Conjecture 108 (Polignac, 1849)

For every natural number k there exist infinitely many numbers p such that p and $p + 2k$ are consecutive primes.

- For $k := 1$, the conjecture by Alphonse de Polignac (1817–1890) is known as the *twin prime conjecture*. As of 19-Sep-2016, the largest known twin primes are $2\,996\,863\,034\,895^{1\,290\,000} \pm 1$; these numbers have 388 342 digits.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Conjecture 108 (Polignac, 1849)

For every natural number k there exist infinitely many numbers p such that p and $p + 2k$ are consecutive primes.

- For $k := 1$, the conjecture by Alphonse de Polignac (1817–1890) is known as the *twin prime conjecture*. As of 19-Sep-2016, the largest known twin primes are $2\,996\,863\,034\,895^{1\,290\,000} \pm 1$; these numbers have 388 342 digits.
- In April 2013, Yitang Zhang proved that their exist infinitely many consecutive prime numbers p_{n+1} and p_n such that $p_{n+1} - p_n < 7 \cdot 10^7$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Conjecture 108 (Polignac, 1849)

For every natural number k there exist infinitely many numbers p such that p and $p + 2k$ are consecutive primes.

- For $k := 1$, the conjecture by Alphonse de Polignac (1817–1890) is known as the *twin prime conjecture*. As of 19-Sep-2016, the largest known twin primes are $2\,996\,863\,034\,895^{1\,290\,000} \pm 1$; these numbers have 388 342 digits.
- In April 2013, Yitang Zhang proved that their exist infinitely many consecutive prime numbers p_{n+1} and p_n such that $p_{n+1} - p_n < 7 \cdot 10^7$.
- In November 2013, James Maynard reduced this bound to 600.
- This bound seems to have been further reduced to 246 by the Polymath project.

A Chance Missed to Become Famous: Fermat's Last Theorem

- A Diophantine equation is an equation for which only integer solutions are sought.
- E.g., $(3, 4, 5)$ is an integer solution triple for $a^2 + b^2 = c^2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

A Chance Missed to Become Famous: Fermat's Last Theorem

- A Diophantine equation is an equation for which only integer solutions are sought.
- E.g., $(3, 4, 5)$ is an integer solution triple for $a^2 + b^2 = c^2$.

Theorem 109 (Wiles&Taylor, 1995)

For every natural number $n > 2$, the Diophantine equation $a^n + b^n = c^n$ has no solution $(a, b, c) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$.

- Dt.: Großer Satz von Fermat.

A Chance Missed to Become Famous: Fermat's Last Theorem

- A Diophantine equation is an equation for which only integer solutions are sought.
- E.g., $(3, 4, 5)$ is an integer solution triple for $a^2 + b^2 = c^2$.

Theorem 109 (Wiles&Taylor, 1995)

For every natural number $n > 2$, the Diophantine equation $a^n + b^n = c^n$ has no solution $(a, b, c) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$.

- Dt.: Großer Satz von Fermat.
- Stated in 1637 by Pierre de Fermat (1607(?)–1665) without proof, but with a famous side remark: "Cuius rei demonstrationem mirabilem sane detexi. Hanc marginis exiguitas non caperet."
- Proved for $n := 4$ by Fermat himself.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

A Chance Missed to Become Famous: Fermat's Last Theorem

- A Diophantine equation is an equation for which only integer solutions are sought.
- E.g., $(3, 4, 5)$ is an integer solution triple for $a^2 + b^2 = c^2$.

Theorem 109 (Wiles&Taylor, 1995)

For every natural number $n > 2$, the Diophantine equation $a^n + b^n = c^n$ has no solution $(a, b, c) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$.

- Dt.: Großer Satz von Fermat.
- Stated in 1637 by Pierre de Fermat (1607(?)–1665) without proof, but with a famous side remark: "Cuius rei demonstrationem mirabilem sane detexi. Hanc marginis exiguitas non caperet."
- Proved for $n := 4$ by Fermat himself.
- Finally proved by Andrew Wiles in 1993; a gap in the proof was fixed by Wiles and his former student Richard Taylor; the full proof was published in 1995.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Quotient and Remainder

Lemma 110

Let $a \in \mathbb{N}$ and $b \in \mathbb{Z}$. Then there exist a unique *quotient* $q \in \mathbb{Z}$ and a unique *remainder* $r \in \mathbb{N}_0$ such that

$$b = a \cdot q + r \quad \text{and} \quad 0 \leqslant r < a.$$

- We will use the operators div and mod for computing the quotient and remainder. That is, q and r of Lemma 110 are given by $q := b \text{ div } a$ and $r := b \bmod a$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Quotient and Remainder

Lemma 110

Let $a \in \mathbb{N}$ and $b \in \mathbb{Z}$. Then there exist a unique *quotient* $q \in \mathbb{Z}$ and a unique *remainder* $r \in \mathbb{N}_0$ such that

$$b = a \cdot q + r \quad \text{and} \quad 0 \leqslant r < a.$$

- We will use the operators div and mod for computing the quotient and remainder. That is, q and r of Lemma 110 are given by $q := b \text{ div } a$ and $r := b \bmod a$.
- In many programming languages the remainder r can be obtained by means of the *modulo* operator. See, e.g., the operator % in C, C++, C#, Java, and Perl.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Quotient and Remainder

Lemma 110

Let $a \in \mathbb{N}$ and $b \in \mathbb{Z}$. Then there exist a unique *quotient* $q \in \mathbb{Z}$ and a unique *remainder* $r \in \mathbb{N}_0$ such that

$$b = a \cdot q + r \quad \text{and} \quad 0 \leqslant r < a.$$

- We will use the operators div and mod for computing the quotient and remainder. That is, q and r of Lemma 110 are given by $q := b \text{ div } a$ and $r := b \bmod a$.
- In many programming languages the remainder r can be obtained by means of the *modulo* operator. See, e.g., the operator % in C, C++, C#, Java, and Perl.
- IEEE 754 defines a remainder function based on the round-to-nearest convention.

Warning

If one or both of a and b are allowed to be negative integers then the sign of the remainder may differ among different implementations!

Real-World Application: Base Conversion

- We know that $25 = (11001)_2$, i.e., $(11001)_2$ is the base-two representation of $25 = (25)_{10}$. (After all, $25 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$.)

Real-World Application: Base Conversion

- We know that $25 = (11001)_2$, i.e., $(11001)_2$ is the base-two representation of $25 = (25)_{10}$. (After all, $25 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$.)
- How can we represent an integer relative to an arbitrary base $b \in \mathbb{N} \setminus \{1\}$?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- We know that $25 = (11001)_2$, i.e., $(11001)_2$ is the base-two representation of $25 = (25)_{10}$. (After all, $25 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$.)
- How can we represent an integer relative to an arbitrary base $b \in \mathbb{N} \backslash \{1\}$?
- Lemma 110 tells us that there exist unique $q_0, r_0 \in \mathbb{N}_0$ such that

$$n = b \cdot q_0 + r_0 \qquad \text{with} \quad 0 \leqslant r_0 < b.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Base Conversion

- We know that $25 = (11001)_2$, i.e., $(11001)_2$ is the base-two representation of $25 = (25)_{10}$. (After all, $25 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$.)
- How can we represent an integer relative to an arbitrary base $b \in \mathbb{N} \setminus \{1\}$?
- Lemma 110 tells us that there exist unique $q_0, r_0 \in \mathbb{N}_0$ such that

$$n = b \cdot q_0 + r_0 \qquad \text{with} \quad 0 \leqslant r_0 < b.$$

- The number r_0 becomes the rightmost digit of the base-b representation of n, and we seek q_1, r_1 such that

$$q_0 = b \cdot q_1 + r_1 \qquad \text{with} \quad 0 \leqslant r_1 < b,$$

and so on until some $q_i = 0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Base Conversion

- We know that $25 = (11001)_2$, i.e., $(11001)_2$ is the base-two representation of $25 = (25)_{10}$. (After all, $25 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$.)
- How can we represent an integer relative to an arbitrary base $b \in \mathbb{N} \setminus \{1\}$?
- Lemma 110 tells us that there exist unique $q_0, r_0 \in \mathbb{N}_0$ such that

$$n = b \cdot q_0 + r_0 \qquad \text{with} \quad 0 \leqslant r_0 < b.$$

- The number r_0 becomes the rightmost digit of the base-b representation of n, and we seek q_1, r_1 such that

$$q_0 = b \cdot q_1 + r_1 \qquad \text{with} \quad 0 \leqslant r_1 < b,$$

  and so on until some $q_i = 0$.
- E.g.,

$$\begin{array}{rcr} 25 & = & 12 \cdot 2 + 1 \\ 12 & = & 6 \cdot 2 + 0 \\ 6 & = & 3 \cdot 2 + 0 \\ 3 & = & 1 \cdot 2 + 1 \\ 1 & = & 0 \cdot 2 + 1 \end{array}$$

  and therefore $25 = (11001)_2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Congruences

- Introduced by Carl Friedrich Gauss (1777–1855) in 1801.

Definition 111 (Congruence, Dt.: Kongruenz)

Let $a, b \in \mathbb{Z}$ and $m \in \mathbb{N}$. We say that a is *congruent* to b *modulo* m, and write

$$a \equiv_m b,$$

if $a - b$ is divisible by m. The term $a \equiv_m b$ is called a *congruence*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Introduced by Carl Friedrich Gauss (1777–1855) in 1801.

Definition 111 (Congruence, Dt.: Kongruenz)

Let $a, b \in \mathbb{Z}$ and $m \in \mathbb{N}$. We say that a is *congruent* to b *modulo* m, and write

$$a \equiv_m b,$$

if $a - b$ is divisible by m. The term $a \equiv_m b$ is called a *congruence*.

- Hence: $a \equiv_m b \quad :\Leftrightarrow \quad m \mid (a - b)$.

- Introduced by Carl Friedrich Gauss (1777–1855) in 1801.

Definition 111 (Congruence, Dt.: Kongruenz)

Let $a, b \in \mathbb{Z}$ and $m \in \mathbb{N}$. We say that a is *congruent* to b *modulo* m, and write

$$a \equiv_m b,$$

if $a - b$ is divisible by m. The term $a \equiv_m b$ is called a *congruence*.

- Hence: $a \equiv_m b \quad :\Leftrightarrow \quad m \mid (a - b)$.
- Note: Some authors prefer to write $a \equiv b \ (m)$ or $a \equiv b \bmod m$ for $a \equiv_m b$.

- Introduced by Carl Friedrich Gauss (1777–1855) in 1801.

Definition 111 (Congruence, Dt.: Kongruenz)

Let $a, b \in \mathbb{Z}$ and $m \in \mathbb{N}$. We say that a is *congruent* to b *modulo* m, and write

$$a \equiv_m b,$$

if $a - b$ is divisible by m. The term $a \equiv_m b$ is called a *congruence*.

- Hence: $a \equiv_m b \quad :\Leftrightarrow \quad m \mid (a - b)$.
- Note: Some authors prefer to write $a \equiv b\ (m)$ or $a \equiv b \bmod m$ for $a \equiv_m b$.

Lemma 112

For all $a, b \in \mathbb{Z}$ and $m \in \mathbb{N}$, we have $a \equiv_m b$ if and only if a and b have the same remainder after dividing by m, i.e., if and only if $a \bmod m = b \bmod m$.

Congruences

$$38 \equiv_{12} 2$$
$$-3 \equiv_5 2$$
$$0 \equiv_3 3$$
$$8 \equiv_3 2$$
$$7 \equiv_3 1$$
$$7 \equiv_3 -8$$

$$\textit{even} + \textit{even} \equiv_2 \textit{even}$$
$$\textit{even} + \textit{odd} \equiv_2 \textit{odd}$$
$$\textit{odd} + \textit{odd} \equiv_2 \textit{even}$$
$$\textit{even} \cdot \textit{even} \equiv_2 \textit{even}$$
$$\textit{even} \cdot \textit{odd} \equiv_2 \textit{even}$$
$$\textit{odd} \cdot \textit{odd} \equiv_2 \textit{odd}$$

Congruences

$$38 \equiv_{12} 2 \qquad even + even \equiv_2 even$$
$$-3 \equiv_5 2 \qquad even + odd \equiv_2 odd$$
$$0 \equiv_3 3 \qquad odd + odd \equiv_2 even$$
$$8 \equiv_3 2 \qquad even \cdot even \equiv_2 even$$
$$7 \equiv_3 1 \qquad even \cdot odd \equiv_2 even$$
$$7 \equiv_3 -8 \qquad odd \cdot odd \equiv_2 odd$$

Lemma 113

For $m \in \mathbb{N}$, the relation $\equiv_m$ is an equivalence relation on $\mathbb{Z}$, i.e., for all $a, b, c \in \mathbb{Z}$,

- **reflexivity** $a \equiv_m a$,
- **symmetry** if $a \equiv_m b$ then $b \equiv_m a$, and
- **transitivity** if $a \equiv_m b$ and $b \equiv_m c$ then $a \equiv_m c$

hold.

Lemma 114

For $m \in \mathbb{N}$, the relation $\equiv_m$ is a congruence relation on $\mathbb{Z}$, i.e., it respects addition, subtraction, and multiplication: Let $a, b, c, d \in \mathbb{Z}$ and $m \in \mathbb{N}$, and suppose that

$$a \equiv_m b \quad \text{and} \quad c \equiv_m d.$$

Then

$$a + c \equiv_m b + d \quad \text{and} \quad a - c \equiv_m b - d \quad \text{and} \quad a \cdot c \equiv_m b \cdot d.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 114

For $m \in \mathbb{N}$, the relation $\equiv_m$ is a congruence relation on $\mathbb{Z}$, i.e., it respects addition, subtraction, and multiplication: Let $a, b, c, d \in \mathbb{Z}$ and $m \in \mathbb{N}$, and suppose that

$$a \equiv_m b \quad \text{and} \quad c \equiv_m d.$$

Then

$$a + c \equiv_m b + d \quad \text{and} \quad a - c \equiv_m b - d \quad \text{and} \quad a \cdot c \equiv_m b \cdot d.$$

Definition 115 (Residue, Dt.: Residuum, Restklasse)

Let $m \in \mathbb{N}$ with $m \geqslant 2$. The equivalence classes of $\mathbb{Z}$ modulo m are called *residues* (or remainders) modulo m. For $a \in \mathbb{Z}$, its equivalence class modulo m is denoted by $[a]_m$. The set of residues modulo m is denoted by $\mathbb{Z}_m$ or $\mathbb{Z}/m\mathbb{Z}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 114

For $m \in \mathbb{N}$, the relation $\equiv_m$ is a congruence relation on $\mathbb{Z}$, i.e., it respects addition, subtraction, and multiplication: Let $a, b, c, d \in \mathbb{Z}$ and $m \in \mathbb{N}$, and suppose that

$$a \equiv_m b \quad \text{and} \quad c \equiv_m d.$$

Then

$$a + c \equiv_m b + d \quad \text{and} \quad a - c \equiv_m b - d \quad \text{and} \quad a \cdot c \equiv_m b \cdot d.$$

Definition 115 (Residue, Dt.: Residuum, Restklasse)

Let $m \in \mathbb{N}$ with $m \geqslant 2$. The equivalence classes of $\mathbb{Z}$ modulo m are called *residues* (or remainders) modulo m. For $a \in \mathbb{Z}$, its equivalence class modulo m is denoted by $[a]_m$. The set of residues modulo m is denoted by $\mathbb{Z}_m$ or $\mathbb{Z}/m\mathbb{Z}$.

Lemma 116

Let $m \in \mathbb{N}$ with $m \geqslant 2$. Then $\mathbb{Z}_m = \{[a]_m : \; a \in \mathbb{N}_0 \; \wedge \; a < m\}$.

Definition 117 (Arithmetic on $\mathbb{Z}_m$)

Let $m \in \mathbb{N}$ with $m \geqslant 2$, and $[a]_m, [b]_m \in \mathbb{Z}_m$. On $\mathbb{Z}_m$ we define an addition $+_m$ and a multiplication $\cdot_m$ as follows.

$$[a]_m +_m [b]_m \quad := \quad [a+b]_m$$

$$[a]_m \cdot_m [b]_m \quad := \quad [a \cdot b]_m$$

Definition 117 (Arithmetic on $\mathbb{Z}_m$)

Let $m \in \mathbb{N}$ with $m \geqslant 2$, and $[a]_m, [b]_m \in \mathbb{Z}_m$. On $\mathbb{Z}_m$ we define an addition $+_m$ and a multiplication $\cdot_m$ as follows.

$$\begin{aligned} [a]_m +_m [b]_m &:= [a+b]_m \\ [a]_m \cdot_m [b]_m &:= [a \cdot b]_m \end{aligned}$$

Lemma 118

Let $m \in \mathbb{N}$ with $m \geqslant 2$. Then addition $+_m$ and multiplication $\cdot_m$ on $\mathbb{Z}_m$ are well-defined. Furthermore, $(\mathbb{Z}_m, +_m, \cdot_m)$ forms a commutative ring.

Definition 117 (Arithmetic on $\mathbb{Z}_m$)

Let $m \in \mathbb{N}$ with $m \geqslant 2$, and $[a]_m, [b]_m \in \mathbb{Z}_m$. On $\mathbb{Z}_m$ we define an addition $+_m$ and a multiplication $\cdot_m$ as follows.

$$\begin{aligned} [a]_m +_m [b]_m &:= [a+b]_m \\ [a]_m \cdot_m [b]_m &:= [a \cdot b]_m \end{aligned}$$

Lemma 118

Let $m \in \mathbb{N}$ with $m \geqslant 2$. Then addition $+_m$ and multiplication $\cdot_m$ on $\mathbb{Z}_m$ are well-defined. Furthermore, $(\mathbb{Z}_m, +_m, \cdot_m)$ forms a commutative ring.

- Often the notation $[a]_m$ is simplified by omitting the modulus m, i.e., by writing $[a]$, or even by simply writing a if it is clear that $a \in \mathbb{Z}_m$. Similarly for $+_m$ and $\cdot_m$.

Definition 117 (Arithmetic on $\mathbb{Z}_m$)

Let $m \in \mathbb{N}$ with $m \geqslant 2$, and $[a]_m, [b]_m \in \mathbb{Z}_m$. On $\mathbb{Z}_m$ we define an addition $+_m$ and a multiplication $\cdot_m$ as follows.

$$\begin{aligned} [a]_m +_m [b]_m &:= [a+b]_m \\ [a]_m \cdot_m [b]_m &:= [a \cdot b]_m \end{aligned}$$

Lemma 118

Let $m \in \mathbb{N}$ with $m \geqslant 2$. Then addition $+_m$ and multiplication $\cdot_m$ on $\mathbb{Z}_m$ are well-defined. Furthermore, $(\mathbb{Z}_m, +_m, \cdot_m)$ forms a commutative ring.

- Often the notation $[a]_m$ is simplified by omitting the modulus m, i.e., by writing $[a]$, or even by simply writing a if it is clear that $a \in \mathbb{Z}_m$. Similarly for $+_m$ and $\cdot_m$.
- It is also common to write

  $$a \bmod m$$

  instead of

  $$[a]_m.$$

Real-World Application: Fermat Primality Test

Theorem 119 (Fermat's Little Theorem)

If $p \in \mathbb{N}$ is prime then $a^p \equiv_p a$ for every $a \in \mathbb{N}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Fermat Primality Test

Theorem 119 (Fermat's Little Theorem)

If $p \in \mathbb{N}$ is prime then $a^p \equiv_p a$ for every $a \in \mathbb{N}$.

- If a is not divisible by p then this yields $a^{p-1} \equiv_p 1$. In particular, this congruence holds for all $1 \leqslant a \leqslant p-1$.

Real-World Application: Fermat Primality Test

Theorem 119 (Fermat's Little Theorem)

If $p \in \mathbb{N}$ is prime then $a^p \equiv_p a$ for every $a \in \mathbb{N}$.

- If a is not divisible by p then this yields $a^{p-1} \equiv_p 1$. In particular, this congruence holds for all $1 \leqslant a \leqslant p-1$.
- Hence, if $a^{n-1} \not\equiv_n 1$ for $n \in \mathbb{N}$ (and $1 \leqslant a \leqslant n-1$) then n is composite, i.e., not a prime.

Real-World Application: Fermat Primality Test

Theorem 119 (Fermat's Little Theorem)

If $p \in \mathbb{N}$ is prime then $a^p \equiv_p a$ for every $a \in \mathbb{N}$.

- If a is not divisible by p then this yields $a^{p-1} \equiv_p 1$. In particular, this congruence holds for all $1 \leqslant a \leqslant p-1$.
- Hence, if $a^{n-1} \not\equiv_n 1$ for $n \in \mathbb{N}$ (and $1 \leqslant a \leqslant n-1$) then n is composite, i.e., not a prime.
- Fermat Primality Test for $n \in \mathbb{N}$:
  1. Pick a random integer a with $1 < a < n-1$.

Theorem 119 (Fermat's Little Theorem)

If $p \in \mathbb{N}$ is prime then $a^p \equiv_p a$ for every $a \in \mathbb{N}$.

- If a is not divisible by p then this yields $a^{p-1} \equiv_p 1$. In particular, this congruence holds for all $1 \leqslant a \leqslant p-1$.
- Hence, if $a^{n-1} \not\equiv_n 1$ for $n \in \mathbb{N}$ (and $1 \leqslant a \leqslant n-1$) then n is composite, i.e., not a prime.
- Fermat Primality Test for $n \in \mathbb{N}$:
  1. Pick a random integer a with $1 < a < n-1$.
  2. If we get $a^{n-1} \not\equiv_n 1$ then a is a *Fermat witness* for the compositeness of n. That is, n is not prime.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 119 (Fermat's Little Theorem)

If $p \in \mathbb{N}$ is prime then $a^p \equiv_p a$ for every $a \in \mathbb{N}$.

- If a is not divisible by p then this yields $a^{p-1} \equiv_p 1$. In particular, this congruence holds for all $1 \leqslant a \leqslant p-1$.
- Hence, if $a^{n-1} \not\equiv_n 1$ for $n \in \mathbb{N}$ (and $1 \leqslant a \leqslant n-1$) then n is composite, i.e., not a prime.
- Fermat Primality Test for $n \in \mathbb{N}$:
  1. Pick a random integer a with $1 < a < n-1$.
  2. If we get $a^{n-1} \not\equiv_n 1$ then a is a *Fermat witness* for the compositeness of n. That is, n is not prime.
  3. Otherwise, repeat the test for some other value of $a \in \{2, 3, \ldots, n-2\}$.

Theorem 119 (Fermat's Little Theorem)

If $p \in \mathbb{N}$ is prime then $a^p \equiv_p a$ for every $a \in \mathbb{N}$.

- If a is not divisible by p then this yields $a^{p-1} \equiv_p 1$. In particular, this congruence holds for all $1 \leqslant a \leqslant p-1$.
- Hence, if $a^{n-1} \not\equiv_n 1$ for $n \in \mathbb{N}$ (and $1 \leqslant a \leqslant n-1$) then n is composite, i.e., not a prime.
- Fermat Primality Test for $n \in \mathbb{N}$:
  1. Pick a random integer a with $1 < a < n-1$.
  2. If we get $a^{n-1} \not\equiv_n 1$ then a is a *Fermat witness* for the compositeness of n. That is, n is not prime.
  3. Otherwise, repeat the test for some other value of $a \in \{2, 3, \ldots, n-2\}$.

  One can prove that the probability for incorrectly classifying n as prime goes to zero (in most cases) as the number of tests is increased.

Real-World Application: Pseudo-Random Numbers

- Since computers cannot flip a coin to obtain a random result, one resorts to algorithms that generate “random” numbers: pseudo-random number generators.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Pseudo-Random Numbers

- Since computers cannot flip a coin to obtain a random result, one resorts to algorithms that generate "random" numbers: pseudo-random number generators.
- *Linear congruential generators* (LCG, [Lehmer 1954]) have been well studied, are easy to implement and used frequently.
- They generate a sequence of non-negative integers less than some specified modulus $m \in \mathbb{N}$ according to the following recursive definition:

$$x_{n+1} := (a \cdot x_n + c) \bmod m,$$

where

$m \in \mathbb{N}$	with	$m > 1$	....	modulus,
$a \in \mathbb{N}$	with	$a < m$	....	multiplier,
$c \in \mathbb{N}_0$	with	$c < m$	....	increment,
$x_0 \in \mathbb{N}_0$	with	$x_0 < m$	....	seed.

Real-World Application: Pseudo-Random Numbers

- Since computers cannot flip a coin to obtain a random result, one resorts to algorithms that generate “random” numbers: pseudo-random number generators.
- *Linear congruential generators* (LCG, [Lehmer 1954]) have been well studied, are easy to implement and used frequently.
- They generate a sequence of non-negative integers less than some specified modulus $m \in \mathbb{N}$ according to the following recursive definition:

  $$x_{n+1} := (a \cdot x_n + c) \bmod m,$$

  where

$m \in \mathbb{N}$	with	$m > 1$	....	modulus,
$a \in \mathbb{N}$	with	$a < m$	....	multiplier,
$c \in \mathbb{N}_0$	with	$c < m$	....	increment,
$x_0 \in \mathbb{N}_0$	with	$x_0 < m$	....	seed.

- E.g., $m := 15$, $a := 1$, $c := 4$ and $x_0 := 2$ yields the following sequence of numbers:

  2 6 10 14 3 7 11 0 4 8 12 1 5 9 13 2 6 ...

Real-World Application: Pseudo-Random Numbers

- Since computers cannot flip a coin to obtain a random result, one resorts to algorithms that generate "random" numbers: pseudo-random number generators.
- *Linear congruential generators* (LCG, [Lehmer 1954]) have been well studied, are easy to implement and used frequently.
- They generate a sequence of non-negative integers less than some specified modulus $m \in \mathbb{N}$ according to the following recursive definition:

$$x_{n+1} := (a \cdot x_n + c) \bmod m,$$

where

$m \in \mathbb{N}$	with	$m > 1$	....	modulus,
$a \in \mathbb{N}$	with	$a < m$	....	multiplier,
$c \in \mathbb{N}_0$	with	$c < m$	....	increment,
$x_0 \in \mathbb{N}_0$	with	$x_0 < m$	....	seed.

- E.g., $m := 15$, $a := 1$, $c := 4$ and $x_0 := 2$ yields the following sequence of numbers:

$$2 \quad 6 \quad 10 \quad 14 \quad 3 \quad 7 \quad 11 \quad 0 \quad 4 \quad 8 \quad 12 \quad 1 \quad 5 \quad 9 \quad 13 \quad 2 \quad 6 \quad \ldots$$

- GCC/glibc: $m := 2^{31} - 1$, $a := 1103515245$, $c := 12345$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Pseudo-Random Numbers

- Since computers cannot flip a coin to obtain a random result, one resorts to algorithms that generate “random” numbers: pseudo-random number generators.
- *Linear congruential generators* (LCG, [Lehmer 1954]) have been well studied, are easy to implement and used frequently.
- They generate a sequence of non-negative integers less than some specified modulus $m \in \mathbb{N}$ according to the following recursive definition:

$$x_{n+1} := (a \cdot x_n + c) \bmod m,$$

where

$m \in \mathbb{N}$	with	$m > 1$	....	modulus,
$a \in \mathbb{N}$	with	$a < m$	....	multiplier,
$c \in \mathbb{N}_0$	with	$c < m$	....	increment,
$x_0 \in \mathbb{N}_0$	with	$x_0 < m$	....	seed.

- E.g., $m := 15$, $a := 1$, $c := 4$ and $x_0 := 2$ yields the following sequence of numbers:

$$2 \quad 6 \quad 10 \quad 14 \quad 3 \quad 7 \quad 11 \quad 0 \quad 4 \quad 8 \quad 12 \quad 1 \quad 5 \quad 9 \quad 13 \quad 2 \quad 6 \quad \ldots$$

- GCC/glibc: $m := 2^{31} - 1$, $a := 1103515245$, $c := 12345$. More advanced pseudo-random number generators exist, e.g., Mersenne twister.

Lemma 120

Let $a, b \in \mathbb{N}$. Then there exists a unique $n \in \mathbb{N}$ such that

1. $n \mid a$ and $n \mid b$, and
2. for all $m \in \mathbb{N}$, if $m \mid a$ and $m \mid b$ then $m \leqslant n$.

Greatest Common Divisor

Lemma 120

Let $a, b \in \mathbb{N}$. Then there exists a unique $n \in \mathbb{N}$ such that

1. $n \mid a$ and $n \mid b$, and
2. for all $m \in \mathbb{N}$, if $m \mid a$ and $m \mid b$ then $m \leqslant n$.

Definition 121 (Greatest common divisor, Dt.: größter gemeinsamer Teiler (ggT))

Let $a, b \in \mathbb{N}$. The unique number $n \in \mathbb{N}$ that exists according to Lem. 120 is called *greatest common divisor* of a and b, and is denoted by $\gcd(a, b)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Greatest Common Divisor

Lemma 120

Let $a, b \in \mathbb{N}$. Then there exists a unique $n \in \mathbb{N}$ such that

1. $n \mid a$ and $n \mid b$, and
2. for all $m \in \mathbb{N}$, if $m \mid a$ and $m \mid b$ then $m \leqslant n$.

Definition 121 (Greatest common divisor, Dt.: größter gemeinsamer Teiler (ggT))

Let $a, b \in \mathbb{N}$. The unique number $n \in \mathbb{N}$ that exists according to Lem. 120 is called *greatest common divisor* of a and b, and is denoted by $\gcd(a, b)$.

- Conventionally, $\gcd(a, 0) = \gcd(0, a) := a$, since 0 is divisible by all integers.

Greatest Common Divisor

Lemma 120

Let $a, b \in \mathbb{N}$. Then there exists a unique $n \in \mathbb{N}$ such that

1. $n \mid a$ and $n \mid b$, and
2. for all $m \in \mathbb{N}$, if $m \mid a$ and $m \mid b$ then $m \leqslant n$.

Definition 121 (Greatest common divisor, Dt.: größter gemeinsamer Teiler (ggT))

Let $a, b \in \mathbb{N}$. The unique number $n \in \mathbb{N}$ that exists according to Lem. 120 is called *greatest common divisor* of a and b, and is denoted by $\gcd(a, b)$.

- Conventionally, $\gcd(a, 0) = \gcd(0, a) := a$, since 0 is divisible by all integers.

Definition 122 (Relatively prime, Dt.: teilerfremd, relativ prim)

The numbers $a, b \in \mathbb{N}$ are *relatively prime*, or *coprime*, if $\gcd(a, b) = 1$.

Greatest Common Divisor

Lemma 120

Let $a, b \in \mathbb{N}$. Then there exists a unique $n \in \mathbb{N}$ such that

1. $n \mid a$ and $n \mid b$, and
2. for all $m \in \mathbb{N}$, if $m \mid a$ and $m \mid b$ then $m \leqslant n$.

Definition 121 (Greatest common divisor, Dt.: größter gemeinsamer Teiler (ggT))

Let $a, b \in \mathbb{N}$. The unique number $n \in \mathbb{N}$ that exists according to Lem. 120 is called *greatest common divisor* of a and b, and is denoted by $\gcd(a, b)$.

- Conventionally, $\gcd(a, 0) = \gcd(0, a) := a$, since 0 is divisible by all integers.

Definition 122 (Relatively prime, Dt.: teilerfremd, relativ prim)

The numbers $a, b \in \mathbb{N}$ are *relatively prime*, or *coprime*, if $\gcd(a, b) = 1$.

Definition 123 (Pairwise relatively prime)

A set S of natural numbers is called *pairwise relatively prime* (or *pairwise coprime* or *mutually coprime*) if all pairs of numbers a and b in S, with $a \neq b$, are relatively prime.

Lemma 124 (Bézout's Identity)

Let $a, b \in \mathbb{N}$. Then there exist $x, y \in \mathbb{Z}$ such that $\gcd(a, b) = a \cdot x + b \cdot y$. Conversely, the smallest positive number $a \cdot x + b \cdot y$, for all $x, y \in \mathbb{Z}$, equals $\gcd(a, b)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 124 (Bézout's Identity)

Let $a, b \in \mathbb{N}$. Then there exist $x, y \in \mathbb{Z}$ such that $\gcd(a, b) = a \cdot x + b \cdot y$. Conversely, the smallest positive number $a \cdot x + b \cdot y$, for all $x, y \in \mathbb{Z}$, equals $\gcd(a, b)$.

- That is, $\gcd(a, b) = \min\left(\mathbb{N} \cap \{a \cdot x + b \cdot y : \; x, y \in \mathbb{Z}\}\right)$.

Lemma 124 (Bézout's Identity)

Let $a, b \in \mathbb{N}$. Then there exist $x, y \in \mathbb{Z}$ such that $\gcd(a, b) = a \cdot x + b \cdot y$. Conversely, the smallest positive number $a \cdot x + b \cdot y$, for all $x, y \in \mathbb{Z}$, equals $\gcd(a, b)$.

- That is, $\gcd(a, b) = \min\left(\mathbb{N} \cap \{a \cdot x + b \cdot y : \ x, y \in \mathbb{Z}\}\right)$.
- For $a, b, d \in \mathbb{Z}$ given, the identity $d = a \cdot x + b \cdot y$ over $\mathbb{Z} \times \mathbb{Z}$ is called a *linear Diophantine equation* in x and y.
- Lemma 124 was first stated by Étienne Bézout (1730–1783), and numbers $x, y \in \mathbb{Z}$ with $\gcd(a, b) = a \cdot x + b \cdot y$ are called Bézout numbers.

Lemma 124 (Bézout's Identity)

Let $a, b \in \mathbb{N}$. Then there exist $x, y \in \mathbb{Z}$ such that $\gcd(a, b) = a \cdot x + b \cdot y$. Conversely, the smallest positive number $a \cdot x + b \cdot y$, for all $x, y \in \mathbb{Z}$, equals $\gcd(a, b)$.

- That is, $\gcd(a, b) = \min\left(\mathbb{N} \cap \{a \cdot x + b \cdot y : \; x, y \in \mathbb{Z}\}\right)$.
- For $a, b, d \in \mathbb{Z}$ given, the identity $d = a \cdot x + b \cdot y$ over $\mathbb{Z} \times \mathbb{Z}$ is called a *linear Diophantine equation* in x and y.
- Lemma 124 was first stated by Étienne Bézout (1730–1783), and numbers $x, y \in \mathbb{Z}$ with $\gcd(a, b) = a \cdot x + b \cdot y$ are called Bézout numbers.
- Note: Bézout numbers are not unique! For instance, $\gcd(10, 15) = 5$, and $10x + 15y = 5$ has the solutions $x := -1$ and $y := 1$, and $x := 2$ and $y := -1$.

Lemma 124 (Bézout's Identity)

Let $a, b \in \mathbb{N}$. Then there exist $x, y \in \mathbb{Z}$ such that $\gcd(a, b) = a \cdot x + b \cdot y$. Conversely, the smallest positive number $a \cdot x + b \cdot y$, for all $x, y \in \mathbb{Z}$, equals $\gcd(a, b)$.

- That is, $\gcd(a, b) = \min\left(\mathbb{N} \cap \{a \cdot x + b \cdot y : \ x, y \in \mathbb{Z}\}\right)$.
- For $a, b, d \in \mathbb{Z}$ given, the identity $d = a \cdot x + b \cdot y$ over $\mathbb{Z} \times \mathbb{Z}$ is called a *linear Diophantine equation* in x and y.
- Lemma 124 was first stated by Étienne Bézout (1730–1783), and numbers $x, y \in \mathbb{Z}$ with $\gcd(a, b) = a \cdot x + b \cdot y$ are called Bézout numbers.
- Note: Bézout numbers are not unique! For instance, $\gcd(10, 15) = 5$, and $10x + 15y = 5$ has the solutions $x := -1$ and $y := 1$, and $x := 2$ and $y := -1$.

Corollary 125

The numbers $a, b \in \mathbb{N}$ are relatively prime if and only if the linear Diophantine equation $a \cdot x + b \cdot y = 1$ has a solution, i.e., if and only if there exist $x, y \in \mathbb{Z}$ such that $a \cdot x + b \cdot y = 1$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Euclidean Algorithm for GCD Computation

Theorem 126 (Euclidean Algorithm)

The following algorithm computes $\gcd(a, b)$ for $a, b \in \mathbb{N}_0$ with $a > b$.

function $\gcd(a, b)$
precondition: $a, b \in \mathbb{N}_0$ with $a > b$.
postcondition: $t = \gcd(a, b)$
  while $b > 0$ **do**
    $t \leftarrow b$
    $b \leftarrow a \bmod b$
    $a \leftarrow t$
  end while
  $t \leftarrow a$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Euclidean Algorithm for GCD Computation: Sample Run

function gcd(a, b)
precondition: $a, b \in \mathbb{N}_0$ with $a > b$.
postcondition: $t = \text{gcd}(a, b)$

```
while b > 0 do
    t ← b
    b ← a mod b
    a ← t
end while
t ← a
```

- We want to compute the gcd of 78 and 99. Hence, $b := 78$ and $a := 99 = 1 \cdot 78 + 21$.

Euclidean Algorithm for GCD Computation: Sample Run

function gcd(a, b)
precondition: $a, b \in \mathbb{N}_0$ with $a > b$.
postcondition: $t = \text{gcd}(a, b)$

```
while b > 0 do
    t ← b
    b ← a mod b
    a ← t
end while
t ← a
```

- We want to compute the gcd of 78 and 99. Hence, $b := 78$ and $a := 99 = 1 \cdot 78 + 21$. We get after different passes through the loop:

$$\text{after 1st pass:} \quad t = 78, \quad b = 21, \quad a = 78 = 3 \cdot 21 + 15$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Euclidean Algorithm for GCD Computation: Sample Run

function gcd(a, b)
precondition: $a, b \in \mathbb{N}_0$ with $a > b$.
postcondition: $t = \text{gcd}(a, b)$

```
while b > 0 do
    t ← b
    b ← a mod b
    a ← t
end while
t ← a
```

- We want to compute the gcd of 78 and 99. Hence, $b := 78$ and $a := 99 = 1 \cdot 78 + 21$. We get after different passes through the loop:

$$\begin{array}{llll} \text{after 1st pass:} & t = 78, & b = 21, & a = 78 = 3 \cdot 21 + 15 \\ \text{after 2nd pass:} & t = 21, & b = 15, & a = 21 = 1 \cdot 15 + 6 \end{array}$$

Euclidean Algorithm for GCD Computation: Sample Run

function gcd(a, b)
precondition: $a, b \in \mathbb{N}_0$ with $a > b$.
postcondition: $t = \text{gcd}(a, b)$

```
while b > 0 do
    t ← b
    b ← a mod b
    a ← t
end while
t ← a
```

- We want to compute the gcd of 78 and 99. Hence, $b := 78$ and $a := 99 = 1 \cdot 78 + 21$. We get after different passes through the loop:

$$
\begin{array}{llll}
\text{after 1st pass:} & t = 78, & b = 21, & a = 78 = 3 \cdot 21 + 15 \\
\text{after 2nd pass:} & t = 21, & b = 15, & a = 21 = 1 \cdot 15 + 6 \\
\text{after 3rd pass:} & t = 15, & b = 6, & a = 15 = 2 \cdot 6 + 3
\end{array}
$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Euclidean Algorithm for GCD Computation: Sample Run

function gcd(a, b)
precondition: $a, b \in \mathbb{N}_0$ with $a > b$.
postcondition: $t = \text{gcd}(a, b)$

```
while b > 0 do
    t ← b
    b ← a mod b
    a ← t
end while
t ← a
```

- We want to compute the gcd of 78 and 99. Hence, $b := 78$ and $a := 99 = 1 \cdot 78 + 21$. We get after different passes through the loop:

after 1st pass:	$t = 78$,	$b = 21$,	$a = 78 = 3 \cdot 21 + 15$
after 2nd pass:	$t = 21$,	$b = 15$,	$a = 21 = 1 \cdot 15 + 6$
after 3rd pass:	$t = 15$,	$b = 6$,	$a = 15 = 2 \cdot 6 + 3$
after 4th pass:	$t = 6$,	$b = 3$,	$a = 6 = 2 \cdot 3 + 0$

Euclidean Algorithm for GCD Computation: Sample Run

function gcd(a, b)
precondition: $a, b \in \mathbb{N}_0$ with $a > b$.
postcondition: $t = \text{gcd}(a, b)$
  while $b > 0$ **do**
    $t \leftarrow b$
    $b \leftarrow a \bmod b$
    $a \leftarrow t$
  end while
  $t \leftarrow a$

- We want to compute the gcd of 78 and 99. Hence, $b := 78$ and $a := 99 = 1 \cdot 78 + 21$. We get after different passes through the loop:

after 1st pass:	$t = 78$,	$b = 21$,	$a = 78 = 3 \cdot 21 + 15$
after 2nd pass:	$t = 21$,	$b = 15$,	$a = 21 = 1 \cdot 15 + 6$
after 3rd pass:	$t = 15$,	$b = 6$,	$a = 15 = 2 \cdot 6 + 3$
after 4th pass:	$t = 6$,	$b = 3$,	$a = 6 = 2 \cdot 3 + 0$
after 5th pass:	$t = 3$,	$b = 0$,	$a = 3$

- Hence, $t = 3 = \text{gcd}(78, 99)$.

Does $(\mathbb{Z}_m, +_m, \cdot_m)$ Form a Field?

Theorem 127

Let $m \in \mathbb{N}$ with $m \geqslant 2$. An element $[a]_m \in \mathbb{Z}_m$ has a multiplicative inverse if and only if a is relatively prime to m.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 127

Let $m \in \mathbb{N}$ with $m \geqslant 2$. An element $[a]_m \in \mathbb{Z}_m$ has a multiplicative inverse if and only if a is relatively prime to m.

Corollary 128

Let $m \in \mathbb{N}$ with $m \geqslant 2$. The ring $(\mathbb{Z}_m, +_m, \cdot_m)$ forms a (finite) field if and only if m is prime.

Does $(\mathbb{Z}_m, +_m, \cdot_m)$ Form a Field?

Theorem 127

Let $m \in \mathbb{N}$ with $m \geqslant 2$. An element $[a]_m \in \mathbb{Z}_m$ has a multiplicative inverse if and only if a is relatively prime to m.

Corollary 128

Let $m \in \mathbb{N}$ with $m \geqslant 2$. The ring $(\mathbb{Z}_m, +_m, \cdot_m)$ forms a (finite) field if and only if m is prime.

- If m is not prime then $(\mathbb{Z}_m, +_m, \cdot_m)$ may contain non-trivial zero divisors.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Does $(\mathbb{Z}_m, +_m, \cdot_m)$ Form a Field?

Theorem 127

Let $m \in \mathbb{N}$ with $m \geqslant 2$. An element $[a]_m \in \mathbb{Z}_m$ has a multiplicative inverse if and only if a is relatively prime to m.

Corollary 128

Let $m \in \mathbb{N}$ with $m \geqslant 2$. The ring $(\mathbb{Z}_m, +_m, \cdot_m)$ forms a (finite) field if and only if m is prime.

- If m is not prime then $(\mathbb{Z}_m, +_m, \cdot_m)$ may contain non-trivial zero divisors.

Lemma 129

Let $m \in \mathbb{N}$ with $m \geqslant 2$ and $[a]_m \in \mathbb{Z}_m$ such that m and a are relatively prime. Let $x, y \in \mathbb{Z}$ such that $a \cdot x + m \cdot y = 1$. Then $[a]_m \cdot_m [x]_m = [1]_m$, i.e., $[x]_m$ is the multiplicative inverse element for $[a]_m$.

Euclidean Algorithm Revisited

- Recursive formulation of the Euclidean Algorithm.

function gcd_recursive(a, b)
precondition: $a, b \in \mathbb{N}$ with $a > b$.
  if (a mod b) $= 0$ **then**
    return b
  else
    return gcd_recursive(b, a mod b)
  end if

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Extended Euclidean Algorithm for GCD Computation

Theorem 130 (Extended Euclidean Algorithm)

The following algorithm computes $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$ for $a, b \in \mathbb{N}_0$ with $a > b$.

function gcd_extended(a, b)
precondition: $a, b \in \mathbb{N}_0$ with $a > b$.
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$

```
if (a mod b) = 0 then
    return (b, 0, 1)
else
    (d, x, y) ← gcd_extended(b, a mod b)
    return (d, y, x − y · (a div b))
end if
```

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Extended Euclidean Algorithm for GCD Computation: Sample Run

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

Extended Euclidean Algorithm for GCD Computation: Sample Run

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	$a \bmod b$	d	x	y
99	78	1	21			

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	$a \bmod b$	d	x	y
99	78	1	21			
78	21	3	15			

Extended Euclidean Algorithm for GCD Computation: Sample Run

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	a mod b	d	x	y
99	78	1	21			
78	21	3	15			
21	15	1	6			

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	a mod b	d	x	y
99	78	1	21			
78	21	3	15			
21	15	1	6			
15	6	2	3			

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Extended Euclidean Algorithm for GCD Computation: Sample Run

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended($b, a \bmod b$)
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	a mod b	d	x	y
99	78	1	21			
78	21	3	15			
21	15	1	6			
15	6	2	3			
6	3	–	0	3	0	1

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	$a \bmod b$	d	x	y
99	78	1	21			
78	21	3	15			
21	15	1	6			
15	6	2	3	3	1	-2
6	3	–	0	3	0	1

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Extended Euclidean Algorithm for GCD Computation: Sample Run

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	$a \bmod b$	d	x	y
99	78	1	21			
78	21	3	15			
21	15	1	6	3	−2	3
15	6	2	3	3	1	−2
6	3	–	0	3	0	1

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Extended Euclidean Algorithm for GCD Computation: Sample Run

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	$a \bmod b$	d	x	y
99	78	1	21			
78	21	3	15	3	3	−11
21	15	1	6	3	−2	3
15	6	2	3	3	1	−2
6	3	–	0	3	0	1

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Extended Euclidean Algorithm for GCD Computation: Sample Run

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended($b, a \bmod b$)
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	$a \bmod b$	d	x	y
99	78	1	21	3	−11	14
78	21	3	15	3	3	−11
21	15	1	6	3	−2	3
15	6	2	3	3	1	−2
6	3	–	0	3	0	1

Extended Euclidean Algorithm for GCD Computation: Sample Run

function gcd_extended(a, b)
postcondition: $(d, x, y) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$ such that $\gcd(a, b) = d = a \cdot x + b \cdot y$
  if $(a \bmod b) = 0$ **then**
    return $(b, 0, 1)$
  else
    $(d, x, y) \leftarrow$ gcd_extended$(b, a \bmod b)$
    return $(d, y, x - y \cdot (a \text{ div } b))$
  end if

- We want to compute $x, y \in \mathbb{Z}$ and $d \in \mathbb{N}$ such that $\gcd(99, 78) = d = 99x + 78y$.

a	b	a div b	$a \bmod b$	d	x	y
99	78	1	21	3	−11	14
78	21	3	15	3	3	−11
21	15	1	6	3	−2	3
15	6	2	3	3	1	−2
6	3	–	0	3	0	1

- Hence, $\gcd(99, 78) = -11 \cdot 99 + 14 \cdot 78 = -1089 + 1092 = 3$.

Chinese Remainder Theorem

- Old Chinese folk tale: A Chinese Emperor used to count his army after a battle by ordering them to form groups of different sizes:

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Chinese Remainder Theorem

- Old Chinese folk tale: A Chinese Emperor used to count his army after a battle by ordering them to form groups of different sizes:
  1. The soldiers should form groups of 3 and report back the number of soldiers that did not end up in a group consisting of 3 soldiers.

Chinese Remainder Theorem

- Old Chinese folk tale: A Chinese Emperor used to count his army after a battle by ordering them to form groups of different sizes:
  1. The soldiers should form groups of 3 and report back the number of soldiers that did not end up in a group consisting of 3 soldiers.
  2. Then the soldiers should form groups of 5 and report back the number of soldiers that did not end up in a group consisting of 5 soldiers.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Chinese Remainder Theorem

- Old Chinese folk tale: A Chinese Emperor used to count his army after a battle by ordering them to form groups of different sizes:
  1. The soldiers should form groups of 3 and report back the number of soldiers that did not end up in a group consisting of 3 soldiers.
  2. Then the soldiers should form groups of 5 and report back the number of soldiers that did not end up in a group consisting of 5 soldiers.
  3. Then the soldiers should form groups of 7 and report back the number of soldiers that could not join a group consisting of 7 soldiers.

Chinese Remainder Theorem

- Old Chinese folk tale: A Chinese Emperor used to count his army after a battle by ordering them to form groups of different sizes:
  1. The soldiers should form groups of 3 and report back the number of soldiers that did not end up in a group consisting of 3 soldiers.
  2. Then the soldiers should form groups of 5 and report back the number of soldiers that did not end up in a group consisting of 5 soldiers.
  3. Then the soldiers should form groups of 7 and report back the number of soldiers that could not join a group consisting of 7 soldiers.
  4. Then the soldiers should form groups of 11 and report back the number of soldiers that did not end up in a group consisting of 11 soldiers.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Chinese Remainder Theorem

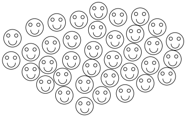
- Old Chinese folk tale: A Chinese Emperor used to count his army after a battle by ordering them to form groups of different sizes:
  1. The soldiers should form groups of 3 and report back the number of soldiers that did not end up in a group consisting of 3 soldiers.
  2. Then the soldiers should form groups of 5 and report back the number of soldiers that did not end up in a group consisting of 5 soldiers.
  3. Then the soldiers should form groups of 7 and report back the number of soldiers that could not join a group consisting of 7 soldiers.
  4. Then the soldiers should form groups of 11 and report back the number of soldiers that did not end up in a group consisting of 11 soldiers.
  5. ...
- Based on this information he was able to figure out the number n of soldiers in his army.

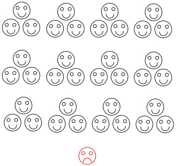
Chinese Remainder Theorem

- Old Chinese folk tale: A Chinese Emperor used to count his army after a battle by ordering them to form groups of different sizes:
  1. The soldiers should form groups of 3 and report back the number of soldiers that did not end up in a group consisting of 3 soldiers.
  2. Then the soldiers should form groups of 5 and report back the number of soldiers that did not end up in a group consisting of 5 soldiers.
  3. Then the soldiers should form groups of 7 and report back the number of soldiers that could not join a group consisting of 7 soldiers.
  4. Then the soldiers should form groups of 11 and report back the number of soldiers that did not end up in a group consisting of 11 soldiers.
  5. ...
- Based on this information he was able to figure out the number n of soldiers in his army.
- Indeed, a mathematical solution was provided by the Chinese mathematician Sun Tzu sometime in the third to fifth century, and republished by Qin Jiushao in 1247!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

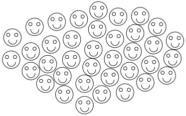
Chinese Remainder Theorem

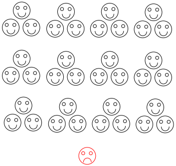




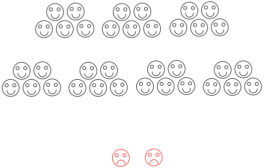
$n \bmod 3 = 1$

Chinese Remainder Theorem



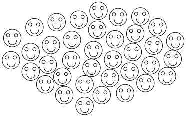


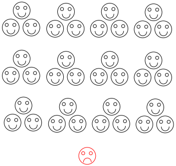
$n \bmod 3 = 1$



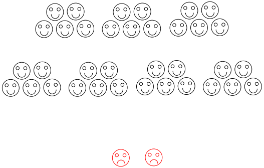
$n \bmod 5 = 2$

Chinese Remainder Theorem

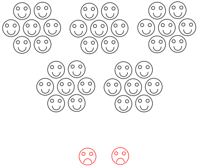




$n \bmod 3 = 1$



$n \bmod 5 = 2$



$n \bmod 7 = 2$

Theorem 131 (Chinese Remainder Theorem, Dt.: Chinesischer Restsatz)

If, for some $k \in \mathbb{N}$, the numbers $m_1, m_2, \cdots, m_k \in \mathbb{N}$ are pairwise relatively prime, then the following system of simultaneous congruences has an integer solution b for all $a_1, a_2, \ldots, a_k \in \mathbb{Z}$ given:

$$\left.\begin{array}{l} b \equiv_{m_1} a_1 \\ b \equiv_{m_2} a_2 \\ \quad\vdots \\ b \equiv_{m_k} a_k \end{array}\right\} (*)$$

Furthermore, all solutions of $(*)$ are congruent modulo $m := \prod_{i=1}^{k} m_i$. That is, the solution is unique if constrained to $\{1, 2, \ldots, m\}$.

Constructive Proof of Chinese Remainder Theorem 131

Proof : We show the existence of an integer solution. Consider $i \in \mathbb{N}$ with $1 \leqslant i \leqslant k$. Since $m_1, m_2, \cdots, m_k$ are pairwise relatively prime, $\gcd(\frac{m}{m_i}, m_i) = 1$. Using the extended Euclidean algorithm (Thm. 130), we can find integers x_i and y_i such that

$$x_i \cdot m_i + y_i \cdot \frac{m}{m_i} = 1. \qquad (\star)$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Constructive Proof of Chinese Remainder Theorem 131

Proof: We show the existence of an integer solution. Consider $i \in \mathbb{N}$ with $1 \leqslant i \leqslant k$. Since $m_1, m_2, \cdots, m_k$ are pairwise relatively prime, $\gcd(\frac{m}{m_i}, m_i) = 1$. Using the extended Euclidean algorithm (Thm. 130), we can find integers x_i and y_i such that

$$x_i \cdot m_i + y_i \cdot \frac{m}{m_i} = 1. \qquad (\star)$$

Let $b_i := y_i \cdot \frac{m}{m_i}$. Equation $(\star)$ guarantees that the remainder of b_i when divided by m_i is 1. On the other hand, for $j \neq i$ every m_j divides b_i evenly. Thus,

$$b_i \equiv_{m_i} 1 \quad \text{and} \quad b_i \equiv_{m_j} 0 \quad \text{for all } j \text{ with } j \neq i \text{ and } 1 \leqslant j \leqslant k.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proof : We show the existence of an integer solution. Consider $i \in \mathbb{N}$ with $1 \leqslant i \leqslant k$. Since $m_1, m_2, \cdots, m_k$ are pairwise relatively prime, $\gcd(\frac{m}{m_i}, m_i) = 1$. Using the extended Euclidean algorithm (Thm. 130), we can find integers x_i and y_i such that

$$x_i \cdot m_i + y_i \cdot \frac{m}{m_i} = 1. \qquad (\star)$$

Let $b_i := y_i \cdot \frac{m}{m_i}$. Equation $(\star)$ guarantees that the remainder of b_i when divided by m_i is 1. On the other hand, for $j \neq i$ every m_j divides b_i evenly. Thus,

$$b_i \equiv_{m_i} 1 \quad \text{and} \quad b_i \equiv_{m_j} 0 \quad \text{for all } j \text{ with } j \neq i \text{ and } 1 \leqslant j \leqslant k.$$

Since congruences respect multiplication, we get

$$a_i \cdot b_i \equiv_{m_i} a_i \quad \text{and} \quad a_i \cdot b_i \equiv_{m_j} 0 \quad \text{for all } j \text{ with } j \neq i \text{ and } 1 \leqslant j \leqslant k.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proof : We show the existence of an integer solution. Consider $i \in \mathbb{N}$ with $1 \leqslant i \leqslant k$. Since $m_1, m_2, \cdots, m_k$ are pairwise relatively prime, $\gcd(\frac{m}{m_i}, m_i) = 1$. Using the extended Euclidean algorithm (Thm. 130), we can find integers x_i and y_i such that

$$x_i \cdot m_i + y_i \cdot \frac{m}{m_i} = 1. \qquad (\star)$$

Let $b_i := y_i \cdot \frac{m}{m_i}$. Equation $(\star)$ guarantees that the remainder of b_i when divided by m_i is 1. On the other hand, for $j \neq i$ every m_j divides b_i evenly. Thus,

$$b_i \equiv_{m_i} 1 \quad \text{and} \quad b_i \equiv_{m_j} 0 \quad \text{for all } j \text{ with } j \neq i \text{ and } 1 \leqslant j \leqslant k.$$

Since congruences respect multiplication, we get

$$a_i \cdot b_i \equiv_{m_i} a_i \quad \text{and} \quad a_i \cdot b_i \equiv_{m_j} 0 \quad \text{for all } j \text{ with } j \neq i \text{ and } 1 \leqslant j \leqslant k.$$

Thus, one solution of the simultaneous congruences is given by

$$b := \sum_{i=1}^{k} a_i \cdot b_i.$$

□

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Helping the Emperor

- The Emperor collected the following information:
  - When the soldiers formed groups of 3, one soldier was left out.
  - When the soldiers formed groups of 5, two soldiers were left out.
  - When the soldiers formed groups of 7, again two soldiers were left out.

Helping the Emperor

- The Emperor collected the following information:
  - When the soldiers formed groups of 3, one soldier was left out.
  - When the soldiers formed groups of 5, two soldiers were left out.
  - When the soldiers formed groups of 7, again two soldiers were left out.
- That is, since $a_1 = 1, a_2 = 2, a_3 = 2$ and $m_1 = 3, m_2 = 5, m_3 = 7$ and $m = 3 \cdot 5 \cdot 7 = 105$:

$$n \equiv_3 1 \qquad n \equiv_5 2 \qquad n \equiv_7 2$$

Helping the Emperor

- The Emperor collected the following information:
  - When the soldiers formed groups of 3, one soldier was left out.
  - When the soldiers formed groups of 5, two soldiers were left out.
  - When the soldiers formed groups of 7, again two soldiers were left out.
- That is, since $a_1 = 1, a_2 = 2, a_3 = 2$ and $m_1 = 3, m_2 = 5, m_3 = 7$ and $m = 3 \cdot 5 \cdot 7 = 105$:

$$n \equiv_3 1 \qquad n \equiv_5 2 \qquad n \equiv_7 2$$

- Hence, we are to find $x_1, x_2, x_3, y_1, y_2, y_3 \in \mathbb{Z}$ such that

$$3x_1 + 35y_1 = 1 \qquad 5x_2 + 21y_2 = 1 \qquad 7x_3 + 15y_3 = 1.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Helping the Emperor

- The Emperor collected the following information:
  - When the soldiers formed groups of 3, one soldier was left out.
  - When the soldiers formed groups of 5, two soldiers were left out.
  - When the soldiers formed groups of 7, again two soldiers were left out.
- That is, since $a_1 = 1, a_2 = 2, a_3 = 2$ and $m_1 = 3, m_2 = 5, m_3 = 7$ and $m = 3 \cdot 5 \cdot 7 = 105$:

$$n \equiv_3 1 \qquad n \equiv_5 2 \qquad n \equiv_7 2$$

- Hence, we are to find $x_1, x_2, x_3, y_1, y_2, y_3 \in \mathbb{Z}$ such that

$$3x_1 + 35y_1 = 1 \qquad 5x_2 + 21y_2 = 1 \qquad 7x_3 + 15y_3 = 1.$$

- We have $x_1 := 12,\ y_1 := -1,\ x_2 := -4,\ y_2 := 1,\ x_3 := -2,\ y_3 := 1$ and, thus,

$$n = (35 \cdot (-1) \cdot 1 + 21 \cdot 1 \cdot 2 + 15 \cdot 1 \cdot 2) \bmod 105 = 37 \bmod 105 = 37.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Secret Sharing

- Secret sharing refers to the distribution of information related to a secret (e.g., a number) among a group of receivers such that the secret can only be reconstructed if all or, at least, a large percentage of the receivers cooperate.
- Ideally, the information received by one individual receiver shall be of no (or very little) help for the receiver to obtain the secret without the help of the others.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Secret Sharing

- Secret sharing refers to the distribution of information related to a secret (e.g., a number) among a group of receivers such that the secret can only be reconstructed if all or, at least, a large percentage of the receivers cooperate.
- Ideally, the information received by one individual receiver shall be of no (or very little) help for the receiver to obtain the secret without the help of the others.
- A secret sharing scheme is called a (t, n) threshold scheme, or t-out-of-n scheme, if at least t of the n receivers have to cooperate. (Of course, $t \leqslant n$.)
- Typically, t is large relative to n but not identical to n.

Real-World Application: Secret Sharing

- Secret sharing refers to the distribution of information related to a secret (e.g., a number) among a group of receivers such that the secret can only be reconstructed if all or, at least, a large percentage of the receivers cooperate.
- Ideally, the information received by one individual receiver shall be of no (or very little) help for the receiver to obtain the secret without the help of the others.
- A secret sharing scheme is called a (t, n) threshold scheme, or t-out-of-n scheme, if at least t of the n receivers have to cooperate. (Of course, $t \leqslant n$.)
- Typically, t is large relative to n but not identical to n.
- Several different variants of schemes for secret sharing are used in practice.
- At least two published schemes rely on the Chinese Remainder Theorem 131.
- We sketch the very basic idea of a scheme based on the Chinese Remainder Theorem 131. (In our simple scheme we have $t := n$.)

Real-World Application: Secret Sharing

- Suppose that the number 1234 is the secret b to be shared by five receivers.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Secret Sharing

- Suppose that the number 1234 is the secret b to be shared by five receivers.
- We choose

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

- Note that

$$m := \prod_{i=1}^{5} m_i = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310 > 1234.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Secret Sharing

- Suppose that the number 1234 is the secret b to be shared by five receivers.
- We choose

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

- Note that

$$m := \prod_{i=1}^{5} m_i = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310 > 1234.$$

- Now consider $a_i := 1234 \bmod m_i$, for $1 \leqslant i \leqslant 5$. This gives us the numbers

$$a_1 = 0, \qquad a_2 = 1, \qquad a_3 = 4, \qquad a_4 = 2, \qquad a_5 = 2.$$

Real-World Application: Secret Sharing

- Suppose that the number 1234 is the secret b to be shared by five receivers.
- We choose

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

- Note that

$$m := \prod_{i=1}^{5} m_i = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310 > 1234.$$

- Now consider $a_i := 1234 \bmod m_i$, for $1 \leqslant i \leqslant 5$. This gives us the numbers

$$a_1 = 0, \qquad a_2 = 1, \qquad a_3 = 4, \qquad a_4 = 2, \qquad a_5 = 2.$$

- The numbers m_i and a_i are passed to the i-th receiver.
- Note that each individual receiver has gained little information about the secret b.

Real-World Application: Secret Sharing

- Suppose that the number 1234 is the secret b to be shared by five receivers.
- We choose

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

- Note that

$$m := \prod_{i=1}^{5} m_i = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310 > 1234.$$

- Now consider $a_i := 1234 \bmod m_i$, for $1 \leqslant i \leqslant 5$. This gives us the numbers

$$a_1 = 0, \qquad a_2 = 1, \qquad a_3 = 4, \qquad a_4 = 2, \qquad a_5 = 2.$$

- The numbers m_i and a_i are passed to the i-th receiver.
- Note that each individual receiver has gained little information about the secret b.
- Rather, in our simple approach, all five receivers need to cooperate in order to recover b: They have to solve the following set of five congruences:

$$b \equiv_2 0 \qquad b \equiv_3 1 \qquad b \equiv_5 4 \qquad b \equiv_7 2 \qquad b \equiv_{11} 2$$

Real-World Application: Secret Sharing

- The five receivers have to solve the following set of five congruences:

$$b \equiv_2 0 \qquad b \equiv_3 1 \qquad b \equiv_5 4 \qquad b \equiv_7 2 \qquad b \equiv_{11} 2$$

- Since $a_1 = 0$, we need to solve only four congruences and get the following four Diophantine equations.

$$3x_2 + 770y_2 = 1 \qquad 5x_3 + 462y_3 = 1 \qquad 7x_4 + 330y_4 = 1 \qquad 11x_5 + 210y_5 = 1$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Secret Sharing

- The five receivers have to solve the following set of five congruences:

$$b \equiv_2 0 \qquad b \equiv_3 1 \qquad b \equiv_5 4 \qquad b \equiv_7 2 \qquad b \equiv_{11} 2$$

- Since $a_1 = 0$, we need to solve only four congruences and get the following four Diophantine equations.

$$3x_2 + 770y_2 = 1 \qquad 5x_3 + 462y_3 = 1 \qquad 7x_4 + 330y_4 = 1 \qquad 11x_5 + 210y_5 = 1$$

- Solving these equations yields the following solutions:

$$x_2 := 257, y_2 := -1 \quad x_3 := 185, y_3 := -2 \quad x_4 := -47, y_4 := 1 \quad x_5 := -19, y_5 := 1$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Secret Sharing

- The five receivers have to solve the following set of five congruences:

$$b \equiv_2 0 \qquad b \equiv_3 1 \qquad b \equiv_5 4 \qquad b \equiv_7 2 \qquad b \equiv_{11} 2$$

- Since $a_1 = 0$, we need to solve only four congruences and get the following four Diophantine equations.

$$3x_2 + 770y_2 = 1 \qquad 5x_3 + 462y_3 = 1 \qquad 7x_4 + 330y_4 = 1 \qquad 11x_5 + 210y_5 = 1$$

- Solving these equations yields the following solutions:

$$x_2 := 257, y_2 := -1 \quad x_3 := 185, y_3 := -2 \quad x_4 := -47, y_4 := 1 \quad x_5 := -19, y_5 := 1$$

- Hence, the secret sought is recovered as

$$b = (-1) \cdot 770 \cdot 1 + (-2) \cdot 462 \cdot 4 + 1 \cdot 330 \cdot 2 + 1 \cdot 210 \cdot 2 = -3386 \equiv_{2310} 1234.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Arithmetic with Large Integers

- Standard integer arithmetic cannot handle arbitrarily large integers.

Real-World Application: Arithmetic with Large Integers

- Standard integer arithmetic cannot handle arbitrarily large integers.
- One way to carry out integer arithmetic with large integers is to apply modulo arithmetic and the Chinese Remainder Theorem 131:
  1. Select k modules $m_1, m_2, \ldots, m_k \in \mathbb{N} \setminus \{1\}$ which are relatively prime, for some $k \in \mathbb{N}$.

Real-World Application: Arithmetic with Large Integers

- Standard integer arithmetic cannot handle arbitrarily large integers.
- One way to carry out integer arithmetic with large integers is to apply modulo arithmetic and the Chinese Remainder Theorem 131:
  1. Select k modules $m_1, m_2, \ldots, m_k \in \mathbb{N} \setminus \{1\}$ which are relatively prime, for some $k \in \mathbb{N}$.
  2. Let $m := m_1 \cdot m_2 \cdot \ldots \cdot m_k$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Arithmetic with Large Integers

- Standard integer arithmetic cannot handle arbitrarily large integers.
- One way to carry out integer arithmetic with large integers is to apply modulo arithmetic and the Chinese Remainder Theorem 131:
  1. Select k modules $m_1, m_2, \ldots, m_k \in \mathbb{N} \backslash \{1\}$ which are relatively prime, for some $k \in \mathbb{N}$.
  2. Let $m := m_1 \cdot m_2 \cdot \ldots \cdot m_k$.
  3. Represent an integer $n < m$ by its k remainders $n_1, n_2, \ldots, n_k$ upon division by $m_1, m_2, \ldots, m_k$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Arithmetic with Large Integers

- Standard integer arithmetic cannot handle arbitrarily large integers.
- One way to carry out integer arithmetic with large integers is to apply modulo arithmetic and the Chinese Remainder Theorem 131:
  1. Select k modules $m_1, m_2, \ldots, m_k \in \mathbb{N} \setminus \{1\}$ which are relatively prime, for some $k \in \mathbb{N}$.
  2. Let $m := m_1 \cdot m_2 \cdot \ldots \cdot m_k$.
  3. Represent an integer $n < m$ by its k remainders $n_1, n_2, \ldots, n_k$ upon division by $m_1, m_2, \ldots, m_k$.
  4. Perform the arithmetic operations of your algorithm on these remainders, with the calculations involving n_i being carried out modulo m_i.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Arithmetic with Large Integers

- Standard integer arithmetic cannot handle arbitrarily large integers.
- One way to carry out integer arithmetic with large integers is to apply modulo arithmetic and the Chinese Remainder Theorem 131:
  1. Select k modules $m_1, m_2, \ldots, m_k \in \mathbb{N} \setminus \{1\}$ which are relatively prime, for some $k \in \mathbb{N}$.
  2. Let $m := m_1 \cdot m_2 \cdot \ldots \cdot m_k$.
  3. Represent an integer $n < m$ by its k remainders $n_1, n_2, \ldots, n_k$ upon division by $m_1, m_2, \ldots, m_k$.
  4. Perform the arithmetic operations of your algorithm on these remainders, with the calculations involving n_i being carried out modulo m_i.
  5. Recover the actual result by applying the Chinese Remainder Theorem 131.

Real-World Application: Arithmetic with Large Integers

- Standard integer arithmetic cannot handle arbitrarily large integers.
- One way to carry out integer arithmetic with large integers is to apply modulo arithmetic and the Chinese Remainder Theorem 131:
  1. Select k modules $m_1, m_2, \ldots, m_k \in \mathbb{N} \setminus \{1\}$ which are relatively prime, for some $k \in \mathbb{N}$.
  2. Let $m := m_1 \cdot m_2 \cdot \ldots \cdot m_k$.
  3. Represent an integer $n < m$ by its k remainders $n_1, n_2, \ldots, n_k$ upon division by $m_1, m_2, \ldots, m_k$.
  4. Perform the arithmetic operations of your algorithm on these remainders, with the calculations involving n_i being carried out modulo m_i.
  5. Recover the actual result by applying the Chinese Remainder Theorem 131.
- This approach works as long as all intermediate results are less than m.
- Advantages:
  - One can use (mostly) standard arithmetic to handle integers larger than those normally handled.
  - One can run the computations for the different remainders in parallel, thus speeding up the computation.

Real-World Application: Arithmetic with Large Integers

- Standard integer arithmetic cannot handle arbitrarily large integers.
- One way to carry out integer arithmetic with large integers is to apply modulo arithmetic and the Chinese Remainder Theorem 131:
  1. Select k modules $m_1, m_2, \dots, m_k \in \mathbb{N}\backslash\{1\}$ which are relatively prime, for some $k \in \mathbb{N}$.
  2. Let $m := m_1 \cdot m_2 \cdot \ldots \cdot m_k$.
  3. Represent an integer $n < m$ by its k remainders $n_1, n_2, \dots, n_k$ upon division by $m_1, m_2, \dots, m_k$.
  4. Perform the arithmetic operations of your algorithm on these remainders, with the calculations involving n_i being carried out modulo m_i.
  5. Recover the actual result by applying the Chinese Remainder Theorem 131.
- This approach works as long as all intermediate results are less than m.
- Advantages:
  - One can use (mostly) standard arithmetic to handle integers larger than those normally handled.
  - One can run the computations for the different remainders in parallel, thus speeding up the computation.
- Standard choices for the modules are numbers of the form $2^i - 1$:
  - One can prove $\gcd(2^i - 1, 2^j - 1) = 2^{\gcd(i,j)} - 1$, which makes it easy to ensure that the modules are relatively prime.

Real-World Application: Arithmetic with Large Integers

- Suppose that we want to limit our arithmetic operations to numbers less than 12.
- We choose the five modules

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

  and remember that $m := m_1 \cdot m_2 \cdot m_3 \cdot m_4 \cdot m_5 = 2310$.
- Hence, we can deal with numbers less than 2310.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Arithmetic with Large Integers

- Suppose that we want to limit our arithmetic operations to numbers less than 12.
- We choose the five modules

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

  and remember that $m := m_1 \cdot m_2 \cdot m_3 \cdot m_4 \cdot m_5 = 2310$.
- Hence, we can deal with numbers less than 2310.
- Recall that $n := 1234$ can be represented by the five remainders $(0, 1, 4, 2, 2)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Arithmetic with Large Integers

- Suppose that we want to limit our arithmetic operations to numbers less than 12.
- We choose the five modules

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

  and remember that $m := m_1 \cdot m_2 \cdot m_3 \cdot m_4 \cdot m_5 = 2310$.
- Hence, we can deal with numbers less than 2310.
- Recall that $n := 1234$ can be represented by the five remainders $(0, 1, 4, 2, 2)$.
- Similarly, 1000 can be represented by the five remainders $(0, 1, 0, 6, 10)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Arithmetic with Large Integers

- Suppose that we want to limit our arithmetic operations to numbers less than 12.
- We choose the five modules

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

and remember that $m := m_1 \cdot m_2 \cdot m_3 \cdot m_4 \cdot m_5 = 2310$.

- Hence, we can deal with numbers less than 2310.
- Recall that $n := 1234$ can be represented by the five remainders $(0, 1, 4, 2, 2)$.
- Similarly, 1000 can be represented by the five remainders $(0, 1, 0, 6, 10)$.
- We get

$$\begin{aligned}(0, 1, 4, 2, 2) + (0, 1, 0, 6, 10) &= (0 \bmod 2, 2 \bmod 3, 4 \bmod 5, 8 \bmod 7, 12 \bmod 11)\\ &= (0, 2, 4, 1, 1).\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Suppose that we want to limit our arithmetic operations to numbers less than 12.
- We choose the five modules

$$m_1 := 2, \qquad m_2 := 3, \qquad m_3 := 5, \qquad m_4 := 7, \qquad m_5 := 11.$$

and remember that $m := m_1 \cdot m_2 \cdot m_3 \cdot m_4 \cdot m_5 = 2310$.

- Hence, we can deal with numbers less than 2310.
- Recall that $n := 1234$ can be represented by the five remainders $(0, 1, 4, 2, 2)$.
- Similarly, 1000 can be represented by the five remainders $(0, 1, 0, 6, 10)$.
- We get

$$\begin{aligned}(0,1,4,2,2) + (0,1,0,6,10) &= (0 \bmod 2, 2 \bmod 3, 4 \bmod 5, 8 \bmod 7, 12 \bmod 11)\\ &= (0,2,4,1,1).\end{aligned}$$

- Thus, $b := 1234 + 1000$ is uniquely determined as the solution of the following set of five congruences:

$$b \equiv_2 0 \qquad b \equiv_3 2 \qquad b \equiv_5 4 \qquad b \equiv_7 1 \qquad b \equiv_{11} 1$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

4 Numbers and Basics of Number Theory

Definition 132 (Rational equivalence)

On $\mathbb{Z} \times \mathbb{N}$ we define the binary relation $\cong_Q$ as follows:

$$(p_1, q_1) \cong_Q (p_2, q_2) \quad :\Leftrightarrow \quad p_1 \cdot q_2 = p_2 \cdot q_1.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 132 (Rational equivalence)

On $\mathbb{Z} \times \mathbb{N}$ we define the binary relation $\cong_Q$ as follows:

$$(p_1, q_1) \cong_Q (p_2, q_2) \quad :\Leftrightarrow \quad p_1 \cdot q_2 = p_2 \cdot q_1.$$

Lemma 133

The relation $\cong_Q$ is an equivalence relation on $\mathbb{Z} \times \mathbb{N}$.

Rational Numbers: $\mathbb{Q}$

Definition 132 (Rational equivalence)

On $\mathbb{Z} \times \mathbb{N}$ we define the binary relation $\cong_Q$ as follows:

$$(p_1, q_1) \cong_Q (p_2, q_2) \quad :\Leftrightarrow \quad p_1 \cdot q_2 = p_2 \cdot q_1 .$$

Lemma 133

The relation $\cong_Q$ is an equivalence relation on $\mathbb{Z} \times \mathbb{N}$.

Definition 134 (Rational numbers)

The *rational numbers* $\mathbb{Q}$ are defined as

$$\mathbb{Q} := \{[(p, q)]_{\cong_Q} : \; p \in \mathbb{Z}, q \in \mathbb{N}\}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 132 (Rational equivalence)

On $\mathbb{Z} \times \mathbb{N}$ we define the binary relation $\cong_Q$ as follows:

$$(p_1, q_1) \cong_Q (p_2, q_2) \quad :\Leftrightarrow \quad p_1 \cdot q_2 = p_2 \cdot q_1.$$

Lemma 133

The relation $\cong_Q$ is an equivalence relation on $\mathbb{Z} \times \mathbb{N}$.

Definition 134 (Rational numbers)

The *rational numbers* $\mathbb{Q}$ are defined as

$$\mathbb{Q} := \{[(p,q)]_{\cong_Q} : \ p \in \mathbb{Z}, q \in \mathbb{N}\}.$$

The *canonical representative* of $[(p,q)]_{\cong_Q}$ is denoted by $\frac{p'}{q'}$, where $p' := p \text{ div } \gcd(|p|, q)$ and $q' := q \text{ div } \gcd(|p|, q)$.

Rational Numbers: $\mathbb{Q}$

- It is easy to define an addition $+_Q$, multiplication $\cdot_Q$ and order $\leqslant_Q$ on $\mathbb{Q}$ that turns $(\mathbb{Q}, +, \cdot)$ into a totally ordered field. E.g.,

$$[(p_1, q_1)]_{\cong_Q} +_Q [(p_2, q_2)]_{\cong_Q} := [(p_1 \cdot q_2 + p_2 \cdot q_1, q_1 \cdot q_2)]_{\cong_Q}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Rational Numbers: $\mathbb{Q}$

- It is easy to define an addition $+_Q$, multiplication $\cdot_Q$ and order $\leqslant_Q$ on $\mathbb{Q}$ that turns $(\mathbb{Q}, +, \cdot)$ into a totally ordered field. E.g.,

$$[(p_1, q_1)]_{\cong_Q} +_Q [(p_2, q_2)]_{\cong_Q} := [(p_1 \cdot q_2 + p_2 \cdot q_1, q_1 \cdot q_2)]_{\cong_Q}$$

- Of course, it is standard to simplify the notation and write

$$\frac{p}{q} \quad \text{instead of} \quad [(p, q)]_{\cong_Q}.$$

But keep in mind that fractions are equivalence classes. Thus,

$$(1, 3) \cong_Q (3, 9) \cong_Q (3000, 9000) \qquad \text{i.e.,} \quad \frac{1}{3} = \frac{3}{9} = \frac{3000}{9000}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- It is easy to define an addition $+_Q$, multiplication $\cdot_Q$ and order $\leqslant_Q$ on $\mathbb{Q}$ that turns $(\mathbb{Q}, +, \cdot)$ into a totally ordered field. E.g.,

$$[(p_1, q_1)]_{\cong_Q} +_Q [(p_2, q_2)]_{\cong_Q} := [(p_1 \cdot q_2 + p_2 \cdot q_1, q_1 \cdot q_2)]_{\cong_Q}$$

- Of course, it is standard to simplify the notation and write

$$\frac{p}{q} \quad \text{instead of} \quad [(p, q)]_{\cong_Q}.$$

But keep in mind that fractions are equivalence classes. Thus,

$$(1, 3) \cong_Q (3, 9) \cong_Q (3000, 9000) \qquad \text{i.e.,} \quad \frac{1}{3} = \frac{3}{9} = \frac{3000}{9000}.$$

- In the sequel we resort to standard knowledge and deal with rational numbers as we learned in school. (However, this could be formalized based on Def. 134!)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 135

The equation $x^2 = 2$ has no solution over $\mathbb{Q}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Properties: $\mathbb{Q}$ Is Not Complete

Theorem 135

The equation $x^2 = 2$ has no solution over $\mathbb{Q}$.

Proof: Suppose that there exists $x \in \mathbb{Q}$ such that $x^2 = 2$. Hence, there exist $p \in \mathbb{Z}$ and $q \in \mathbb{N}$ such that

$$\gcd(|p|, q) = 1 \quad \text{and} \quad 2 = \left(\frac{p}{q}\right)^2 = \frac{p^2}{q^2}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 135

The equation $x^2 = 2$ has no solution over $\mathbb{Q}$.

Proof : Suppose that there exists $x \in \mathbb{Q}$ such that $x^2 = 2$. Hence, there exist $p \in \mathbb{Z}$ and $q \in \mathbb{N}$ such that

$$\gcd(|p|, q) = 1 \quad \text{and} \quad 2 = \left(\frac{p}{q}\right)^2 = \frac{p^2}{q^2}.$$

The second equation is equivalent to $2q^2 = p^2$, implying that $p^2 \equiv_2 0$, and, thus, also $p \equiv_2 0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Properties: $\mathbb{Q}$ Is Not Complete

Theorem 135

The equation $x^2 = 2$ has no solution over $\mathbb{Q}$.

Proof : Suppose that there exists $x \in \mathbb{Q}$ such that $x^2 = 2$. Hence, there exist $p \in \mathbb{Z}$ and $q \in \mathbb{N}$ such that

$$\gcd(|p|, q) = 1 \quad \text{and} \quad 2 = \left(\frac{p}{q}\right)^2 = \frac{p^2}{q^2}.$$

The second equation is equivalent to $2q^2 = p^2$, implying that $p^2 \equiv_2 0$, and, thus, also $p \equiv_2 0$. This in turn implies $q^2 \equiv_2 0$, and, therefore, also $q \equiv_2 0$. We have a contradiction to $\gcd(|p|, q) = 1$. □

Theorem 135

The equation $x^2 = 2$ has no solution over $\mathbb{Q}$.

Proof: Suppose that there exists $x \in \mathbb{Q}$ such that $x^2 = 2$. Hence, there exist $p \in \mathbb{Z}$ and $q \in \mathbb{N}$ such that

$$\gcd(|p|, q) = 1 \quad \text{and} \quad 2 = \left(\frac{p}{q}\right)^2 = \frac{p^2}{q^2}.$$

The second equation is equivalent to $2q^2 = p^2$, implying that $p^2 \equiv_2 0$, and, thus, also $p \equiv_2 0$. This in turn implies $q^2 \equiv_2 0$, and, therefore, also $q \equiv_2 0$. We have a contradiction to $\gcd(|p|, q) = 1$. □

- Hence, $\sqrt{2} \notin \mathbb{Q}$.

Theorem 135

The equation $x^2 = 2$ has no solution over $\mathbb{Q}$.

Proof : Suppose that there exists $x \in \mathbb{Q}$ such that $x^2 = 2$. Hence, there exist $p \in \mathbb{Z}$ and $q \in \mathbb{N}$ such that

$$\gcd(|p|, q) = 1 \quad \text{and} \quad 2 = \left(\frac{p}{q}\right)^2 = \frac{p^2}{q^2}.$$

The second equation is equivalent to $2q^2 = p^2$, implying that $p^2 \equiv_2 0$, and, thus, also $p \equiv_2 0$. This in turn implies $q^2 \equiv_2 0$, and, therefore, also $q \equiv_2 0$. We have a contradiction to $\gcd(|p|, q) = 1$. □

- Hence, $\sqrt{2} \notin \mathbb{Q}$.

Lemma 136

There exists a rational number between any two distinct rational numbers.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Properties: $\mathbb{Q}$ Is Countably Infinite

Theorem 137

$\mathbb{Q}$ is a countably infinite set.

Theorem 137

$\mathbb{Q}$ is a countably infinite set.

Proof by Cantor : Construct a bijection between $\mathbb{N}$ and $\mathbb{Z} \times \mathbb{N}$ (as a "superset" of $\mathbb{Q}$) .

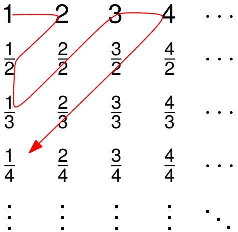
$$\begin{array}{ccccc}
1 & 2 & 3 & 4 & \cdots \\
\frac{1}{2} & \frac{2}{2} & \frac{3}{2} & \frac{4}{2} & \cdots \\
\frac{1}{3} & \frac{2}{3} & \frac{3}{3} & \frac{4}{3} & \cdots \\
\frac{1}{4} & \frac{2}{4} & \frac{3}{4} & \frac{4}{4} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

Properties: $\mathbb{Q}$ Is Countably Infinite

Theorem 137

$\mathbb{Q}$ is a countably infinite set.

Proof by Cantor: Construct a bijection between $\mathbb{N}$ and $\mathbb{Z} \times \mathbb{N}$ (as a “superset” of $\mathbb{Q}$) .



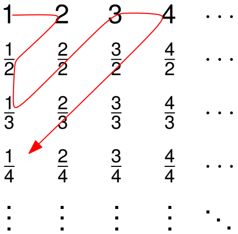
This gives the sequence $1, 2, \frac{1}{2}, \frac{1}{3}, \frac{2}{2}, 3, \ldots$.

Properties: $\mathbb{Q}$ Is Countably Infinite

Theorem 137

$\mathbb{Q}$ is a countably infinite set.

Proof by Cantor: Construct a bijection between $\mathbb{N}$ and $\mathbb{Z} \times \mathbb{N}$ (as a "superset" of $\mathbb{Q}$) .



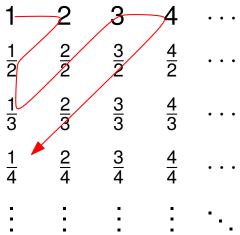
This gives the sequence $1, 2, \frac{1}{2}, \frac{1}{3}, \frac{2}{2}, 3, \ldots$. Now start with zero and include every number's negative number, thus obtaining a systematic enumeration of $\mathbb{Z} \times \mathbb{N}$:

$$0 \quad 1 \quad -1 \quad 2 \quad -2 \quad \tfrac{1}{2} \quad -\tfrac{1}{2} \quad \tfrac{1}{3} \quad -\tfrac{1}{3} \quad \tfrac{2}{2} \quad -\tfrac{2}{2} \quad 3 \quad -3 \quad \ldots$$

Theorem 137

$\mathbb{Q}$ is a countably infinite set.

Proof by Cantor: Construct a bijection between $\mathbb{N}$ and $\mathbb{Z} \times \mathbb{N}$ (as a "superset" of $\mathbb{Q}$) .



$$\begin{array}{ccccc} 1 & 2 & 3 & 4 & \cdots \\ \frac{1}{2} & \frac{2}{2} & \frac{3}{2} & \frac{4}{2} & \cdots \\ \frac{1}{3} & \frac{2}{3} & \frac{3}{3} & \frac{4}{3} & \cdots \\ \frac{1}{4} & \frac{2}{4} & \frac{3}{4} & \frac{4}{4} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

This gives the sequence $1, 2, \frac{1}{2}, \frac{1}{3}, \frac{2}{2}, 3, \ldots$. Now start with zero and include every number's negative number, thus obtaining a systematic enumeration of $\mathbb{Z} \times \mathbb{N}$:

$$0 \quad 1 \quad -1 \quad 2 \quad -2 \quad \frac{1}{2} \quad -\frac{1}{2} \quad \frac{1}{3} \quad -\frac{1}{3} \quad \frac{2}{2} \quad -\frac{2}{2} \quad 3 \quad -3 \quad \ldots$$

Numbering this sequence yields a bijection from $\mathbb{N}$ onto $\mathbb{Z} \times \mathbb{N}$, and Cor. 87 implies the claim.

4 Numbers and Basics of Number Theory

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real Numbers: $\mathbb{R}$

- Intuitively, the reals comprise both rational and irrational numbers like $\sqrt{2}$ or π.
- A formal introduction of the reals, $\mathbb{R}$, based on $\mathbb{Q}$ — e.g., based on Dedekind cuts or based on equivalence classes of Cauchy sequences — is beyond the scope of this lecture.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real Numbers: $\mathbb{R}$

- Intuitively, the reals comprise both rational and irrational numbers like $\sqrt{2}$ or π.
- A formal introduction of the reals, $\mathbb{R}$, based on $\mathbb{Q}$ — e.g., based on Dedekind cuts or based on equivalence classes of Cauchy sequences — is beyond the scope of this lecture.
- Convenient notations for intervals of real numbers:
  $\forall a, b \in \mathbb{R} \quad [a, b] := \{x \in \mathbb{R} : a \leqslant x \leqslant b\};$
  $\forall a, b \in \mathbb{R} \quad]a, b[:= \{x \in \mathbb{R} : a < x < b\};$
  $\forall a, b \in \mathbb{R} \quad [a, b[:= \{x \in \mathbb{R} : a \leqslant x < b\};$
  $\forall a, b \in \mathbb{R} \quad]a, b] := \{x \in \mathbb{R} : a < x \leqslant b\}.$
- Note: Some authors prefer to denote the open interval $]a, b[$ by (a, b).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real Numbers: $\mathbb{R}$

- Intuitively, the reals comprise both rational and irrational numbers like $\sqrt{2}$ or π.
- A formal introduction of the reals, $\mathbb{R}$, based on $\mathbb{Q}$ — e.g., based on Dedekind cuts or based on equivalence classes of Cauchy sequences — is beyond the scope of this lecture.
- Convenient notations for intervals of real numbers:
  $\forall a, b \in \mathbb{R} \quad [a, b] := \{x \in \mathbb{R} : a \leqslant x \leqslant b\}$;
  $\forall a, b \in \mathbb{R} \quad]a, b[:= \{x \in \mathbb{R} : a < x < b\}$;
  $\forall a, b \in \mathbb{R} \quad [a, b[:= \{x \in \mathbb{R} : a \leqslant x < b\}$;
  $\forall a, b \in \mathbb{R} \quad]a, b] := \{x \in \mathbb{R} : a < x \leqslant b\}$.
- Note: Some authors prefer to denote the open interval $]a, b[$ by (a, b).
- Floor and ceiling function (Dt.: Ab- und Aufrundungsfunktion): For $x \in \mathbb{R}$,

$$\lfloor x \rfloor := \max\{k \in \mathbb{Z} : k \leqslant x\},$$

$$\lceil x \rceil := \min\{k \in \mathbb{Z} : k \geqslant x\}.$$

Real Numbers: $\mathbb{R}$

- Intuitively, the reals comprise both rational and irrational numbers like $\sqrt{2}$ or π.
- A formal introduction of the reals, $\mathbb{R}$, based on $\mathbb{Q}$ — e.g., based on Dedekind cuts or based on equivalence classes of Cauchy sequences — is beyond the scope of this lecture.
- Convenient notations for intervals of real numbers:
  $\forall a, b \in \mathbb{R} \quad [a, b] := \{x \in \mathbb{R} : a \leqslant x \leqslant b\};$
  $\forall a, b \in \mathbb{R} \quad]a, b[:= \{x \in \mathbb{R} : a < x < b\};$
  $\forall a, b \in \mathbb{R} \quad [a, b[:= \{x \in \mathbb{R} : a \leqslant x < b\};$
  $\forall a, b \in \mathbb{R} \quad]a, b] := \{x \in \mathbb{R} : a < x \leqslant b\}.$
- Note: Some authors prefer to denote the open interval $]a, b[$ by (a, b).
- Floor and ceiling function (Dt.: Ab- und Aufrundungsfunktion): For $x \in \mathbb{R}$,

$$\lfloor x \rfloor := \max\{k \in \mathbb{Z} : k \leqslant x\},$$

$$\lceil x \rceil := \min\{k \in \mathbb{Z} : k \geqslant x\}.$$

- Gauß introduced the square-bracket notation $[x]$ (“Gaussklammer”) in 1808. The names “floor” and “ceiling” and the corresponding notations were introduced by Iverson in 1962 in his book on APL.
- We have $[x] = \lfloor x \rfloor$ for all $x \in \mathbb{R}$.

Decimal Notation

Definition 138 (Decimal representation, Dt.: Dezimalzahl)

A real number $x \in \mathbb{R}_0^+$ is in *decimal representation* (or a *decimal number*) if it is represented as a sum of (negative) powers of ten:

$$x = x_0 + \sum_{i=1}^{\infty} \frac{x_i}{10^i}, \qquad \text{with an integer part } x_0 \in \mathbb{N}_0 \text{ and with } 0 \leqslant x_i \leqslant 9 \text{ for all } i \in \mathbb{N}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Decimal Notation

Definition 138 (Decimal representation, Dt.: Dezimalzahl)

A real number $x \in \mathbb{R}_0^+$ is in *decimal representation* (or a *decimal number*) if it is represented as a sum of (negative) powers of ten:

$$x = x_0 + \sum_{i=1}^{\infty} \frac{x_i}{10^i}, \qquad \text{with an integer part } x_0 \in \mathbb{N}_0 \text{ and with } 0 \leqslant x_i \leqslant 9 \text{ for all } i \in \mathbb{N}.$$

The decimal representation is *finite* if, for some $n_0 \in \mathbb{N}_0$, we have $x_i = 0$ for all $i \geqslant n_0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Decimal Notation

Definition 138 (Decimal representation, Dt.: Dezimalzahl)

A real number $x \in \mathbb{R}_0^+$ is in *decimal representation* (or a *decimal number*) if it is represented as a sum of (negative) powers of ten:

$$x = x_0 + \sum_{i=1}^{\infty} \frac{x_i}{10^i}, \qquad \text{with an integer part } x_0 \in \mathbb{N}_0 \text{ and with } 0 \leqslant x_i \leqslant 9 \text{ for all } i \in \mathbb{N}.$$

The decimal representation is *finite* if, for some $n_0 \in \mathbb{N}_0$, we have $x_i = 0$ for all $i \geqslant n_0$.

- It is straightforward to extend Def. 138 to negative reals.

Definition 139 (Recurring decimal, Dt.: periodische Dezimalzahl)

A decimal representation of a real number is a *recurring decimal* (or *repeating decimal*) if it becomes periodic at some point: a finite subsequence of the digits after the decimal separator is repeated indefinitely.

Definition 139 (Recurring decimal, Dt.: periodische Dezimalzahl)

A decimal representation of a real number is a *recurring decimal* (or *repeating decimal*) if it becomes periodic at some point: a finite subsequence of the digits after the decimal separator is repeated indefinitely.

- *Recurring decimals*, e.g.,

$$\frac{1}{3} = 0.333\cdots$$

or

$$\frac{1}{7} = 0.142857142857142857\cdots$$

are written as $0.\overline{3}$ or $0.\dot{3}$, and $0.\overline{142857}$. (The horizontal line is known as *vinculum*.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Decimal Notation

Definition 139 (Recurring decimal, Dt.: periodische Dezimalzahl)

A decimal representation of a real number is a *recurring decimal* (or *repeating decimal*) if it becomes periodic at some point: a finite subsequence of the digits after the decimal separator is repeated indefinitely.

- *Recurring decimals*, e.g.,

$$\frac{1}{3} = 0.333\cdots$$

  or

$$\frac{1}{7} = 0.142857142857142857\cdots$$

  are written as $0.\overline{3}$ or $0.\dot{3}$, and $0.\overline{142857}$. (The horizontal line is known as *vinculum*.)

- Note: The decimal representation is not unique: we have $1.0 = 0.\dot{9} = 0.9999\ldots$, where the ellipsis "…" represents an infinite sequence of the digit 9.
- In fact, every non-zero, finitely represented decimal number has an alternate representation with trailing 9s, such as 123.4567 as $123.4566\dot{9}$.

Decimal Notation: Is It a Rational Number?

Lemma 140

A real number has a finite or recurring decimal representation if and only if it is a rational number.

Decimal Notation: Is It a Rational Number?

Lemma 140

A real number has a finite or recurring decimal representation if and only if it is a rational number.

- We proceed as follows to convert $0.43\overline{21}$ to a rational number.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Decimal Notation: Is It a Rational Number?

Lemma 140

A real number has a finite or recurring decimal representation if and only if it is a rational number.

- We proceed as follows to convert $0.43\overline{21}$ to a rational number.
- Let $x := 0.00\overline{21}$. Then $100x = 0.\overline{21}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Decimal Notation: Is It a Rational Number?

Lemma 140

A real number has a finite or recurring decimal representation if and only if it is a rational number.

- We proceed as follows to convert $0.4\overline{321}$ to a rational number.
- Let $x := 0.00\overline{21}$. Then $100x = 0.\overline{21}$. This gives

$$99x = 100x - x = 0.\overline{21} - 0.00\overline{21} = 0.21 = \frac{21}{100}.$$

Lemma 140

A real number has a finite or recurring decimal representation if and only if it is a rational number.

- We proceed as follows to convert $0.4\overline{321}$ to a rational number.
- Let $x := 0.00\overline{21}$. Then $100x = 0.\overline{21}$. This gives

$$99x = 100x - x = 0.\overline{21} - 0.00\overline{21} = 0.21 = \frac{21}{100}.$$

- We get

$$x = \frac{21}{99 \cdot 100} = \frac{21}{9900} = \frac{7}{3300}.$$

Decimal Notation: Is It a Rational Number?

Lemma 140

A real number has a finite or recurring decimal representation if and only if it is a rational number.

- We proceed as follows to convert $0.43\overline{21}$ to a rational number.
- Let $x := 0.00\overline{21}$. Then $100x = 0.\overline{21}$. This gives

$$99x = 100x - x = 0.\overline{21} - 0.00\overline{21} = 0.21 = \frac{21}{100}.$$

- We get

$$x = \frac{21}{99 \cdot 100} = \frac{21}{9900} = \frac{7}{3300}.$$

- Hence,

$$0.43\overline{21} = 0.43 + x = \frac{43}{100} + \frac{7}{3300} = \frac{1426}{3300} = \frac{713}{1650}.$$

Decimal Notation: Is It a Rational Number?

Lemma 140

A real number has a finite or recurring decimal representation if and only if it is a rational number.

- We proceed as follows to convert $0.43\overline{21}$ to a rational number.
- Let $x := 0.00\overline{21}$. Then $100x = 0.\overline{21}$. This gives

$$99x = 100x - x = 0.\overline{21} - 0.00\overline{21} = 0.21 = \frac{21}{100}.$$

- We get

$$x = \frac{21}{99 \cdot 100} = \frac{21}{9900} = \frac{7}{3300}.$$

- Hence,

$$0.43\overline{21} = 0.43 + x = \frac{43}{100} + \frac{7}{3300} = \frac{1426}{3300} = \frac{713}{1650}.$$

Definition 141 (Irrational)

A number $x \in \mathbb{R} \backslash \mathbb{Q}$ is called *irrational*.

Decimal Notation

Definition 142 (Decimal separator)

The decimal separator is a symbol which is used to mark the boundary between the integer part and the fractional part of a number in decimal representation.

Decimal Notation

Definition 142 (Decimal separator)

The decimal separator is a symbol which is used to mark the boundary between the integer part and the fractional part of a number in decimal representation.

Warning

A least two symbols are in wide-spread use for the decimal separator!

- Most of Europe, most of South America and French Canada use the comma, while the UK, USA, Australia, English Canada and several Asian countries use a dot (“period”, “full stop”). The dot also prevails in English-language publications.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 142 (Decimal separator)

The decimal separator is a symbol which is used to mark the boundary between the integer part and the fractional part of a number in decimal representation.

Warning

A least two symbols are in wide-spread use for the decimal separator!

- Most of Europe, most of South America and French Canada use the comma, while the UK, USA, Australia, English Canada and several Asian countries use a dot ("period", "full stop"). The dot also prevails in English-language publications.
- Dots or commas are frequently used to group three digits into groups within the integer part. However, this practice is discouraged by ISO!

Well-Ordering the Reals

- By definition, $(\mathbb{N}, \leqslant)$ is well-ordered. And we have already hinted at well-orderings for $\mathbb{Z}$ and $\mathbb{Q}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Well-Ordering the Reals

- By definition, $(\mathbb{N}, \leqslant)$ is well-ordered. And we have already hinted at well-orderings for $\mathbb{Z}$ and $\mathbb{Q}$.
- Question: Can the reals be well-ordered?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Well-Ordering the Reals

- By definition, $(\mathbb{N}, \leqslant)$ is well-ordered. And we have already hinted at well-orderings for $\mathbb{Z}$ and $\mathbb{Q}$.
- Question: Can the reals be well-ordered?
- Answer: We don't know it for sure!

Well-Ordering the Reals

- By definition, $(\mathbb{N}, \leqslant)$ is well-ordered. And we have already hinted at well-orderings for $\mathbb{Z}$ and $\mathbb{Q}$.
- Question: Can the reals be well-ordered?
- Answer: We don't know it for sure!
- It has been proved that it is impossible to write down an explicit well-ordering for the reals.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Well-Ordering the Reals

- By definition, $(\mathbb{N}, \leqslant)$ is well-ordered. And we have already hinted at well-orderings for $\mathbb{Z}$ and $\mathbb{Q}$.
- Question: Can the reals be well-ordered?
- Answer: We don't know it for sure!
- It has been proved that it is impossible to write down an explicit well-ordering for the reals.

Well-Order "Theorem"

Every set can be well-ordered.

- In 1883, Georg Cantor stated that the Well-Order Theorem is a ”fundamental law of thought”. This statement started a mathematical flame war!

Well-Ordering the Reals

- By definition, $(\mathbb{N}, \leqslant)$ is well-ordered. And we have already hinted at well-orderings for $\mathbb{Z}$ and $\mathbb{Q}$.
- Question: Can the reals be well-ordered?
- Answer: We don't know it for sure!
- It has been proved that it is impossible to write down an explicit well-ordering for the reals.

Well-Order "Theorem"

Every set can be well-ordered.

- In 1883, Georg Cantor stated that the Well-Order Theorem is a "fundamental law of thought". This statement started a mathematical flame war!
- In any case, this "theorem" can only be taken as an axiom, since it has been proved that it does not follow from any of the other commonly accepted axioms of set theory.
- In first-order logic, the Well-Order Theorem is equivalent to the Axiom of Choice (Dt.: Auswahlaxiom) and to Zorn's Lemma, in the sense that either one of them together with the Zermelo-Fraenkel Axioms allows to deduce the other ones.

Theorem 143

The real numbers are uncountable, i.e., there exists no bijection from $\mathbb{N}$ onto $\mathbb{R}$.

Theorem 143

The real numbers are uncountable, i.e., there exists no bijection from $\mathbb{N}$ onto $\mathbb{R}$.

Proof by Cantor (1891) : Suppose to the contrary that there exists a bijection $a : \mathbb{N} \to \mathbb{R}$. We show that we can construct a number r which is not in the list $a_1, a_2, a_3, \ldots$:

The Reals are Not Countable

Theorem 143

The real numbers are uncountable, i.e., there exists no bijection from $\mathbb{N}$ onto $\mathbb{R}$.

Proof by Cantor (1891) : Suppose to the contrary that there exists a bijection $a : \mathbb{N} \to \mathbb{R}$. We show that we can construct a number r which is not in the list $a_1, a_2, a_3, \ldots$: For $k \in \mathbb{N}$ let d_k be the k-th digit after the decimal separator in a_k if a_k has at least k digits after the decimal separator, and $d_k := 0$ otherwise.

$$\begin{aligned} a_1 &= \text{—}.d_1 - - - - \ldots \\ a_2 &= \text{—}.- d_2 - - - \ldots \\ a_3 &= \text{—}.- - d_3 - - \ldots \\ &\vdots \end{aligned}$$

Theorem 143

The real numbers are uncountable, i.e., there exists no bijection from $\mathbb{N}$ onto $\mathbb{R}$.

Proof by Cantor (1891) : Suppose to the contrary that there exists a bijection $a : \mathbb{N} \to \mathbb{R}$. We show that we can construct a number r which is not in the list $a_1, a_2, a_3, \ldots$: For $k \in \mathbb{N}$ let d_k be the k-th digit after the decimal separator in a_k if a_k has at least k digits after the decimal separator, and $d_k := 0$ otherwise.

$$\begin{array}{lll} a_1 & = & \text{—}.d_1 - - - - \ldots \\ a_2 & = & \text{—}.- d_2 - - - \ldots \\ a_3 & = & \text{—}.- - d_3 - - \ldots \\ & \vdots & \end{array}$$

If $d_k = 1$ then $r_k := 2$ else $r_k := 1$. Now regard r_k as the k-th digit of a number $r \in \mathbb{R}$: we have $r = 0.r_1r_2r_3r_4\ldots$. Since at least the k-th digit of r differs from the k-th digit of a_k, we conclude that $r \neq a_n$ for all $n \in \mathbb{N}$. □

Theorem 143

The real numbers are uncountable, i.e., there exists no bijection from $\mathbb{N}$ onto $\mathbb{R}$.

Proof by Cantor (1891) : Suppose to the contrary that there exists a bijection $a : \mathbb{N} \to \mathbb{R}$. We show that we can construct a number r which is not in the list $a_1, a_2, a_3, \ldots$: For $k \in \mathbb{N}$ let d_k be the k-th digit after the decimal separator in a_k if a_k has at least k digits after the decimal separator, and $d_k := 0$ otherwise.

$$\begin{array}{lcl} a_1 & = & \text{—}.d_1 - - - - \ldots \\ a_2 & = & \text{—}.- d_2 - - - \ldots \\ a_3 & = & \text{—}.- - d_3 - - \ldots \\ & \vdots & \end{array}$$

If $d_k = 1$ then $r_k := 2$ else $r_k := 1$. Now regard r_k as the k-th digit of a number $r \in \mathbb{R}$: we have $r = 0.r_1 r_2 r_3 r_4 \ldots$. Since at least the k-th digit of r differs from the k-th digit of a_k, we conclude that $r \neq a_n$ for all $n \in \mathbb{N}$. □

- Hence, $|\mathbb{N}| < |\mathbb{R}|$.

Theorem 144

For every $x \in \mathbb{R}$, every arbitrarily small neighborhood of x contains a rational number.

Sketch of proof: Let $x \in \mathbb{R}$ and $\varepsilon \in \mathbb{R}^+$ be arbitrary but fixed. W.l.o.g, $0 < x < 1$. Let $k \in \mathbb{N}$ such that $10^{-k} < \varepsilon$.

Theorem 144

For every $x \in \mathbb{R}$, every arbitrarily small neighborhood of x contains a rational number.

Sketch of proof: Let $x \in \mathbb{R}$ and $\varepsilon \in \mathbb{R}^+$ be arbitrary but fixed. W.l.o.g, $0 < x < 1$. Let $k \in \mathbb{N}$ such that $10^{-k} < \varepsilon$.
We define the rational number p/q as follows:

$$p := \lfloor x \cdot 10^k \rfloor \qquad q := 10^k$$

Theorem 144

For every $x \in \mathbb{R}$, every arbitrarily small neighborhood of x contains a rational number.

Sketch of proof: Let $x \in \mathbb{R}$ and $\varepsilon \in \mathbb{R}^+$ be arbitrary but fixed. W.l.o.g, $0 < x < 1$. Let $k \in \mathbb{N}$ such that $10^{-k} < \varepsilon$.
We define the rational number p/q as follows:

$$p := \lfloor x \cdot 10^k \rfloor \qquad q := 10^k$$

This gives

$$\left| x - \frac{p}{q} \right| \leqslant \frac{1}{10^k} < \varepsilon.$$

□

Theorem 144

For every $x \in \mathbb{R}$, every arbitrarily small neighborhood of x contains a rational number.

Sketch of proof: Let $x \in \mathbb{R}$ and $\varepsilon \in \mathbb{R}^+$ be arbitrary but fixed. W.l.o.g, $0 < x < 1$. Let $k \in \mathbb{N}$ such that $10^{-k} < \varepsilon$.
We define the rational number p/q as follows:

$$p := \left\lfloor x \cdot 10^k \right\rfloor \qquad\qquad q := 10^k$$

This gives

$$\left| x - \frac{p}{q} \right| \leqslant \frac{1}{10^k} < \varepsilon.$$

□

- E.g., $\pi \approx 3.1415 = \frac{31\,415}{10\,000}$ with $|\pi - \frac{31\,415}{10\,000}| \leqslant \frac{1}{10\,000}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 144

For every $x \in \mathbb{R}$, every arbitrarily small neighborhood of x contains a rational number.

Sketch of proof: Let $x \in \mathbb{R}$ and $\varepsilon \in \mathbb{R}^+$ be arbitrary but fixed. W.l.o.g, $0 < x < 1$. Let $k \in \mathbb{N}$ such that $10^{-k} < \varepsilon$.
We define the rational number p/q as follows:

$$p := \lfloor x \cdot 10^k \rfloor \qquad q := 10^k$$

This gives

$$\left| x - \frac{p}{q} \right| \leqslant \frac{1}{10^k} < \varepsilon. \qquad \square$$

- E.g., $\pi \approx 3.1415 = \frac{31\,415}{10\,000}$ with $|\pi - \frac{31\,415}{10\,000}| \leqslant \frac{1}{10\,000}$.
- Thus, we can approximate a real number by a rational number p/q.
- If the denominator q is a power of 10 then we can guarantee the error to be at most $1/q$. Otherwise, if we allow an arbitrary integer q as denominator, we can guarantee the error to be at most $1/q^2$.

More on Cardinalities

Theorem 145

No (non-empty) set A has the same cardinality as its power set $\mathcal{P}(A)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 145

No (non-empty) set A has the same cardinality as its power set $\mathcal{P}(A)$.

- This implies that $|\mathbb{N}| < |\mathcal{P}(\mathbb{N})| < |\mathcal{P}(\mathcal{P}(\mathbb{N}))| < \ldots$

More on Cardinalities

Theorem 145

No (non-empty) set A has the same cardinality as its power set $\mathcal{P}(A)$.

- This implies that $|\mathbb{N}| < |\mathcal{P}(\mathbb{N})| < |\mathcal{P}(\mathcal{P}(\mathbb{N}))| < \ldots$
- The cardinality of $\mathbb{N}$ is denoted by $\aleph_0$.

Theorem 146

$|\mathbb{R}| = |\mathcal{P}(\mathbb{N})| = 2^{\aleph_0} > \aleph_0 = |\mathbb{N}| = |\mathbb{Z}| = |\mathbb{Q}|$.

More on Cardinalities

Theorem 145

No (non-empty) set A has the same cardinality as its power set $\mathcal{P}(A)$.

- This implies that $|\mathbb{N}| < |\mathcal{P}(\mathbb{N})| < |\mathcal{P}(\mathcal{P}(\mathbb{N}))| < \dots$
- The cardinality of $\mathbb{N}$ is denoted by $\aleph_0$.

Theorem 146

$|\mathbb{R}| = |\mathcal{P}(\mathbb{N})| = 2^{\aleph_0} > \aleph_0 = |\mathbb{N}| = |\mathbb{Z}| = |\mathbb{Q}|$.

- *Continuum hypothesis*: There is no set with cardinality strictly between that of the integers and the reals.

More on Cardinalities

Theorem 145

No (non-empty) set A has the same cardinality as its power set $\mathcal{P}(A)$.

- This implies that $|\mathbb{N}| < |\mathcal{P}(\mathbb{N})| < |\mathcal{P}(\mathcal{P}(\mathbb{N}))| < \dots$
- The cardinality of $\mathbb{N}$ is denoted by $\aleph_0$.

Theorem 146

$|\mathbb{R}| = |\mathcal{P}(\mathbb{N})| = 2^{\aleph_0} > \aleph_0 = |\mathbb{N}| = |\mathbb{Z}| = |\mathbb{Q}|$.

- *Continuum hypothesis*: There is no set with cardinality strictly between that of the integers and the reals.
- The continuum hypothesis started out as a conjecture, until it was shown to be consistent with the usual axioms of the reals (by Gödel in 1940), and independent of those axioms (by Cohen in 1963).

More on Cardinalities

Theorem 145

No (non-empty) set A has the same cardinality as its power set $\mathcal{P}(A)$.

- This implies that $|\mathbb{N}| < |\mathcal{P}(\mathbb{N})| < |\mathcal{P}(\mathcal{P}(\mathbb{N}))| < \ldots$
- The cardinality of $\mathbb{N}$ is denoted by $\aleph_0$.

Theorem 146

$|\mathbb{R}| = |\mathcal{P}(\mathbb{N})| = 2^{\aleph_0} > \aleph_0 = |\mathbb{N}| = |\mathbb{Z}| = |\mathbb{Q}|$.

- *Continuum hypothesis*: There is no set with cardinality strictly between that of the integers and the reals.
- The continuum hypothesis started out as a conjecture, until it was shown to be consistent with the usual axioms of the reals (by Gödel in 1940), and independent of those axioms (by Cohen in 1963).
- Under this hypothesis, the cardinality of $\mathbb{R}$ equals $\aleph_1$, and we have $2^{\aleph_0} = \aleph_1$.
- Furthermore, $|\mathcal{P}(\mathbb{R})| =: \aleph_2$, etc.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

4 Numbers and Basics of Number Theory

- Algebraic Structures
- Natural Numbers
- Integers
- Rational Numbers
- Real Numbers
- More Proof Techniques
  - Pigeonhole Principle
  - Well-founded Induction
  - Structural Induction

The Pigeonhole Principle

- In 1834, Johann Dirichlet noted that if there are five objects in four drawers then there is a drawer with two or more objects.
- Pigeonhole Principle: If n letters are posted to m pigeonholes, then
  - at least one pigeonhole receives more than one letter if $n > m$.
  - at least one pigeonhole remains empty if $n < m$.
  - each pigeonhole might receive exactly one letter if $n = m$.

The Pigeonhole Principle

- In 1834, Johann Dirichlet noted that if there are five objects in four drawers then there is a drawer with two or more objects.
- Pigeonhole Principle: If n letters are posted to m pigeonholes, then
  - at least one pigeonhole receives more than one letter if $n > m$.
  - at least one pigeonhole remains empty if $n < m$.
  - each pigeonhole might receive exactly one letter if $n = m$.

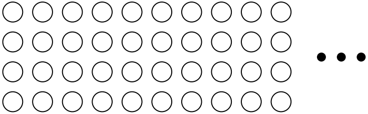
Theorem 147 (Pigeonhole Principle, Dt.: Schubfachschluss)

Consider two finite sets A and B. If A has more elements then B then every mapping from A to B will cause at least one element of B to be the target of two or more elements of A.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

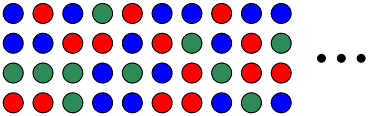
Lemma 148

Consider a rectangular grid of points which consists of four rows and 100 columns.



Lemma 148

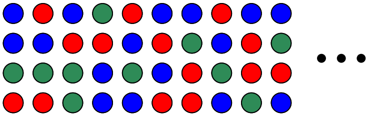
Consider a rectangular grid of points which consists of four rows and 100 columns. Each point is colored with a color which is picked randomly among red, green and blue.



The Pigeonhole Principle: Sample Application

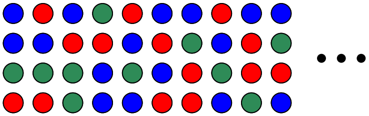
Lemma 148

Consider a rectangular grid of points which consists of four rows and 100 columns. Each point is colored with a color which is picked randomly among red, green and blue. Prove that there always exist four points of the same color that form the corners of a rectangle (with sides parallel to the grid), no matter how the coloring is done.



Lemma 148

Consider a rectangular grid of points which consists of four rows and 100 columns. Each point is colored with a color which is picked randomly among red, green and blue. Prove that there always exist four points of the same color that form the corners of a rectangle (with sides parallel to the grid), no matter how the coloring is done.

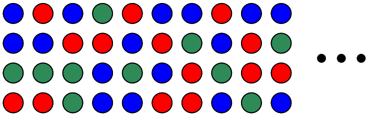


Proof: A column pattern is the top-to-bottom sequence of colors assigned to the four points of a column of the grid.

□

Lemma 148

Consider a rectangular grid of points which consists of four rows and 100 columns. Each point is colored with a color which is picked randomly among red, green and blue. Prove that there always exist four points of the same color that form the corners of a rectangle (with sides parallel to the grid), no matter how the coloring is done.

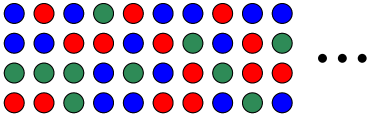


Proof: A column pattern is the top-to-bottom sequence of colors assigned to the four points of a column of the grid. There are exactly $3^4 = 81$ different column patterns. □

The Pigeonhole Principle: Sample Application

Lemma 148

Consider a rectangular grid of points which consists of four rows and 100 columns. Each point is colored with a color which is picked randomly among red, green and blue. Prove that there always exist four points of the same color that form the corners of a rectangle (with sides parallel to the grid), no matter how the coloring is done.



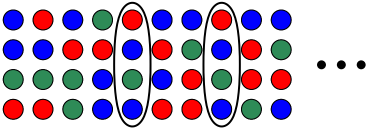
Proof : A column pattern is the top-to-bottom sequence of colors assigned to the four points of a column of the grid. There are exactly $3^4 = 81$ different column patterns. Since there are more than 81 columns, we are guaranteed to have at least two columns with the same column pattern.

□

The Pigeonhole Principle: Sample Application

Lemma 148

Consider a rectangular grid of points which consists of four rows and 100 columns. Each point is colored with a color which is picked randomly among red, green and blue. Prove that there always exist four points of the same color that form the corners of a rectangle (with sides parallel to the grid), no matter how the coloring is done.

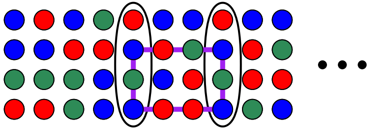


Proof : A column pattern is the top-to-bottom sequence of colors assigned to the four points of a column of the grid. There are exactly $3^4 = 81$ different column patterns. Since there are more than 81 columns, we are guaranteed to have at least two columns with the same column pattern. Consider two such columns.

□

Lemma 148

Consider a rectangular grid of points which consists of four rows and 100 columns. Each point is colored with a color which is picked randomly among red, green and blue. Prove that there always exist four points of the same color that form the corners of a rectangle (with sides parallel to the grid), no matter how the coloring is done.

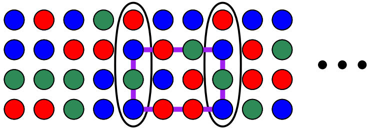


Proof : A column pattern is the top-to-bottom sequence of colors assigned to the four points of a column of the grid. There are exactly $3^4 = 81$ different column patterns. Since there are more than 81 columns, we are guaranteed to have at least two columns with the same column pattern. Consider two such columns. Since there are four rows but only three colors, we conclude that two of the rows have the same color, thus giving us the four corners of the rectangle sought. □

The Pigeonhole Principle: Sample Application

Lemma 148

Consider a rectangular grid of points which consists of four rows and 100 columns. Each point is colored with a color which is picked randomly among red, green and blue. Prove that there always exist four points of the same color that form the corners of a rectangle (with sides parallel to the grid), no matter how the coloring is done.

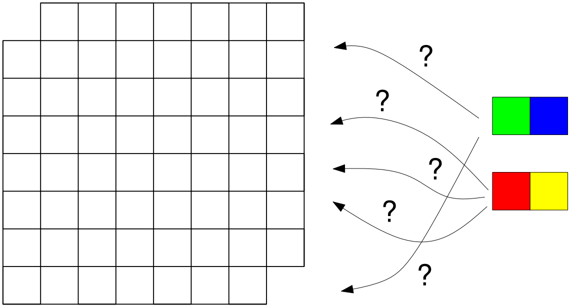


Proof : A column pattern is the top-to-bottom sequence of colors assigned to the four points of a column of the grid. There are exactly $3^4 = 81$ different column patterns. Since there are more than 81 columns, we are guaranteed to have at least two columns with the same column pattern. Consider two such columns. Since there are four rows but only three colors, we conclude that two of the rows have the same color, thus giving us the four corners of the rectangle sought. □

- Note: Just 19 columns suffice to guarantee the existance of such a rectangle.

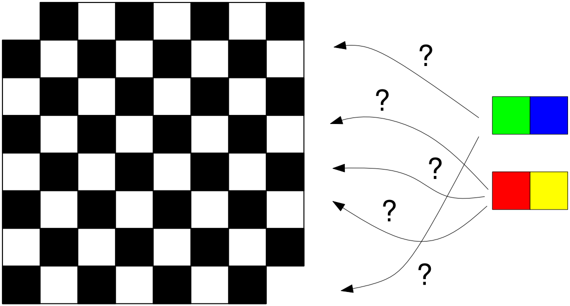
Real-World Application: Chessboard Tilings Revisited

- Question: Can our modified chessboard be covered completely by 31 domino blocks of arbitrary color combinations?



Real-World Application: Chessboard Tilings Revisited

- Question: Can our modified chessboard be covered completely by 31 domino blocks of arbitrary color combinations?

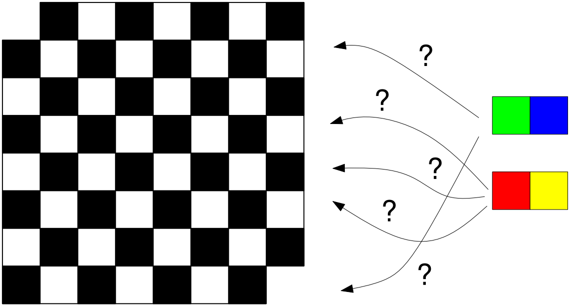


- We observe that every permissible domino placement covers exactly one black square and one white square of the chessboard.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Chessboard Tilings Revisited

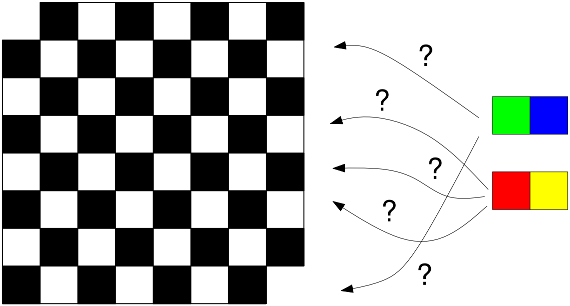
- Question: Can our modified chessboard be covered completely by 31 domino blocks of arbitrary color combinations?



- We observe that every permissible domino placement covers exactly one black square and one white square of the chessboard.
- Thus, all domino placements would establish a one-to-one mapping between black and white squares.

Real-World Application: Chessboard Tilings Revisited

- Question: Can our modified chessboard be covered completely by 31 domino blocks of arbitrary color combinations?



- We observe that every permissible domino placement covers exactly one black square and one white square of the chessboard.
- Thus, all domino placements would establish a one-to-one mapping between black and white squares. However, there are 32 black squares and only 30 white squares! We conclude that our chessboard cannot be covered completely by domino blocks.

Real-World Application: Analysis of Lossless Data Compression

- Could one design an algorithm for lossless data compression that is guaranteed not to increase the file size of some input file while achieving a genuine compression for at least one other file?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Could one design an algorithm for lossless data compression that is guaranteed not to increase the file size of some input file while achieving a genuine compression for at least one other file? No!

Real-World Application: Analysis of Lossless Data Compression

- Could one design an algorithm for lossless data compression that is guaranteed not to increase the file size of some input file while achieving a genuine compression for at least one other file? No!
- Assume that every file is represented as a string of bits of some arbitrary length. Suppose further that there exists a compression algorithm that transforms every file into a distinct file which is no longer than the original file, and that at least one file will be compressed into something that is shorter than itself.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Lossless Data Compression

- Could one design an algorithm for lossless data compression that is guaranteed not to increase the file size of some input file while achieving a genuine compression for at least one other file? No!
- Assume that every file is represented as a string of bits of some arbitrary length. Suppose further that there exists a compression algorithm that transforms every file into a distinct file which is no longer than the original file, and that at least one file will be compressed into something that is shorter than itself.
- Let m be the least number such that there is a file f with length m bits that gets compressed to something shorter. Let n be the number of bits of the compressed version of f. Hence, $n < m$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Lossless Data Compression

- Could one design an algorithm for lossless data compression that is guaranteed not to increase the file size of some input file while achieving a genuine compression for at least one other file? No!
- Assume that every file is represented as a string of bits of some arbitrary length. Suppose further that there exists a compression algorithm that transforms every file into a distinct file which is no longer than the original file, and that at least one file will be compressed into something that is shorter than itself.
- Let m be the least number such that there is a file f with length m bits that gets compressed to something shorter. Let n be the number of bits of the compressed version of f. Hence, $n < m$.
- Since $n < m$, every file of length n keeps its size during compression. There are 2^n many such files. Together with f we would have $2^n + 1$ files which all compress into one of the 2^n files of length n.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Lossless Data Compression

- Could one design an algorithm for lossless data compression that is guaranteed not to increase the file size of some input file while achieving a genuine compression for at least one other file? No!
- Assume that every file is represented as a string of bits of some arbitrary length. Suppose further that there exists a compression algorithm that transforms every file into a distinct file which is no longer than the original file, and that at least one file will be compressed into something that is shorter than itself.
- Let m be the least number such that there is a file f with length m bits that gets compressed to something shorter. Let n be the number of bits of the compressed version of f. Hence, $n < m$.
- Since $n < m$, every file of length n keeps its size during compression. There are 2^n many such files. Together with f we would have $2^n + 1$ files which all compress into one of the 2^n files of length n.
- By the pigeonhole principle there must exist some file f' of length n which is the output of the compression algorithm for two different inputs. That file f' cannot be decompressed reliably, which contradicts the assumption that the algorithm is lossless.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Lossless Data Compression

- Could one design an algorithm for lossless data compression that is guaranteed not to increase the file size of some input file while achieving a genuine compression for at least one other file? No!
- Assume that every file is represented as a string of bits of some arbitrary length. Suppose further that there exists a compression algorithm that transforms every file into a distinct file which is no longer than the original file, and that at least one file will be compressed into something that is shorter than itself.
- Let m be the least number such that there is a file f with length m bits that gets compressed to something shorter. Let n be the number of bits of the compressed version of f. Hence, $n < m$.
- Since $n < m$, every file of length n keeps its size during compression. There are 2^n many such files. Together with f we would have $2^n + 1$ files which all compress into one of the 2^n files of length n.
- By the pigeonhole principle there must exist some file f' of length n which is the output of the compression algorithm for two different inputs. That file f' cannot be decompressed reliably, which contradicts the assumption that the algorithm is lossless.
- Hence, every compression algorithm will increase the size of at least some file, or keep the sizes of all files unchanged.

Definition 149 (Well-founded order, Dt.: wohlfundierte Ordnung)

A strict partial order $\prec$ on M is called *well-founded* if every $X \subseteq M$, with $X \neq \emptyset$, has at least one minimal element relative to $\prec$. A poset $(M, \prec)$ is called a well-founded poset if $\prec$ is well-founded.

Definition 149 (Well-founded order, Dt.: wohlfundierte Ordnung)

A strict partial order $\prec$ on M is called *well-founded* if every $X \subseteq M$, with $X \neq \emptyset$, has at least one minimal element relative to $\prec$. A poset $(M, \prec)$ is called a well-founded poset if $\prec$ is well-founded.

- Of course, $(\mathbb{N}, <)$ is well-founded.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Well-founded Order

Definition 149 (Well-founded order, Dt.: wohlfundierte Ordnung)

A strict partial order $\prec$ on M is called *well-founded* if every $X \subseteq M$, with $X \neq \emptyset$, has at least one minimal element relative to $\prec$. A poset $(M, \prec)$ is called a well-founded poset if $\prec$ is well-founded.

- Of course, $(\mathbb{N}, <)$ is well-founded.
- Some authors call a well-founded order also a *Noetherian order*, named after Emmy Noether (1882-1935).
- Not to be confused with a well-order (Dt.: Wohlordnung).

Definition 149 (Well-founded order, Dt.: wohlfundierte Ordnung)

A strict partial order $\prec$ on M is called *well-founded* if every $X \subseteq M$, with $X \neq \emptyset$, has at least one minimal element relative to $\prec$. A poset $(M, \prec)$ is called a well-founded poset if $\prec$ is well-founded.

- Of course, $(\mathbb{N}, <)$ is well-founded.
- Some authors call a well-founded order also a *Noetherian order*, named after Emmy Noether (1882-1935).
- Not to be confused with a well-order (Dt.: Wohlordnung).

Lemma 150

The poset $(M, \prec)$ is well-founded if and only if no infinite strictly decreasing sequence in M exists, i.e., if an $a : \mathbb{N} \to M$ with $a_{i+1} \prec a_i$ for all $i \in \mathbb{N}$ does not exist.

Lexicographical Order

Definition 151

Let $(M_1, \prec_1)$ and $(M_2, \prec_2)$ be two posets. The *lexicographical ordering* $(\prec_1, \prec_2)_{lex}$ on $M_1 \times M_2$ is defined as

$$(a_1, b_1)\ (\prec_1, \prec_2)_{lex}\ (a_2, b_2) \quad :\Leftrightarrow \quad \big((a_1 \prec_1 a_2) \vee ((a_1 = a_2) \wedge (b_1 \prec_2 b_2))\big),$$

where $(a_1, b_1), (a_2, b_2) \in M_1 \times M_2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 151

Let $(M_1, \prec_1)$ and $(M_2, \prec_2)$ be two posets. The *lexicographical ordering* $(\prec_1, \prec_2)_{lex}$ on $M_1 \times M_2$ is defined as

$$(a_1, b_1)\ (\prec_1, \prec_2)_{lex}\ (a_2, b_2) \quad :\Leftrightarrow \quad \big((a_1 \prec_1 a_2) \vee ((a_1 = a_2) \wedge (b_1 \prec_2 b_2))\big),$$

where $(a_1, b_1), (a_2, b_2) \in M_1 \times M_2$.

Lemma 152

Let $(M_1, \prec_1)$ and $(M_2, \prec_2)$ be two posets. Then $M_1 \times M_2$ together with the lexicographical order $(\prec_1, \prec_2)_{lex}$ is a poset.

- Similarly for a non-strict partial order $\preceq$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 151

Let $(M_1, \prec_1)$ and $(M_2, \prec_2)$ be two posets. The *lexicographical ordering* $(\prec_1, \prec_2)_{lex}$ on $M_1 \times M_2$ is defined as

$$(a_1, b_1)\ (\prec_1, \prec_2)_{lex}\ (a_2, b_2) \quad :\Leftrightarrow \quad \big((a_1 \prec_1 a_2) \vee ((a_1 = a_2) \wedge (b_1 \prec_2 b_2))\big),$$

where $(a_1, b_1), (a_2, b_2) \in M_1 \times M_2$.

Lemma 152

Let $(M_1, \prec_1)$ and $(M_2, \prec_2)$ be two posets. Then $M_1 \times M_2$ together with the lexicographical order $(\prec_1, \prec_2)_{lex}$ is a poset.

- Similarly for a non-strict partial order $\preceq$.

Lemma 153

The posets $(M_1, \prec_1)$ and $(M_2, \prec_2)$ are well-founded if and only if $(M_1 \times M_2, (\prec_1, \prec_2)_{lex})$ is well-founded.

Induction Revisited

- Consider a predicate P over $\mathbb{N}$ and recall the Strong Induction Principle (Thm 79): If $P(1)$ and if

$$\forall k \in \mathbb{N} \ \left[\left(\forall (m \in \mathbb{N}, m \leqslant k) \ \ P(m)\right) \ \Rightarrow \ P(k+1)\right]$$

then

$$\forall n \in \mathbb{N} \ \ P(n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- And yet another version with "implicit" base:
  If

$$\forall k \in \mathbb{N} \ \left[\big(\forall (m \in \mathbb{N}, m < k) \ \ P(m) \big) \ \Rightarrow \ P(k) \right]$$

then

$$\forall n \in \mathbb{N} \ \ P(n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Induction Revisited

- And yet another version with "implicit" base:
  If
  $$\forall k \in \mathbb{N} \; \left[\left(\forall (m \in \mathbb{N}, m < k) \;\; P(m) \right) \;\Rightarrow\; P(k) \right]$$
  then
  $$\forall n \in \mathbb{N} \;\; P(n).$$
- Note: The base case was not lost! Rather, it is included since we have to prove $P(1)$ using the "helpful knowledge" that $P(m)$ holds for all $m \in \mathbb{N}$ with $m < 1$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 154 (Principle of Well-founded Induction, Dt.: wohlfundierte Induktion)

Let $(M, \prec)$ be well-founded and P be a predicate on M.
If

$$\forall k \in M \ \left[\left(\forall (m \in M, m \prec k) \ P(m)\right) \ \Rightarrow \ P(k)\right]$$

then

$$\forall m \in M \ \ P(m).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Well-founded Induction

Theorem 154 (Principle of Well-founded Induction, Dt.: wohlfundierte Induktion)

Let $(M, \prec)$ be well-founded and P be a predicate on M.
If

$$\forall k \in M \;\; \big[\big(\forall (m \in M, m \prec k) \;\; P(m)\big) \;\Rightarrow\; P(k)\big]$$

then

$$\forall m \in M \;\; P(m).$$

- That is, as inductive step we have to prove that the predicate holds for k if it holds for all predecessors m of k relative to $\prec$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Well-founded Induction

Theorem 154 (Principle of Well-founded Induction, Dt.: wohlfundierte Induktion)

Let $(M, \prec)$ be well-founded and P be a predicate on M.
If

$$\forall k \in M \ \left[\left(\forall (m \in M, m \prec k) \ \ P(m)\right) \ \Rightarrow \ P(k)\right]$$

then

$$\forall m \in M \ \ P(m).$$

- That is, as inductive step we have to prove that the predicate holds for k if it holds for all predecessors m of k relative to $\prec$.

Proof: Let $X := \{m \in M : P(m) \text{ is false}\}$, and suppose $X \neq \emptyset$. Since $(M, \prec)$ is well-founded, X has a minimal element n. Thus, $\forall m \in M$ with $m \prec n$ the predicate $P(m)$ holds. The inductive step

$$\left(\forall (m \in M, m \prec n) \ \ P(m)\right) \ \Rightarrow \ P(n)$$

yields that $P(n)$ holds, in contradiction to $n \in X$. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Well-founded Induction

- We give a proof of the existance claim made by the Fundamental Theorem of Arithmetic (Thm. 102): Every natural number $n > 1$ is either a prime number or has a prime factorization.

Sample Well-founded Induction

- We give a proof of the existance claim made by the Fundamental Theorem of Arithmetic (Thm. 102): Every natural number $n > 1$ is either a prime number or has a prime factorization.

Proof : We begin with observing that the relation "is genuine divisor of" (Def. 93) over $\mathbb{N}\backslash\{1\}$ is well-founded. The minimal elements relative to this relation are the primes.

Sample Well-founded Induction

- We give a proof of the existance claim made by the Fundamental Theorem of Arithmetic (Thm. 102): Every natural number $n > 1$ is either a prime number or has a prime factorization.

Proof : We begin with observing that the relation "is genuine divisor of" (Def. 93) over $\mathbb{N}\backslash\{1\}$ is well-founded. The minimal elements relative to this relation are the primes.

We consider an arbitrary but fixed $k \in \mathbb{N}\backslash\{1\}$ and assume as inductive hypothesis that the claim holds for all $m \in \mathbb{N}\backslash\{1\}$ that are smaller than k relative to this order.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Well-founded Induction

- We give a proof of the existance claim made by the Fundamental Theorem of Arithmetic (Thm. 102): Every natural number $n > 1$ is either a prime number or has a prime factorization.

Proof : We begin with observing that the relation “is genuine divisor of” (Def. 93) over $\mathbb{N}\backslash\{1\}$ is well-founded. The minimal elements relative to this relation are the primes.

We consider an arbitrary but fixed $k \in \mathbb{N}\backslash\{1\}$ and assume as inductive hypothesis that the claim holds for all $m \in \mathbb{N}\backslash\{1\}$ that are smaller than k relative to this order.

Of course, if k is prime then the claim given by the theorem holds.

Sample Well-founded Induction

- We give a proof of the existance claim made by the Fundamental Theorem of Arithmetic (Thm. 102): Every natural number $n > 1$ is either a prime number or has a prime factorization.

Proof: We begin with observing that the relation "is genuine divisor of" (Def. 93) over $\mathbb{N}\backslash\{1\}$ is well-founded. The minimal elements relative to this relation are the primes.

We consider an arbitrary but fixed $k \in \mathbb{N}\backslash\{1\}$ and assume as inductive hypothesis that the claim holds for all $m \in \mathbb{N}\backslash\{1\}$ that are smaller than k relative to this order.

Of course, if k is prime then the claim given by the theorem holds.
So suppose that k is not prime. By definition of primality, this means that there exist $m_1, m_2 \in \mathbb{N}\backslash\{1\}$ such that $k = m_1 \cdot m_2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Well-founded Induction

- We give a proof of the existance claim made by the Fundamental Theorem of Arithmetic (Thm. 102): Every natural number $n > 1$ is either a prime number or has a prime factorization.

Proof: We begin with observing that the relation “is genuine divisor of” (Def. 93) over $\mathbb{N}\backslash\{1\}$ is well-founded. The minimal elements relative to this relation are the primes.

We consider an arbitrary but fixed $k \in \mathbb{N}\backslash\{1\}$ and assume as inductive hypothesis that the claim holds for all $m \in \mathbb{N}\backslash\{1\}$ that are smaller than k relative to this order.

Of course, if k is prime then the claim given by the theorem holds.
So suppose that k is not prime. By definition of primality, this means that there exist $m_1, m_2 \in \mathbb{N}\backslash\{1\}$ such that $k = m_1 \cdot m_2$.

Now we have

$$m_1 \text{ is genuine divisor of } k \quad \text{and} \quad m_2 \text{ is genuine divisor of } k.$$

Hence, both m_1 and m_2 are predecessors of k.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- We give a proof of the existance claim made by the Fundamental Theorem of Arithmetic (Thm. 102): Every natural number $n > 1$ is either a prime number or has a prime factorization.

Proof : We begin with observing that the relation “is genuine divisor of” (Def. 93) over $\mathbb{N}\backslash\{1\}$ is well-founded. The minimal elements relative to this relation are the primes.

We consider an arbitrary but fixed $k \in \mathbb{N}\backslash\{1\}$ and assume as inductive hypothesis that the claim holds for all $m \in \mathbb{N}\backslash\{1\}$ that are smaller than k relative to this order.

Of course, if k is prime then the claim given by the theorem holds.
So suppose that k is not prime. By definition of primality, this means that there exist $m_1, m_2 \in \mathbb{N}\backslash\{1\}$ such that $k = m_1 \cdot m_2$.

Now we have

$$m_1 \text{ is genuine divisor of } k \quad \text{and} \quad m_2 \text{ is genuine divisor of } k.$$

Hence, both m_1 and m_2 are predecessors of k. By the inductive hypothesis, we know that m_1 is either prime or has a prime factorization; same for m_2. Thus, also k has a prime factorization, which establishes the inductive step. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Partial Order on Recursive Structures

- Many structures in computer science are defined recursively:
  1. There are one or more base cases that allow to create an instance of that structure from scratch.

Partial Order on Recursive Structures

- Many structures in computer science are defined recursively:
  1. There are one or more base cases that allow to create an instance of that structure from scratch.
  2. There are recursive rules (“constructors”) that take multiple instances of that structure and combine them to form a new instance of that structure.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Partial Order on Recursive Structures

- Many structures in computer science are defined recursively:
  1. There are one or more base cases that allow to create an instance of that structure from scratch.
  2. There are recursive rules (“constructors”) that take multiple instances of that structure and combine them to form a new instance of that structure.
- E.g., recall the definition of words (Def. 35).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Partial Order on Recursive Structures

- Many structures in computer science are defined recursively:
  1. There are one or more base cases that allow to create an instance of that structure from scratch.
  2. There are recursive rules (“constructors”) that take multiple instances of that structure and combine them to form a new instance of that structure.
- E.g., recall the definition of words (Def. 35).
- A key aspect is that every instance of the structure is obtained by applying a finite number of constructors.

Partial Order on Recursive Structures

- Many structures in computer science are defined recursively:
  1. There are one or more base cases that allow to create an instance of that structure from scratch.
  2. There are recursive rules ("constructors") that take multiple instances of that structure and combine them to form a new instance of that structure.
- E.g., recall the definition of words (Def. 35).
- A key aspect is that every instance of the structure is obtained by applying a finite number of constructors.
- The fact that "complex" instances of such a structure are obtained from "simpler" instances by means of constructors suggests that one can define a comparison among them.

Partial Order on Recursive Structures

- Many structures in computer science are defined recursively:
  1. There are one or more base cases that allow to create an instance of that structure from scratch.
  2. There are recursive rules ("constructors") that take multiple instances of that structure and combine them to form a new instance of that structure.
- E.g., recall the definition of words (Def. 35).
- A key aspect is that every instance of the structure is obtained by applying a finite number of constructors.
- The fact that "complex" instances of such a structure are obtained from "simpler" instances by means of constructors suggests that one can define a comparison among them.
- E.g., for $a \in \Sigma$ and $\sigma, \sigma' \in \Sigma^*$, if $\sigma = a\sigma'$ then we could regard σ' to be "smaller" than σ.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Partial Order on Recursive Structures

- Many structures in computer science are defined recursively:
  1. There are one or more base cases that allow to create an instance of that structure from scratch.
  2. There are recursive rules (“constructors”) that take multiple instances of that structure and combine them to form a new instance of that structure.
- E.g., recall the definition of words (Def. 35).
- A key aspect is that every instance of the structure is obtained by applying a finite number of constructors.
- The fact that “complex” instances of such a structure are obtained from “simpler” instances by means of constructors suggests that one can define a comparison among them.
- E.g., for $a \in \Sigma$ and $\sigma, \sigma' \in \Sigma^*$, if $\sigma = a\sigma'$ then we could regard σ' to be “smaller” than σ.
- More generally, $\sigma' <_\Sigma \sigma$ if and only if σ can be obtained from σ' and other words over Σ by applying constructors finitely often. (Hence, in this case σ' is a *sub-string* of σ.)
- Easy to prove: $<_\Sigma$ is a well-founded partial order on Σ^*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 155 (Structural induction)

Let S be a recursively defined structure, and P be a predicate on S.

Structural Induction

Theorem 155 (Structural induction)

Let S be a recursively defined structure, and P be a predicate on S.
If

$P(s)$ for every instance $s \in S$ specified in the base case(s),

Theorem 155 (Structural induction)

Let S be a recursively defined structure, and P be a predicate on S.
If

> *$P(s)$ for every instance $s \in S$ specified in the base case(s),*

and if

> *$P(s)$ for every instance $s \in S$ under the assumption $P(s_1), P(s_2), \ldots, P(s_k)$, for some suitable $k \in \mathbb{N}$, if s can be obtained in one recursive construction step from $s_1, s_2, \ldots, s_k \in S$,*

Theorem 155 (Structural induction)

Let S be a recursively defined structure, and P be a predicate on S.
If

$P(s)$ for every instance $s \in S$ specified in the base case(s),

and if

$P(s)$ for every instance $s \in S$ under the assumption $P(s_1), P(s_2), \ldots, P(s_k)$, for some suitable $k \in \mathbb{N}$, if s can be obtained in one recursive construction step from $s_1, s_2, \ldots, s_k \in S$,

then

$\forall s \in S \;\; P(s)$.

Theorem 155 (Structural induction)

Let S be a recursively defined structure, and P be a predicate on S.
If

$P(s)$ for every instance $s \in S$ specified in the base case(s),

and if

$P(s)$ for every instance $s \in S$ under the assumption $P(s_1), P(s_2), \ldots, P(s_k)$, for some suitable $k \in \mathbb{N}$, if s can be obtained in one recursive construction step from $s_1, s_2, \ldots, s_k \in S$,

then

$\forall s \in S \;\; P(s)$.

- Structural induction can be seen as a special case of a well-founded induction.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Structural Induction

Lemma 156

Let Σ be a finite set. For every $\sigma \in \Sigma^*$ we have $\sigma \bullet \epsilon = \epsilon \bullet \sigma = \sigma$.

Proof: Def. 37 immediately gives $\epsilon \bullet \sigma = \sigma$ for all $\sigma \in \Sigma^*$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Structural Induction

Lemma 156

Let Σ be a finite set. For every $\sigma \in \Sigma^*$ we have $\sigma \bullet \epsilon = \epsilon \bullet \sigma = \sigma$.

Proof: Def. 37 immediately gives $\epsilon \bullet \sigma = \sigma$ for all $\sigma \in \Sigma^*$. We prove $\sigma \bullet \epsilon = \sigma$ by means of structural induction.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Structural Induction

Lemma 156

Let Σ be a finite set. For every $\sigma \in \Sigma^*$ we have $\sigma \bullet \epsilon = \epsilon \bullet \sigma = \sigma$.

Proof: Def. 37 immediately gives $\epsilon \bullet \sigma = \sigma$ for all $\sigma \in \Sigma^*$. We prove $\sigma \bullet \epsilon = \sigma$ by means of structural induction.

The empty word ϵ is the only minimal element stated in the base case of the definition of Σ^*, and we have

$$\epsilon \bullet \epsilon \stackrel{\text{Def. } 37}{=} \epsilon.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Structural Induction

Lemma 156

Let Σ be a finite set. For every $\sigma \in \Sigma^*$ we have $\sigma \bullet \epsilon = \epsilon \bullet \sigma = \sigma$.

Proof: Def. 37 immediately gives $\epsilon \bullet \sigma = \sigma$ for all $\sigma \in \Sigma^*$. We prove $\sigma \bullet \epsilon = \sigma$ by means of structural induction.
The empty word ϵ is the only minimal element stated in the base case of the definition of Σ^*, and we have

$$\epsilon \bullet \epsilon \overset{Def.\ 37}{=} \epsilon.$$

Now consider an arbitrary but fixed word $\sigma \in \Sigma^*$ with $\sigma \neq \epsilon$. Then there exist $a \in \Sigma$ and $\sigma' \in \Sigma^*$ such that $\sigma = a\sigma' = (a, \sigma')$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 156

Let Σ be a finite set. For every $\sigma \in \Sigma^*$ we have $\sigma \bullet \epsilon = \epsilon \bullet \sigma = \sigma$.

Proof: Def. 37 immediately gives $\epsilon \bullet \sigma = \sigma$ for all $\sigma \in \Sigma^*$. We prove $\sigma \bullet \epsilon = \sigma$ by means of structural induction.
The empty word ϵ is the only minimal element stated in the base case of the definition of Σ^*, and we have

$$\epsilon \bullet \epsilon \stackrel{Def.\ 37}{=} \epsilon.$$

Now consider an arbitrary but fixed word $\sigma \in \Sigma^*$ with $\sigma \neq \epsilon$. Then there exist $a \in \Sigma$ and $\sigma' \in \Sigma^*$ such that $\sigma = a\sigma' = (a, \sigma')$. Suppose as I.H. that $\sigma' \bullet \epsilon = \sigma'$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 156

Let Σ be a finite set. For every $\sigma \in \Sigma^*$ we have $\sigma \bullet \epsilon = \epsilon \bullet \sigma = \sigma$.

Proof: Def. 37 immediately gives $\epsilon \bullet \sigma = \sigma$ for all $\sigma \in \Sigma^*$. We prove $\sigma \bullet \epsilon = \sigma$ by means of structural induction.
The empty word ϵ is the only minimal element stated in the base case of the definition of Σ^*, and we have

$$\epsilon \bullet \epsilon \stackrel{Def. 37}{=} \epsilon.$$

Now consider an arbitrary but fixed word $\sigma \in \Sigma^*$ with $\sigma \neq \epsilon$. Then there exist $a \in \Sigma$ and $\sigma' \in \Sigma^*$ such that $\sigma = a\sigma' = (a, \sigma')$. Suppose as I.H. that $\sigma' \bullet \epsilon = \sigma'$. We get

$$\sigma \bullet \epsilon = (a, \sigma') \bullet \epsilon \stackrel{Def. 37}{=} (a, \sigma' \bullet \epsilon) \stackrel{I.H.}{=} (a, \sigma') = \sigma.$$

□

Real-World Application: Functional Completeness of NAND

Theorem 157 (Functional completeness of NAND)

The NAND junctor, $\uparrow$, is (truth-functionally) complete.

- Thus, every formula of propositional logic has a logically equivalent formula that uses only NAND junctors.
- Hence, any digital circuit can be realized by using only one type of gate: NAND gates. (This is also true for the NOR inverter.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Functional Completeness of NAND

Theorem 157 (Functional completeness of NAND)

The NAND junctor, $\uparrow$, is (truth-functionally) complete.

- Thus, every formula of propositional logic has a logically equivalent formula that uses only NAND junctors.
- Hence, any digital circuit can be realized by using only one type of gate: NAND gates. (This is also true for the NOR inverter.)

Lemma 158

Let p, q denote two Boolean variables. The following logical equivalences hold:

$$\neg p \equiv (p \uparrow p) \qquad (p \wedge q) \equiv ((p \uparrow q) \uparrow (p \uparrow q)) \qquad (p \vee q) \equiv ((p \uparrow p) \uparrow (q \uparrow q))$$

$$(p \Rightarrow q) \equiv (\neg p \vee q) \qquad (p \Leftrightarrow q) \equiv ((p \Rightarrow q) \wedge (q \Rightarrow p))$$

$$\top \equiv (p \uparrow (p \uparrow p)) \qquad \bot \equiv (\top \uparrow \top)$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Functional Completeness of NAND

Proof of Thm. 157: Recall Def. 2: Propositional formulas (over some fixed set of n propositional variables $p_1, p_2, \dots, p_n$) follow a rigid recursive construction scheme.

Real-World Application: Functional Completeness of NAND

Proof of Thm. 157 : Recall Def. 2: Propositional formulas (over some fixed set of n propositional variables $p_1, p_2, \ldots, p_n$) follow a rigid recursive construction scheme. Hence, we may use structural induction:

1. The minimal elements of the base case are given by the variables $p_1, p_2, \ldots, p_n$ and the constants $\perp$ and $\top$. Lem. 158 tells us that $\perp$ and $\top$ can be expressed using NAND junctors.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Functional Completeness of NAND

Proof of Thm. 157: Recall Def. 2: Propositional formulas (over some fixed set of n propositional variables $p_1, p_2, \ldots, p_n$) follow a rigid recursive construction scheme. Hence, we may use structural induction:

1. The minimal elements of the base case are given by the variables $p_1, p_2, \ldots, p_n$ and the constants $\bot$ and $\top$. Lem. 158 tells us that $\bot$ and $\top$ can be expressed using NAND junctors.
2. Consider an arbitrary but fixed propositional formula ϕ_0 that contains at least one junctor.

Real-World Application: Functional Completeness of NAND

Proof of Thm. 157: Recall Def. 2: Propositional formulas (over some fixed set of n propositional variables $p_1, p_2, \ldots, p_n$) follow a rigid recursive construction scheme. Hence, we may use structural induction:

1. The minimal elements of the base case are given by the variables $p_1, p_2, \ldots, p_n$ and the constants $\perp$ and $\top$. Lem. 158 tells us that $\perp$ and $\top$ can be expressed using NAND junctors.
2. Consider an arbitrary but fixed propositional formula ϕ_0 that contains at least one junctor. By the construction scheme of propositional formulas, the formula ϕ_0 is of the form $(\neg\phi_1)$ or $(\phi_1 \# \phi_2)$, for suitable propositional formulas ϕ_1, ϕ_2 and where $\#$ is one of the junctors $\wedge, \vee, \Leftrightarrow, \Rightarrow$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proof of Thm. 157 : Recall Def. 2: Propositional formulas (over some fixed set of n propositional variables $p_1, p_2, \ldots, p_n$) follow a rigid recursive construction scheme. Hence, we may use structural induction:

1. The minimal elements of the base case are given by the variables $p_1, p_2, \ldots, p_n$ and the constants $\bot$ and $\top$. Lem. 158 tells us that $\bot$ and $\top$ can be expressed using NAND junctors.
2. Consider an arbitrary but fixed propositional formula ϕ_0 that contains at least one junctor. By the construction scheme of propositional formulas, the formula ϕ_0 is of the form $(\neg\phi_1)$ or $(\phi_1 \# \phi_2)$, for suitable propositional formulas ϕ_1, ϕ_2 and where $\#$ is one of the junctors $\wedge, \vee, \Leftrightarrow, \Rightarrow$.

   Assume as inductive hypothesis that ϕ_1, ϕ_2 can be expressed using only NAND junctors (or are simply variables).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Functional Completeness of NAND

Proof of Thm. 157 : Recall Def. 2: Propositional formulas (over some fixed set of n propositional variables $p_1, p_2, \ldots, p_n$) follow a rigid recursive construction scheme. Hence, we may use structural induction:

1. The minimal elements of the base case are given by the variables $p_1, p_2, \ldots, p_n$ and the constants $\bot$ and $\top$. Lem. 158 tells us that $\bot$ and $\top$ can be expressed using NAND junctors.
2. Consider an arbitrary but fixed propositional formula ϕ_0 that contains at least one junctor. By the construction scheme of propositional formulas, the formula ϕ_0 is of the form $(\neg\phi_1)$ or $(\phi_1 \# \phi_2)$, for suitable propositional formulas ϕ_1, ϕ_2 and where $\#$ is one of the junctors $\wedge, \vee, \Leftrightarrow, \Rightarrow$.

   Assume as inductive hypothesis that ϕ_1, ϕ_2 can be expressed using only NAND junctors (or are simply variables).

   By using the scheme outlined in Lem. 158, also ϕ_0 can be expressed using only NAND junctors.

□

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

5 Principles of Elementary Counting and Combinatorics

- Sum and Product Rule
- Inclusion-Exclusion Principle
- Binomial Coefficient
- Permutations
- Ordered Selection (Variation)
- Unordered Selection (Combination)

5 Principles of Elementary Counting and Combinatorics

- Sum and Product Rule
- Inclusion-Exclusion Principle
- Binomial Coefficient
- Permutations
- Ordered Selection (Variation)
- Unordered Selection (Combination)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sum and Product Rule

Theorem 159 (Sum rule, Dt.: Additionsprinzip)

Let A, B be two finite sets with $A \cap B = \emptyset$. Then

$$|A \cup B| = |A| + |B|.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sum and Product Rule

Theorem 159 (Sum rule, Dt.: Additionsprinzip)

Let A, B be two finite sets with $A \cap B = \emptyset$. Then

$$|A \cup B| = |A| + |B|.$$

Corollary 160

For $n \in \mathbb{N}$, let $A_1, A_2, \ldots, A_n$ be n finite sets that are pairwise disjoint. Then

$$|A_1 \cup A_2 \cup \ldots \cup A_n| = \sum_{i=1}^{n} |A_i|.$$

Theorem 159 (Sum rule, Dt.: Additionsprinzip)

Let A, B be two finite sets with $A \cap B = \emptyset$. Then

$$|A \cup B| = |A| + |B|.$$

Corollary 160

For $n \in \mathbb{N}$, let $A_1, A_2, \ldots, A_n$ be n finite sets that are pairwise disjoint. Then

$$|A_1 \cup A_2 \cup \ldots \cup A_n| = \sum_{i=1}^{n} |A_i|.$$

Theorem 161 (Product rule, Dt.: Multiplikationsprinzip)

Let A, B be two finite sets. Then

$$|A \times B| = |A| \cdot |B|.$$

Sum and Product Rule

Proof of Theorem 161 :

- We observe that

$$A \times B = \bigcup_{b \in B} (A \times \{b\}), \quad \text{with } (A \times \{b_1\}) \cap (A \times \{b_2\}) = \emptyset \text{ if } b_1 \neq b_2.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proof of Theorem 161 :

- We observe that

$$A \times B = \bigcup_{b \in B} (A \times \{b\}), \quad \text{with } (A \times \{b_1\}) \cap (A \times \{b_2\}) = \emptyset \text{ if } b_1 \neq b_2.$$

- There exists a bijective mapping between A and $A \times \{b\}$ for every $b \in B$. Thus, $|A| = |A \times \{b\}|$, and the theorem is a consequence of Corollary 160. □

Proof of Theorem 161 :

- We observe that

$$A \times B = \bigcup_{b \in B} (A \times \{b\}), \quad \text{with } (A \times \{b_1\}) \cap (A \times \{b_2\}) = \emptyset \text{ if } b_1 \neq b_2.$$

- There exists a bijective mapping between A and $A \times \{b\}$ for every $b \in B$. Thus, $|A| = |A \times \{b\}|$, and the theorem is a consequence of Corollary 160. □

Corollary 162

For $n \in \mathbb{N}$, let $A_1, A_2, \ldots, A_n$ be n finite sets. Then

$$|A_1 \times A_2 \times \ldots \times A_n| = \prod_{i=1}^{n} |A_i|.$$

Proof of Theorem 161 :

- We observe that

$$A \times B = \bigcup_{b \in B} (A \times \{b\}), \quad \text{with } (A \times \{b_1\}) \cap (A \times \{b_2\}) = \emptyset \text{ if } b_1 \neq b_2.$$

- There exists a bijective mapping between A and $A \times \{b\}$ for every $b \in B$. Thus, $|A| = |A \times \{b\}|$, and the theorem is a consequence of Corollary 160. □

Corollary 162

For $n \in \mathbb{N}$, let $A_1, A_2, \ldots, A_n$ be n finite sets. Then

$$|A_1 \times A_2 \times \ldots \times A_n| = \prod_{i=1}^{n} |A_i|.$$

Corollary 163

For a propositional formula that contains n variables, 2^n evaluations are necessary in order to test all possible combinations of truth assignments to its variables.

Characteristic Function and Cardinality of Power Set

Definition 164 (Characteristic function, Dt.: Indikatorfunktion)

Let A be a finite set, and $B \subseteq A$. The *characteristic function* $1_B : A \to \{0, 1\}$ indicates membership of an element of A in B:

$$1_B(a) := \begin{cases} 1 & \text{if} \quad a \in B, \\ 0 & \text{if} \quad a \notin B. \end{cases}$$

Characteristic Function and Cardinality of Power Set

Definition 164 (Characteristic function, Dt.: Indikatorfunktion)

Let A be a finite set, and $B \subseteq A$. The *characteristic function* $1_B : A \to \{0, 1\}$ indicates membership of an element of A in B:

$$1_B(a) := \begin{cases} 1 & \text{if} \quad a \in B, \\ 0 & \text{if} \quad a \notin B. \end{cases}$$

Lemma 165

A finite set A has $2^{|A|}$ many different subsets. That is, $|\mathcal{P}(A)| = 2^{|A|}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Characteristic Function and Cardinality of Power Set

Definition 164 (Characteristic function, Dt.: Indikatorfunktion)

Let A be a finite set, and $B \subseteq A$. The *characteristic function* $1_B : A \to \{0, 1\}$ indicates membership of an element of A in B:

$$1_B(a) := \begin{cases} 1 & \text{if} \quad a \in B, \\ 0 & \text{if} \quad a \notin B. \end{cases}$$

Lemma 165

A finite set A has $2^{|A|}$ many different subsets. That is, $|\mathcal{P}(A)| = 2^{|A|}$.

Proof : We observe that every subset of A, including $\emptyset$ and A itself, has a one-to-one correspondance to a characteristic function. Thus, every subset of A corresponds to a sequence of n 0's and 1's, where $n := |A|$. We conclude that the power set $\mathcal{P}(A)$ has 2^n members. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 164 (Characteristic function, Dt.: Indikatorfunktion)

Let A be a finite set, and $B \subseteq A$. The *characteristic function* $1_B : A \to \{0, 1\}$ indicates membership of an element of A in B:

$$1_B(a) := \begin{cases} 1 & \text{if} \quad a \in B, \\ 0 & \text{if} \quad a \notin B. \end{cases}$$

Lemma 165

A finite set A has $2^{|A|}$ many different subsets. That is, $|\mathcal{P}(A)| = 2^{|A|}$.

Proof: We observe that every subset of A, including $\emptyset$ and A itself, has a one-to-one correspondance to a characteristic function. Thus, every subset of A corresponds to a sequence of n 0's and 1's, where $n := |A|$. We conclude that the power set $\mathcal{P}(A)$ has 2^n members. □

Lemma 166

Let A be a finite set, and $B \subseteq A$. Then $|B| = \sum_{a \in A} 1_B(a)$.

Real-World Application: Counting Strings

- How many 3-element strings s can be formed over the standard Latin alphabet — 26 lower-case letters — such that every string contains at least one x?

Real-World Application: Counting Strings

- How many 3-element strings s can be formed over the standard Latin alphabet — 26 lower-case letters — such that every string contains at least one x?
- Obviously such a 3-element string s is in exactly one of the following sets:

$$A_1 := \{s : \text{ first } x \text{ in first place of } s\},$$
$$A_2 := \{s : \text{ first } x \text{ in second place of } s\},$$
$$A_3 := \{s : \text{ first } x \text{ in third place of } s\}.$$

Real-World Application: Counting Strings

- How many 3-element strings s can be formed over the standard Latin alphabet — 26 lower-case letters — such that every string contains at least one x?
- Obviously such a 3-element string s is in exactly one of the following sets:

$$A_1 := \{s : \text{ first } x \text{ in first place of } s\},$$
$$A_2 := \{s : \text{ first } x \text{ in second place of } s\},$$
$$A_3 := \{s : \text{ first } x \text{ in third place of } s\}.$$

- Applying the Product Rule 161 yields

$$|A_1| = |\{x\} \times \{a, b, \ldots, z\} \times \{a, b, \ldots, z\}| = 1 \cdot 26 \cdot 26,$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Counting Strings

- How many 3-element strings s can be formed over the standard Latin alphabet — 26 lower-case letters — such that every string contains at least one x?
- Obviously such a 3-element string s is in exactly one of the following sets:

$$A_1 := \{s : \text{ first } x \text{ in first place of } s\},$$
$$A_2 := \{s : \text{ first } x \text{ in second place of } s\},$$
$$A_3 := \{s : \text{ first } x \text{ in third place of } s\}.$$

- Applying the Product Rule 161 yields

$$|A_1| = |\{x\} \times \{a, b, \ldots, z\} \times \{a, b, \ldots, z\}| = 1 \cdot 26 \cdot 26,$$
$$|A_2| = |(\{a, b, \ldots, z\} \setminus \{x\}) \times \{x\} \times \{a, b, \ldots, z\}| = 25 \cdot 1 \cdot 26,$$

Real-World Application: Counting Strings

- How many 3-element strings s can be formed over the standard Latin alphabet — 26 lower-case letters — such that every string contains at least one x?
- Obviously such a 3-element string s is in exactly one of the following sets:

$$A_1 := \{s : \text{ first } x \text{ in first place of } s\},$$
$$A_2 := \{s : \text{ first } x \text{ in second place of } s\},$$
$$A_3 := \{s : \text{ first } x \text{ in third place of } s\}.$$

- Applying the Product Rule 161 yields

$$|A_1| = |\{x\} \times \{a, b, \ldots, z\} \times \{a, b, \ldots, z\}| = 1 \cdot 26 \cdot 26,$$
$$|A_2| = |(\{a, b, \ldots, z\} \backslash \{x\}) \times \{x\} \times \{a, b, \ldots, z\}| = 25 \cdot 1 \cdot 26,$$
$$|A_3| = |(\{a, b, \ldots, z\} \backslash \{x\}) \times (\{a, b, \ldots, z\} \backslash \{x\}) \times \{x\}| = 25 \cdot 25 \cdot 1.$$

Real-World Application: Counting Strings

- How many 3-element strings s can be formed over the standard Latin alphabet — 26 lower-case letters — such that every string contains at least one x?
- Obviously such a 3-element string s is in exactly one of the following sets:

$$A_1 := \{s : \text{ first } x \text{ in first place of } s\},$$
$$A_2 := \{s : \text{ first } x \text{ in second place of } s\},$$
$$A_3 := \{s : \text{ first } x \text{ in third place of } s\}.$$

- Applying the Product Rule 161 yields

$$|A_1| = |\{x\} \times \{a, b, \ldots, z\} \times \{a, b, \ldots, z\}| = 1 \cdot 26 \cdot 26,$$
$$|A_2| = |(\{a, b, \ldots, z\} \backslash \{x\}) \times \{x\} \times \{a, b, \ldots, z\}| = 25 \cdot 1 \cdot 26,$$
$$|A_3| = |(\{a, b, \ldots, z\} \backslash \{x\}) \times (\{a, b, \ldots, z\} \backslash \{x\}) \times \{x\}| = 25 \cdot 25 \cdot 1.$$

- Since A_1, A_2, A_3 are pairwise disjoint, the Sum Rule 159 implies

$$|A_1 \cup A_2 \cup A_3| = |A_1| + |A_2| + |A_3| = 26 \cdot 26 + 25 \cdot 26 + 25 \cdot 25 = 1951.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Counting Passwords

- Suppose that passwords are limited to strings of six to eight characters, where each character is one of the 26 uppercase letters or a digit. Every password has to contain at least one digit.
- How many different passwords do exist under these restrictions?

Real-World Application: Counting Passwords

- Suppose that passwords are limited to strings of six to eight characters, where each character is one of the 26 uppercase letters or a digit. Every password has to contain at least one digit.
- How many different passwords do exist under these restrictions?
- Let N be the total number of passwords, and let N_6, N_7, N_8 denote the number of passwords with six (seven, eight, resp.) characters.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Counting Passwords

- Suppose that passwords are limited to strings of six to eight characters, where each character is one of the 26 uppercase letters or a digit. Every password has to contain at least one digit.
- How many different passwords do exist under these restrictions?
- Let N be the total number of passwords, and let N_6, N_7, N_8 denote the number of passwords with six (seven, eight, resp.) characters.
- By the Product Rule 161, the total number of six-character strings (over the 26 letters and the 10 digits) is 36^6, with 26^6 of them containing no digit at all. Hence,

$$N_6 = 36^6 - 26^6 = 1\,867\,866\,560.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Counting Passwords

- Suppose that passwords are limited to strings of six to eight characters, where each character is one of the 26 uppercase letters or a digit. Every password has to contain at least one digit.
- How many different passwords do exist under these restrictions?
- Let N be the total number of passwords, and let N_6, N_7, N_8 denote the number of passwords with six (seven, eight, resp.) characters.
- By the Product Rule 161, the total number of six-character strings (over the 26 letters and the 10 digits) is 36^6, with 26^6 of them containing no digit at all. Hence,

$$N_6 = 36^6 - 26^6 = 1\,867\,866\,560.$$

- Similarly,

$$N_7 = 36^7 - 26^7 = 70\,332\,353\,920$$

and

$$N_8 = 36^8 - 26^8 = 2\,612\,282\,842\,880.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Counting Passwords

- Suppose that passwords are limited to strings of six to eight characters, where each character is one of the 26 uppercase letters or a digit. Every password has to contain at least one digit.
- How many different passwords do exist under these restrictions?
- Let N be the total number of passwords, and let N_6, N_7, N_8 denote the number of passwords with six (seven, eight, resp.) characters.
- By the Product Rule 161, the total number of six-character strings (over the 26 letters and the 10 digits) is 36^6, with 26^6 of them containing no digit at all. Hence,

$$N_6 = 36^6 - 26^6 = 1\,867\,866\,560.$$

- Similarly,

$$N_7 = 36^7 - 26^7 = 70\,332\,353\,920$$

and

$$N_8 = 36^8 - 26^8 = 2\,612\,282\,842\,880.$$

- Hence, by the Sum Rule 159,

$$N = N_6 + N_7 + N_8 = 2\,684\,483\,063\,360.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

5 Principles of Elementary Counting and Combinatorics

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Inclusion-Exclusion Principle

Theorem 167 (Inclusion-exclusion principle, Dt.: Siebprinzip, Poincaré-Formel)

Let $A_1, A_2, \ldots, A_n$ be finite sets. Then

$$|\bigcup_{i=1}^{n} A_i| = \sum_{\substack{I \neq \emptyset \\ I \subseteq \{1,\ldots,n\}}} (-1)^{|I|+1} \, |\bigcap_{i \in I} A_i|.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 167 (Inclusion-exclusion principle, Dt.: Siebprinzip, Poincaré-Formel)

Let $A_1, A_2, \ldots, A_n$ be finite sets. Then

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{\substack{I \neq \emptyset \\ I \subseteq \{1,\ldots,n\}}} (-1)^{|I|+1} \left|\bigcap_{i \in I} A_i\right|.$$

- For $|I| = 1$:

$$\sum_{1 \leqslant i \leqslant n} (-1)^{1+1} |A_i| = \sum_{i=1}^{n} |A_i|.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 167 (Inclusion-exclusion principle, Dt.: Siebprinzip, Poincaré-Formel)

Let $A_1, A_2, \ldots, A_n$ be finite sets. Then

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{\substack{I \neq \emptyset \\ I \subseteq \{1,\ldots,n\}}} (-1)^{|I|+1} \left|\bigcap_{i \in I} A_i\right|.$$

- For $|I| = 1$:

$$\sum_{1 \leqslant i \leqslant n} (-1)^{1+1} |A_i| = \sum_{i=1}^{n} |A_i|.$$

- For $|I| = 2$:

$$\sum_{1 \leqslant i < j \leqslant n} (-1)^{2+1} |A_i \cap A_j| = -\sum_{1 \leqslant i < j \leqslant n} |A_i \cap A_j|.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 167 (Inclusion-exclusion principle, Dt.: Siebprinzip, Poincaré-Formel)

Let $A_1, A_2, \ldots, A_n$ be finite sets. Then

$$|\bigcup_{i=1}^{n} A_i| = \sum_{\substack{I \neq \emptyset \\ I \subseteq \{1,\ldots,n\}}} (-1)^{|I|+1} \, |\bigcap_{i \in I} A_i|.$$

- For $|I| = 1$:

$$\sum_{1 \leqslant i \leqslant n} (-1)^{1+1} \, |A_i| = \sum_{i=1}^{n} |A_i|.$$

- For $|I| = 2$:

$$\sum_{1 \leqslant i < j \leqslant n} (-1)^{2+1} \, |A_i \cap A_j| = - \sum_{1 \leqslant i < j \leqslant n} |A_i \cap A_j|.$$

- In particular:

$$|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|$$

$$|A_1 \cup A_2 \cup A_3| = |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |A_1 \cap A_3| - |A_2 \cap A_3| + |A_1 \cap A_2 \cap A_3|$$

Real-World Application: Counting Bit Strings

- How many bit strings of length eight either start with 1 as first bit or end in 00 as the two last bits? (This is a non-exclusive or!)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Counting Bit Strings

- How many bit strings of length eight either start with 1 as first bit or end in 00 as the two last bits? (This is a non-exclusive or!)
- Let A_1 be the set of 8-bit strings that start with 1. Similarly, let A_2 be the set of 8-bit strings that end in 00.
- Then the number sought equals $|A_1 \cup A_2|$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Counting Bit Strings

- How many bit strings of length eight either start with 1 as first bit or end in 00 as the two last bits? (This is a non-exclusive or!)
- Let A_1 be the set of 8-bit strings that start with 1. Similarly, let A_2 be the set of 8-bit strings that end in 00.
- Then the number sought equals $|A_1 \cup A_2|$.
- By the Product Rule 161,

$$|A_1| = 2^7 = 128 \quad \text{and} \quad |A_2| = 2^6 = 64 \quad \text{and} \quad |A_1 \cap A_2| = 2^5 = 32.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Counting Bit Strings

- How many bit strings of length eight either start with 1 as first bit or end in 00 as the two last bits? (This is a non-exclusive or!)
- Let A_1 be the set of 8-bit strings that start with 1. Similarly, let A_2 be the set of 8-bit strings that end in 00.
- Then the number sought equals $|A_1 \cup A_2|$.
- By the Product Rule 161,

$$|A_1| = 2^7 = 128 \quad \text{and} \quad |A_2| = 2^6 = 64 \quad \text{and} \quad |A_1 \cap A_2| = 2^5 = 32.$$

- Hence, by the Inclusion-Exclusion Principle (Thm. 167),

$$|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2| = 128 + 64 - 32 = 160.$$

5 Principles of Elementary Counting and Combinatorics

Definition 168 (Binomial coefficient, Dt.: Binomialkoeffizient)

Let $n \in \mathbb{N}_0$ and $k \in \mathbb{Z}$. The *binomial coefficient* $\binom{n}{k}$ of n and k is defined as follows:

$$\binom{n}{k} := \begin{cases} 0 & \text{if} \quad k < 0, \\ \dfrac{n!}{k! \cdot (n-k)!} & \text{if} \quad 0 \leqslant k \leqslant n, \\ 0 & \text{if} \quad k > n. \end{cases}$$

Definition 168 (Binomial coefficient, Dt.: Binomialkoeffizient)

Let $n \in \mathbb{N}_0$ and $k \in \mathbb{Z}$. The *binomial coefficient* $\binom{n}{k}$ of n and k is defined as follows:

$$\binom{n}{k} := \begin{cases} 0 & \text{if} \quad k < 0, \\ \dfrac{n!}{k! \cdot (n-k)!} & \text{if} \quad 0 \leqslant k \leqslant n, \\ 0 & \text{if} \quad k > n. \end{cases}$$

- Recall $k! := 1$ for $k := 0$.
- The binomial coefficient $\binom{n}{k}$ is pronounced as "n choose k"; Dt.: "n über k".

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 168 (Binomial coefficient, Dt.: Binomialkoeffizient)

Let $n \in \mathbb{N}_0$ and $k \in \mathbb{Z}$. The *binomial coefficient* $\binom{n}{k}$ of n and k is defined as follows:

$$\binom{n}{k} := \begin{cases} 0 & \text{if} \quad k < 0, \\ \dfrac{n!}{k! \cdot (n-k)!} & \text{if} \quad 0 \leqslant k \leqslant n, \\ 0 & \text{if} \quad k > n. \end{cases}$$

- Recall $k! := 1$ for $k := 0$.
- The binomial coefficient $\binom{n}{k}$ is pronounced as "n choose k"; Dt.: "n über k".

Lemma 169

Let $n \in \mathbb{N}_0$ and $k \in \mathbb{Z}$.

$$\binom{n}{0} = \binom{n}{n} = 1 \qquad \binom{n}{1} = \binom{n}{n-1} = n \qquad \binom{n}{k} = \binom{n}{n-k}$$

Binomial Coefficients

- The following table contains the non-zero values of $\binom{n}{k}$ for $0 \leqslant n, k \leqslant 6$.

	k						
n	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	1						
1	1	1					
2	1	2	1				
3	1	3	3	1			
4	1	4	6	4	1		
5	1	5	10	10	5	1	
6	1	6	15	20	15	6	1

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Binomial Coefficients

- The following table contains the non-zero values of $\binom{n}{k}$ for $0 \leqslant n, k \leqslant 6$.

	k						
n	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	1						
1	1	1					
2	1	2	1				
3	1	3	3	1			
4	1	4	6	4	1		
5	1	5	10	10	5	1	
6	1	6	15	20	15	6	1

- Trivial to observe:
  - Each row begins and ends with 1.
  - Initially each row contains increasing numbers till its middle but then the numbers start to decrease.
  - Each row's first half is exactly the mirror image of its second half.

Binomial Coefficients: Pascal's Triangle

- A simple rearrangement of the previous table yields what is known as *Pascal's Triangle* in the Western world (Blaise Pascal, 1623–1662). But it was already studied in India in the 10th century, and discussed by Omar Khayyam (1048–1131)!

						1						
					1		1					
				1		2		1				
			1		3		3		1			
		1		4		6		4		1		
	1		5		10		10		5		1	
1		6		15		20		15		6		1

Binomial Coefficients: Pascal’s Triangle

- A simple rearrangement of the previous table yields what is known as *Pascal’s Triangle* in the Western world (Blaise Pascal, 1623–1662). But it was already studied in India in the 10th century, and discussed by Omar Khayyam (1048–1131)!

						1						
					1		1					
				1		2		1				
			1		3		3		1			
		1		4		6		4		1		
	1		5		10		10		5		1	
1		6		15		20		15		6		1

- All entries in this triangle, except for the left-most and right-most entries per row, are the sum of the two entries above them in the previous row.

Binomial Coefficients: Pascal's Triangle

- A simple rearrangement of the previous table yields what is known as *Pascal's Triangle* in the Western world (Blaise Pascal, 1623–1662). But it was already studied in India in the 10th century, and discussed by Omar Khayyam (1048–1131)!

						1						
					1		1					
				1		2		1				
			1		3		3		1			
		1		4		6		4		1		
	1		5		10		10		5		1	
1		6		15		20		15		6		1

- All entries in this triangle, except for the left-most and right-most entries per row, are the sum of the two entries above them in the previous row.

Theorem 170 (Khayyam, Yang Hui, Tartaglia, Pascal)

For $n \in \mathbb{N}$ and $k \in \mathbb{Z}$,

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}.$$

Binomial Theorem

- We know: $(a+b)^2 = a^2 + 2ab + b^2$ and $(a+b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$.

Binomial Theorem

- We know: $(a+b)^2 = a^2 + 2ab + b^2$ and $(a+b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$.

Theorem 171 (Binomial Theorem, Dt.: Binomischer Lehrsatz)

For all $n \in \mathbb{N}_0$ and $a, b \in \mathbb{R}$,

$$(a+b)^n = \binom{n}{0}a^n + \binom{n}{1}a^{n-1}b + \cdots + \binom{n}{n}b^n$$

or, equivalently,

$$(a+b)^n = \sum_{i=0}^{n}\binom{n}{i}a^{n-i}b^i.$$

Binomial Theorem

- We know: $(a+b)^2 = a^2 + 2ab + b^2$ and $(a+b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$.

Theorem 171 (Binomial Theorem, Dt.: Binomischer Lehrsatz)

For all $n \in \mathbb{N}_0$ and $a, b \in \mathbb{R}$,

$$(a+b)^n = \binom{n}{0} a^n + \binom{n}{1} a^{n-1} b + \cdots + \binom{n}{n} b^n$$

or, equivalently,

$$(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^{n-i} b^i.$$

Corollary 172

For all $n \in \mathbb{N}$ and all $x \in \mathbb{R}$:

$$\sum_{i=0}^{n} \binom{n}{i} x^i = (1+x)^n \qquad \sum_{i=0}^{n} \binom{n}{i} = 2^n \qquad \sum_{i=0}^{n} (-1)^i \binom{n}{i} = 0$$

5 Principles of Elementary Counting and Combinatorics

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 173 (Permutation)

Let A be a finite set. A *permutation* of A is a bijective function from A to A.

Definition 173 (Permutation)

Let A be a finite set. A *permutation* of A is a bijective function from A to A.

- A permutation on a finite set A of cardinality n can be regarded as an (ordered) sequence of length n in which every element of A appears exactly once.

Definition 173 (Permutation)

Let A be a finite set. A *permutation* of A is a bijective function from A to A.

- A permutation on a finite set A of cardinality n can be regarded as an (ordered) sequence of length n in which every element of A appears exactly once.
- Many encryption schemes used in cryptography can be seen as permutations.

Definition 173 (Permutation)

Let A be a finite set. A *permutation* of A is a bijective function from A to A.

- A permutation on a finite set A of cardinality n can be regarded as an (ordered) sequence of length n in which every element of A appears exactly once.
- Many encryption schemes used in cryptography can be seen as permutations.
- Standard notation for a permutation π of $\{1, 2, \ldots, n\}$:

$$\begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ \pi(1) & \pi(2) & \pi(3) & \ldots & \pi(n) \end{pmatrix}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 173 (Permutation)

Let A be a finite set. A *permutation* of A is a bijective function from A to A.

- A permutation on a finite set A of cardinality n can be regarded as an (ordered) sequence of length n in which every element of A appears exactly once.
- Many encryption schemes used in cryptography can be seen as permutations.
- Standard notation for a permutation π of $\{1, 2, \ldots, n\}$:

$$\begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ \pi(1) & \pi(2) & \pi(3) & \ldots & \pi(n) \end{pmatrix}$$

- E.g., for $n := 4$:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 1 & 4 \end{pmatrix}$$

Definition 173 (Permutation)

Let A be a finite set. A *permutation* of A is a bijective function from A to A.

- A permutation on a finite set A of cardinality n can be regarded as an (ordered) sequence of length n in which every element of A appears exactly once.
- Many encryption schemes used in cryptography can be seen as permutations.
- Standard notation for a permutation π of $\{1, 2, \ldots, n\}$:

$$\begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ \pi(1) & \pi(2) & \pi(3) & \ldots & \pi(n) \end{pmatrix}$$

- E.g., for $n := 4$:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 1 & 4 \end{pmatrix}$$

Definition 174 (Product of permutations)

Let A be a finite set together with two permutations α, β. Then the *product* (or *composition*) $\alpha \circ \beta$ is the function

$$\alpha \circ \beta : A \to A \quad \text{with} \quad (\alpha \circ \beta)(a) := \alpha(\beta(a)) \quad \text{for all} \quad a \in A.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Permutations

- The product of two permutations is itself a bijective function, i.e., a permutation.
- Note: It is common to drop $\circ$ in $\alpha \circ \beta$ and simply write $\alpha\beta$.

Permutations

- The product of two permutations is itself a bijective function, i.e., a permutation.
- Note: It is common to drop $\circ$ in $\alpha \circ \beta$ and simply write $\alpha\beta$.
- The product of two permutations is not commutative.

$$\alpha := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 3 & 4 & 2 \end{pmatrix} \qquad \beta := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 3 & 4 \end{pmatrix}$$

$$\alpha\beta = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 1 & 4 & 2 \end{pmatrix} \qquad \beta\alpha = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \end{pmatrix}$$

Permutations

- The product of two permutations is itself a bijective function, i.e., a permutation.
- Note: It is common to drop $\circ$ in $\alpha \circ \beta$ and simply write $\alpha\beta$.
- The product of two permutations is not commutative.

$$\alpha := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 3 & 4 & 2 \end{pmatrix} \qquad \beta := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 3 & 4 \end{pmatrix}$$

$$\alpha\beta = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 1 & 4 & 2 \end{pmatrix} \qquad \beta\alpha = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \end{pmatrix}$$

Lemma 175

For all $n \in \mathbb{N}$ and all finite sets A with $n = |A|$, the set of all permutations, S_n, over A together with $\circ$ as operation forms a group, the so-called *symmetric group*.

Permutations

- The product of two permutations is itself a bijective function, i.e., a permutation.
- Note: It is common to drop $\circ$ in $\alpha \circ \beta$ and simply write $\alpha\beta$.
- The product of two permutations is not commutative.

$$\alpha := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 3 & 4 & 2 \end{pmatrix} \qquad \beta := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 3 & 4 \end{pmatrix}$$

$$\alpha\beta = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 1 & 4 & 2 \end{pmatrix} \qquad \beta\alpha = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \end{pmatrix}$$

Lemma 175

For all $n \in \mathbb{N}$ and all finite sets A with $n = |A|$, the set of all permutations, S_n, over A together with $\circ$ as operation forms a group, the so-called *symmetric group*.

- Common assumption when talking about S_n: We have $A := \{1, 2, \ldots, n\}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Permutations

- The product of two permutations is itself a bijective function, i.e., a permutation.
- Note: It is common to drop $\circ$ in $\alpha \circ \beta$ and simply write $\alpha\beta$.
- The product of two permutations is not commutative.

$$\alpha := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 3 & 4 & 2 \end{pmatrix} \qquad \beta := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 3 & 4 \end{pmatrix}$$

$$\alpha\beta = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 1 & 4 & 2 \end{pmatrix} \qquad \beta\alpha = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \end{pmatrix}$$

Lemma 175

For all $n \in \mathbb{N}$ and all finite sets A with $n = |A|$, the set of all permutations, S_n, over A together with $\circ$ as operation forms a group, the so-called *symmetric group*.

- Common assumption when talking about S_n: We have $A := \{1, 2, \ldots, n\}$.

Lemma 176

For all $n \in \mathbb{N}$ and all finite sets A with $n = |A|$, the group $(S_n, \circ)$ is a finite group with exactly $n!$ members.

Definition 177 (Cycle, Dt.: Zyklus)

Let A be a finite set of cardinality n. A permutation π of A is a *cycle of length* $k \leqslant n$ if there exists a set $B \subseteq A$ with $|B| = k$ such that, with $B := \{b_1, b_2, \ldots, b_k\}$,

$$\pi(b_1) = b_2, \quad \pi(b_2) = b_3, \quad \ldots, \quad \pi(b_{k-1}) = b_k, \quad \pi(b_k) = b_1,$$

and $\pi(a) = a$ for all $a \in A \backslash B$. In this case this k-cycle is written as

$$(b_1 \;\; b_2 \;\; \ldots \;\; b_k) \qquad \text{or as} \qquad b_1 \mapsto b_2 \mapsto \ldots \mapsto b_k \mapsto b_1.$$

A cycle is *non-trivial* if $k \geqslant 2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 177 (Cycle, Dt.: Zyklus)

Let A be a finite set of cardinality n. A permutation π of A is a *cycle of length* $k \leqslant n$ if there exists a set $B \subseteq A$ with $|B| = k$ such that, with $B := \{b_1, b_2, \ldots, b_k\}$,

$$\pi(b_1) = b_2, \quad \pi(b_2) = b_3, \quad \ldots, \quad \pi(b_{k-1}) = b_k, \quad \pi(b_k) = b_1,$$

and $\pi(a) = a$ for all $a \in A \backslash B$. In this case this k-cycle is written as

$$(b_1 \quad b_2 \quad \ldots \quad b_k) \qquad \text{or as} \qquad b_1 \mapsto b_2 \mapsto \ldots \mapsto b_k \mapsto b_1.$$

A cycle is *non-trivial* if $k \geqslant 2$.

Definition 178 (Transposition)

A *transposition* is a cycle of length two, aka 2-cycle.

Definition 177 (Cycle, Dt.: Zyklus)

Let A be a finite set of cardinality n. A permutation π of A is a *cycle of length* $k \leqslant n$ if there exists a set $B \subseteq A$ with $|B| = k$ such that, with $B := \{b_1, b_2, \ldots, b_k\}$,

$$\pi(b_1) = b_2, \quad \pi(b_2) = b_3, \quad \ldots, \quad \pi(b_{k-1}) = b_k, \quad \pi(b_k) = b_1,$$

and $\pi(a) = a$ for all $a \in A \backslash B$. In this case this k-cycle is written as

$$(b_1 \;\; b_2 \;\; \ldots \;\; b_k) \quad \text{or as} \quad b_1 \mapsto b_2 \mapsto \ldots \mapsto b_k \mapsto b_1.$$

A cycle is *non-trivial* if $k \geqslant 2$.

Definition 178 (Transposition)

A *transposition* is a cycle of length two, aka 2-cycle.

Lemma 179

Every permutation (of two or more elements) can be written as

(1) a product of cycles,

(2) a product of transpositions.

Lemma 180

If two different products of transpositions correspond to the same permutation then both products consist of either an even or an odd number of transpositions.

Lemma 180

If two different products of transpositions correspond to the same permutation then both products consist of either an even or an odd number of transpositions.

Definition 181 (Signature, Dt.: Signum)

The *signature* of a permutation is $+1$, and the permutation is *even*, if it consists of an even number of transpositions. Otherwise, the signature is -1 and the permutation is *odd*.

Lemma 180

If two different products of transpositions correspond to the same permutation then both products consist of either an even or an odd number of transpositions.

Definition 181 (Signature, Dt.: Signum)

The *signature* of a permutation is $+1$, and the permutation is *even*, if it consists of an even number of transpositions. Otherwise, the signature is -1 and the permutation is *odd*.

Definition 182 (Derangement, Dt.: Permutation ohne Fixpunkt)

A permutation π of A is a *derangement* if $\pi(a) \neq a$ for all $a \in A$.

Lemma 180

If two different products of transpositions correspond to the same permutation then both products consist of either an even or an odd number of transpositions.

Definition 181 (Signature, Dt.: Signum)

The *signature* of a permutation is $+1$, and the permutation is *even*, if it consists of an even number of transpositions. Otherwise, the signature is -1 and the permutation is *odd*.

Definition 182 (Derangement, Dt.: Permutation ohne Fixpunkt)

A permutation π of A is a *derangement* if $\pi(a) \neq a$ for all $a \in A$.

- The Christmas tradition “Secret Santa” (Dt.: Wichteln) is based on an (unknown) derangement.

Lemma 180

If two different products of transpositions correspond to the same permutation then both products consist of either an even or an odd number of transpositions.

Definition 181 (Signature, Dt.: Signum)

The *signature* of a permutation is $+1$, and the permutation is *even*, if it consists of an even number of transpositions. Otherwise, the signature is -1 and the permutation is *odd*.

Definition 182 (Derangement, Dt.: Permutation ohne Fixpunkt)

A permutation π of A is a *derangement* if $\pi(a) \neq a$ for all $a \in A$.

- The Christmas tradition “Secret Santa” (Dt.: Wichteln) is based on an (unknown) derangement.

Definition 183 (Inversion, Dt.: Inversion, Fehlstand)

A permutation $\pi \in S_n$ has an *inversion* (i, j) if $\pi(i) > \pi(j)$ for $1 \leqslant i < j \leqslant n$.

5 Principles of Elementary Counting and Combinatorics

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 184 (Ordered selection without repetition, Dt.: Variation ohne Zurücklegen, Variation ohne Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. An *ordered selection without repetition* of k elements from A is a k-tuple

$$(a_1, a_2, \ldots, a_k) \quad \text{with} \quad a_i \in A \quad \text{for} \quad i = 1, 2, \ldots, k \quad \text{and} \quad a_i \neq a_j \quad \text{for} \quad 1 \leqslant i < j \leqslant k.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 184 (Ordered selection without repetition, Dt.: Variation ohne Zurücklegen, Variation ohne Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. An *ordered selection without repetition* of k elements from A is a k-tuple

$$(a_1, a_2, \ldots, a_k) \quad \text{with} \quad a_i \in A \quad \text{for} \quad i = 1, 2, \ldots, k \quad \text{and} \quad a_i \neq a_j \quad \text{for} \quad 1 \leqslant i < j \leqslant k.$$

Lemma 185

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. There exist

$$V_k^n := \frac{n!}{(n-k)!}$$

many different ordered selections without repetition of k elements from A.

Ordered Selection

Definition 184 (Ordered selection without repetition, Dt.: Variation ohne Zurücklegen, Variation ohne Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. An *ordered selection without repetition* of k elements from A is a k-tuple

$$(a_1, a_2, \ldots, a_k) \quad \text{with } a_i \in A \text{ for } i = 1, 2, \ldots, k \text{ and } a_i \neq a_j \text{ for } 1 \leqslant i < j \leqslant k.$$

Lemma 185

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. There exist

$$V_k^n := \frac{n!}{(n-k)!}$$

many different ordered selections without repetition of k elements from A.

- Convention: $V_k^n := 0$ for $k > n$.
- V_k^n is the number of injective functions from I_k to A.
- Sometimes, $V(n,k)$ is written instead of V_k^n. English-language textbooks often speak of a k-permutation rather than of an ordered selection without repetition.

Definition 186 (Ordered selection with repetition, Dt.: Variation mit Zurücklegen, Variation mit Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. An *ordered selection with repetition* of k elements from A is a k-tuple

$$(a_1, a_2, \ldots, a_k) \quad \text{with} \quad a_i \in A \quad \text{for} \quad i = 1, 2, \ldots, k.$$

Ordered Selection

Definition 186 (Ordered selection with repetition, Dt.: Variation mit Zurücklegen, Variation mit Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. An *ordered selection with repetition* of k elements from A is a k-tuple

$$(a_1, a_2, \ldots, a_k) \quad \text{with} \quad a_i \in A \quad \text{for} \quad i = 1, 2, \ldots, k.$$

Lemma 187

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. There exist

$$^{r}V_k^n := n^k$$

many different ordered selections with repetition of k elements from A.

Definition 186 (Ordered selection with repetition, Dt.: Variation mit Zurücklegen, Variation mit Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. An *ordered selection with repetition* of k elements from A is a k-tuple

$$(a_1, a_2, \ldots, a_k) \quad \text{with} \quad a_i \in A \quad \text{for} \quad i = 1, 2, \ldots, k.$$

Lemma 187

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. There exist

$$^{r}V_k^n := n^k$$

many different ordered selections with repetition of k elements from A.

- Note: $^{r}V_k^n = |A^k|$.
- Sometimes, $V_r(n, k)$ is written instead of $^{r}V_k^n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

5 Principles of Elementary Counting and Combinatorics

- Sum and Product Rule
- Inclusion-Exclusion Principle
- Binomial Coefficient
- Permutations
- Ordered Selection (Variation)
- Unordered Selection (Combination)

Definition 188 (Unordered selection without repetition, Dt.: Kombination ohne Zurücklegen, Kombination ohne Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. An *unordered selection without repetition* of k elements from A is a set B such that

$$B \subseteq A \quad \text{with} \quad |B| = k.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 188 (Unordered selection without repetition, Dt.: Kombination ohne Zurücklegen, Kombination ohne Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. An *unordered selection without repetition* of k elements from A is a set B such that

$$B \subseteq A \quad \text{with} \quad |B| = k.$$

Lemma 189

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. There exist

$$C_k^n := \binom{n}{k}$$

many different unordered selections without repetition of k elements from A.

Definition 188 (Unordered selection without repetition, Dt.: Kombination ohne Zurücklegen, Kombination ohne Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. An *unordered selection without repetition* of k elements from A is a set B such that

$$B \subseteq A \quad \text{with} \quad |B| = k.$$

Lemma 189

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, with $k \leqslant n$, and A be a finite set of cardinality n. There exist

$$C_k^n := \binom{n}{k}$$

many different unordered selections without repetition of k elements from A.

- Convention: $C_k^n := 0$ for $k > n$. Sometimes, $C(n, k)$ is written instead of C_k^n.
- Lemma 189 yields an alternate proof of $|\mathcal{P}(A)| = 2^n$. It also implies that there exist $\binom{n}{k}$ different binary sequences where exactly k elements are 1.

Unordered Selection

Definition 190 (Unordered selection with repetition, Dt.: Kombination mit Zurücklegen, Kombination mit Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. An *unordered selection with repetition* of k elements from A is a k-element *multiset*, i.e., a set $B \subseteq A$ together with a *multiplicity function*, $\text{mult}\colon A \to \mathbb{N}_0$, such that

$$\text{mult}(a) = 0 \text{ for all } a \in A \backslash B \quad \text{and} \quad \text{mult}(b) > 0 \text{ for all } b \in B \quad \text{and} \quad \sum_{b \in B} \text{mult}(b) = k.$$

Unordered Selection

Definition 190 (Unordered selection with repetition, Dt.: Kombination mit Zurücklegen, Kombination mit Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. An *unordered selection with repetition* of k elements from A is a k-element *multiset*, i.e., a set $B \subseteq A$ together with a *multiplicity function*, $\text{mult}\colon A \to \mathbb{N}_0$, such that

$$\text{mult}(a) = 0 \text{ for all } a \in A \backslash B \quad \text{and} \quad \text{mult}(b) > 0 \text{ for all } b \in B \quad \text{and} \quad \sum_{b \in B} \text{mult}(b) = k.$$

Lemma 191

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. There exist

$$^{r}C_k^n := \binom{n+k-1}{k}$$

many different unordered selections with repetition of k elements from A.

Unordered Selection

Definition 190 (Unordered selection with repetition, Dt.: Kombination mit Zurücklegen, Kombination mit Wiederholung)

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. An *unordered selection with repetition* of k elements from A is a k-element *multiset*, i.e., a set $B \subseteq A$ together with a *multiplicity function*, $\text{mult}\colon A \to \mathbb{N}_0$, such that

$$\text{mult}(a) = 0 \text{ for all } a \in A \backslash B \quad \text{and} \quad \text{mult}(b) > 0 \text{ for all } b \in B \quad \text{and} \quad \sum_{b \in B} \text{mult}(b) = k.$$

Lemma 191

Let $n \in \mathbb{N}$ and $k \in \mathbb{N}_0$, and A be a finite set of cardinality n. There exist

$${}^{r}C_k^n := \binom{n+k-1}{k}$$

many different unordered selections with repetition of k elements from A.

- Sometimes, $C_r(n, k)$ is written instead of ${}^{r}C_k^n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proofs of Lemmas 185–191

Proof of Lemma 185 : We have n options for a_1, leaving $n-1$ options for a_2, etc. Thus, we have $n \cdot (n-1) \cdot \ldots \cdot (n-k+1) = \frac{n!}{(n-k)!}$ options. □

Proof of Lemma 187 : We have n options for every selection. Thus, we have n^k options in total. □

Proof of Lemma 189 : We know that $V_k^n = \frac{n!}{(n-k)!}$. There are $k!$ many different ordered selections that correspond to the same unordered selection. Thus, $C_k^n = V_k^n / k! = \frac{n!}{(n-k)!k!} = \binom{n}{k}$. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Proofs of Lemmas 185–191

Proof of Lemma 185: We have n options for a_1, leaving $n-1$ options for a_2, etc. Thus, we have $n \cdot (n-1) \cdot \ldots \cdot (n-k+1) = \frac{n!}{(n-k)!}$ options. □

Proof of Lemma 187: We have n options for every selection. Thus, we have n^k options in total. □

Proof of Lemma 189: We know that $V_k^n = \frac{n!}{(n-k)!}$. There are $k!$ many different ordered selections that correspond to the same unordered selection. Thus, $C_k^n = V_k^n / k! = \frac{n!}{(n-k)!k!} = \binom{n}{k}$. □

Proof of Lemma 191: Let $a_1, \ldots, a_n$ be the n elements of A, and $k \in \mathbb{N}_0$. We encode such an unordered selection with repetition of k elements from A as a sequence of length $n+k-1$ of k crosses $\times$ which are separated by $n-1$ vertical bars $|$, where i crosses between the j-th vertical bar and the $(j+1)$-st vertical bar, for $1 \leqslant j \leqslant n-2$, indicate that element a_{j+1} was chosen with multiplicity i.

Proof of Lemma 185: We have n options for a_1, leaving $n-1$ options for a_2, etc. Thus, we have $n \cdot (n-1) \cdot \ldots \cdot (n-k+1) = \frac{n!}{(n-k)!}$ options. □

Proof of Lemma 187: We have n options for every selection. Thus, we have n^k options in total. □

Proof of Lemma 189: We know that $V_k^n = \frac{n!}{(n-k)!}$. There are $k!$ many different ordered selections that correspond to the same unordered selection. Thus, $C_k^n = V_k^n / k! = \frac{n!}{(n-k)!k!} = \binom{n}{k}$. □

Proof of Lemma 191: Let $a_1, \ldots, a_n$ be the n elements of A, and $k \in \mathbb{N}_0$. We encode such an unordered selection with repetition of k elements from A as a sequence of length $n+k-1$ of k crosses $\times$ which are separated by $n-1$ vertical bars $|$, where i crosses between the j-th vertical bar and the $(j+1)$-st vertical bar, for $1 \leqslant j \leqslant n-2$, indicate that element a_{j+1} was chosen with multiplicity i. Similarly for the multiplicities of a_1 and a_n for crosses before the first and after the last vertical bar.
We note that we have exactly

$$C_k^{n+k-1} = \binom{n+k-1}{k}$$

ways to choose the positions of the k crosses within this sequence.

Real-World Application: Elementary Probability

- What is the probability to win in the Austrian “6-aus-45” lottery after choosing one set of six numbers?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Elementary Probability

- What is the probability to win in the Austrian "6-aus-45" lottery after choosing one set of six numbers?
- As usual, we define the probability of an event among (finitely many) equally-likely outcomes as the number of favorable outcomes divided by the total number of possible outcomes.

Real-World Application: Elementary Probability

- What is the probability to win in the Austrian “6-aus-45” lottery after choosing one set of six numbers?
- As usual, we define the probability of an event among (finitely many) equally-likely outcomes as the number of favorable outcomes divided by the total number of possible outcomes.
- Assuming that the lottery is fair and, thus, that all combinations are equally likely to win, we get

$$\frac{1}{C_6^{45}} = \frac{1}{\binom{45}{6}} = \frac{1}{8\,145\,060} \approx 1.22774 \cdot 10^{-7}$$

as probability for having all six numbers right.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Elementary Probability

- A standard deck of cards contains 52 cards grouped into four suits (Dt.: Farben) — diamonds (Dt.: Schelle, Karo), clubs (Dt.: Eichel, Kreuz), hearts (Dt.: Herz), and spades (Dt.: Laub, Pik) — with 13 cards in each suit (ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, jack, queen, king).

Real-World Application: Elementary Probability

- A standard deck of cards contains 52 cards grouped into four suits (Dt.: Farben) — diamonds (Dt.: Schelle, Karo), clubs (Dt.: Eichel, Kreuz), hearts (Dt.: Herz), and spades (Dt.: Laub, Pik) — with 13 cards in each suit (ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, jack, queen, king).
- What is the probability that all hearts appear in consecutive (but arbitrary) order after a decent shuffling of the deck?

Real-World Application: Elementary Probability

- A standard deck of cards contains 52 cards grouped into four suits (Dt.: Farben) — diamonds (Dt.: Schelle, Karo), clubs (Dt.: Eichel, Kreuz), hearts (Dt.: Herz), and spades (Dt.: Laub, Pik) — with 13 cards in each suit (ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, jack, queen, king).
- What is the probability that all hearts appear in consecutive (but arbitrary) order after a decent shuffling of the deck?
- There are 52! different permutations of the 52 cards.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Elementary Probability

- A standard deck of cards contains 52 cards grouped into four suits (Dt.: Farben) — diamonds (Dt.: Schelle, Karo), clubs (Dt.: Eichel, Kreuz), hearts (Dt.: Herz), and spades (Dt.: Laub, Pik) — with 13 cards in each suit (ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, jack, queen, king).
- What is the probability that all hearts appear in consecutive (but arbitrary) order after a decent shuffling of the deck?
- There are 52! different permutations of the 52 cards.
- There are 40! different permutations of the block of 13 hearts and the other 39 cards, and 13! many permutations of the 13 hearts within that block.

Real-World Application: Elementary Probability

- A standard deck of cards contains 52 cards grouped into four suits (Dt.: Farben) — diamonds (Dt.: Schelle, Karo), clubs (Dt.: Eichel, Kreuz), hearts (Dt.: Herz), and spades (Dt.: Laub, Pik) — with 13 cards in each suit (ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, jack, queen, king).
- What is the probability that all hearts appear in consecutive (but arbitrary) order after a decent shuffling of the deck?
- There are 52! different permutations of the 52 cards.
- There are 40! different permutations of the block of 13 hearts and the other 39 cards, and 13! many permutations of the 13 hearts within that block.
- Hence, the probability that all hearts are consecutive is given by

$$\frac{40! \cdot 13!}{52!} \approx 6.29908 \cdot 10^{-11}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

6 Complexity Analysis and Recurrence Relations

- Growth Rates
- Bachmann-Landau (Asymptotic) Notation
- Recurrence Relations
- Master Theorem

6 Complexity Analysis and Recurrence Relations

- Growth Rates
- Bachmann-Landau (Asymptotic) Notation
- Recurrence Relations
- Master Theorem

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Growth Rate of Functions

- Algorithms/codes tend to process inputs of small sizes instantaneously. Therefore we are most interested in how an algorithm performs as the input size n gets large: *asymptotic complexity analysis*.

Growth Rate of Functions

- Algorithms/codes tend to process inputs of small sizes instantaneously. Therefore we are most interested in how an algorithm performs as the input size n gets large: *asymptotic complexity analysis*.
- Determine the dominating term in the complexity function — it gives the order of magnitude of the asymptotic behavior.

$$1,\ \log n,\ \log^2 n,\ \sqrt{n},\ n,\ n\log n,\ n\log^2 n,\ n^{\frac{7}{6}},\ n^2,\ n^3,\ \ldots,\ 2^n,\ 3^n,\ 2^{(2^n)},\ \ldots$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Growth Rate of Functions

- Algorithms/codes tend to process inputs of small sizes instantaneously. Therefore we are most interested in how an algorithm performs as the input size n gets large: *asymptotic complexity analysis*.
- Determine the dominating term in the complexity function — it gives the order of magnitude of the asymptotic behavior.

$$1,\ \log n,\ \log^2 n,\ \sqrt{n},\ n,\ n\log n,\ n\log^2 n,\ n^{\frac{7}{6}},\ n^2,\ n^3,\ \ldots,\ 2^n,\ 3^n,\ 2^{(2^n)},\ \ldots$$

Convention regarding logarithms

In this course, $\log n$ will always denote the logarithm of n to the base 2, i.e., $\log n := \log_2 n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Growth Rate of Functions

- Algorithms/codes tend to process inputs of small sizes instantaneously. Therefore we are most interested in how an algorithm performs as the input size n gets large: *asymptotic complexity analysis*.
- Determine the dominating term in the complexity function — it gives the order of magnitude of the asymptotic behavior.

$$1,\ \log n,\ \log^2 n,\ \sqrt{n},\ n,\ n\log n,\ n\log^2 n,\ n^{\frac{7}{6}},\ n^2,\ n^3,\ \ldots,\ 2^n,\ 3^n,\ 2^{(2^n)},\ \ldots$$

Convention regarding logarithms

In this course, $\log n$ will always denote the logarithm of n to the base 2, i.e., $\log n := \log_2 n$.

- Recall that $\log_\alpha n = \frac{1}{\log_2 \alpha} \log_2 n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Growth Rates: Bachmann-Landau Notation

- Let's consider $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n$ and $g(n) := 9n + 20$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Growth Rates: Bachmann-Landau Notation

- Let's consider $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n$ and $g(n) := 9n + 20$.
- We get for all $n \in \mathbb{N}$ with $n \geqslant 20$

$$g(n) = 9n + 20 \leqslant 9n + n = 10n = 10f(n),$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Growth Rates: Bachmann-Landau Notation

- Let's consider $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n$ and $g(n) := 9n + 20$.
- We get for all $n \in \mathbb{N}$ with $n \geqslant 20$

$$g(n) = 9n + 20 \leqslant 9n + n = 10n = 10f(n), \quad \text{that is } g(n) \leqslant 10f(n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Let's consider $f, g \colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n$ and $g(n) := 9n + 20$.
- We get for all $n \in \mathbb{N}$ with $n \geqslant 20$

$$g(n) = 9n + 20 \leqslant 9n + n = 10n = 10f(n), \quad \text{that is } g(n) \leqslant 10f(n).$$

- Also for all $n \in \mathbb{N}$

$$f(n) \leqslant g(n).$$

- Let's consider $f, g \colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n$ and $g(n) := 9n + 20$.
- We get for all $n \in \mathbb{N}$ with $n \geqslant 20$

$$g(n) = 9n + 20 \leqslant 9n + n = 10n = 10f(n), \quad \text{that is } \underbrace{g(n) \leqslant 10f(n)}.$$

- Also for all $n \in \mathbb{N}$

$$\underbrace{f(n) \leqslant g(n)}.$$

Thus, we have $$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n) \quad \begin{cases} \text{for all } n \geq n_0 \\ \text{where } n_0 := 20, \\ c_1 := 1,\ c_2 := 10. \end{cases}$$

- Let's consider $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n$ and $g(n) := 9n + 20$.
- We get for all $n \in \mathbb{N}$ with $n \geqslant 20$

$$g(n) = 9n + 20 \leqslant 9n + n = 10n = 10f(n), \quad \text{that is } g(n) \leqslant 10f(n).$$

- Also for all $n \in \mathbb{N}$

$$f(n) \leqslant g(n).$$

Thus, we have $c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n)$ $\begin{cases} \text{for all } n \geq n_0 \\ \text{where } n_0 := 20, \\ c_1 := 1,\ c_2 := 10. \end{cases}$

g grows at most as fast as $c_2 \cdot f$

f is an asymptotic upper bound on g

we'll say that $g \in O(f)$

- Let's consider $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n$ and $g(n) := 9n + 20$.
- We get for all $n \in \mathbb{N}$ with $n \geqslant 20$

$$g(n) = 9n + 20 \leqslant 9n + n = 10n = 10f(n), \quad \text{that is } g(n) \leqslant 10f(n).$$

- Also for all $n \in \mathbb{N}$

$$f(n) \leqslant g(n).$$

Thus, we have $c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n)$ $\begin{cases} \text{for all } n \geq n_0 \\ \text{where } n_0 := 20, \\ c_1 := 1,\ c_2 := 10. \end{cases}$

g grows at least as fast as $c_1 \cdot f$

f is an asymptotic lower bound on g

we'll say that $g \in \Omega(f)$

- Let's consider $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n$ and $g(n) := 9n + 20$.
- We get for all $n \in \mathbb{N}$ with $n \geqslant 20$

$$g(n) = 9n + 20 \leqslant 9n + n = 10n = 10f(n), \quad \text{that is } g(n) \leqslant 10f(n).$$

- Also for all $n \in \mathbb{N}$

$$f(n) \leqslant g(n).$$

Thus, we have $c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n)$ $\begin{cases} \text{for all } n \geq n_0 \\ \text{where } n_0 := 20, \\ c_1 := 1,\ c_2 := 10. \end{cases}$

g grows at least as fast as $c_1 \cdot f$

f is an asymptotic lower bound on g

we'll say that $g \in \Omega(f)$

g grows at most as fast as $c_2 \cdot f$

f is an asymptotic upper bound on g

we'll say that $g \in O(f)$

g has same growth rate as f

we'll say that $g \in \Theta(f)$

6 Complexity Analysis and Recurrence Relations

- Growth Rates
- Bachmann-Landau (Asymptotic) Notation
  - Bachmann-Landau Symbols
  - Limit of a Sequence
  - Basic Facts
  - Conditional Asymptotic Notation and Smoothness Rule
- Recurrence Relations
- Master Theorem

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Big-O

$$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n) \qquad \begin{cases} \text{for all } n \geq n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g grows at most as fast as $c_2 \cdot f$

f is an asymptotic upper bound on g

we'll say that $g \in O(f)$

$$c_1 \cdot f(n) \le g(n) \le c_2 \cdot f(n) \quad \begin{cases} \text{for all } n \ge n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g grows at most as fast as $c_2 \cdot f$

f is an asymptotic upper bound on g

we'll say that $g \in O(f)$

Definition 192 (Big-O, Dt.: Groß-O)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $O(f)$ is defined as

$$O(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c_2 \cdot f(n)\}.$$

$$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n) \qquad \begin{cases} \text{for all } n \geq n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g grows at most as fast as $c_2 \cdot f$

f is an asymptotic upper bound on g

we'll say that $g \in O(f)$

Definition 192 (Big-O, Dt.: Groß-O)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $O(f)$ is defined as

$$O(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \exists c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \quad g(n) \leqslant c_2 \cdot f(n)\}.$$

- Some authors prefer to use the symbol $\mathcal{O}$ instead of O.

Asymptotic Notation: Big-O

$$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n) \qquad \begin{cases} \text{for all } n \geq n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g grows at most as fast as $c_2 \cdot f$

f is an asymptotic upper bound on g

we'll say that $g \in O(f)$

Definition 192 (Big-O, Dt.: Groß-O)

Let $f : \mathbb{N} \to \mathbb{R}^+$. Then the set $O(f)$ is defined as

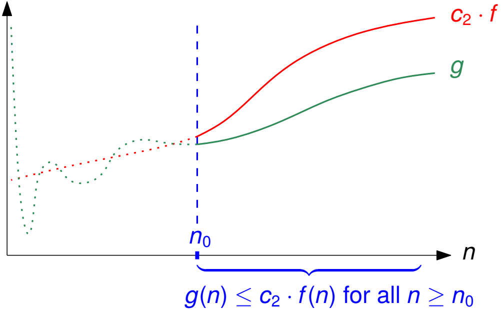
$$O(f) := \{g : \mathbb{N} \to \mathbb{R}^+ \mid \exists c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c_2 \cdot f(n)\}.$$

- Some authors prefer to use the symbol $\mathcal{O}$ instead of O.
- Note: $O(f)$ is a set of functions! Definitions of the form
  $$O(f(n)) := \{g : \mathbb{N} \to \mathbb{R}^+ \mid \exists c_2 \in \mathbb{R}^+ \ \exists n_0 \in \mathbb{N} \ \forall n \geqslant n_0 \quad g(n) \leqslant c_2 \cdot f(n)\}$$
  are a (wide-spread) formal nonsense.

Definition 192 (Big-O, Dt.: Groß-O)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $O(f)$ is defined as

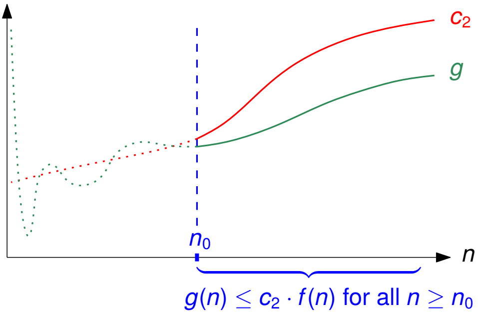
$$O(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \exists c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c_2 \cdot f(n)\}.$$



Definition 192 (Big-O, Dt.: Groß-O)

Let $f: \mathbb{N} \to \mathbb{R}^+$. Then the set $O(f)$ is defined as

$$O(f) := \{g: \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c_2 \cdot f(n)\}.$$



- Equivalent definition used by some authors:

$$O(f) := \left\{g: \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad \frac{g(n)}{f(n)} \leqslant c_2 \right\}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Why Don't We Care About Constants?

- Note that this notation hides all lower-order terms and multiplicative constants. Why don't we care?

Why Don't We Care About Constants?

- Note that this notation hides all lower-order terms and multiplicative constants. Why don't we care?
- Since it doesn't matter for large values of n.

Why Don't We Care About Constants?

- Note that this notation hides all lower-order terms and multiplicative constants. Why don't we care?
- Since it doesn't matter for large values of n.
- Consider the following two nested for-loops:

for $i = 1$ **to** n **do**
  for $j = i$ **to** n **do**
    Compute(i, j)
  end for
end for

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Why Don't We Care About Constants?

- Note that this notation hides all lower-order terms and multiplicative constants. Why don't we care?
- Since it doesn't matter for large values of n.
- Consider the following two nested for-loops:

for $i = 1$ **to** n **do**
  for $j = i$ **to** n **do**
    Compute(i, j)
  end for
end for

- How often is Compute() being called?
  Let $g\colon \mathbb{N} \to \mathbb{R}^+$ be the function that models the number of calls in dependence on n.

Why Don't We Care About Constants?

- Note that this notation hides all lower-order terms and multiplicative constants. Why don't we care?
- Since it doesn't matter for large values of n.
- Consider the following two nested for-loops:

for $i = 1$ **to** n **do**
  for $j = i$ **to** n **do**
    Compute(i, j)
  end for
end for

- How often is Compute() being called? Let $g\colon \mathbb{N} \to \mathbb{R}^+$ be the function that models the number of calls in dependence on n.
- We get

$$\begin{aligned} g(n) &= n + (n-1) + \ldots + 2 + 1 \\ &= \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n. \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Why Don't We Care About Constants?

- Note that this notation hides all lower-order terms and multiplicative constants. Why don't we care?
- Since it doesn't matter for large values of n.
- Consider the following two nested for-loops:

for $i = 1$ **to** n **do**
  for $j = i$ **to** n **do**
    Compute(i, j)
  end for
end for

- How often is Compute() being called? Let $g\colon \mathbb{N} \to \mathbb{R}^+$ be the function that models the number of calls in dependence on n.
- We get

$$g(n) = n + (n-1) + \ldots + 2 + 1$$
$$= \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n.$$

- Consider $f\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Why Don't We Care About Constants?

- Note that this notation hides all lower-order terms and multiplicative constants. Why don't we care?
- Since it doesn't matter for large values of n.
- Consider the following two nested for-loops:

for $i = 1$ **to** n **do**
  for $j = i$ **to** n **do**
    Compute(i, j)
  end for
end for

- How often is Compute() being called? Let $g\colon \mathbb{N} \to \mathbb{R}^+$ be the function that models the number of calls in dependence on n.
- We get

$$g(n) = n + (n-1) + \ldots + 2 + 1 = \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n.$$

- Consider $f\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$.
- Let's compare the growth rates of f and g when we double n:

n	$g(n)$	$f(n)$
5	15	25
10	55	100
20	210	400
40	820	1600
80	3240	6400

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Why Don't We Care About Constants?

- Note that this notation hides all lower-order terms and multiplicative constants. Why don't we care?
- Since it doesn't matter for large values of n.
- Consider the following two nested for-loops:

for $i = 1$ **to** n **do**
  for $j = i$ **to** n **do**
    Compute(i, j)
  end for
end for

- How often is Compute() being called? Let $g\colon \mathbb{N} \to \mathbb{R}^+$ be the function that models the number of calls in dependence on n.
- We get

$$g(n) = n + (n-1) + \ldots + 2 + 1 = \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n.$$

- Consider $f\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$.
- Let's compare the growth rates of f and g when we double n:

n	$g(n)$	$f(n)$
5	15	25
10	55	100
20	210	400
40	820	1600
80	3240	6400

- Doubling n causes both $f(n)$ and $g(n)$ to (roughly) quadruple!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

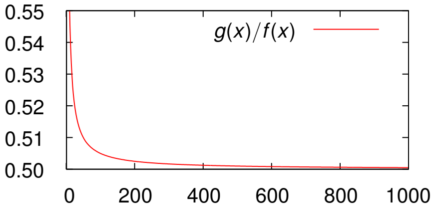
Why Don't We Care About Constants?

- We plot the growth ratio $\frac{g(n)}{f(n)}$ for $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

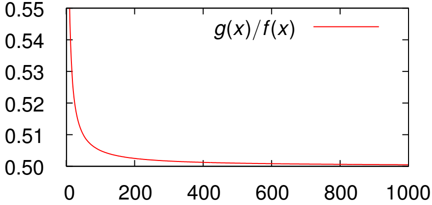
Why Don't We Care About Constants?

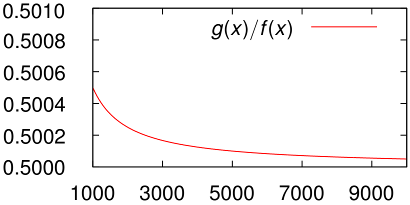
- We plot the growth ratio $\frac{g(n)}{f(n)}$ for $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.



Why Don't We Care About Constants?

- We plot the growth ratio $\frac{g(n)}{f(n)}$ for $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

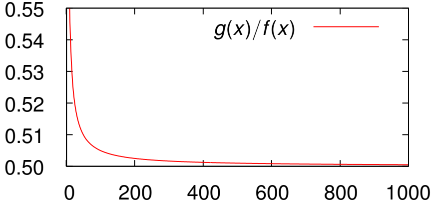


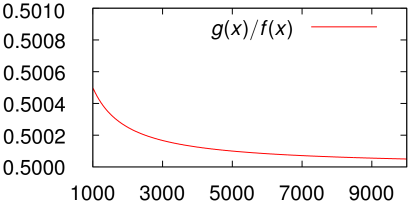


UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Why Don't We Care About Constants?

- We plot the growth ratio $\frac{g(n)}{f(n)}$ for $f, g \colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

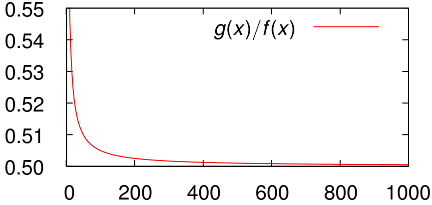


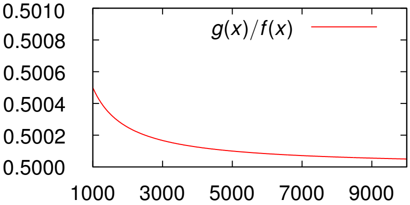


- The plots suggest $\frac{g(n)}{f(n)} \leqslant 1$ for all $n \geqslant 200$, that is, $g(n) \leqslant f(n)$, which would imply $g \in O(f)$.

Why Don't We Care About Constants?

- We plot the growth ratio $\frac{g(n)}{f(n)}$ for $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

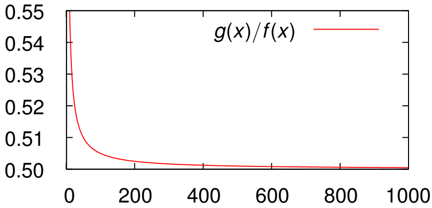


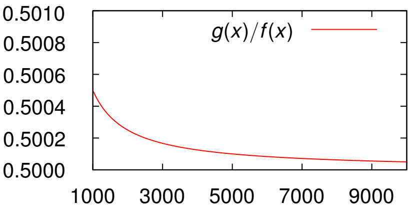


- The plots suggest $\frac{g(n)}{f(n)} \leqslant 1$ for all $n \geqslant 200$, that is, $g(n) \leqslant f(n)$, which would imply $g \in O(f)$.
- More precisely, they suggest $\frac{g(n)}{f(n)} \leqslant \frac{1}{2} + \varepsilon$ for any positive ε and all sufficiently large values of n.

Why Don't We Care About Constants?

- We plot the growth ratio $\frac{g(n)}{f(n)}$ for $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

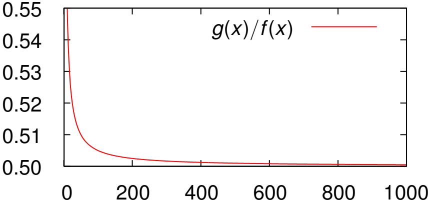


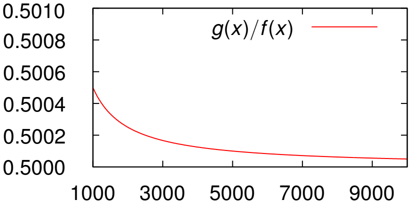


- The plots suggest $\frac{g(n)}{f(n)} \leqslant 1$ for all $n \geqslant 200$, that is, $g(n) \leqslant f(n)$, which would imply $g \in O(f)$.
- More precisely, they suggest $\frac{g(n)}{f(n)} \leqslant \frac{1}{2} + \varepsilon$ for any positive ε and all sufficiently large values of n.
- The plots also suggest $\frac{g(n)}{f(n)} \geqslant \frac{1}{2}$, which would imply $g \in \Omega(f)$.

Why Don't We Care About Constants?

- We plot the growth ratio $\frac{g(n)}{f(n)}$ for $f, g\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.





- The plots suggest $\frac{g(n)}{f(n)} \leqslant 1$ for all $n \geqslant 200$, that is, $g(n) \leqslant f(n)$, which would imply $g \in O(f)$.
- More precisely, they suggest $\frac{g(n)}{f(n)} \leqslant \frac{1}{2} + \varepsilon$ for any positive ε and all sufficiently large values of n.
- The plots also suggest $\frac{g(n)}{f(n)} \geqslant \frac{1}{2}$, which would imply $g \in \Omega(f)$.
- Hence $g(n) \approx \frac{1}{2}f(n)$, which would imply $g \in \Theta(f)$.

$$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n) \qquad \begin{cases} \text{for all } n \geq n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g grows at least as fast as $c_1 \cdot f$

f is an asymptotic lower bound on g

we'll say that $g \in \Omega(f)$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Big-Omega

$$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n) \qquad \begin{cases} \text{for all } n \geq n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g grows at least as fast as $c_1 \cdot f$

f is an asymptotic lower bound on g

we'll say that $g \in \Omega(f)$

Definition 193 (Big-Omega, Dt.: Groß-Omega)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $\Omega(f)$ is defined as

$$\Omega(f) \;:=\; \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c_1 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad c_1 \cdot f(n) \leqslant g(n)\}.$$

$$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n) \qquad \begin{cases} \text{for all } n \geq n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g grows at least as fast as $c_1 \cdot f$

f is an asymptotic lower bound on g

we'll say that $g \in \Omega(f)$

Definition 193 (Big-Omega, Dt.: Groß-Omega)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $\Omega(f)$ is defined as

$$\Omega(f) \;:=\; \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \;\; \exists c_1 \in \mathbb{R}^+ \;\; \exists n_0 \in \mathbb{N} \;\; \forall n \geqslant n_0 \qquad c_1 \cdot f(n) \leqslant g(n)\}.$$

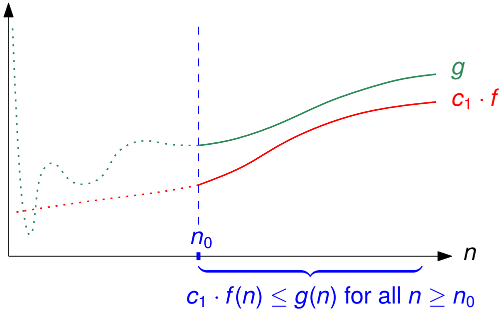
- Equivalently,

$$\Omega(f) \;:=\; \left\{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \;\; \exists c_1 \in \mathbb{R}^+ \;\; \exists n_0 \in \mathbb{N} \;\; \forall n \geqslant n_0 \qquad c_1 \leqslant \frac{g(n)}{f(n)}\right\}.$$

Definition 193 (Big-Omega, Dt.: Groß-Omega)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $\Omega(f)$ is defined as

$$\Omega(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c_1 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad c_1 \cdot f(n) \leqslant g(n)\}.$$



$$\underbrace{c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n)}_{} \quad \begin{cases} \text{for all } n \geq n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g has same growth rate as f

we'll say that $g \in \Theta(f)$

$$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n) \qquad \begin{cases} \text{for all } n \geq n_0 \text{ and} \\ \text{fixed } c_1, c_2 \in \mathbb{R}^+. \end{cases}$$

g has same growth rate as *f*

we'll say that $g \in \Theta(f)$

Definition 194 (Big-Theta, Dt.: Groß-Theta)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $\Theta(f)$ is defined as

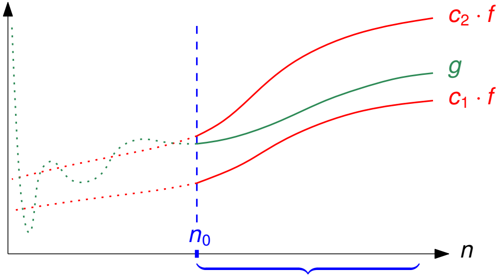
$$\Theta(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \exists c_1, c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \quad c_1 \cdot f(n) \leqslant g(n) \leqslant c_2 \cdot f(n)\}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 194 (Big-Theta, Dt.: Groß-Theta)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $\Theta(f)$ is defined as

$$\Theta(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \exists c_1, c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \quad c_1 \cdot f(n) \leqslant g(n) \leqslant c_2 \cdot f(n)\}.$$



$c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n)$ for all $n \geq n_0$

which is equivalent to $c_1 \leq \frac{g(n)}{f(n)} \leq c_2$ for all $n \geq n_0$

Sample Proof of $g \in \Theta(f)$

- We prove $g \in \Theta(f)$ for $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- We prove $g \in \Theta(f)$ for $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

  Proof :

  - We get, for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \leqslant \frac{1}{2}n^2 + \frac{1}{2}n^2 = n^2 = f(n),$$

Sample Proof of $g \in \Theta(f)$

- We prove $g \in \Theta(f)$ for $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

  Proof :

  - We get, for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \leqslant \frac{1}{2}n^2 + \frac{1}{2}n^2 = n^2 = f(n), \quad \text{that is } g(n) \leqslant f(n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Proof of $g \in \Theta(f)$

- We prove $g \in \Theta(f)$ for $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

  Proof :

  - We get, for all $n \in \mathbb{N}$,

  $$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \leqslant \frac{1}{2}n^2 + \frac{1}{2}n^2 = n^2 = f(n), \quad \text{that is } g(n) \leqslant f(n).$$

  - Thus, $g \in O(f)$ with $c_2 := 1$ and $n_0 := 1$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Proof of $g \in \Theta(f)$

- We prove $g \in \Theta(f)$ for $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

  Proof :

  - We get, for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \leqslant \frac{1}{2}n^2 + \frac{1}{2}n^2 = n^2 = f(n), \quad \text{that is } g(n) \leqslant f(n).$$

  - Thus, $g \in O(f)$ with $c_2 := 1$ and $n_0 := 1$.
  - Now we prove $g \in \Omega(f)$ and get, again for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \geqslant \frac{1}{2}n^2 = \frac{1}{2}f(n),$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Proof of $g \in \Theta(f)$

- We prove $g \in \Theta(f)$ for $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

  Proof :

  - We get, for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \leqslant \frac{1}{2}n^2 + \frac{1}{2}n^2 = n^2 = f(n), \quad \text{that is } g(n) \leqslant f(n).$$

  - Thus, $g \in O(f)$ with $c_2 := 1$ and $n_0 := 1$.
  - Now we prove $g \in \Omega(f)$ and get, again for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \geqslant \frac{1}{2}n^2 = \frac{1}{2}f(n), \quad \text{that is } \frac{1}{2}f(n) \leqslant g(n).$$

Sample Proof of $g \in \Theta(f)$

- We prove $g \in \Theta(f)$ for $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

  Proof:

  - We get, for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \leqslant \frac{1}{2}n^2 + \frac{1}{2}n^2 = n^2 = f(n), \quad \text{that is } g(n) \leqslant f(n).$$

  - Thus, $g \in O(f)$ with $c_2 := 1$ and $n_0 := 1$.
  - Now we prove $g \in \Omega(f)$ and get, again for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \geqslant \frac{1}{2}n^2 = \frac{1}{2}f(n), \quad \text{that is } \frac{1}{2}f(n) \leqslant g(n).$$

  - Thus, $g \in \Omega(f)$ with $c_1 := \frac{1}{2}$ and $n_0 := 1$. Def. 194 or Lemma 201 yield $g \in \Theta(f)$. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sample Proof of $g \in \Theta(f)$

- We prove $g \in \Theta(f)$ for $f(n) := n^2$ and $g(n) := \frac{1}{2}n^2 + \frac{1}{2}n$.

  Proof:
  - We get, for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \leqslant \frac{1}{2}n^2 + \frac{1}{2}n^2 = n^2 = f(n), \quad \text{that is } g(n) \leqslant f(n).$$

  - Thus, $g \in O(f)$ with $c_2 := 1$ and $n_0 := 1$.
  - Now we prove $g \in \Omega(f)$ and get, again for all $n \in \mathbb{N}$,

$$g(n) = \frac{1}{2}n^2 + \frac{1}{2}n \geqslant \frac{1}{2}n^2 = \frac{1}{2}f(n), \quad \text{that is } \frac{1}{2}f(n) \leqslant g(n).$$

  - Thus, $g \in \Omega(f)$ with $c_1 := \frac{1}{2}$ and $n_0 := 1$. Def. 194 or Lemma 201 yield $g \in \Theta(f)$. □

Don't be overly zealous!

There is no need to try to obtain the "best-possible" values for n_0 and c_1, c_2!

Definition 195 (Small-Oh, Dt.: Klein-O)

Let $f: \mathbb{N} \to \mathbb{R}^+$. Then the set $o(f)$ is defined as

$$o(f) := \{g: \mathbb{N} \to \mathbb{R}^+ \mid \quad \forall c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}.$$

Definition 195 (Small-Oh, Dt.: Klein-O)

Let $f: \mathbb{N} \to \mathbb{R}^+$. Then the set $o(f)$ is defined as

$$o(f) := \{g: \mathbb{N} \to \mathbb{R}^+ \mid \quad \forall c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}.$$

Mind the difference

$$O(f) := \{g: \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}$$

$$o(f) := \{g: \mathbb{N} \to \mathbb{R}^+ \mid \quad \forall c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}$$

Definition 195 (Small-Oh, Dt.: Klein-O)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $o(f)$ is defined as

$$o(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \forall c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}.$$

Mind the difference

$$O(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}$$
$$o(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \forall c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}$$

- Similarly, $\omega(f)$ can be defined relative to $\Omega(f)$.

Definition 195 (Small-Oh, Dt.: Klein-O)

Let $f\colon \mathbb{N} \to \mathbb{R}^+$. Then the set $o(f)$ is defined as

$$o(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \forall c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}.$$

Mind the difference

$$O(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}$$
$$o(f) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \forall c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad g(n) \leqslant c \cdot f(n)\}$$

- Similarly, $\omega(f)$ can be defined relative to $\Omega(f)$.
- It is trivial to extend Definitions 192–195 such that $\mathbb{N}_0$ rather than $\mathbb{N}$ is taken as the domain.
- We can also replace the codomain $\mathbb{R}^+$ by $\mathbb{R}_0^+$ (or even $\mathbb{R}$) provided that all functions are eventually positive.
- The same comments apply to the subsequent slides.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 196 (Sequence, Dt.: Folge)

A (real) *sequence* is a function from $\mathbb{N}$ (or $\mathbb{N}_0$) to $\mathbb{R}$. For $x : \mathbb{N} \to \mathbb{R}$ it is common to write the sequence as $(x_n)_{n\in\mathbb{N}}$ or $\langle x_n \rangle_{n\in\mathbb{N}}$, or simply (x_n) or $\langle x_n \rangle$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 196 (Sequence, Dt.: Folge)

A (real) *sequence* is a function from $\mathbb{N}$ (or $\mathbb{N}_0$) to $\mathbb{R}$. For $x : \mathbb{N} \to \mathbb{R}$ it is common to write the sequence as $(x_n)_{n\in\mathbb{N}}$ or $\langle x_n \rangle_{n\in\mathbb{N}}$, or simply (x_n) or $\langle x_n \rangle$.

Definition 197 (Limit, Dt. Grenzwert)

The value $\bar{x} \in \mathbb{R}$ is the limit of the (real) sequence (x_n), denoted by $\lim_{n\to\infty} x_n = \bar{x}$, if

$$\forall \varepsilon \in \mathbb{R}^+ \ \exists n_0 \in \mathbb{N} \ \forall n \geqslant n_0 \quad |x_n - \bar{x}| < \varepsilon.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 196 (Sequence, Dt.: Folge)

A (real) *sequence* is a function from $\mathbb{N}$ (or $\mathbb{N}_0$) to $\mathbb{R}$. For $x : \mathbb{N} \to \mathbb{R}$ it is common to write the sequence as $(x_n)_{n\in\mathbb{N}}$ or $\langle x_n \rangle_{n\in\mathbb{N}}$, or simply (x_n) or $\langle x_n \rangle$.

Definition 197 (Limit, Dt. Grenzwert)

The value $\bar{x} \in \mathbb{R}$ is the limit of the (real) sequence (x_n), denoted by $\lim_{n\to\infty} x_n = \bar{x}$, if

$$\forall \varepsilon \in \mathbb{R}^+ \;\; \exists n_0 \in \mathbb{N} \;\; \forall n \geqslant n_0 \qquad |x_n - \bar{x}| < \varepsilon.$$

Lemma 198

If $z_n = x_n + y_n$ for three sequences $(x_n), (y_n), (z_n)$ and if $\lim_{n\to\infty} x_n$ and $\lim_{n\to\infty} y_n$ exist, then $\lim_{n\to\infty} z_n$ exists and we have $\lim_{n\to\infty} z_n = \lim_{n\to\infty} x_n + \lim_{n\to\infty} y_n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 199 (Squeeze theorem, Dt.: Einschnürungssatz)

Consider three real sequences $(x_n), (y_n), (z_n)$ and suppose that $x_n \leqslant y_n \leqslant z_n$ for all $n \geqslant n_0$ for some $n_0 \in \mathbb{N}$. If the limits of (x_n) and (z_n) exist such that

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} z_n,$$

then the limit of (y_n) exists with

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} y_n = \lim_{n\to\infty} z_n.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 199 (Squeeze theorem, Dt.: Einschnürungssatz)

Consider three real sequences $(x_n), (y_n), (z_n)$ and suppose that $x_n \leqslant y_n \leqslant z_n$ for all $n \geqslant n_0$ for some $n_0 \in \mathbb{N}$. If the limits of (x_n) and (z_n) exist such that

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} z_n,$$

then the limit of (y_n) exists with

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} y_n = \lim_{n\to\infty} z_n.$$

- For $z_n := \frac{8}{n}$ it is easy to see that $\lim_{n\to\infty} z_n = 0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 199 (Squeeze theorem, Dt.: Einschnürungssatz)

Consider three real sequences $(x_n), (y_n), (z_n)$ and suppose that $x_n \leqslant y_n \leqslant z_n$ for all $n \geqslant n_0$ for some $n_0 \in \mathbb{N}$. If the limits of (x_n) and (z_n) exist such that

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} z_n,$$

then the limit of (y_n) exists with

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} y_n = \lim_{n\to\infty} z_n.$$

- For $z_n := \frac{8}{n}$ it is easy to see that $\lim_{n\to\infty} z_n = 0$.
- Now consider the following sequences:

$$x_n := 0 \qquad y_n := \frac{\log n + 7\sqrt{n} - 10}{n^2} \qquad z_n := \frac{8}{n}$$

Theorem 199 (Squeeze theorem, Dt.: Einschnürungssatz)

Consider three real sequences $(x_n), (y_n), (z_n)$ and suppose that $x_n \leqslant y_n \leqslant z_n$ for all $n \geqslant n_0$ for some $n_0 \in \mathbb{N}$. If the limits of (x_n) and (z_n) exist such that

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} z_n,$$

then the limit of (y_n) exists with

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} y_n = \lim_{n\to\infty} z_n.$$

- For $z_n := \frac{8}{n}$ it is easy to see that $\lim_{n\to\infty} z_n = 0$.
- Now consider the following sequences:

$$x_n := 0 \qquad y_n := \frac{\log n + 7\sqrt{n} - 10}{n^2} \qquad z_n := \frac{8}{n}$$

- We have for all $n \in \mathbb{N}\setminus\{1, 2, 3\}$

$$x_n \leqslant y_n \leqslant z_n \quad \text{and} \quad \lim_{n\to\infty} x_n = 0 = \lim_{n\to\infty} z_n.$$

Thus, $\lim_{n\to\infty} y_n = 0$.

Asymptotic Notation: Limit of a Sequence

- The following theorem (by Guillaume de l'Hôpital, 1661–1704) allows to handle limits that involve indeterminate terms of the form

$$\frac{0}{0} \qquad \text{or} \qquad \frac{\infty}{\infty}.$$

- The following theorem (by Guillaume de l'Hôpital, 1661–1704) allows to handle limits that involve indeterminate terms of the form

$$\frac{0}{0} \qquad \text{or} \qquad \frac{\infty}{\infty}.$$

Theorem 200 (L'Hôpital's rule)

Consider two real functions f and g, and a real value c.

- The following theorem (by Guillaume de l'Hôpital, 1661–1704) allows to handle limits that involve indeterminate terms of the form

$$\frac{0}{0} \qquad \text{or} \qquad \frac{\infty}{\infty}.$$

Theorem 200 (L'Hôpital's rule)

Consider two real functions f and g, and a real value c.
If

1. $\lim_{x \to c} f(x) = 0 = \lim_{x \to c} g(x)$ or $\lim_{x \to c} f(x) = \pm\infty = \lim_{x \to c} g(x)$,

Asymptotic Notation: Limit of a Sequence

- The following theorem (by Guillaume de l'Hôpital, 1661–1704) allows to handle limits that involve indeterminate terms of the form

$$\frac{0}{0} \quad \text{or} \quad \frac{\infty}{\infty}.$$

Theorem 200 (L'Hôpital's rule)

Consider two real functions f and g, and a real value c.
If

1. $\lim_{x\to c} f(x) = 0 = \lim_{x\to c} g(x)$ or $\lim_{x\to c} f(x) = \pm\infty = \lim_{x\to c} g(x)$,
2. f and g are differentiable in an open interval I with $c \in I$, except possibly at c itself,

- The following theorem (by Guillaume de l'Hôpital, 1661–1704) allows to handle limits that involve indeterminate terms of the form

$$\frac{0}{0} \qquad \text{or} \qquad \frac{\infty}{\infty}.$$

Theorem 200 (L'Hôpital's rule)

Consider two real functions f and g, and a real value c.
If

1. $\lim_{x\to c} f(x) = 0 = \lim_{x\to c} g(x)$ or $\lim_{x\to c} f(x) = \pm\infty = \lim_{x\to c} g(x)$,
2. f and g are differentiable in an open interval I with $c \in I$, except possibly at c itself,
3. $g'(x) \neq 0$ for all $x \in I \backslash \{c\}$,

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Limit of a Sequence

- The following theorem (by Guillaume de l'Hôpital, 1661–1704) allows to handle limits that involve indeterminate terms of the form

$$\frac{0}{0} \qquad \text{or} \qquad \frac{\infty}{\infty}.$$

Theorem 200 (L'Hôpital's rule)

Consider two real functions f and g, and a real value c.
If

1. $\lim_{x\to c} f(x) = 0 = \lim_{x\to c} g(x)$ or $\lim_{x\to c} f(x) = \pm\infty = \lim_{x\to c} g(x)$,
2. f and g are differentiable in an open interval I with $c \in I$, except possibly at c itself,
3. $g'(x) \neq 0$ for all $x \in I \backslash \{c\}$, and if
4. $\lim_{x\to c} \frac{f'(x)}{g'(x)}$ exists,

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Limit of a Sequence

- The following theorem (by Guillaume de l'Hôpital, 1661–1704) allows to handle limits that involve indeterminate terms of the form

$$\frac{0}{0} \qquad \text{or} \qquad \frac{\infty}{\infty}.$$

Theorem 200 (L'Hôpital's rule)

Consider two real functions f and g, and a real value c.
If

1. $\lim_{x\to c} f(x) = 0 = \lim_{x\to c} g(x)$ or $\lim_{x\to c} f(x) = \pm\infty = \lim_{x\to c} g(x)$,
2. f and g are differentiable in an open interval I with $c \in I$, except possibly at c itself,
3. $g'(x) \neq 0$ for all $x \in I \setminus \{c\}$, and if
4. $\lim_{x\to c} \frac{f'(x)}{g'(x)}$ exists,

then

$$\lim_{x\to c} \frac{f(x)}{g(x)} = \lim_{x\to c} \frac{f'(x)}{g'(x)}.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Basic Facts

Lemma 201

Let $f_1, f_2, g_1, g_2 \colon \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $\big(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)\big) \quad \Rightarrow \quad g_1 + g_2 \in O(f_1 + f_2)$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 201

Let $f_1, f_2, g_1, g_2 \colon \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $\big(g_1 \in O(f_1) \quad \wedge \quad g_2 \in O(f_2)\big) \quad \Rightarrow \quad g_1 + g_2 \in O(f_1 + f_2)$
2. $\big(g_1 \in O(f_1) \quad \wedge \quad g_2 \in O(f_2)\big) \quad \Rightarrow \quad g_1 \cdot g_2 \in O(f_1 \cdot f_2)$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 201

Let $f_1, f_2, g_1, g_2 \colon \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $\big(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)\big) \quad\Rightarrow\quad g_1 + g_2 \in O(f_1 + f_2)$
2. $\big(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)\big) \quad\Rightarrow\quad g_1 \cdot g_2 \in O(f_1 \cdot f_2)$
3. $f_2 \cdot O(f_1) \subseteq O(f_1 \cdot f_2)$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 201

Let $f_1, f_2, g_1, g_2 \colon \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $\big(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)\big) \quad\Rightarrow\quad g_1 + g_2 \in O(f_1 + f_2)$
2. $\big(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)\big) \quad\Rightarrow\quad g_1 \cdot g_2 \in O(f_1 \cdot f_2)$
3. $f_2 \cdot O(f_1) \subseteq O(f_1 \cdot f_2)$
4. $O(c \cdot f_1) = O(f_1)$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 201

Let $f_1, f_2, g_1, g_2 \colon \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)) \quad\Rightarrow\quad g_1 + g_2 \in O(f_1 + f_2)$
2. $(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)) \quad\Rightarrow\quad g_1 \cdot g_2 \in O(f_1 \cdot f_2)$
3. $f_2 \cdot O(f_1) \subseteq O(f_1 \cdot f_2)$
4. $O(c \cdot f_1) = O(f_1)$
5. $g_1 \in O(f_1) \quad\Rightarrow\quad c \cdot g_1 \in O(f_1)$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 201

Let $f_1, f_2, g_1, g_2 : \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)) \quad\Rightarrow\quad g_1 + g_2 \in O(f_1 + f_2)$
2. $(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)) \quad\Rightarrow\quad g_1 \cdot g_2 \in O(f_1 \cdot f_2)$
3. $f_2 \cdot O(f_1) \subseteq O(f_1 \cdot f_2)$
4. $O(c \cdot f_1) = O(f_1)$
5. $g_1 \in O(f_1) \quad\Rightarrow\quad c \cdot g_1 \in O(f_1)$
6. $\Theta(f_1) \;=\; O(f_1) \;\cap\; \Omega(f_1)$

Lemma 201

Let $f_1, f_2, g_1, g_2 \colon \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)) \quad\Rightarrow\quad g_1 + g_2 \in O(f_1 + f_2)$
2. $(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)) \quad\Rightarrow\quad g_1 \cdot g_2 \in O(f_1 \cdot f_2)$
3. $f_2 \cdot O(f_1) \subseteq O(f_1 \cdot f_2)$
4. $O(c \cdot f_1) = O(f_1)$
5. $g_1 \in O(f_1) \quad\Rightarrow\quad c \cdot g_1 \in O(f_1)$
6. $\Theta(f_1) \;=\; O(f_1) \;\cap\; \Omega(f_1)$
7. $g_1 \in \Theta(f_1) \quad\Leftrightarrow\quad f_1 \in \Theta(g_1)$

Lemma 201

Let $f_1, f_2, g_1, g_2 \colon \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)) \quad \Rightarrow \quad g_1 + g_2 \in O(f_1 + f_2)$
2. $(g_1 \in O(f_1) \;\wedge\; g_2 \in O(f_2)) \quad \Rightarrow \quad g_1 \cdot g_2 \in O(f_1 \cdot f_2)$
3. $f_2 \cdot O(f_1) \subseteq O(f_1 \cdot f_2)$
4. $O(c \cdot f_1) = O(f_1)$
5. $g_1 \in O(f_1) \quad \Rightarrow \quad c \cdot g_1 \in O(f_1)$
6. $\Theta(f_1) \;=\; O(f_1) \;\cap\; \Omega(f_1)$
7. $g_1 \in \Theta(f_1) \quad \Leftrightarrow \quad f_1 \in \Theta(g_1)$
8. $(g_1 \in O(f_1) \;\wedge\; g_1 \in \Omega(f_1)) \quad \Rightarrow \quad g_1 \in \Theta(f_1)$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Basic Facts

Lemma 201

Let $f_1, f_2, g_1, g_2 \colon \mathbb{N} \to \mathbb{R}^+$, and $c \in \mathbb{R}^+$. Then the following relations hold:

1. $(g_1 \in O(f_1) \ \wedge \ g_2 \in O(f_2)) \quad \Rightarrow \quad g_1 + g_2 \in O(f_1 + f_2)$
2. $(g_1 \in O(f_1) \ \wedge \ g_2 \in O(f_2)) \quad \Rightarrow \quad g_1 \cdot g_2 \in O(f_1 \cdot f_2)$
3. $f_2 \cdot O(f_1) \subseteq O(f_1 \cdot f_2)$
4. $O(c \cdot f_1) = O(f_1)$
5. $g_1 \in O(f_1) \quad \Rightarrow \quad c \cdot g_1 \in O(f_1)$
6. $\Theta(f_1) = O(f_1) \cap \Omega(f_1)$
7. $g_1 \in \Theta(f_1) \quad \Leftrightarrow \quad f_1 \in \Theta(g_1)$
8. $(g_1 \in O(f_1) \ \wedge \ g_1 \in \Omega(f_1)) \quad \Rightarrow \quad g_1 \in \Theta(f_1)$
9. $(g_1 \in \Theta(f_1) \ \wedge \ g_2 \in \Theta(f_1)) \quad \Rightarrow \quad g_1 \in \Theta(g_2)$

Asymptotic Notation: Basic Facts

Lemma 202

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$ and $c \in \mathbb{R}^+$. Then:

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = c \quad \Rightarrow \quad g \in \Theta(f),$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 202

Let $f, g \colon \mathbb{N} \to \mathbb{R}^+$ and $c \in \mathbb{R}^+$. Then:

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = c \quad \Rightarrow \quad g \in \Theta(f),$$

and

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0 \quad \Leftrightarrow \quad g \in o(f).$$

Asymptotic Notation: Basic Facts

Lemma 202

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$ and $c \in \mathbb{R}^+$. Then:

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = c \quad \Rightarrow \quad g \in \Theta(f),$$

and

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0 \quad \Leftrightarrow \quad g \in o(f).$$

- For example, let $f, g, h\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2 - 7n$, $g(n) := 3n^2 + 5n\sqrt{n}$ and $h(n) := n^2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 202

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$ and $c \in \mathbb{R}^+$. Then:

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = c \quad \Rightarrow \quad g \in \Theta(f),$$

and

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0 \quad \Leftrightarrow \quad g \in o(f).$$

- For example, let $f, g, h\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2 - 7n$, $g(n) := 3n^2 + 5n\sqrt{n}$ and $h(n) := n^2$.
  We have $g \in \Theta(f)$ since $f \in \Theta(h)$ and $g \in \Theta(h)$:

$$\lim_{n\to\infty} \frac{f(n)}{h(n)} = \lim_{n\to\infty} \frac{n^2 - 7n}{n^2} = \lim_{n\to\infty} \left(1 - \frac{7}{n}\right) = 1$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 202

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$ and $c \in \mathbb{R}^+$. Then:

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = c \quad \Rightarrow \quad g \in \Theta(f),$$

and

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0 \quad \Leftrightarrow \quad g \in o(f).$$

- For example, let $f, g, h\colon \mathbb{N} \to \mathbb{R}^+$ with $f(n) := n^2 - 7n$, $g(n) := 3n^2 + 5n\sqrt{n}$ and $h(n) := n^2$.
  We have $g \in \Theta(f)$ since $f \in \Theta(h)$ and $g \in \Theta(h)$:

$$\lim_{n\to\infty} \frac{f(n)}{h(n)} = \lim_{n\to\infty} \frac{n^2 - 7n}{n^2} = \lim_{n\to\infty} \left(1 - \frac{7}{n}\right) = 1$$

$$\lim_{n\to\infty} \frac{g(n)}{h(n)} = \lim_{n\to\infty} \frac{3n^2 + 5n\sqrt{n}}{n^2} = \lim_{n\to\infty} \left(3 + \frac{5}{\sqrt{n}}\right) = 3$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Wide-spread Notational Abuse

- It is convenient to be a bit sloppy and write, e.g.,

$$g(n) = O(n^2) \quad \text{or} \quad g \in O(n^2)$$

rather than to resort to the λ-quantifier and write $g \in O(\lambda n.n^2)$, or

$$g \in O(f) \quad \text{with } f : \mathbb{N} \to \mathbb{R}^+,\ n \mapsto n^2.$$

Asymptotic Notation: Wide-spread Notational Abuse

- It is convenient to be a bit sloppy and write, e.g.,

$$g(n) = O(n^2) \quad \text{or} \quad g \in O(n^2)$$

rather than to resort to the λ-quantifier and write $g \in O(\lambda n.n^2)$, or

$$g \in O(f) \quad \text{with } f: \mathbb{N} \to \mathbb{R}^+,\ n \mapsto n^2.$$

- Similarly,

$$g(n) = h(n) + O(n^3)$$

means

$$|g - h| \in O(f) \quad \text{with } f: \mathbb{N} \to \mathbb{R}^+,\ n \mapsto n^3.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Wide-spread Notational Abuse

- It is convenient to be a bit sloppy and write, e.g.,

$$g(n) = O(n^2) \quad \text{or} \quad g \in O(n^2)$$

rather than to resort to the λ-quantifier and write $g \in O(\lambda n.n^2)$, or

$$g \in O(f) \quad \text{with } f : \mathbb{N} \to \mathbb{R}^+,\ n \mapsto n^2.$$

- Similarly,

$$g(n) = h(n) + O(n^3)$$

means

$$|g - h| \in O(f) \quad \text{with } f : \mathbb{N} \to \mathbb{R}^+,\ n \mapsto n^3.$$

- Furthermore,

$$g(n) = n^{O(1)}$$

indicates that

$$g \in O(f) \quad \text{with } f : \mathbb{N} \to \mathbb{R}^+,\ n \mapsto n^c$$

for some constant $c \in \mathbb{R}^+$.

Warning

1. In the equation-based notation the equality sign does not assert the equality of two functions or sets!

Asymptotic Notation: Wide-spread Notational Abuse — Caveats!

Warning

1. In the equation-based notation the equality sign does not assert the equality of two functions or sets!
2. The property expressed by this equality sign is not symmetric! That is,

$$O(n^2) = O(n^3) \quad \text{but} \quad O(n^3) \neq O(n^2).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Warning

1. In the equation-based notation the equality sign does not assert the equality of two functions or sets!
2. The property expressed by this equality sign is not symmetric! That is,

$$O(n^2) = O(n^3) \quad \text{but} \quad O(n^3) \neq O(n^2).$$

3. Stipulating

$$g(m) = O(m^n)$$

is not the same as stipulating

$$g(n) = O(m^n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Asymptotic Notation: Wide-spread Notational Abuse — Caveats!

Warning

1. In the equation-based notation the equality sign does not assert the equality of two functions or sets!
2. The property expressed by this equality sign is not symmetric! That is,
$$O(n^2) = O(n^3) \quad \text{but} \quad O(n^3) \neq O(n^2).$$
3. Stipulating
$$g(m) = O(m^n)$$
is not the same as stipulating
$$g(n) = O(m^n).$$

- So, keep in mind that an *is-element-of* or *subset relation* is meant even if an equality sign is used!
- Unfortunately, several textbooks are fuzzy about this important distinction . . .

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Conditional Asymptotic Notation

Definition 203 (Conditional Asymptotic Notation)

Consider a function $f : \mathbb{N} \to \mathbb{R}^+$

$$O(f \quad) := \{g : \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \quad : \quad g(n) \leqslant c \cdot f(n)\}\,.$$

Definition 203 (Conditional Asymptotic Notation)

Consider a function $f\colon \mathbb{N} \to \mathbb{R}^+$ and a predicate $P\colon \mathbb{N} \to \{F, T\}$.

$$O(f \quad) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \; : \qquad g(n) \leqslant c \cdot f(n)\}.$$

- E.g., let $P(n) \;:\Leftrightarrow\; n \equiv_2 0$, or $P(n) \;:\Leftrightarrow\; (\exists k \in \mathbb{N}_0 \;\; n = 2^k)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Conditional Asymptotic Notation

Definition 203 (Conditional Asymptotic Notation)

Consider a function $f\colon \mathbb{N} \to \mathbb{R}^+$ and a predicate $P\colon \mathbb{N} \to \{F, T\}$.

$$O(f \mid P) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 : \quad P(n) \Rightarrow g(n) \leqslant c \cdot f(n)\}.$$

- E.g., let $P(n) \;:\Leftrightarrow\; n \equiv_2 0$, or $P(n) \;:\Leftrightarrow\; (\exists k \in \mathbb{N}_0 \;\; n = 2^k)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 203 (Conditional Asymptotic Notation)

Consider a function $f\colon \mathbb{N} \to \mathbb{R}^+$ and a predicate $P\colon \mathbb{N} \to \{F, T\}$.

$$O(f \mid P) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 : \quad P(n) \Rightarrow g(n) \leqslant c \cdot f(n)\}.$$

$$\Omega(f \mid P) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 : \quad P(n) \Rightarrow g(n) \geqslant c \cdot f(n)\}.$$

$$\Theta(f \mid P) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \exists c_1, c_2 \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 : \quad P(n) \Rightarrow c_1 \cdot f(n) \leqslant g(n) \leqslant c_2 \cdot f(n)\}.$$

$$o(f \mid P) := \{g\colon \mathbb{N} \to \mathbb{R}^+ \mid \quad \forall c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 : \quad P(n) \Rightarrow g(n) \leqslant c \cdot f(n)\}.$$

- E.g., let $P(n) \;:\Leftrightarrow\; n \equiv_2 0$, or $P(n) \;:\Leftrightarrow\; (\exists k \in \mathbb{N}_0 \;\; n = 2^k)$.

Smoothness

Definition 204 (Eventually non-decreasing, Dt.: schlussendlich nicht abnehmend)

A function $f : \mathbb{N} \to \mathbb{R}^+$ is *eventually non-decreasing* exactly if

$$\exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad f(n) \leqslant f(n+1).$$

Smoothness

Definition 204 (Eventually non-decreasing, Dt.: schlussendlich nicht abnehmend)

A function $f\colon \mathbb{N} \to \mathbb{R}^+$ is *eventually non-decreasing* exactly if

$$\exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad f(n) \leqslant f(n+1).$$

Definition 205 (b-smooth, Dt.: b-glatt)

A function $f\colon \mathbb{N} \to \mathbb{R}^+$ is *b-smooth* for some integer $b \geqslant 2$ exactly if f is eventually non-decreasing and if

$$\exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad f(b \cdot n) \leqslant c \cdot f(n).$$

Smoothness

Definition 204 (Eventually non-decreasing, Dt.: schlussendlich nicht abnehmend)

A function $f\colon \mathbb{N} \to \mathbb{R}^+$ is *eventually non-decreasing* exactly if

$$\exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad f(n) \leqslant f(n+1).$$

Definition 205 (b-smooth, Dt.: b-glatt)

A function $f\colon \mathbb{N} \to \mathbb{R}^+$ is *b-smooth* for some integer $b \geqslant 2$ exactly if f is eventually non-decreasing and if

$$\exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad f(b \cdot n) \leqslant c \cdot f(n).$$

Definition 206 (smooth, Dt.: glatt)

A function $f\colon \mathbb{N} \to \mathbb{R}^+$ is *smooth* if it is b-smooth for all integers $b \geqslant 2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Smoothness

Definition 204 (Eventually non-decreasing, Dt.: schlussendlich nicht abnehmend)

A function $f\colon \mathbb{N} \to \mathbb{R}^+$ is *eventually non-decreasing* exactly if

$$\exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad f(n) \leqslant f(n+1).$$

Definition 205 (b-smooth, Dt.: b-glatt)

A function $f\colon \mathbb{N} \to \mathbb{R}^+$ is *b-smooth* for some integer $b \geqslant 2$ exactly if f is eventually non-decreasing and if

$$\exists c \in \mathbb{R}^+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geqslant n_0 \qquad f(b \cdot n) \leqslant c \cdot f(n).$$

Definition 206 (smooth, Dt.: glatt)

A function $f\colon \mathbb{N} \to \mathbb{R}^+$ is *smooth* if it is b-smooth for all integers $b \geqslant 2$.

Lemma 207

If $f\colon \mathbb{N} \to \mathbb{R}^+$ is b'-smooth for some integer $b' \geqslant 2$ then it is smooth.

Smoothness Rule

Theorem 208 (Smoothness Rule)

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$, and consider an integer $b \geqslant 2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 208 (Smoothness Rule)

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$, and consider an integer $b \geqslant 2$. If

1. f is a smooth function,

Smoothness Rule

Theorem 208 (Smoothness Rule)

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$, and consider an integer $b \geqslant 2$. If

1. f is a smooth function,
2. $g \in O(f \mid$ "is power of b"$)$,

Smoothness Rule

Theorem 208 (Smoothness Rule)

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$, and consider an integer $b \geqslant 2$. If

1. f is a smooth function,
2. $g \in O(f \mid$ "is power of b"$)$, and if
3. g is an eventually non-decreasing function,

then $g \in O(f)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Smoothness Rule

Theorem 208 (Smoothness Rule)

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$, and consider an integer $b \geqslant 2$. If

1. f is a smooth function,
2. $g \in O(f \mid \text{“is power of } b\text{”})$, and if
3. g is an eventually non-decreasing function,

then $g \in O(f)$.

- Similarly for $\Omega(f)$ and $\Theta(f)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Smoothness Rule

Theorem 208 (Smoothness Rule)

Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$, and consider an integer $b \geqslant 2$. If

1. f is a smooth function,
2. $g \in O(f \mid \text{“is power of } b\text{”})$, and if
3. g is an eventually non-decreasing function,

then $g \in O(f)$.

- Similarly for $\Omega(f)$ and $\Theta(f)$.
- Again, it is trivial to extend the definitions and lemmas such that $\mathbb{N}_0$ rather than $\mathbb{N}$ is taken as the base set. Similarly, we can replace $\mathbb{R}^+$ by $\mathbb{R}_0^+$ or even by $\mathbb{R}$ provided that all functions are eventually positive.
- The same comments apply to the subsequent slides.

Smoothness Rule: Sample Application

- For $a, b \in \mathbb{R}_0^+$ we define $g\colon \mathbb{N} \to \mathbb{R}_0^+$ as

$$g(n) := \begin{cases} a & \text{if } n = 1, \\ 4 \cdot g\left(\left\lceil \frac{n}{2} \right\rceil\right) + b \cdot n & \text{otherwise.} \end{cases}$$

Smoothness Rule: Sample Application

- For $a, b \in \mathbb{R}_0^+$ we define $g\colon \mathbb{N} \to \mathbb{R}_0^+$ as

$$g(n) := \begin{cases} a & \text{if } n = 1, \\ 4 \cdot g\left(\left\lceil \frac{n}{2} \right\rceil\right) + b \cdot n & \text{otherwise.} \end{cases}$$

- Note that $\left\lceil \frac{n}{2} \right\rceil = 2^{k-1}$ if $n = 2^k$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Smoothness Rule: Sample Application

- For $a, b \in \mathbb{R}_0^+$ we define $g \colon \mathbb{N} \to \mathbb{R}_0^+$ as

$$g(n) := \begin{cases} a & \text{if } n = 1, \\ 4 \cdot g\left(\left\lceil \frac{n}{2} \right\rceil\right) + b \cdot n & \text{otherwise.} \end{cases}$$

- Note that $\left\lceil \frac{n}{2} \right\rceil = 2^{k-1}$ if $n = 2^k$.
- We would like to show that $g \in \Theta(n^2)$:

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Smoothness Rule: Sample Application

- For $a, b \in \mathbb{R}_0^+$ we define $g\colon \mathbb{N} \to \mathbb{R}_0^+$ as

$$g(n) := \begin{cases} a & \text{if } n = 1, \\ 4 \cdot g\left(\left\lceil \frac{n}{2} \right\rceil\right) + b \cdot n & \text{otherwise.} \end{cases}$$

- Note that $\left\lceil \frac{n}{2} \right\rceil = 2^{k-1}$ if $n = 2^k$.
- We would like to show that $g \in \Theta(n^2)$:
  It suffices to
  - prove that f, with $f(n) := n^2$, is smooth,

Smoothness Rule: Sample Application

- For $a, b \in \mathbb{R}_0^+$ we define $g\colon \mathbb{N} \to \mathbb{R}_0^+$ as

$$g(n) := \begin{cases} a & \text{if } n = 1, \\ 4 \cdot g\left(\left\lceil \frac{n}{2} \right\rceil\right) + b \cdot n & \text{otherwise.} \end{cases}$$

- Note that $\left\lceil \frac{n}{2} \right\rceil = 2^{k-1}$ if $n = 2^k$.
- We would like to show that $g \in \Theta(n^2)$:
  It suffices to
  - prove that f, with $f(n) := n^2$, is smooth,
  - prove that $g \in \Theta(f \mid \text{“is power of 2”})$,

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Smoothness Rule: Sample Application

- For $a, b \in \mathbb{R}_0^+$ we define $g\colon \mathbb{N} \to \mathbb{R}_0^+$ as

$$g(n) := \begin{cases} a & \text{if } n = 1, \\ 4 \cdot g\left(\left\lceil \frac{n}{2} \right\rceil\right) + b \cdot n & \text{otherwise.} \end{cases}$$

- Note that $\left\lceil \frac{n}{2} \right\rceil = 2^{k-1}$ if $n = 2^k$.
- We would like to show that $g \in \Theta(n^2)$:
  It suffices to
  - prove that f, with $f(n) := n^2$, is smooth,
  - prove that $g \in \Theta(f \mid \text{“is power of 2”})$,
  - prove that g is eventually non-decreasing.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Smoothness Rule: Sample Application

- For $a, b \in \mathbb{R}_0^+$ we define $g\colon \mathbb{N} \to \mathbb{R}_0^+$ as

$$g(n) := \begin{cases} a & \text{if } n = 1, \\ 4 \cdot g\left(\left\lceil \frac{n}{2} \right\rceil\right) + b \cdot n & \text{otherwise.} \end{cases}$$

- Note that $\left\lceil \frac{n}{2} \right\rceil = 2^{k-1}$ if $n = 2^k$.
- We would like to show that $g \in \Theta(n^2)$:
  It suffices to
  - prove that f, with $f(n) := n^2$, is smooth,
  - prove that $g \in \Theta(f \mid$ "is power of 2"$)$,
  - prove that g is eventually non-decreasing.

- Standard application in computer science: Solving the recurrence relation

$$T(n) = T\left(\left\lceil \frac{n}{2} \right\rceil\right) + T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + b \cdot n,$$

e.g., as derived when analyzing the complexity of merge sort.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

6 Complexity Analysis and Recurrence Relations

- Growth Rates
- Bachmann-Landau (Asymptotic) Notation
- Recurrence Relations
  - Heuristics for Solving Recurrences
  - Solving Linear Recurrence Relations
- Master Theorem

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Recurrence Relations

- Sample sequence $t\colon \mathbb{N}_0 \to \mathbb{R}$: $(1, 2, 4, 8, 16, 32, 64, 128, 256, \ldots)$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Sample sequence $t\colon \mathbb{N}_0 \to \mathbb{R}$: $(1, 2, 4, 8, 16, 32, 64, 128, 256, \ldots)$

Definition 209 (Recurrence relation, Dt.: Rekurrenzgleichung)

A *recurrence relation* for a sequence t is an equation that relates elements of t. It is of order k, for some $k \in \mathbb{N}$, if t_n can be expressed in terms of n and $t_{n-1}, t_{n-2}, \ldots, t_{n-k}$, i.e., if t_n is of the form $t_n = f(t_{n-1}, t_{n-2}, \ldots, t_{n-k}, n)$ for $f\colon \mathbb{R}^k \times \mathbb{N} \to \mathbb{R}$ (or for $f\colon \mathbb{R}^k \times \mathbb{N}_0 \to \mathbb{R}$).

- Sample sequence $t\colon \mathbb{N}_0 \to \mathbb{R}$: $(1, 2, 4, 8, 16, 32, 64, 128, 256, \ldots)$

Definition 209 (Recurrence relation, Dt.: Rekurrenzgleichung)

A *recurrence relation* for a sequence t is an equation that relates elements of t. It is of order k, for some $k \in \mathbb{N}$, if t_n can be expressed in terms of n and $t_{n-1}, t_{n-2}, \ldots, t_{n-k}$, i.e., if t_n is of the form $t_n = f(t_{n-1}, t_{n-2}, \ldots, t_{n-k}, n)$ for $f\colon \mathbb{R}^k \times \mathbb{N} \to \mathbb{R}$ (or for $f\colon \mathbb{R}^k \times \mathbb{N}_0 \to \mathbb{R}$).

- Recurrence relation (of order 1) for the sample sequence given above:

$$t_n := \begin{cases} 1 & \text{if } n = 0, \\ 2 \cdot t_{n-1} & \text{if } n > 0. \end{cases}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Sample sequence $t\colon \mathbb{N}_0 \to \mathbb{R}$: $(1, 2, 4, 8, 16, 32, 64, 128, 256, \ldots)$

Definition 209 (Recurrence relation, Dt.: Rekurrenzgleichung)

A *recurrence relation* for a sequence t is an equation that relates elements of t. It is of order k, for some $k \in \mathbb{N}$, if t_n can be expressed in terms of n and $t_{n-1}, t_{n-2}, \ldots, t_{n-k}$, i.e., if t_n is of the form $t_n = f(t_{n-1}, t_{n-2}, \ldots, t_{n-k}, n)$ for $f\colon \mathbb{R}^k \times \mathbb{N} \to \mathbb{R}$ (or for $f\colon \mathbb{R}^k \times \mathbb{N}_0 \to \mathbb{R}$).

- Recurrence relation (of order 1) for the sample sequence given above:

$$t_n := \begin{cases} 1 & \text{if } n = 0, \\ 2 \cdot t_{n-1} & \text{if } n > 0. \end{cases}$$

- Easy to see: $t_n = 2^n$ for all $n \in \mathbb{N}_0$.

- Sample sequence $t\colon \mathbb{N}_0 \to \mathbb{R}$: $(1, 2, 4, 8, 16, 32, 64, 128, 256, \ldots)$

Definition 209 (Recurrence relation, Dt.: Rekurrenzgleichung)

A *recurrence relation* for a sequence t is an equation that relates elements of t. It is of order k, for some $k \in \mathbb{N}$, if t_n can be expressed in terms of n and $t_{n-1}, t_{n-2}, \ldots, t_{n-k}$, i.e., if t_n is of the form $t_n = f(t_{n-1}, t_{n-2}, \ldots, t_{n-k}, n)$ for $f\colon \mathbb{R}^k \times \mathbb{N} \to \mathbb{R}$ (or for $f\colon \mathbb{R}^k \times \mathbb{N}_0 \to \mathbb{R}$).

- Recurrence relation (of order 1) for the sample sequence given above:

$$t_n := \begin{cases} 1 & \text{if } n = 0, \\ 2 \cdot t_{n-1} & \text{if } n > 0. \end{cases}$$

- Easy to see: $t_n = 2^n$ for all $n \in \mathbb{N}_0$.

Note

We will freely mix the notations t_k and $t(k)$ for denoting the k-th element of a sequence $(t_n)_{n\in\mathbb{N}}$ or $(t_n)_{n\in\mathbb{N}_0}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- According to legend, life on Earth will end once the Brahmin priests managed to move the last disk in their 64-disk Tower-of-Hanoi problem . . .

Recurrence Relations: The Tower-of-Hanoi Recurrence

- According to legend, life on Earth will end once the Brahmin priests managed to move the last disk in their 64-disk Tower-of-Hanoi problem . . .
- Also according to legend, the priests apply a recursive algorithm, thereby moving
  1. the top $n-1$ disks (recursively) from pole I to the auxiliary pole III,
  2. the largest (bottom-most) disk from pole I to pole II,
  3. the top $n-1$ disks (recursively) from pole III to pole II.

Recurrence Relations: The Tower-of-Hanoi Recurrence

- According to legend, life on Earth will end once the Brahmin priests managed to move the last disk in their 64-disk Tower-of-Hanoi problem ...
- Also according to legend, the priests apply a recursive algorithm, thereby moving
  1. the top $n-1$ disks (recursively) from pole I to the auxiliary pole III,
  2. the largest (bottom-most) disk from pole I to pole II,
  3. the top $n-1$ disks (recursively) from pole III to pole II.
- If $T(n)$ denotes the number of moves for the n-disk ToH problem, the priests need two times $T(n-1)$ moves for the recursive Steps (1) and (3), and one move for getting the largest disk from pole I to II in Step (2).
- Of course, $T(1) = 1$.

- According to legend, life on Earth will end once the Brahmin priests managed to move the last disk in their 64-disk Tower-of-Hanoi problem ...
- Also according to legend, the priests apply a recursive algorithm, thereby moving
  1. the top $n-1$ disks (recursively) from pole I to the auxiliary pole III,
  2. the largest (bottom-most) disk from pole I to pole II,
  3. the top $n-1$ disks (recursively) from pole III to pole II.
- If $T(n)$ denotes the number of moves for the n-disk ToH problem, the priests need two times $T(n-1)$ moves for the recursive Steps (1) and (3), and one move for getting the largest disk from pole I to II in Step (2).
- Of course, $T(1) = 1$.
- Hence, we get the recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1$$

for the number T of moves for solving the Tower-of-Hanoi problem recursively.

Recurrence Relations: The Tower-of-Hanoi Recurrence

- According to legend, life on Earth will end once the Brahmin priests managed to move the last disk in their 64-disk Tower-of-Hanoi problem ...
- Also according to legend, the priests apply a recursive algorithm, thereby moving
  1. the top $n-1$ disks (recursively) from pole I to the auxiliary pole III,
  2. the largest (bottom-most) disk from pole I to pole II,
  3. the top $n-1$ disks (recursively) from pole III to pole II.
- If $T(n)$ denotes the number of moves for the n-disk ToH problem, the priests need two times $T(n-1)$ moves for the recursive Steps (1) and (3), and one move for getting the largest disk from pole I to II in Step (2).
- Of course, $T(1) = 1$.
- Hence, we get the recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1$$

  for the number T of moves for solving the Tower-of-Hanoi problem recursively.
- A solution of this recurrence relation tells us when life on Earth might end ...
- So, is it already time for an apocalyptic mood?
- We start with heuristics for solving recurrence relations.

Heuristics for Solving Recurrences

- Constructive Induction:
  - First ”guess” a solution.
  - Use ”constructive” induction to verify that the solution guessed is correct.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences

- Constructive Induction:
  - First ”guess” a solution.
  - Use ”constructive” induction to verify that the solution guessed is correct.
- Cascading:
  - Restate the recurrence relation for $t_n, t_{n-1}, t_{n-2}, \ldots$.
  - Manipulate and rearrange the individual equations such that summing over all equations yields a closed-form expression for t_n.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences

- Constructive Induction:
  - First "guess" a solution.
  - Use "constructive" induction to verify that the solution guessed is correct.
- Cascading:
  - Restate the recurrence relation for $t_n, t_{n-1}, t_{n-2}, \ldots$.
  - Manipulate and rearrange the individual equations such that summing over all equations yields a closed-form expression for t_n.
- Iteration:
  - Expand the recurrence relation.
  - Derive a closed-form solution.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences

- Constructive Induction:
  - First ”guess” a solution.
  - Use ”constructive” induction to verify that the solution guessed is correct.
- Cascading:
  - Restate the recurrence relation for $t_n, t_{n-1}, t_{n-2}, \ldots$.
  - Manipulate and rearrange the individual equations such that summing over all equations yields a closed-form expression for t_n.
- Iteration:
  - Expand the recurrence relation.
  - Derive a closed-form solution.

Note

All heuristics require induction to prove that the result obtained is indeed correct!

Heuristics for Solving Recurrences: Constructive Induction

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- *Guess:* $t \in O(f)$ for $f(n) := n^2$.

Heuristics for Solving Recurrences: Constructive Induction

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- *Guess:* $t \in O(f)$ for $f(n) := n^2$.
- Our guess could be verified by showing $t_n \leqslant a \cdot n^2$ for all $n \in \mathbb{N}_0$ for a suitable (but yet unknown) $a \in \mathbb{R}^+$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Constructive Induction

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- *Guess:* $t \in O(f)$ for $f(n) := n^2$.
- Our guess could be verified by showing $t_n \leqslant a \cdot n^2$ for all $n \in \mathbb{N}_0$ for a suitable (but yet unknown) $a \in \mathbb{R}^+$.
- If we assume $t_n \leqslant a \cdot n^2$ then we get

$$\begin{aligned} t_{n+1} &= t_n + (n+1) \\ &\leqslant a \cdot n^2 + (n+1) \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Constructive Induction

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- *Guess:* $t \in O(f)$ for $f(n) := n^2$.
- Our guess could be verified by showing $t_n \leqslant a \cdot n^2$ for all $n \in \mathbb{N}_0$ for a suitable (but yet unknown) $a \in \mathbb{R}^+$.
- If we assume $t_n \leqslant a \cdot n^2$ then we get

$$\begin{aligned} t_{n+1} &= t_n + (n+1) \\ &\leqslant a \cdot n^2 + (n+1) \\ &\leqslant a \cdot n^2 + 4n + 2 \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Constructive Induction

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- *Guess:* $t \in O(f)$ for $f(n) := n^2$.
- Our guess could be verified by showing $t_n \leqslant a \cdot n^2$ for all $n \in \mathbb{N}_0$ for a suitable (but yet unknown) $a \in \mathbb{R}^+$.
- If we assume $t_n \leqslant a \cdot n^2$ then we get

$$\begin{aligned} t_{n+1} &= t_n + (n+1) \\ &\leqslant a \cdot n^2 + (n+1) \\ &\leqslant a \cdot n^2 + 4n + 2 \\ &= 2(\frac{a}{2} \cdot n^2 + 2n + 1) \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Constructive Induction

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- *Guess:* $t \in O(f)$ for $f(n) := n^2$.
- Our guess could be verified by showing $t_n \leqslant a \cdot n^2$ for all $n \in \mathbb{N}_0$ for a suitable (but yet unknown) $a \in \mathbb{R}^+$.
- If we assume $t_n \leqslant a \cdot n^2$ then we get

$$\begin{aligned} t_{n+1} &= t_n + (n+1) \\ &\leqslant a \cdot n^2 + (n+1) \\ &\leqslant a \cdot n^2 + 4n + 2 \\ &= 2(\frac{a}{2} \cdot n^2 + 2n + 1) \\ &\overset{a:=2}{=} 2(n^2 + 2n + 1) \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Constructive Induction

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- *Guess:* $t \in O(f)$ for $f(n) := n^2$.
- Our guess could be verified by showing $t_n \leqslant a \cdot n^2$ for all $n \in \mathbb{N}_0$ for a suitable (but yet unknown) $a \in \mathbb{R}^+$.
- If we assume $t_n \leqslant a \cdot n^2$ then we get

$$\begin{aligned} t_{n+1} &= t_n + (n+1) \\ &\leqslant a \cdot n^2 + (n+1) \\ &\leqslant a \cdot n^2 + 4n + 2 \\ &= 2(\frac{a}{2} \cdot n^2 + 2n + 1) \\ &\overset{a:=2}{=} 2(n^2 + 2n + 1) \\ &= 2(n+1)^2. \end{aligned}$$

- Now use standard induction to show that $t_n \leqslant 2n^2$ is indeed correct for all $n \in \mathbb{N}_0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Cascading

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Restating the recurrence yields the following set of equations:

$$t_n = t_{n-1} + n$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Cascading

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Restating the recurrence yields the following set of equations:

$$\begin{aligned} t_n &= t_{n-1} + n \\ t_{n-1} &= t_{n-2} + n - 1 \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Cascading

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Restating the recurrence yields the following set of equations:

$$\begin{aligned} t_n &= t_{n-1} + n \\ t_{n-1} &= t_{n-2} + n - 1 \\ t_{n-2} &= t_{n-3} + n - 2 \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Cascading

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Restating the recurrence yields the following set of equations:

$$\begin{aligned}
t_n &= t_{n-1} + n \\
t_{n-1} &= t_{n-2} + n - 1 \\
t_{n-2} &= t_{n-3} + n - 2 \\
&\vdots \\
t_2 &= t_1 + 2 \\
t_1 &= t_0 + 1
\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Cascading

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Restating the recurrence yields the following set of equations:

$$\begin{aligned}
t_n &= t_{n-1} + n \\
t_{n-1} &= t_{n-2} + n - 1 \\
t_{n-2} &= t_{n-3} + n - 2 \\
&\vdots \\
t_2 &= t_1 + 2 \\
t_1 &= t_0 + 1 \\
\hline
t_n &= t_0 + 1 + 2 + \cdots + (n-2) + (n-1) + n
\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Cascading

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Restating the recurrence yields the following set of equations:

$$\begin{array}{rcl} t_n &=& t_{n-1} + n \\ t_{n-1} &=& t_{n-2} + n - 1 \\ t_{n-2} &=& t_{n-3} + n - 2 \\ & \vdots & \\ t_2 &=& t_1 + 2 \\ t_1 &=& t_0 + 1 \\ \hline t_n &=& t_0 + 1 + 2 + \cdots + (n-2) + (n-1) + n \\ &=& 0 + 1 + 2 + \cdots + (n-2) + (n-1) + n \end{array}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Cascading

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Restating the recurrence yields the following set of equations:

$$
\begin{aligned}
t_n &= t_{n-1} + n \\
t_{n-1} &= t_{n-2} + n - 1 \\
t_{n-2} &= t_{n-3} + n - 2 \\
&\vdots \\
t_2 &= t_1 + 2 \\
t_1 &= t_0 + 1 \\
\hline
t_n &= t_0 + 1 + 2 + \cdots + (n-2) + (n-1) + n \\
&= 0 + 1 + 2 + \cdots + (n-2) + (n-1) + n
\end{aligned}
$$

- This indicates that

$$t_n = \sum_{i=0}^{n} i = \frac{n(n+1)}{2} \in \Theta(n^2),$$

which is proved by induction.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Iteration

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Iterating the recurrence yields

$$t_n = t_{n-1} + n$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Iteration

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Iterating the recurrence yields

$$\begin{aligned} t_n &= t_{n-1} + n \\ &= \big(t_{n-2} + (n-1)\big) + n \end{aligned}$$

Heuristics for Solving Recurrences: Iteration

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Iterating the recurrence yields

$$\begin{aligned} t_n &= t_{n-1} + n \\ &= \big(t_{n-2} + (n-1)\big) + n = t_{n-2} + \big((n-1)) + n\big) \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Iteration

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Iterating the recurrence yields

$$\begin{aligned} t_n &= t_{n-1} + n \\ &= \big(t_{n-2} + (n-1)\big) + n = t_{n-2} + \big((n-1)) + n\big) \\ &= \big(t_{n-3} + (n-2)\big) + \big((n-1) + n\big) \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Iteration

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Iterating the recurrence yields

$$\begin{aligned} t_n &= t_{n-1} + n \\ &= \big(t_{n-2} + (n-1)\big) + n = t_{n-2} + \big((n-1)) + n\big) \\ &= \big(t_{n-3} + (n-2)\big) + \big((n-1) + n\big) = t_{n-3} + \big((n-2) + (n-1) + n\big) \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Iteration

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Iterating the recurrence yields

$$\begin{aligned} t_n &= t_{n-1} + n \\ &= \big(t_{n-2} + (n-1)\big) + n = t_{n-2} + \big((n-1)) + n\big) \\ &= \big(t_{n-3} + (n-2)\big) + \big((n-1) + n\big) = t_{n-3} + \big((n-2) + (n-1) + n\big) \\ &= \big(t_{n-4} + (n-3)\big) + \big((n-2) + (n-1) + n\big) \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Iteration

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Iterating the recurrence yields

$$\begin{aligned}
t_n &= t_{n-1} + n \\
&= \big(t_{n-2} + (n-1)\big) + n = t_{n-2} + \big((n-1)) + n\big) \\
&= \big(t_{n-3} + (n-2)\big) + \big((n-1) + n\big) = t_{n-3} + \big((n-2) + (n-1) + n\big) \\
&= \big(t_{n-4} + (n-3)\big) + \big((n-2) + (n-1) + n\big) \\
&\ \vdots \\
&= t_0 + 1 + 2 + \cdots + (n-1) + n \\
&= 0 + 1 + 2 + \cdots + (n-1) + n.
\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Heuristics for Solving Recurrences: Iteration

- Solve the recurrence relation $t_n = t_{n-1} + n$, with $t_0 := 0$.
- Iterating the recurrence yields

$$
\begin{aligned}
t_n &= t_{n-1} + n \\
&= \big(t_{n-2} + (n-1)\big) + n = t_{n-2} + \big((n-1)) + n\big) \\
&= \big(t_{n-3} + (n-2)\big) + \big((n-1) + n\big) = t_{n-3} + \big((n-2) + (n-1) + n\big) \\
&= \big(t_{n-4} + (n-3)\big) + \big((n-2) + (n-1) + n\big) \\
&\vdots \\
&= t_0 + 1 + 2 + \cdots + (n-1) + n \\
&= 0 + 1 + 2 + \cdots + (n-1) + n.
\end{aligned}
$$

- Again, this indicates that

$$t_n = \sum_{i=0}^{n} i = \frac{n(n+1)}{2} \in \Theta(n^2),$$

which is proved by induction.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

- Iteration yields the following identities:

$$T(n) = 2T(n-1) + 1 = 2^1 T(n-1) + 2^0$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

- Iteration yields the following identities:

$$\begin{aligned} T(n) &= 2T(n-1) + 1 = 2^1 T(n-1) + 2^0 \\ &= 2(2^1 T(n-2) + 2^0) + 2^0 \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

- Iteration yields the following identities:

$$\begin{aligned} T(n) &= 2T(n-1) + 1 = 2^1 T(n-1) + 2^0 \\ &= 2(2^1 T(n-2) + 2^0) + 2^0 = 2^2 T(n-2) + 2^1 + 2^0 \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

- Iteration yields the following identities:

$$\begin{aligned} T(n) &= 2T(n-1) + 1 = 2^1 T(n-1) + 2^0 \\ &= 2(2^1 T(n-2) + 2^0) + 2^0 = 2^2 T(n-2) + 2^1 + 2^0 \\ &= 2^2 (2^1 T(n-3) + 2^0) + 2^1 + 2^0 \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

- Iteration yields the following identities:

$$\begin{aligned} T(n) &= 2T(n-1) + 1 = 2^1 T(n-1) + 2^0 \\ &= 2(2^1 T(n-2) + 2^0) + 2^0 = 2^2 T(n-2) + 2^1 + 2^0 \\ &= 2^2 (2^1 T(n-3) + 2^0) + 2^1 + 2^0 = 2^3 T(n-3) + 2^2 + 2^1 + 2^0 \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

- Iteration yields the following identities:

$$\begin{aligned}
T(n) &= 2T(n-1) + 1 = 2^1 T(n-1) + 2^0 \\
&= 2(2^1 T(n-2) + 2^0) + 2^0 = 2^2 T(n-2) + 2^1 + 2^0 \\
&= 2^2(2^1 T(n-3) + 2^0) + 2^1 + 2^0 = 2^3 T(n-3) + 2^2 + 2^1 + 2^0 \\
&\;\;\vdots \\
&= 2^{n-1} T(n-(n-1)) + 2^{n-2} + \ldots + 2^2 + 2^1 + 2^0 \\
&= 2^{n-1} + 2^{n-2} + \ldots + 2^2 + 2^1 + 2^0
\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

- Iteration yields the following identities:

$$\begin{aligned}
T(n) &= 2T(n-1) + 1 = 2^1 T(n-1) + 2^0 \\
&= 2(2^1 T(n-2) + 2^0) + 2^0 = 2^2 T(n-2) + 2^1 + 2^0 \\
&= 2^2(2^1 T(n-3) + 2^0) + 2^1 + 2^0 = 2^3 T(n-3) + 2^2 + 2^1 + 2^0 \\
&\vdots \\
&= 2^{n-1} T(n-(n-1)) + 2^{n-2} + \ldots + 2^2 + 2^1 + 2^0 \\
&= 2^{n-1} + 2^{n-2} + \ldots + 2^2 + 2^1 + 2^0 \\
&= 2^n - 1
\end{aligned}$$

Real-World Problem: When Will Life on Earth End?

- We have the Tower-of-Hanoi recurrence relation

$$T(n) = 2T(n-1) + 1 \qquad \text{with} \quad T(1) := 1.$$

- Iteration yields the following identities:

$$\begin{aligned}
T(n) &= 2T(n-1) + 1 = 2^1 T(n-1) + 2^0 \\
&= 2(2^1 T(n-2) + 2^0) + 2^0 = 2^2 T(n-2) + 2^1 + 2^0 \\
&= 2^2 (2^1 T(n-3) + 2^0) + 2^1 + 2^0 = 2^3 T(n-3) + 2^2 + 2^1 + 2^0 \\
&\vdots \\
&= 2^{n-1} T(n-(n-1)) + 2^{n-2} + \ldots + 2^2 + 2^1 + 2^0 \\
&= 2^{n-1} + 2^{n-2} + \ldots + 2^2 + 2^1 + 2^0 \\
&= 2^n - 1
\end{aligned}$$

- Hence, if the priests manage to move one disk per second then we would have to expect the end of Earth $2^{64} - 1$ seconds after they started, i.e., roughly within $5 \cdot 10^{11}$ years ...

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Recurrence Relations

Definition 210 (Homogeneous recurrence, Dt.: homogene Rekurrenz)

A recurrence relation of order k is *homogeneous* if it is satisfied by the zero sequence.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Types of Recurrence Relations

Definition 210 (Homogeneous recurrence, Dt.: homogene Rekurrenz)

A recurrence relation of order k is *homogeneous* if it is satisfied by the zero sequence.

- E.g., $t_n := 3 \cdot n^2 \cdot t_{n-1} \cdot t_{n-2}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 210 (Homogeneous recurrence, Dt.: homogene Rekurrenz)

A recurrence relation of order k is *homogeneous* if it is satisfied by the zero sequence.

- E.g., $t_n := 3 \cdot n^2 \cdot t_{n-1} \cdot t_{n-2}$.

Definition 211 (Linear homogeneous recurrence)

A homogeneous recurrence relation of order k is *linear* if $t_n = \sum_{i=1}^{k} a_i(n) \cdot t_{n-i}$, where $a_i : \mathbb{N} \to \mathbb{R}$ for $i = 1, 2, \ldots, k$.

Definition 210 (Homogeneous recurrence, Dt.: homogene Rekurrenz)

A recurrence relation of order k is *homogeneous* if it is satisfied by the zero sequence.

- E.g., $t_n := 3 \cdot n^2 \cdot t_{n-1} \cdot t_{n-2}$.

Definition 211 (Linear homogeneous recurrence)

A homogeneous recurrence relation of order k is *linear* if $t_n = \sum_{i=1}^{k} a_i(n) \cdot t_{n-i}$, where $a_i \colon \mathbb{N} \to \mathbb{R}$ for $i = 1, 2, \ldots, k$.

- E.g., $t_n := n^2 \cdot t_{n-1} + 3 \cdot t_{n-2}$.

Types of Recurrence Relations

Definition 210 (Homogeneous recurrence, Dt.: homogene Rekurrenz)

A recurrence relation of order k is *homogeneous* if it is satisfied by the zero sequence.

- E.g., $t_n := 3 \cdot n^2 \cdot t_{n-1} \cdot t_{n-2}$.

Definition 211 (Linear homogeneous recurrence)

A homogeneous recurrence relation of order k is *linear* if $t_n = \sum_{i=1}^{k} a_i(n) \cdot t_{n-i}$, where $a_i \colon \mathbb{N} \to \mathbb{R}$ for $i = 1, 2, \ldots, k$.

- E.g., $t_n := n^2 \cdot t_{n-1} + 3 \cdot t_{n-2}$.

Definition 212 (Linear homogeneous recurrence with constant coefficients)

A linear homogeneous recurrence relation of order k has *constant coefficients* if $t_n = \sum_{i=1}^{k} a_i \cdot t_{n-i}$, where $a_1, a_2, \ldots, a_k \in \mathbb{R}$.

Definition 210 (Homogeneous recurrence, Dt.: homogene Rekurrenz)

A recurrence relation of order k is *homogeneous* if it is satisfied by the zero sequence.

- E.g., $t_n := 3 \cdot n^2 \cdot t_{n-1} \cdot t_{n-2}$.

Definition 211 (Linear homogeneous recurrence)

A homogeneous recurrence relation of order k is *linear* if $t_n = \sum_{i=1}^{k} a_i(n) \cdot t_{n-i}$, where $a_i : \mathbb{N} \to \mathbb{R}$ for $i = 1, 2, \ldots, k$.

- E.g., $t_n := n^2 \cdot t_{n-1} + 3 \cdot t_{n-2}$.

Definition 212 (Linear homogeneous recurrence with constant coefficients)

A linear homogeneous recurrence relation of order k has *constant coefficients* if $t_n = \sum_{i=1}^{k} a_i \cdot t_{n-i}$, where $a_1, a_2, \ldots, a_k \in \mathbb{R}$.

- E.g., $t_n := 2 \cdot t_{n-1} + 3 \cdot t_{n-2}$.

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

Lemma 213

Consider the recurrence relation $a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0$, with $a_i \in \mathbb{R}$. If (f_n) and (g_n) satisfy the recurrence relation then $(\alpha f_n + \beta g_n)$ satisfies the recurrence relation for all $\alpha, \beta \in \mathbb{R}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 213

Consider the recurrence relation $a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0$, with $a_i \in \mathbb{R}$. If (f_n) and (g_n) satisfy the recurrence relation then $(\alpha f_n + \beta g_n)$ satisfies the recurrence relation for all $\alpha, \beta \in \mathbb{R}$.

Proof: Suppose that

$$a_0 f_n + a_1 f_{n-1} + \cdots + a_k f_{n-k} = \sum_{i=0}^{k} a_i f_{n-i} = 0 \qquad \text{and} \qquad \sum_{i=0}^{k} a_i g_{n-i} = 0$$

for all $n \geqslant k$.

Lemma 213

Consider the recurrence relation $a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0$, with $a_i \in \mathbb{R}$. If (f_n) and (g_n) satisfy the recurrence relation then $(\alpha f_n + \beta g_n)$ satisfies the recurrence relation for all $\alpha, \beta \in \mathbb{R}$.

Proof: Suppose that

$$a_0 f_n + a_1 f_{n-1} + \cdots + a_k f_{n-k} = \sum_{i=0}^{k} a_i f_{n-i} = 0 \quad \text{and} \quad \sum_{i=0}^{k} a_i g_{n-i} = 0$$

for all $n \geqslant k$. Let $\alpha, \beta \in \mathbb{R}$ arbitrary but fixed and consider $(\alpha f_n + \beta g_n)$.

Lemma 213

Consider the recurrence relation $a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0$, with $a_i \in \mathbb{R}$. If (f_n) and (g_n) satisfy the recurrence relation then $(\alpha f_n + \beta g_n)$ satisfies the recurrence relation for all $\alpha, \beta \in \mathbb{R}$.

Proof: Suppose that

$$a_0 f_n + a_1 f_{n-1} + \cdots + a_k f_{n-k} = \sum_{i=0}^{k} a_i f_{n-i} = 0 \qquad \text{and} \qquad \sum_{i=0}^{k} a_i g_{n-i} = 0$$

for all $n \geqslant k$. Let $\alpha, \beta \in \mathbb{R}$ arbitrary but fixed and consider $(\alpha f_n + \beta g_n)$. We get

$$\sum_{i=0}^{k} a_i(\alpha f_{n-i} + \beta g_{n-i}) = \alpha \sum_{i=0}^{k} a_i f_{n-i} + \beta \sum_{i=0}^{k} a_i g_{n-i} = 0.$$

□

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- So, consider $a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0$
- Guess $t_n = x^n$ for some unknown $x \in \mathbb{R}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- So, consider $a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0$
- Guess $t_n = x^n$ for some unknown $x \in \mathbb{R}$.
- Then $a_0 x^n + a_1 x^{n-1} + \cdots + a_k x^{n-k} = 0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- So, consider $a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0$
- Guess $t_n = x^n$ for some unknown $x \in \mathbb{R}$.
- Then $a_0 x^n + a_1 x^{n-1} + \cdots + a_k x^{n-k} = 0$.
- Further $x^{n-k}(a_0 x^k + a_1 x^{k-1} + \cdots + a_k) = 0$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- So, consider $a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0$
- Guess $t_n = x^n$ for some unknown $x \in \mathbb{R}$.
- Then $a_0 x^n + a_1 x^{n-1} + \cdots + a_k x^{n-k} = 0$.
- Further $x^{n-k}(a_0 x^k + a_1 x^{k-1} + \cdots + a_k) = 0$.
- If we ignore the trivial solution $x := 0$ then we get

  $$a_0 x^k + a_1 x^{k-1} + \cdots + a_k = 0$$

  as the so-called *characteristic equation* of the recurrence relation

  $$a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = 0.$$

- Hence, any root r of this equation serves as a partial solution of the recurrence relation, with $t_n := r^n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- Suppose that the characteristic equation has k distinct roots $r_1, \ldots, r_k$ such that all roots are real numbers. I.e., the characteristic equation is given as

$$\prod_{i=1}^{k}(x - r_i) = 0.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- Suppose that the characteristic equation has k distinct roots $r_1, \ldots, r_k$ such that all roots are real numbers. I.e., the characteristic equation is given as

$$\prod_{i=1}^{k} (x - r_i) = 0.$$

- Then, the general solution of the recurrence relation is of the form

$$t_n = \sum_{i=1}^{k} c_i \cdot r_i^n,$$

for some constants $c_1, c_2, \ldots, c_k \in \mathbb{R}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- Suppose that the characteristic equation has k distinct roots $r_1, \ldots, r_k$ such that all roots are real numbers. I.e., the characteristic equation is given as

$$\prod_{i=1}^{k}(x - r_i) = 0.$$

- Then, the general solution of the recurrence relation is of the form

$$t_n = \sum_{i=1}^{k} c_i \cdot r_i^n,$$

for some constants $c_1, c_2, \ldots, c_k \in \mathbb{R}$.

- The constants c_i are determined based on the initial condition(s).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients: Fibonacci Sequence

- Consider the *Fibonacci* sequence (over $\mathbb{N}_0$)

$$F_n := \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n \geqslant 2. \end{cases}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients: Fibonacci Sequence

- Consider the *Fibonacci* sequence (over $\mathbb{N}_0$)

$$F_n := \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n \geqslant 2. \end{cases}$$

- Hence, $F_n - F_{n-1} - F_{n-2} = 0$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients: Fibonacci Sequence

- Consider the *Fibonacci* sequence (over $\mathbb{N}_0$)

$$F_n := \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n \geqslant 2. \end{cases}$$

- Hence, $F_n - F_{n-1} - F_{n-2} = 0$, and we get

$$x^2 - x - 1 = 0$$

as the characteristic equation.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients: Fibonacci Sequence

- Consider the *Fibonacci* sequence (over $\mathbb{N}_0$)

$$F_n := \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n \geqslant 2. \end{cases}$$

- Hence, $F_n - F_{n-1} - F_{n-2} = 0$, and we get

$$x^2 - x - 1 = 0$$

as the characteristic equation.

- This characteristic equation has the roots

$$r_1 := \frac{1+\sqrt{5}}{2} \quad \text{and} \quad r_2 := \frac{1-\sqrt{5}}{2}.$$

- Note: r_1 is known as the *golden ratio*, ϕ, with $\phi \approx 1.618$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients: Fibonacci Sequence

- Consider the *Fibonacci* sequence (over $\mathbb{N}_0$)

$$F_n := \begin{cases} 0 & \text{if } n = 0, \\ 1 & \text{if } n = 1, \\ F_{n-1} + F_{n-2} & \text{if } n \geqslant 2. \end{cases}$$

- Hence, $F_n - F_{n-1} - F_{n-2} = 0$, and we get

$$x^2 - x - 1 = 0$$

as the characteristic equation.

- This characteristic equation has the roots

$$r_1 := \frac{1+\sqrt{5}}{2} \quad \text{and} \quad r_2 := \frac{1-\sqrt{5}}{2}.$$

- Note: r_1 is known as the *golden ratio*, ϕ, with $\phi \approx 1.618$.
- This yields

$$F_n = c_1 \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n + c_2 \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients: Fibonacci Sequence

- This yields

$$F_n = c_1 \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n + c_2 \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n.$$

- The constants c_1, c_2 are determined by resorting to the initial conditions.

$$n := 0: \qquad F_0 = 0 = c_1 + c_2$$

$$n := 1: \qquad F_1 = 1 = c_1 \cdot \frac{1+\sqrt{5}}{2} + c_2 \cdot \frac{1-\sqrt{5}}{2}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients: Fibonacci Sequence

- This yields

$$F_n = c_1 \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n + c_2 \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n.$$

- The constants c_1, c_2 are determined by resorting to the initial conditions.

$$n := 0: \qquad F_0 = 0 = c_1 + c_2$$

$$n := 1: \qquad F_1 = 1 = c_1 \cdot \frac{1+\sqrt{5}}{2} + c_2 \cdot \frac{1-\sqrt{5}}{2}$$

- By solving this linear system we obtain $c_1 = -c_2 = \frac{1}{\sqrt{5}}$.
- Hence,

$$F_n = \frac{1}{\sqrt{5}} \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}} \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- *Multiple roots*: Suppose that the characteristic equation has s distinct roots $r_1, \ldots, r_s$ of multiplicities $m_1, \ldots, m_s$ such that all roots are real numbers. I.e., the characteristic equation is given as

$$\prod_{i=1}^{s}(x - r_i)^{m_i} = 0.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- *Multiple roots*: Suppose that the characteristic equation has s distinct roots $r_1, \ldots, r_s$ of multiplicities $m_1, \ldots, m_s$ such that all roots are real numbers. I.e., the characteristic equation is given as

$$\prod_{i=1}^{s} (x - r_i)^{m_i} = 0.$$

- Then we have

$$t_n = \sum_{i=1}^{s} \sum_{j=0}^{m_i - 1} c_{ij} \cdot n^j \cdot r_i^n,$$

for constants $c_{ij} \in \mathbb{R}$.

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- *Multiple roots*: Suppose that the characteristic equation has s distinct roots $r_1, \ldots, r_s$ of multiplicities $m_1, \ldots, m_s$ such that all roots are real numbers. I.e., the characteristic equation is given as

$$\prod_{i=1}^{s} (x - r_i)^{m_i} = 0.$$

- Then we have

$$t_n = \sum_{i=1}^{s} \sum_{j=0}^{m_i - 1} c_{ij} \cdot n^j \cdot r_i^n,$$

  for constants $c_{ij} \in \mathbb{R}$.

- E.g., for the characteristic equation $(x-1) \cdot (x-2)^2 = 0$ we have $s = 2$, $r_1 = 1$, $r_2 = 2$, $m_1 = 1$, $m_2 = 2$,

Solving Linear Homogeneous Recurrence Relations With Constant Coefficients

- *Multiple roots*: Suppose that the characteristic equation has s distinct roots $r_1, \ldots, r_s$ of multiplicities $m_1, \ldots, m_s$ such that all roots are real numbers. I.e., the characteristic equation is given as

$$\prod_{i=1}^{s}(x - r_i)^{m_i} = 0.$$

- Then we have

$$t_n = \sum_{i=1}^{s} \sum_{j=0}^{m_i - 1} c_{ij} \cdot n^j \cdot r_i^n,$$

  for constants $c_{ij} \in \mathbb{R}$.

- E.g., for the characteristic equation $(x-1)\cdot(x-2)^2 = 0$ we have $s = 2$, $r_1 = 1$, $r_2 = 2$, $m_1 = 1$, $m_2 = 2$, and get

$$t_n = c_{10} \cdot n^0 \cdot 1^n + c_{20} \cdot n^0 \cdot 2^n + c_{21} \cdot n^1 \cdot 2^n = c_{10} + c_{20} \cdot 2^n + c_{21} \cdot n \cdot 2^n.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients

- Assume we have an inhomogeneous recurrence relation of the following form:

$$a_0 \cdot t_n + a_1 \cdot t_{n-1} + \cdots + a_k \cdot t_{n-k} = b_1^n \cdot p_1(n) + b_2^n \cdot p_2(n) + \cdots + b_t^n \cdot p_t(n),$$

where $t \in \mathbb{N}_0$ and b_i is constant and p_i is a polynomial in n of degree $d_i \in \mathbb{N}_0$ for each $1 \leqslant i \leqslant t$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients

- Assume we have an inhomogeneous recurrence relation of the following form:

$$a_0 \cdot t_n + a_1 \cdot t_{n-1} + \cdots + a_k \cdot t_{n-k} = b_1^n \cdot p_1(n) + b_2^n \cdot p_2(n) + \cdots + b_t^n \cdot p_t(n),$$

  where $t \in \mathbb{N}_0$ and b_i is constant and p_i is a polynomial in n of degree $d_i \in \mathbb{N}_0$ for each $1 \leqslant i \leqslant t$.
- Then the characteristic polynomial is

$$(a_0 \cdot x^k + a_1 \cdot x^{k-1} + \cdots + a_k) \cdot \prod_{i=1}^{t} (x - b_i)^{d_i+1} = 0.$$

- Now proceed as in the homogeneous case.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients

Theorem 214

Consider the linear inhomogeneous recurrence relation

$$a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = \sum_{i=1}^{t} b_i^n \cdot p_i(n),$$

where $t \in \mathbb{N}_0$, and b_i is constant and p_i is a polynomial in n of degree $d_i \in \mathbb{N}_0$ for each $1 \leqslant i \leqslant t$,

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients

Theorem 214

Consider the linear inhomogeneous recurrence relation

$$a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = \sum_{i=1}^{t} b_i^n \cdot p_i(n),$$

where $t \in \mathbb{N}_0$, and b_i is constant and p_i is a polynomial in n of degree $d_i \in \mathbb{N}_0$ for each $1 \leqslant i \leqslant t$, and suppose that its characteristic equation

$$(a_0 x^k + a_1 x^{k-1} + \cdots + a_k) \cdot \prod_{i=1}^{t} (x - b_i)^{d_i+1} = 0$$

has s distinct roots $r_1, \ldots, r_s$ of multiplicities $m_1, \ldots, m_s$ such that all roots are real numbers.

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients

Theorem 214

Consider the linear inhomogeneous recurrence relation

$$a_0 t_n + a_1 t_{n-1} + \cdots + a_k t_{n-k} = \sum_{i=1}^{t} b_i^n \cdot p_i(n),$$

where $t \in \mathbb{N}_0$, and b_i is constant and p_i is a polynomial in n of degree $d_i \in \mathbb{N}_0$ for each $1 \leqslant i \leqslant t$, and suppose that its characteristic equation

$$(a_0 x^k + a_1 x^{k-1} + \cdots + a_k) \cdot \prod_{i=1}^{t} (x - b_i)^{d_i+1} = 0$$

has s distinct roots $r_1, \ldots, r_s$ of multiplicities $m_1, \ldots, m_s$ such that all roots are real numbers. Then the general solution of the recurrence relation is given by

$$t_n = \sum_{i=1}^{s} \sum_{j=0}^{m_i - 1} c_{ij} \cdot n^j \cdot r_i^n,$$

for constants $c_{ij} \in \mathbb{R}$.

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- Consider

$$t_n := \begin{cases} 0 & \text{if } n = 0, \\ 2t_{n-1} + n + 2^n & \text{otherwise.} \end{cases}$$

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- Consider

$$t_n := \begin{cases} 0 & \text{if } n = 0, \\ 2t_{n-1} + n + 2^n & \text{otherwise.} \end{cases}$$

- The standard form of this recurrence is

$$t_n - 2t_{n-1} = n + 2^n$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- Consider

$$t_n := \begin{cases} 0 & \text{if } n = 0, \\ 2t_{n-1} + n + 2^n & \text{otherwise.} \end{cases}$$

- The standard form of this recurrence is

$$t_n - 2t_{n-1} = n + 2^n = 1^n \cdot n^1 + 2^n \cdot n^0.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- Consider

$$t_n := \begin{cases} 0 & \text{if } n = 0, \\ 2t_{n-1} + n + 2^n & \text{otherwise.} \end{cases}$$

- The standard form of this recurrence is

$$t_n - 2t_{n-1} = n + 2^n = 1^n \cdot n^1 + 2^n \cdot n^0.$$

- Hence, relative to Thm. 214, we get

$$k = 1 \qquad a_0 = 1 \qquad a_1 = -2 \qquad t = 2$$

$$b_1 = 1 \qquad p_1(n) = n \qquad d_1 = 1 \qquad b_2 = 2 \qquad p_2(n) = 1 \qquad d_2 = 0.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- Consider

$$t_n := \begin{cases} 0 & \text{if } n = 0, \\ 2t_{n-1} + n + 2^n & \text{otherwise.} \end{cases}$$

- The standard form of this recurrence is

$$t_n - 2t_{n-1} = n + 2^n = 1^n \cdot n^1 + 2^n \cdot n^0.$$

- Hence, relative to Thm. 214, we get

$$k = 1 \qquad a_0 = 1 \qquad a_1 = -2 \qquad t = 2$$

$$b_1 = 1 \qquad p_1(n) = n \qquad d_1 = 1 \qquad b_2 = 2 \qquad p_2(n) = 1 \qquad d_2 = 0.$$

- This results in

$$0 = (x-2) \cdot (x-1)^2 \cdot (x-2)^1 = (x-1)^2 \cdot (x-2)^2$$

as the characteristic equation,

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- Consider

$$t_n := \begin{cases} 0 & \text{if } n = 0, \\ 2t_{n-1} + n + 2^n & \text{otherwise.} \end{cases}$$

- The standard form of this recurrence is

$$t_n - 2t_{n-1} = n + 2^n = 1^n \cdot n^1 + 2^n \cdot n^0.$$

- Hence, relative to Thm. 214, we get

$$k = 1 \qquad a_0 = 1 \qquad a_1 = -2 \qquad t = 2$$

$$b_1 = 1 \qquad p_1(n) = n \qquad d_1 = 1 \qquad b_2 = 2 \qquad p_2(n) = 1 \qquad d_2 = 0.$$

- This results in

$$0 = (x-2) \cdot (x-1)^2 \cdot (x-2)^1 = (x-1)^2 \cdot (x-2)^2$$

as the characteristic equation, and we get, with $r_1 := 1, r_2 := 2, m_1 = m_2 := 2$,

$$\begin{aligned} t_n &= c_{10} \cdot n^0 \cdot 1^n + c_{11} \cdot n^1 \cdot 1^n + c_{20} \cdot n^0 \cdot 2^n + c_{21} \cdot n^1 \cdot 2^n \\ &= c_{10} + c_{11} \cdot n + c_{20} \cdot 2^n + c_{21} \cdot n \cdot 2^n. \end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- So, we know that

$$t_n = c_{10} + c_{11} \cdot n + c_{20} \cdot 2^n + c_{21} \cdot n \cdot 2^n.$$

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- So, we know that

$$t_n = c_{10} + c_{11} \cdot n + c_{20} \cdot 2^n + c_{21} \cdot n \cdot 2^n.$$

- The constants $c_{10}, c_{11}, c_{20}, c_{21}$ are determined by resorting to the initial conditions:

$$\begin{aligned}
n := 0: \quad 0 &= c_{10} + c_{11} \cdot 0 + c_{20} \cdot 2^0 + c_{21} \cdot 0 \cdot 2^0 = c_{10} + c_{20} \\
n := 1: \quad 3 &= c_{10} + c_{11} + 2 \cdot c_{20} + 2 \cdot c_{21} \\
n := 2: \quad 12 &= c_{10} + 2 \cdot c_{11} + 4 \cdot c_{20} + 8 \cdot c_{21} \\
n := 3: \quad 35 &= c_{10} + 3 \cdot c_{11} + 8 \cdot c_{20} + 24 \cdot c_{21}
\end{aligned}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- So, we know that

$$t_n = c_{10} + c_{11} \cdot n + c_{20} \cdot 2^n + c_{21} \cdot n \cdot 2^n.$$

- The constants $c_{10}, c_{11}, c_{20}, c_{21}$ are determined by resorting to the initial conditions:

$$\begin{aligned}
n := 0: &\quad 0 = c_{10} + c_{11} \cdot 0 + c_{20} \cdot 2^0 + c_{21} \cdot 0 \cdot 2^0 = c_{10} + c_{20} \\
n := 1: &\quad 3 = c_{10} + c_{11} + 2 \cdot c_{20} + 2 \cdot c_{21} \\
n := 2: &\quad 12 = c_{10} + 2 \cdot c_{11} + 4 \cdot c_{20} + 8 \cdot c_{21} \\
n := 3: &\quad 35 = c_{10} + 3 \cdot c_{11} + 8 \cdot c_{20} + 24 \cdot c_{21}
\end{aligned}$$

- Solving this system of four linear equations for $c_{10}, c_{11}, c_{20}, c_{21}$ yields

$$c_{10} = -2, \qquad c_{11} = -1, \qquad c_{20} = 2, \qquad c_{21} = 1.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Solving Linear Inhomogeneous Recurrence Relations With Constant Coefficients: Sample Solution

- So, we know that

$$t_n = c_{10} + c_{11} \cdot n + c_{20} \cdot 2^n + c_{21} \cdot n \cdot 2^n.$$

- The constants $c_{10}, c_{11}, c_{20}, c_{21}$ are determined by resorting to the initial conditions:

$$\begin{aligned} n := 0: &\quad 0 = c_{10} + c_{11} \cdot 0 + c_{20} \cdot 2^0 + c_{21} \cdot 0 \cdot 2^0 = c_{10} + c_{20} \\ n := 1: &\quad 3 = c_{10} + c_{11} + 2 \cdot c_{20} + 2 \cdot c_{21} \\ n := 2: &\quad 12 = c_{10} + 2 \cdot c_{11} + 4 \cdot c_{20} + 8 \cdot c_{21} \\ n := 3: &\quad 35 = c_{10} + 3 \cdot c_{11} + 8 \cdot c_{20} + 24 \cdot c_{21} \end{aligned}$$

- Solving this system of four linear equations for $c_{10}, c_{11}, c_{20}, c_{21}$ yields

$$c_{10} = -2, \qquad c_{11} = -1, \qquad c_{20} = 2, \qquad c_{21} = 1.$$

- We conclude that

$$t_n = -2 - n + 2 \cdot 2^n + n \cdot 2^n, \qquad \text{i.e.,} \quad t_n = -2 - n + 2^{n+1} + n \cdot 2^n.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

6 Complexity Analysis and Recurrence Relations

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 215 (Master theorem, Dt.: Hauptsatz der Laufzeitfunktionen)

Consider constants $c \in \mathbb{R}^+$, $k, n_0 \in \mathbb{N}$ and $a, b \in \mathbb{N}$ with $b \geqslant 2$, and let $T: \mathbb{N} \to \mathbb{R}_0^+$ be an eventually non-decreasing function such that

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + c \cdot n^k$$

for all $n \in \mathbb{N}$ with $n \geqslant n_0$, where we interpret $T(\frac{n}{b})$ as (a combination of) $T(\lceil \frac{n}{b} \rceil)$ or $T(\lfloor \frac{n}{b} \rfloor)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 215 (Master theorem, Dt.: Hauptsatz der Laufzeitfunktionen)

Consider constants $c \in \mathbb{R}^+$, $k, n_0 \in \mathbb{N}$ and $a, b \in \mathbb{N}$ with $b \geqslant 2$, and let $T\colon \mathbb{N} \to \mathbb{R}_0^+$ be an eventually non-decreasing function such that

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + c \cdot n^k$$

for all $n \in \mathbb{N}$ with $n \geqslant n_0$, where we interpret $T(\frac{n}{b})$ as (a combination of) $T(\lceil \frac{n}{b} \rceil)$ or $T(\lfloor \frac{n}{b} \rfloor)$.

Then we have

$$T \in \begin{cases} \Theta(n^k) & \text{if } a < b^k, \\ \Theta(n^k \log n) & \text{if } a = b^k, \\ \Theta(n^{\log_b a}) & \text{if } a > b^k. \end{cases}$$

Theorem 215 (Master theorem, Dt.: Hauptsatz der Laufzeitfunktionen)

Consider constants $c \in \mathbb{R}^+$, $k, n_0 \in \mathbb{N}$ and $a, b \in \mathbb{N}$ with $b \geqslant 2$, and let $T\colon \mathbb{N} \to \mathbb{R}_0^+$ be an eventually non-decreasing function such that

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + c \cdot n^k$$

for all $n \in \mathbb{N}$ with $n \geqslant n_0$, where we interpret $T(\frac{n}{b})$ as (a combination of) $T(\lceil \frac{n}{b} \rceil)$ or $T(\lfloor \frac{n}{b} \rfloor)$.

Then we have

$$T \in \begin{cases} \Theta(n^k) & \text{if } a < b^k, \\ \Theta(n^k \log n) & \text{if } a = b^k, \\ \Theta(n^{\log_b a}) & \text{if } a > b^k. \end{cases}$$

- E.g., we get $T \in \Theta(n \log n)$ for T defined as follows:

$$T(n) = T\left(\left\lceil \frac{n}{2} \right\rceil\right) + T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + c \cdot n.$$

Master Theorem (Asymptotic Version)

Theorem 216

Consider constants $k, n_0 \in \mathbb{N}$ and $a, b \in \mathbb{N}$ with $b \geqslant 2$, and a function $f\colon \mathbb{N} \to \mathbb{R}_0^+$ with $f \in \Theta(n^k)$. Let $T\colon \mathbb{N} \to \mathbb{R}_0^+$ be an eventually non-decreasing function such that

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$

for all $n \in \mathbb{N}$ with $n \geqslant n_0$, where we interpret $T(\frac{n}{b})$ as (a combination of) $T(\lceil \frac{n}{b} \rceil)$ or $T(\lfloor \frac{n}{b} \rfloor)$.

Then we have

$$T \in \begin{cases} \Theta(n^k) & \text{if } a < b^k, \\ \Theta(n^k \log n) & \text{if } a = b^k, \\ \Theta(n^{\log_b a}) & \text{if } a > b^k. \end{cases}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Master Theorem (Refined Asymptotic Version)

Theorem 217

Consider constants $n_0 \in \mathbb{N}$ and $a \in \mathbb{N}$, $b \in \mathbb{R}$ with $b > 1$, and a function $f: \mathbb{N} \to \mathbb{R}_0^+$. Let $T: \mathbb{N} \to \mathbb{R}_0^+$ be an eventually non-decreasing function such that

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$

for all $n \in \mathbb{N}$ with $n \geqslant n_0$, where we interpret $T(\frac{n}{b})$ as (a combination of) $T(\lceil \frac{n}{b} \rceil)$ or $T(\lfloor \frac{n}{b} \rfloor)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Master Theorem (Refined Asymptotic Version)

Theorem 217

Consider constants $n_0 \in \mathbb{N}$ and $a \in \mathbb{N}$, $b \in \mathbb{R}$ with $b > 1$, and a function $f : \mathbb{N} \to \mathbb{R}_0^+$. Let $T : \mathbb{N} \to \mathbb{R}_0^+$ be an eventually non-decreasing function such that

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$

for all $n \in \mathbb{N}$ with $n \geqslant n_0$, where we interpret $T(\frac{n}{b})$ as (a combination of) $T(\lceil \frac{n}{b} \rceil)$ or $T(\lfloor \frac{n}{b} \rfloor)$.
Then we have

$$T \in \begin{cases} \Theta(f) & \text{if } \begin{cases} f \in \Omega(n^{(\log_b a)+\varepsilon}) \text{ for some } \varepsilon \in \mathbb{R}^+, \\ \text{and if the following regularity condition holds} \\ \text{for some } 0 < s < 1 \text{ and all sufficiently large n:} \\ \quad a \cdot f(n/b) \leqslant s \cdot f(n), \end{cases} \\ \Theta\left(n^{\log_b a} \log n\right) & \text{if } f \in \Theta(n^{\log_b a}), \\ \Theta(n^{\log_b a}) & \text{if } f \in O(n^{(\log_b a)-\varepsilon}) \text{ for some } \varepsilon \in \mathbb{R}^+. \end{cases}$$

- This is a simplified version of the Akra-Bazzi Theorem [Akra&Bazzi 1998].

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Fast Integer Multiplication

- The standard multiplication of two integers a, b represented as binary numbers with $2n$ bits each requires $\Theta(n^2)$ many additions and shifts of bits.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Fast Integer Multiplication

- The standard multiplication of two integers a, b represented as binary numbers with $2n$ bits each requires $\Theta(n^2)$ many additions and shifts of bits.
- Can we do any better and achieve $o(n^2)$ time?

Real-World Application: Analysis of Fast Integer Multiplication

- The standard multiplication of two integers a, b represented as binary numbers with $2n$ bits each requires $\Theta(n^2)$ many additions and shifts of bits.
- Can we do any better and achieve $o(n^2)$ time? Yes!
- [Karatsuba (1960–1963)]: Let

$$(a_{2n-1}a_{2n-2}\cdots a_1a_0)_2 \qquad \text{and} \qquad (b_{2n-1}b_{2n-2}\cdots b_1b_0)_2$$

be the $2n$-bit binary representations of a and b. Hence, $a = \sum_{i=0}^{2n-1} a_i 2^i$ and $b = \sum_{i=0}^{2n-1} b_i 2^i$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- The standard multiplication of two integers a, b represented as binary numbers with $2n$ bits each requires $\Theta(n^2)$ many additions and shifts of bits.
- Can we do any better and achieve $o(n^2)$ time? Yes!
- [Karatsuba (1960–1963)]: Let

$$(a_{2n-1}a_{2n-2}\cdots a_1a_0)_2 \qquad \text{and} \qquad (b_{2n-1}b_{2n-2}\cdots b_1b_0)_2$$

  be the $2n$-bit binary representations of a and b. Hence, $a = \sum_{i=0}^{2n-1} a_i 2^i$ and $b = \sum_{i=0}^{2n-1} b_i 2^i$.
- We have

$$a \sim 2^n A_1 + A_0 \qquad \text{and} \qquad b \sim 2^n B_1 + B_0$$

  with

$$A_1 := (a_{2n-1}a_{2n-2}\cdots a_{n+1}, a_n)_2, \quad A_0 := (a_{n-1}a_{n-2}\cdots a_1a_0)_2,$$

$$B_1 := (b_{2n-1}b_{2n-2}\cdots b_{n+1}, b_n)_2, \quad B_0 := (b_{n-1}b_{n-2}\cdots b_1b_0)_2.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Fast Integer Multiplication

- The standard multiplication of two integers a, b represented as binary numbers with $2n$ bits each requires $\Theta(n^2)$ many additions and shifts of bits.
- Can we do any better and achieve $o(n^2)$ time? Yes!
- [Karatsuba (1960–1963)]: Let

$$(a_{2n-1}a_{2n-2}\cdots a_1a_0)_2 \qquad \text{and} \qquad (b_{2n-1}b_{2n-2}\cdots b_1b_0)_2$$

  be the $2n$-bit binary representations of a and b. Hence, $a = \sum_{i=0}^{2n-1} a_i 2^i$ and $b = \sum_{i=0}^{2n-1} b_i 2^i$.
- We have

$$a \sim 2^n A_1 + A_0 \qquad \text{and} \qquad b \sim 2^n B_1 + B_0$$

  with

$$A_1 := (a_{2n-1}a_{2n-2}\cdots a_{n+1}, a_n)_2, \quad A_0 := (a_{n-1}a_{n-2}\cdots a_1a_0)_2,$$

$$B_1 := (b_{2n-1}b_{2n-2}\cdots b_{n+1}, b_n)_2, \quad B_0 := (b_{n-1}b_{n-2}\cdots b_1b_0)_2.$$

- We get

$$a \cdot b \sim 2^{2n} A_1 \cdot B_1 + 2^n(A_1 \cdot B_0 + A_0 \cdot B_1) + A_0 \cdot B_0.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Fast Integer Multiplication

- We get

$$a \cdot b \sim 2^{2n} A_1 \cdot B_1 + 2^n (A_1 \cdot B_0 + A_0 \cdot B_1) + A_0 \cdot B_0,$$

which can be rewritten as

$$a \cdot b \sim (2^{2n} + 2^n) A_1 \cdot B_1 + 2^n (A_1 - A_0) \cdot (B_0 - B_1) + (2^n + 1) A_0 \cdot B_0.$$

- We get

  $$a \cdot b \sim 2^{2n} A_1 \cdot B_1 + 2^n (A_1 \cdot B_0 + A_0 \cdot B_1) + A_0 \cdot B_0,$$

  which can be rewritten as

  $$a \cdot b \sim (2^{2n} + 2^n) A_1 \cdot B_1 + 2^n (A_1 - A_0) \cdot (B_0 - B_1) + (2^n + 1) A_0 \cdot B_0.$$

- Thus, the multiplication of two $2n$-bit binary numbers can be carried out recursively by computing
  1. three multiplications of n-bit binary numbers, plus
  2. a constant number of additions and shifts on n-bit binary numbers.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Fast Integer Multiplication

- We get

$$a \cdot b \sim 2^{2n} A_1 \cdot B_1 + 2^n (A_1 \cdot B_0 + A_0 \cdot B_1) + A_0 \cdot B_0,$$

  which can be rewritten as

$$a \cdot b \sim (2^{2n} + 2^n) A_1 \cdot B_1 + 2^n (A_1 - A_0) \cdot (B_0 - B_1) + (2^n + 1) A_0 \cdot B_0.$$

- Thus, the multiplication of two $2n$-bit binary numbers can be carried out recursively by computing
  1. three multiplications of n-bit binary numbers, plus
  2. a constant number of additions and shifts on n-bit binary numbers.
- Hence, if $T(n)$ denotes the total number of bit operations used by this recursive algorithm for n-bit binary numbers, then

$$T(n) = 3T\left(\frac{n}{2}\right) + f(n) \qquad \text{with } f \in \Theta(n).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Fast Integer Multiplication

- We get

$$a \cdot b \sim 2^{2n} A_1 \cdot B_1 + 2^n (A_1 \cdot B_0 + A_0 \cdot B_1) + A_0 \cdot B_0,$$

which can be rewritten as

$$a \cdot b \sim (2^{2n} + 2^n) A_1 \cdot B_1 + 2^n (A_1 - A_0) \cdot (B_0 - B_1) + (2^n + 1) A_0 \cdot B_0.$$

- Thus, the multiplication of two $2n$-bit binary numbers can be carried out recursively by computing
  1. three multiplications of n-bit binary numbers, plus
  2. a constant number of additions and shifts on n-bit binary numbers.
- Hence, if $T(n)$ denotes the total number of bit operations used by this recursive algorithm for n-bit binary numbers, then

$$T(n) = 3T\left(\frac{n}{2}\right) + f(n) \qquad \text{with } f \in \Theta(n).$$

- The asymptotic version of the Master Theorem 216 allows us to conclude that

$$T \in \Theta(n^{\log_2 3}), \qquad \text{i.e., that } T \in \Theta(n^{1.58496\ldots}) \text{ and, thus, } T \in o(n^2).$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Analysis of Fast Integer Multiplication

- We get
$$a \cdot b \sim 2^{2n} A_1 \cdot B_1 + 2^n (A_1 \cdot B_0 + A_0 \cdot B_1) + A_0 \cdot B_0,$$
which can be rewritten as
$$a \cdot b \sim (2^{2n} + 2^n) A_1 \cdot B_1 + 2^n (A_1 - A_0) \cdot (B_0 - B_1) + (2^n + 1) A_0 \cdot B_0.$$
- Thus, the multiplication of two $2n$-bit binary numbers can be carried out recursively by computing
  1. three multiplications of n-bit binary numbers, plus
  2. a constant number of additions and shifts on n-bit binary numbers.
- Hence, if $T(n)$ denotes the total number of bit operations used by this recursive algorithm for n-bit binary numbers, then
$$T(n) = 3T\left(\frac{n}{2}\right) + f(n) \qquad \text{with } f \in \Theta(n).$$
- The asymptotic version of the Master Theorem 216 allows us to conclude that
$$T \in \Theta(n^{\log_2 3}), \qquad \text{i.e., that } T \in \Theta(n^{1.58496\ldots}) \text{ and, thus, } T \in o(n^2).$$
- [Schönhage&Strassen (1971), Fürer (2007)]: Faster methods based on Fast Fourier Transform.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- We get
$$a \cdot b \sim 2^{2n} A_1 \cdot B_1 + 2^n (A_1 \cdot B_0 + A_0 \cdot B_1) + A_0 \cdot B_0,$$
which can be rewritten as
$$a \cdot b \sim (2^{2n} + 2^n) A_1 \cdot B_1 + 2^n (A_1 - A_0) \cdot (B_0 - B_1) + (2^n + 1) A_0 \cdot B_0.$$
- Thus, the multiplication of two $2n$-bit binary numbers can be carried out recursively by computing
  1. three multiplications of n-bit binary numbers, plus
  2. a constant number of additions and shifts on n-bit binary numbers.
- Hence, if $T(n)$ denotes the total number of bit operations used by this recursive algorithm for n-bit binary numbers, then
$$T(n) = 3T\left(\frac{n}{2}\right) + f(n) \qquad \text{with } f \in \Theta(n).$$
- The asymptotic version of the Master Theorem 216 allows us to conclude that
$$T \in \Theta(n^{\log_2 3}), \qquad \text{i.e., that } T \in \Theta(n^{1.58496\ldots}) \text{ and, thus, } T \in o(n^2).$$
- [Schönhage&Strassen (1971), Fürer (2007)]: Faster methods based on Fast Fourier Transform.
- [Harvey&van der Hoeven (2021)]: Achieved $O(n \log n)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

7 Graph Theory

- What is a (Directed) Graph?
- Paths
- Trees
- Special Graphs
- Graph Coloring

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

7 Graph Theory

- What is a (Directed) Graph?
  - Undirected and Directed Graph
  - Applications: Hasse Diagram and Precedence Graphs
  - Adjacency and Degree
  - Euler’s Handshaking Lemma
- Paths
- Trees
- Special Graphs
- Graph Coloring

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Basic Definitions: Undirected Graph

Definition 218 (Graph, Dt.: (schlichter endlicher ungerichteter) Graph)

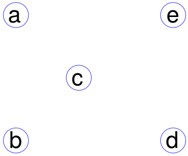
For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite undirected) graph* $\mathcal{G} := (V, E)$ with *n vertices* (aka *nodes*) and *m edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *unordered* pair of distinct vertices:

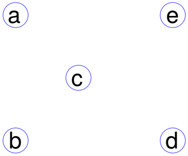
$$E \subseteq \{\{u, v\} : u, v \in V \text{ and } u \neq v\}.$$

Definition 218 (Graph, Dt.: (schlichter endlicher ungerichteter) Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite undirected) graph* $\mathcal{G} := (V, E)$ with *n vertices* (aka *nodes*) and *m edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *unordered* pair of distinct vertices:

$$E \subseteq \{\{u, v\} : \ u, v \in V \text{ and } u \neq v\}.$$

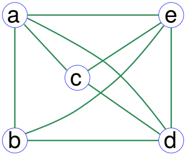


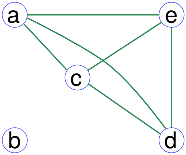


Definition 218 (Graph, Dt.: (schlichter endlicher ungerichteter) Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite undirected) graph* $\mathcal{G} := (V, E)$ with *n vertices* (aka *nodes*) and *m edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *unordered* pair of distinct vertices:

$$E \subseteq \{\{u, v\} : u, v \in V \text{ and } u \neq v\}.$$





Definition 218 (Graph, Dt.: (schlichter endlicher ungerichteter) Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite undirected) graph* $\mathcal{G} := (V, E)$ with *n vertices* (aka *nodes*) and *m edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *unordered* pair of distinct vertices:

$$E \subseteq \{\{u, v\} : u, v \in V \text{ and } u \neq v\}.$$

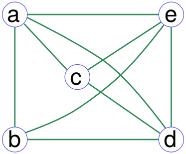
- It is common to mix the terms "*node*" (Dt.: Knoten) and "*vertex*" (Dt.: Ecke) freely.

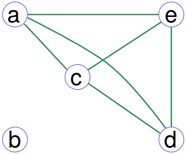
Basic Definitions: Undirected Graph

Definition 218 (Graph, Dt.: (schlichter endlicher ungerichteter) Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite undirected) graph* $\mathcal{G} := (V, E)$ with *n vertices* (aka *nodes*) and *m edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *unordered* pair of distinct vertices:

$$E \subseteq \{\{u, v\} : u, v \in V \text{ and } u \neq v\}.$$



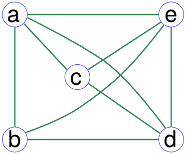


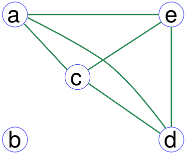
- It is common to mix the terms "*node*" (Dt.: Knoten) and "*vertex*" (Dt.: Ecke) freely.
- An edge $\{u, v\}$ is often denoted by uv.

Definition 218 (Graph, Dt.: (schlichter endlicher ungerichteter) Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite undirected) graph* $\mathcal{G} := (V, E)$ with *n vertices* (aka *nodes*) and *m edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *unordered* pair of distinct vertices:

$$E \subseteq \{\{u, v\} : u, v \in V \text{ and } u \neq v\}.$$





- It is common to mix the terms "*node*" (Dt.: Knoten) and "*vertex*" (Dt.: Ecke) freely.
- An edge $\{u, v\}$ is often denoted by uv.
- If we allow edges of the form uu then we get a *loop* (Dt.: Schlinge, Schleife) and the graph is no longer simple (Dt.: schlicht, einfach).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 218 (Graph, Dt.: (schlichter endlicher ungerichteter) Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite undirected) graph* $\mathcal{G} := (V, E)$ with *n vertices* (aka *nodes*) and *m edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *unordered* pair of distinct vertices:

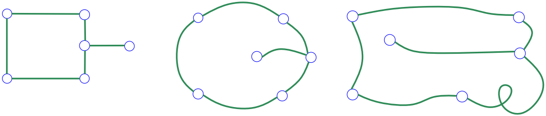
$$E \subseteq \{\{u, v\} : u, v \in V \text{ and } u \neq v\}.$$

- It is common to mix the terms "*node*" (Dt.: Knoten) and "*vertex*" (Dt.: Ecke) freely.
- An edge $\{u, v\}$ is often denoted by uv.
- If we allow edges of the form uu then we get a *loop* (Dt.: Schlinge, Schleife) and the graph is no longer simple (Dt.: schlicht, einfach).
- If we allow multiple edges between two vertices then we get a *multigraph*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

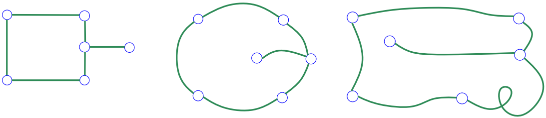
Basic Definitions: Graphical Representation

- Graphical representation of a graph:
  - Denote the vertices by markers of the same form (circles, dots, squares, . . .).
  - For every pair of vertex markers, draw a curve between them if the graph contains an edge between the corresponding vertices.



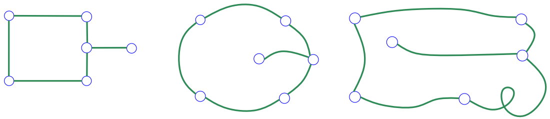
Basic Definitions: Graphical Representation

- Graphical representation of a graph:
  - Denote the vertices by markers of the same form (circles, dots, squares, . . .).
  - For every pair of vertex markers, draw a curve between them if the graph contains an edge between the corresponding vertices.
- The edges drawn may be curved and may intersect.



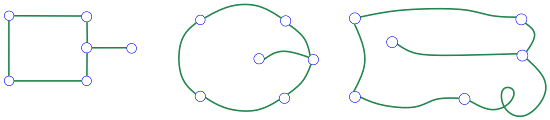
Basic Definitions: Graphical Representation

- Graphical representation of a graph:
  - Denote the vertices by markers of the same form (circles, dots, squares, . . .).
  - For every pair of vertex markers, draw a curve between them if the graph contains an edge between the corresponding vertices.
- The edges drawn may be curved and may intersect.
- However, it is poor practice to let an edge pass or touch any other vertex in addition to its two defining vertices.

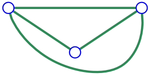


Basic Definitions: Graphical Representation

- Graphical representation of a graph:
  - Denote the vertices by markers of the same form (circles, dots, squares, . . .).
  - For every pair of vertex markers, draw a curve between them if the graph contains an edge between the corresponding vertices.
- The edges drawn may be curved and may intersect.
- However, it is poor practice to let an edge pass or touch any other vertex in addition to its two defining vertices.
- Use arrows to denote directed edges.

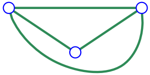


- Which of the following drawings show simple graphs?

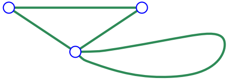


multigraph

- Which of the following drawings show simple graphs?



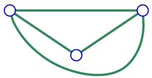
multigraph



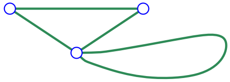
not a simple graph: loop!

Basic Definitions: Graphical Representation

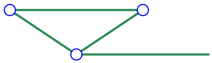
- Which of the following drawings show simple graphs?



multigraph

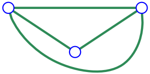


not a simple graph: loop!

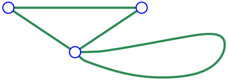


not a graph

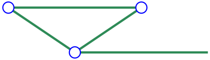
- Which of the following drawings show simple graphs?



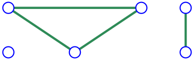
multigraph



not a simple graph: loop!



not a graph

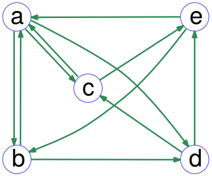


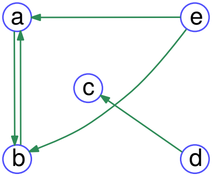
this is a graph!

Definition 219 (Directed graph, Dt.: (schlichter endlicher) gerichteter Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite) directed graph*, or *digraph*, $\mathcal{G} := (V, E)$ with n *vertices* (aka *nodes*) and m *edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *ordered* pair of distinct vertices:

$$E \subseteq \{(u, v) : u, v \in V \text{ and } u \neq v\}.$$

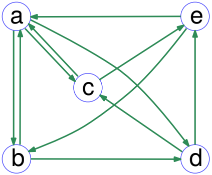


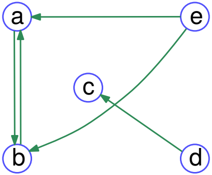


Definition 219 (Directed graph, Dt.: (schlichter endlicher) gerichteter Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite) directed graph*, or *digraph*, $\mathcal{G} := (V, E)$ with n *vertices* (aka *nodes*) and m *edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *ordered* pair of distinct vertices:

$$E \subseteq \{(u, v) : u, v \in V \text{ and } u \neq v\}.$$



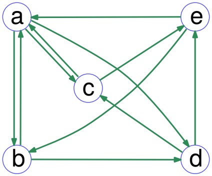


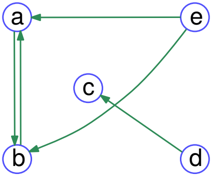
- For a digraph, uv indicates the edge (u, v), i.e., an edge where u is the *tail* and v is the *head*.

Definition 219 (Directed graph, Dt.: (schlichter endlicher) gerichteter Graph)

For $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$, a *(simple finite) directed graph*, or *digraph*, $\mathcal{G} := (V, E)$ with n *vertices* (aka *nodes*) and m *edges* consists of a vertex set $V := \{v_1, v_2, \ldots, v_n\}$ and an edge set $E := \{e_1, e_2, \ldots, e_m\}$, where $V \cap E = \emptyset$ and each edge is an *ordered* pair of distinct vertices:

$$E \subseteq \{(u, v) : u, v \in V \text{ and } u \neq v\}.$$





- For a digraph, uv indicates the edge (u, v), i.e., an edge where u is the *tail* and v is the *head*.
- In this lecture we will always specify a directed graph explicitly; that is, the term "graph" without the qualifier "directed" shall mean "undirected graph".

Basic Definitions: How to Deal with $V = \emptyset$

- There is no consensus on whether or not to allow $V = \emptyset$ in the definition of a graph. (Of course, if $V = \emptyset$ then $E = \emptyset$.)
- And, indeed, there are pros and cons of allowing $V = \emptyset$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Basic Definitions: How to Deal with $V = \emptyset$

- There is no consensus on whether or not to allow $V = \emptyset$ in the definition of a graph. (Of course, if $V = \emptyset$ then $E = \emptyset$.)
- And, indeed, there are pros and cons of allowing $V = \emptyset$.
- Furthermore, if $V = \emptyset$ is allowed then there is little consensus on how to call such a graph:
  - Common terms are *order-zero graph*, K_0, and *null graph*.
  - Some authors also use the term *empty graph* to indicate $V = \emptyset$ while other authors prefer to reserve this term for a graph with $E = \emptyset$ but $V \neq \emptyset$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Basic Definitions: How to Deal with $V = \emptyset$

- There is no consensus on whether or not to allow $V = \emptyset$ in the definition of a graph. (Of course, if $V = \emptyset$ then $E = \emptyset$.)
- And, indeed, there are pros and cons of allowing $V = \emptyset$.
- Furthermore, if $V = \emptyset$ is allowed then there is little consensus on how to call such a graph:
  - Common terms are *order-zero graph*, K_0, and *null graph*.
  - Some authors also use the term *empty graph* to indicate $V = \emptyset$ while other authors prefer to reserve this term for a graph with $E = \emptyset$ but $V \neq \emptyset$.

Convention

We will always assume that every (directed) graph has at least one node.

Basic Definitions — Warning!

No common terminology

The terminology in graph theory lacks a rigorous standardization, both in the German and in the English literature.

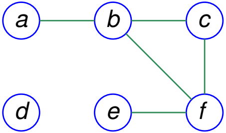
No common terminology

The terminology in graph theory lacks a rigorous standardization, both in the German and in the English literature.

- In several cases the meanings of different terms coincide for simple undirected graphs, which seems to serve as a justification for authors to freely mix and match terms.
- Thus, always make sure to check how some author defines standard terms of graph theory . . .

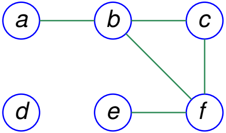
Undirected Graphs as Directed Graphs

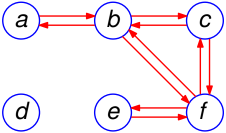
- It is straightforward to represent an undirected graph as a directed graph.



Undirected Graphs as Directed Graphs

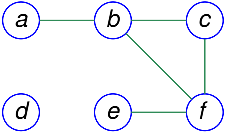
- It is straightforward to represent an undirected graph as a directed graph.

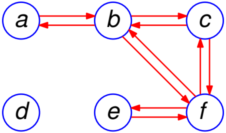




Undirected Graphs as Directed Graphs

- It is straightforward to represent an undirected graph as a directed graph.
- Hence, undirected graphs can be seen as a special case of directed graphs, and most algorithms that work for directed graphs are applicable to undirected graphs, too.



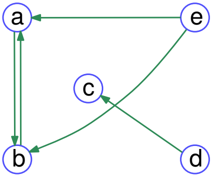


- There is an elementary mapping from relations to digraphs!

- There is an elementary mapping from relations to digraphs!
- E.g., the relation R on the set $\{a, b, c, d, e\}$, with

$$R := \{(a, b), (b, a), (d, c), (e, a), (e, b)\},$$

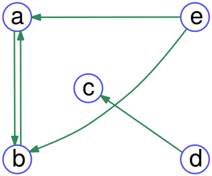
corresponds to the following directed graph:



- There is an elementary mapping from relations to digraphs!
- E.g., the relation R on the set $\{a, b, c, d, e\}$, with

$$R := \{(a, b), (b, a), (d, c), (e, a), (e, b)\},$$

corresponds to the following directed graph:



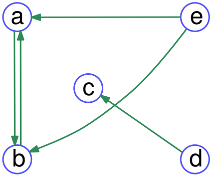
- Hence, statements about relations can be translated to statements about digraphs, and vice versa.

Directed Graphs and Relations

- There is an elementary mapping from relations to digraphs!
- E.g., the relation R on the set $\{a, b, c, d, e\}$, with

$$R := \{(a, b), (b, a), (d, c), (e, a), (e, b)\},$$

corresponds to the following directed graph:



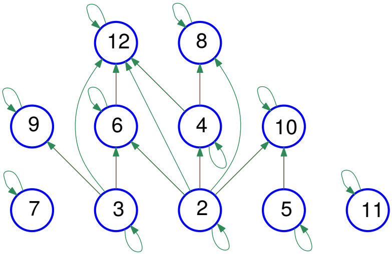
- Hence, statements about relations can be translated to statements about digraphs, and vice versa.
- Note, though, that the digraph corresponding to a relation
  - need not be simple but might contain loops,
  - need not have a finite vertex set.
- Simplified representation of the digraph of an order relation: Hasse diagram.

Directed Graphs and Relations: Hasse Diagram

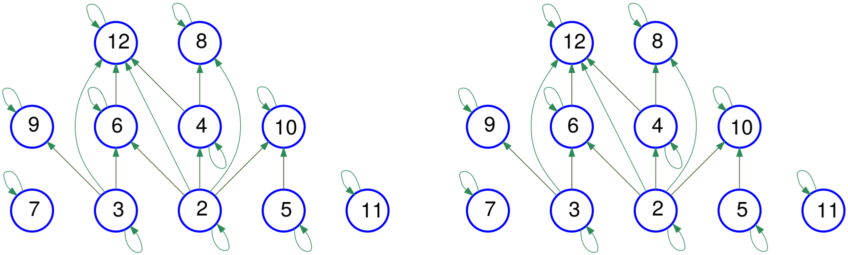
- Consider the poset (S, R), where $S := \{n \in \mathbb{N} : 1 < n \leqslant 12\}$ and R denotes the partial order of divisibility on S. (That is, for $a, b \in S$, we have $a\,R\,b$ iff $a \mid b$.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Consider the poset (S, R), where $S := \{n \in \mathbb{N} : 1 < n \leqslant 12\}$ and R denotes the partial order of divisibility on S. (That is, for $a, b \in S$, we have $a\,R\,b$ iff $a \mid b$.)



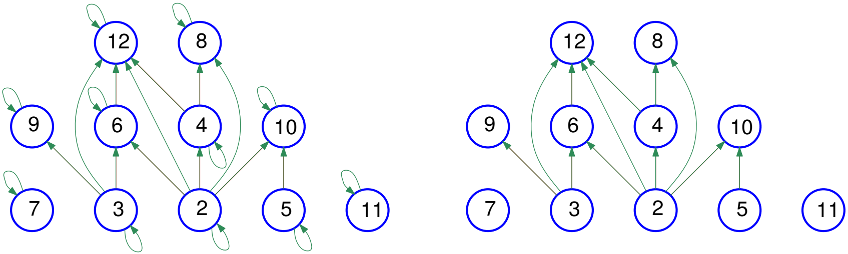
- Consider the poset (S, R), where $S := \{n \in \mathbb{N} : 1 < n \leqslant 12\}$ and R denotes the partial order of divisibility on S. (That is, for $a, b \in S$, we have $a\,R\,b$ iff $a \mid b$.)



(1) Redraw the digraph such that all oriented (non-loop) edges point upwards.

Directed Graphs and Relations: Hasse Diagram

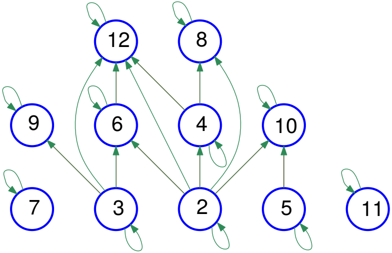
- Consider the poset (S, R), where $S := \{n \in \mathbb{N} : 1 < n \leqslant 12\}$ and R denotes the partial order of divisibility on S. (That is, for $a, b \in S$, we have $a\,R\,b$ iff $a \mid b$.)

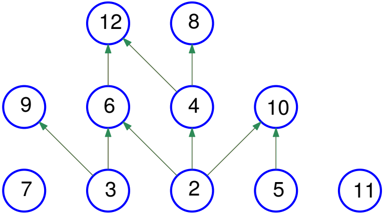


1. Redraw the digraph such that all oriented (non-loop) edges point upwards.
2. Now remove all loops (that result from the reflexivity of the partial order).

Directed Graphs and Relations: Hasse Diagram

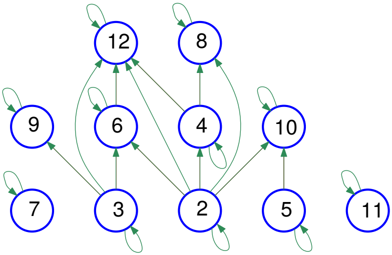
- Consider the poset (S, R), where $S := \{n \in \mathbb{N} : 1 < n \leqslant 12\}$ and R denotes the partial order of divisibility on S. (That is, for $a, b \in S$, we have $a\,R\,b$ iff $a \mid b$.)

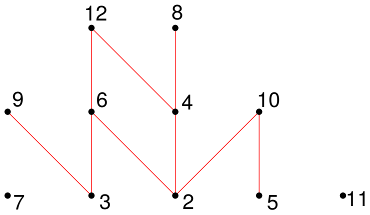




1. Redraw the digraph such that all oriented (non-loop) edges point upwards.
2. Now remove all loops (that result from the reflexivity of the partial order).
3. Next, remove all edges implied by transitivity.

- Consider the poset (S, R), where $S := \{n \in \mathbb{N} : 1 < n \leqslant 12\}$ and R denotes the partial order of divisibility on S. (That is, for $a, b \in S$, we have $a\,R\,b$ iff $a \mid b$.)

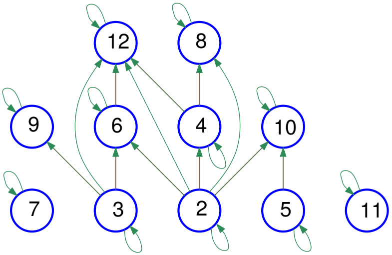


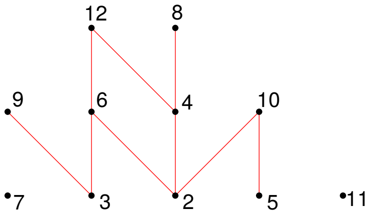


1. Redraw the digraph such that all oriented (non-loop) edges point upwards.
2. Now remove all loops (that result from the reflexivity of the partial order).
3. Next, remove all edges implied by transitivity.
4. Finally, shrink all node markers to dots.

Directed Graphs and Relations: Hasse Diagram

- Consider the poset (S, R), where $S := \{n \in \mathbb{N} : 1 < n \leqslant 12\}$ and R denotes the partial order of divisibility on S. (That is, for $a, b \in S$, we have $a\,R\,b$ iff $a \mid b$.)





Hasse diagram

1. Redraw the digraph such that all oriented (non-loop) edges point upwards.
2. Now remove all loops (that result from the reflexivity of the partial order).
3. Next, remove all edges implied by transitivity.
4. Finally, shrink all node markers to dots.

Definition 220 (Hasse diagram)

The graph obtained after carrying out Steps (1)–(4) is the *Hasse diagram* of the poset.

Real-World Application: Precedence Graph

- Typically, some statements of a computer program could be executed in parallel.

(1) $a := 1$

(2) $b := 2$

(3) $c := 3$

(4) $d := a + 2$

(5) $e := 2a + b$

(6) $f := d + c$

(7) $g := c + e$

(8) $h := d + e + f$

Real-World Application: Precedence Graph

- Typically, some statements of a computer program could be executed in parallel.
- A *precedence graph* is a directed graph that models dependences. E.g., the dependence of statements of a computer program on other statements:
  - Each statement is represented by a vertex.
  - There is an edge from vertex u to vertex v if the statement that corresponds to v has to be executed after the statement of u.

(1) $a := 1$

(2) $b := 2$

(3) $c := 3$

(4) $d := a + 2$

(5) $e := 2a + b$

(6) $f := d + c$

(7) $g := c + e$

(8) $h := d + e + f$

Real-World Application: Precedence Graph

- Typically, some statements of a computer program could be executed in parallel.
- A *precedence graph* is a directed graph that models dependences. E.g., the dependence of statements of a computer program on other statements:
  - Each statement is represented by a vertex.
  - There is an edge from vertex u to vertex v if the statement that corresponds to v has to be executed after the statement of u.

(1) $a := 1$

(2) $b := 2$

(3) $c := 3$

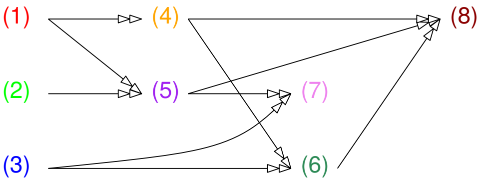
(4) $d := a + 2$

(5) $e := 2a + b$

(6) $f := d + c$

(7) $g := c + e$

(8) $h := d + e + f$



Real-World Application: Precedence Graph

- Typically, some statements of a computer program could be executed in parallel.
- A *precedence graph* is a directed graph that models dependences. E.g., the dependence of statements of a computer program on other statements:
  - Each statement is represented by a vertex.
  - There is an edge from vertex u to vertex v if the statement that corresponds to v has to be executed after the statement of u.
- Precedence graphs are used in all sorts of scheduling tasks: E.g., job scheduling, concurrency control and instruction scheduling, resolving linker dependencies, data serialization, automated parallelization of sequential code.

(1) $a := 1$

(2) $b := 2$

(3) $c := 3$

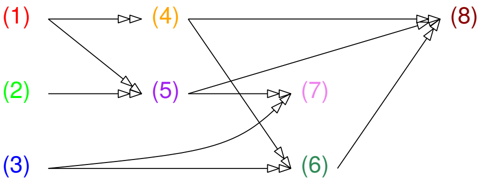
(4) $d := a + 2$

(5) $e := 2a + b$

(6) $f := d + c$

(7) $g := c + e$

(8) $h := d + e + f$



Basic Definitions: Adjacency and Degree

Definition 221 (Adjacent, Dt.: benachbart)

Two vertices $u, v \in V$ of a graph $\mathcal{G} := (V, E)$ are *adjacent* if $uv \in E$; the edge uv is *incident* to the vertices u and v.

Basic Definitions: Adjacency and Degree

Definition 221 (Adjacent, Dt.: benachbart)

Two vertices $u, v \in V$ of a graph $\mathcal{G} := (V, E)$ are *adjacent* if $uv \in E$; the edge uv is *incident* to the vertices u and v.

Definition 222 (Degree, Dt.: Grad)

The *degree* (aka *valence*) of a vertex u of a graph $\mathcal{G} := (V, E)$ is the number of edges incident to u. It is denoted by $\deg(u)$.

Basic Definitions: Adjacency and Degree

Definition 221 (Adjacent, Dt.: benachbart)

Two vertices $u, v \in V$ of a graph $\mathcal{G} := (V, E)$ are *adjacent* if $uv \in E$; the edge uv is *incident* to the vertices u and v.

Definition 222 (Degree, Dt.: Grad)

The *degree* (aka *valence*) of a vertex u of a graph $\mathcal{G} := (V, E)$ is the number of edges incident to u. It is denoted by $\deg(u)$.
For directed graphs, it is common to distinguish between the *in-degree*, $\deg^-(u)$, i.e., the number of edges vu for $v \in V$, and the *out-degree*, $\deg^+(u)$, i.e., the number of edges uv for $v \in V$.

Definition 221 (Adjacent, Dt.: benachbart)

Two vertices $u, v \in V$ of a graph $\mathcal{G} := (V, E)$ are *adjacent* if $uv \in E$; the edge uv is *incident* to the vertices u and v.

Definition 222 (Degree, Dt.: Grad)

The *degree* (aka *valence*) of a vertex u of a graph $\mathcal{G} := (V, E)$ is the number of edges incident to u. It is denoted by $\deg(u)$.
For directed graphs, it is common to distinguish between the *in-degree*, $\deg^-(u)$, i.e., the number of edges vu for $v \in V$, and the *out-degree*, $\deg^+(u)$, i.e., the number of edges uv for $v \in V$.
The *degree of a graph* is the maximum of the degrees of its vertices.

Basic Definitions: Adjacency and Degree

Definition 221 (Adjacent, Dt.: benachbart)

Two vertices $u, v \in V$ of a graph $\mathcal{G} := (V, E)$ are *adjacent* if $uv \in E$; the edge uv is *incident* to the vertices u and v.

Definition 222 (Degree, Dt.: Grad)

The *degree* (aka *valence*) of a vertex u of a graph $\mathcal{G} := (V, E)$ is the number of edges incident to u. It is denoted by $\deg(u)$.
For directed graphs, it is common to distinguish between the *in-degree*, $\deg^-(u)$, i.e., the number of edges vu for $v \in V$, and the *out-degree*, $\deg^+(u)$, i.e., the number of edges uv for $v \in V$.
The *degree of a graph* is the maximum of the degrees of its vertices.

Definition 223 (Subgraph, Dt.: Teilgraph)

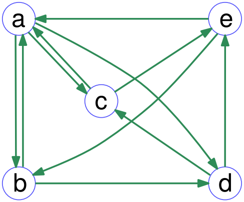
A graph $\mathcal{G}' := (V', E')$ is a *subgraph* of a (directed) graph $\mathcal{G} := (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$ such that all edges of E' are formed by vertices of V'.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 224 (Adjacency matrix, Dt.: Adjazenzmatrix)

The *adjacency matrix* of a (directed) graph $\mathcal{G} := (V, E)$ is an $n \times n$ matrix $\mathbf{M}$, where $n := |V|$ and

$$m_{ij} := \begin{cases} 1 & \text{if } v_i v_j \in E, \\ 0 & \text{otherwise.} \end{cases}$$

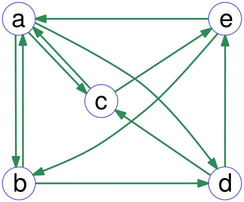


i \ j	a	b	c	d	e
a	0	1	1	1	0
b	1	0	0	1	0
c	1	0	0	0	1
d	0	0	1	0	1
e	1	1	0	0	0

Definition 224 (Adjacency matrix, Dt.: Adjazenzmatrix)

The *adjacency matrix* of a (directed) graph $\mathcal{G} := (V, E)$ is an $n \times n$ matrix **M**, where $n := |V|$ and

$$m_{ij} := \begin{cases} 1 & \text{if } v_i v_j \in E, \\ 0 & \text{otherwise.} \end{cases}$$



i \ j	a	b	c	d	e
a	0	1	1	1	0
b	1	0	0	1	0
c	1	0	0	0	1
d	0	0	1	0	1
e	1	1	0	0	0

- The adjacency matrix **M** is symmetric for undirected graphs, and all diagonal elements are zero for simple graphs.

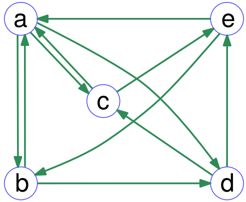
UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Basic Definitions: Adjacency Matrix

Definition 224 (Adjacency matrix, Dt.: Adjazenzmatrix)

The *adjacency matrix* of a (directed) graph $\mathcal{G} := (V, E)$ is an $n \times n$ matrix $\mathbf{M}$, where $n := |V|$ and

$$m_{ij} := \begin{cases} 1 & \text{if } v_i v_j \in E, \\ 0 & \text{otherwise.} \end{cases}$$



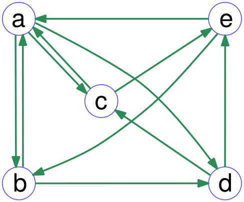
i \ j	a	b	c	d	e
a	0	1	1	1	0
b	1	0	0	1	0
c	1	0	0	0	1
d	0	0	1	0	1
e	1	1	0	0	0

- The adjacency matrix $\mathbf{M}$ is symmetric for undirected graphs, and all diagonal elements are zero for simple graphs.
- Note: Storing $\mathbf{M}$ (as an $n \times n$ array) requires $\Theta(n^2)$ memory!

Definition 224 (Adjacency matrix, Dt.: Adjazenzmatrix)

The *adjacency matrix* of a (directed) graph $\mathcal{G} := (V, E)$ is an $n \times n$ matrix $\mathbf{M}$, where $n := |V|$ and

$$m_{ij} := \begin{cases} 1 & \text{if } v_i v_j \in E, \\ 0 & \text{otherwise.} \end{cases}$$



j / i	a	b	c	d	e
a	0	1	1	1	0
b	1	0	0	1	0
c	1	0	0	0	1
d	0	0	1	0	1
e	1	1	0	0	0

- The adjacency matrix $\mathbf{M}$ is symmetric for undirected graphs, and all diagonal elements are zero for simple graphs.
- Note: Storing $\mathbf{M}$ (as an $n \times n$ array) requires $\Theta(n^2)$ memory!
- Adjacency lists (and their variants) help to preserve memory if $|E| \ll |V|^2$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

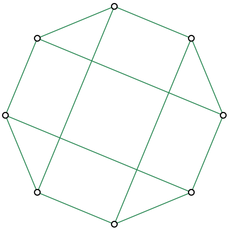
Definition 225 (Regular graph, Dt.: regulärer Graph)

A graph $\mathcal{G}$ is *regular* if every vertex of $\mathcal{G}$ has the same degree. A regular graph with vertices of degree k is called a k-regular graph or regular graph of degree k.

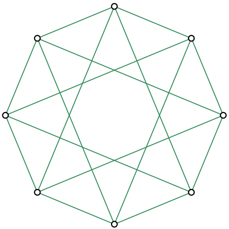
Definition 225 (Regular graph, Dt.: regulärer Graph)

A graph $\mathcal{G}$ is *regular* if every vertex of $\mathcal{G}$ has the same degree. A regular graph with vertices of degree k is called a k-regular graph or regular graph of degree k.

- A 3-regular graph is known as a cubic graph, and a 4-regular graph is known as a quartic graph.



3-regular

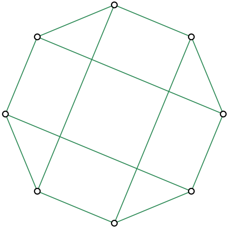


4-regular

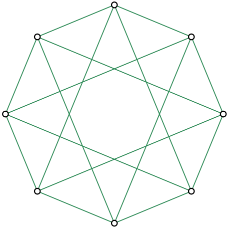
Definition 225 (Regular graph, Dt.: regulärer Graph)

A graph $\mathcal{G}$ is *regular* if every vertex of $\mathcal{G}$ has the same degree. A regular graph with vertices of degree k is called a k-regular graph or regular graph of degree k.

- A 3-regular graph is known as a cubic graph, and a 4-regular graph is known as a quartic graph.
- For directed regular graphs it is common to demand that the in-degree and the out-degree of each vertex is identical.



3-regular



4-regular

Lemma 226 (Degree sum formula)

The sum over all degrees of vertices of a graph $\mathcal{G} := (V, E)$ equals twice the number of its edges, i.e., $\sum_{\nu \in V} \deg(\nu) = 2|E|$.

Lemma 226 (Degree sum formula)

The sum over all degrees of vertices of a graph $\mathcal{G} := (V, E)$ equals twice the number of its edges, i.e., $\sum_{\nu \in V} \deg(\nu) = 2|E|$.

Sketch of proof: Adding one edge increases the sum of the degrees by two. □

Lemma 226 (Degree sum formula)

The sum over all degrees of vertices of a graph $\mathcal{G} := (V, E)$ equals twice the number of its edges, i.e., $\sum_{v \in V} \deg(v) = 2|E|$.

Sketch of proof: Adding one edge increases the sum of the degrees by two. □

Corollary 227 (Euler's Handshaking Lemma, Dt.: Handschlag-Lemma)

In every graph the number of vertices of odd degree is even.

Basic Properties of Graphs

Lemma 226 (Degree sum formula)

The sum over all degrees of vertices of a graph $\mathcal{G} := (V, E)$ equals twice the number of its edges, i.e., $\sum_{\nu \in V} \deg(\nu) = 2|E|$.

Sketch of proof : Adding one edge increases the sum of the degrees by two. □

Corollary 227 (Euler's Handshaking Lemma, Dt.: Handschlag-Lemma)

In every graph the number of vertices of odd degree is even.

- Simple application of Euler's Handshaking Lemma:
  - Suppose that a party is attended by 15 guests. Is it possible that every guest at the party knows all others except for precisely one guest?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Basic Properties of Graphs

Lemma 226 (Degree sum formula)

The sum over all degrees of vertices of a graph $\mathcal{G} := (V, E)$ equals twice the number of its edges, i.e., $\sum_{\nu \in V} \deg(\nu) = 2|E|$.

Sketch of proof : Adding one edge increases the sum of the degrees by two. □

Corollary 227 (Euler's Handshaking Lemma, Dt.: Handschlag-Lemma)

In every graph the number of vertices of odd degree is even.

- Simple application of Euler's Handshaking Lemma:
  - Suppose that a party is attended by 15 guests. Is it possible that every guest at the party knows all others except for precisely one guest?
  - No: Consider a graph with 15 nodes (guests) where two nodes are linked by an edge if the corresponding guests do not know each other. Hence, we would get 15 nodes of degree one, in contradiction to Cor. 227.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

7 Graph Theory

Definition 228 (Walk, Dt.: Wanderung, Kantenfolge)

A *walk* of length k, with $k \in \mathbb{N}_0$, on $\mathcal{G} := (V, E)$ is an alternating sequence

$$v_0 e_1 v_1 e_2 v_2 \ldots e_k v_k$$

of $k + 1$ vertices $v_0, v_1, \ldots, v_k \in V$ and k edges $e_1, \ldots, e_k \in E$ such that

$$\forall (1 \leqslant i \leqslant k) \;\; e_i = v_{i-1} v_i.$$

Walks

Definition 228 (Walk, Dt.: Wanderung, Kantenfolge)

A *walk* of length k, with $k \in \mathbb{N}_0$, on $\mathcal{G} := (V, E)$ is an alternating sequence

$$v_0 e_1 v_1 e_2 v_2 \ldots e_k v_k$$

of $k + 1$ vertices $v_0, v_1, \ldots, v_k \in V$ and k edges $e_1, \ldots, e_k \in E$ such that

$$\forall (1 \leqslant i \leqslant k) \;\; e_i = v_{i-1} v_i.$$

- Often, a walk of length k is written simply as

$$v_0 v_1 v_2 \ldots v_k.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 228 (Walk, Dt.: Wanderung, Kantenfolge)

A *walk* of length k, with $k \in \mathbb{N}_0$, on $\mathcal{G} := (V, E)$ is an alternating sequence

$$v_0 e_1 v_1 e_2 v_2 \dots e_k v_k$$

of $k + 1$ vertices $v_0, v_1, \dots, v_k \in V$ and k edges $e_1, \dots, e_k \in E$ such that

$$\forall (1 \leqslant i \leqslant k) \;\; e_i = v_{i-1} v_i.$$

- Often, a walk of length k is written simply as

$$v_0 v_1 v_2 \dots v_k.$$

- Conventionally, v_0 is called the *start vertex* (or *initial vertex*) of the walk, and v_k is called its *end vertex* (or *terminal vertex*). Note that $v_{i-1} \neq v_i$ for $i \in \{1, 2, \dots, k\}$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Walks

Definition 228 (Walk, Dt.: Wanderung, Kantenfolge)

A *walk* of length k, with $k \in \mathbb{N}_0$, on $\mathcal{G} := (V, E)$ is an alternating sequence

$$v_0 e_1 v_1 e_2 v_2 \ldots e_k v_k$$

of $k + 1$ vertices $v_0, v_1, \ldots, v_k \in V$ and k edges $e_1, \ldots, e_k \in E$ such that

$$\forall (1 \leqslant i \leqslant k) \;\; e_i = v_{i-1} v_i.$$

- Often, a walk of length k is written simply as

$$v_0 v_1 v_2 \ldots v_k.$$

- Conventionally, v_0 is called the *start vertex* (or *initial vertex*) of the walk, and v_k is called its *end vertex* (or *terminal vertex*). Note that $v_{i-1} \neq v_i$ for $i \in \{1, 2, \ldots, k\}$.

Definition 229 (Closed walk, Dt.: geschlossene Wanderung)

A walk is called *closed* if the start vertex and the end vertex are identical. A closed walk of length k is called *trivial* if $k \leqslant 2$.

Definition 230 (Trail, Dt.: Weg)

A *trail* in a (directed) graph $\mathcal{G}$ is a walk in which all edges are distinct.

Paths, Trails, Tours and Cycles

Definition 230 (Trail, Dt.: Weg)

A *trail* in a (directed) graph $\mathcal{G}$ is a walk in which all edges are distinct.

Definition 231 (Path, Dt.: Pfad)

A *path* in a (directed) graph $\mathcal{G}$ is a walk in which all vertices are distinct.

Paths, Trails, Tours and Cycles

Definition 230 (Trail, Dt.: Weg)

A *trail* in a (directed) graph $\mathcal{G}$ is a walk in which all edges are distinct.

Definition 231 (Path, Dt.: Pfad)

A *path* in a (directed) graph $\mathcal{G}$ is a walk in which all vertices are distinct.

Definition 232 (Tour, Dt.: Tour)

A *tour* in a (directed) graph $\mathcal{G}$ is a closed trail.

Paths, Trails, Tours and Cycles

Definition 230 (Trail, Dt.: Weg)

A *trail* in a (directed) graph $\mathcal{G}$ is a walk in which all edges are distinct.

Definition 231 (Path, Dt.: Pfad)

A *path* in a (directed) graph $\mathcal{G}$ is a walk in which all vertices are distinct.

Definition 232 (Tour, Dt.: Tour)

A *tour* in a (directed) graph $\mathcal{G}$ is a closed trail.

Definition 233 (Cycle, Dt.: Zyklus, Kreis)

A *cycle* in a (directed) graph $\mathcal{G}$ is a non-trivial closed walk in which all but the start and the end vertices are distinct.

Paths, Trails, Tours and Cycles

Definition 230 (Trail, Dt.: Weg)

A *trail* in a (directed) graph $\mathcal{G}$ is a walk in which all edges are distinct.

Definition 231 (Path, Dt.: Pfad)

A *path* in a (directed) graph $\mathcal{G}$ is a walk in which all vertices are distinct.

Definition 232 (Tour, Dt.: Tour)

A *tour* in a (directed) graph $\mathcal{G}$ is a closed trail.

Definition 233 (Cycle, Dt.: Zyklus, Kreis)

A *cycle* in a (directed) graph $\mathcal{G}$ is a non-trivial closed walk in which all but the start and the end vertices are distinct.

- Note: Distinct vertices implies distinct edges; i.e., every path is a trail and every cycle is a tour.

Paths, Trails, Tours and Cycles

Definition 230 (Trail, Dt.: Weg)

A *trail* in a (directed) graph $\mathcal{G}$ is a walk in which all edges are distinct.

Definition 231 (Path, Dt.: Pfad)

A *path* in a (directed) graph $\mathcal{G}$ is a walk in which all vertices are distinct.

Definition 232 (Tour, Dt.: Tour)

A *tour* in a (directed) graph $\mathcal{G}$ is a closed trail.

Definition 233 (Cycle, Dt.: Zyklus, Kreis)

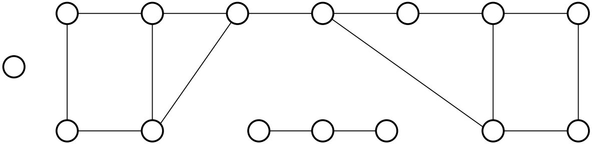
A *cycle* in a (directed) graph $\mathcal{G}$ is a non-trivial closed walk in which all but the start and the end vertices are distinct.

- Note: Distinct vertices implies distinct edges; i.e., every path is a trail and every cycle is a tour.
- Note that some authors prefer to use the terms “path”, “simple path”, “cycle” and “simple cycle” instead of “trail”, “path”, “tour” and “cycle” . . .

Connectedness

Definition 234 (Connected component, Dt.: Zusammenhangskomponente)

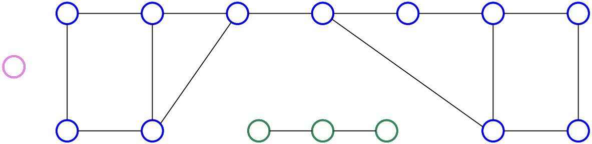
A *connected component* of a graph $\mathcal{G} := (V, E)$ is a maximal subgraph $\mathcal{G}' := (V', E')$ of $\mathcal{G}$ such that for every unordered pair $\{u, v\}$, with $u, v \in V'$ and $u \neq v$, there exists a path between u and v within $\mathcal{G}'$.



Connectedness

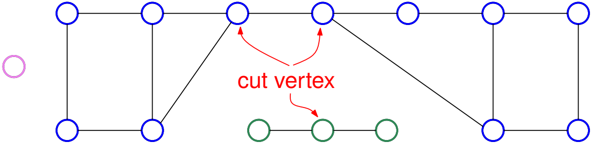
Definition 234 (Connected component, Dt.: Zusammenhangskomponente)

A *connected component* of a graph $\mathcal{G} := (V, E)$ is a maximal subgraph $\mathcal{G}' := (V', E')$ of $\mathcal{G}$ such that for every unordered pair $\{u, v\}$, with $u, v \in V'$ and $u \neq v$, there exists a path between u and v within $\mathcal{G}'$.



Definition 234 (Connected component, Dt.: Zusammenhangskomponente)

A *connected component* of a graph $\mathcal{G} := (V, E)$ is a maximal subgraph $\mathcal{G}' := (V', E')$ of $\mathcal{G}$ such that for every unordered pair $\{u, v\}$, with $u, v \in V'$ and $u \neq v$, there exists a path between u and v within $\mathcal{G}'$.

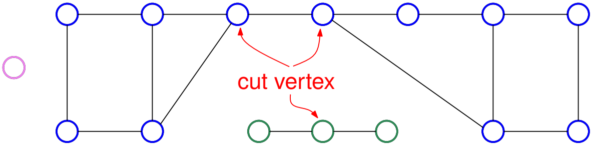


Definition 235 (Cut vertex, Dt.: Artikulationspunkt, Schnittknoten)

A *cut vertex* of a graph $\mathcal{G} := (V, E)$ is a vertex $v \in V$ such that the removal of v and of all edges incident to v would increase the number of connected components.

Definition 234 (Connected component, Dt.: Zusammenhangskomponente)

A *connected component* of a graph $\mathcal{G} := (V, E)$ is a maximal subgraph $\mathcal{G}' := (V', E')$ of $\mathcal{G}$ such that for every unordered pair $\{u, v\}$, with $u, v \in V'$ and $u \neq v$, there exists a path between u and v within $\mathcal{G}'$.



Definition 235 (Cut vertex, Dt.: Artikulationspunkt, Schnittknoten)

A *cut vertex* of a graph $\mathcal{G} := (V, E)$ is a vertex $v \in V$ such that the removal of v and of all edges incident to v would increase the number of connected components.

Definition 236 (Connected, Dt.: zusammenhängend)

A graph is *connected* if it contains only one connected component.

Definition 237 (Weakly connected, Dt.: schwach zusammenhängend)

A directed graph is *weakly connected* if replacing all its directed edges by undirected edges results in a connected (undirected) graph.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 237 (Weakly connected, Dt.: schwach zusammenhängend)

A directed graph is *weakly connected* if replacing all its directed edges by undirected edges results in a connected (undirected) graph.

Definition 238 (Strong component, Dt.: starke Zusammenhangskomponente)

A *strong component* (aka *strongly connected component*) of a directed graph $\mathcal{G} := (V, E)$ is a maximal subgraph $\mathcal{G}' = (V', E')$ of $\mathcal{G}$ such that for every ordered pair (u, v), with $u, v \in V'$ and $u \neq v$, there exists a path from u to v within $\mathcal{G}'$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 237 (Weakly connected, Dt.: schwach zusammenhängend)

A directed graph is *weakly connected* if replacing all its directed edges by undirected edges results in a connected (undirected) graph.

Definition 238 (Strong component, Dt.: starke Zusammenhangskomponente)

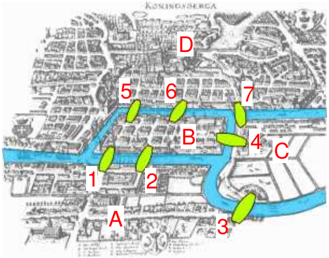
A *strong component* (aka *strongly connected component*) of a directed graph $\mathcal{G} := (V, E)$ is a maximal subgraph $\mathcal{G}' = (V', E')$ of $\mathcal{G}$ such that for every ordered pair (u, v), with $u, v \in V'$ and $u \neq v$, there exists a path from u to v within $\mathcal{G}'$.

Definition 239 (Strongly connected, Dt.: stark zusammenhängend)

A directed graph $\mathcal{G} := (V, E)$ is *strongly connected* if it consists of only one strong component, i.e., if for every ordered pair (u, v), with $u, v \in V$ and $u \neq v$, there exists a path from u to v.

Seven Bridges of Königsberg

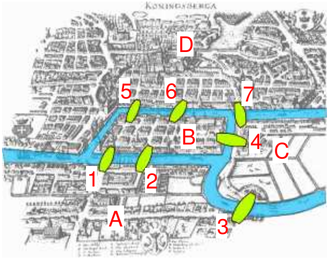
- Early 18th century: Does there exist a trail (or even a tour) through the city of Königsberg that crosses every of its seven bridges exactly once? (Of course, every bridge had to be crossed fully, and no other means to get across the river Pregel were allowed.)

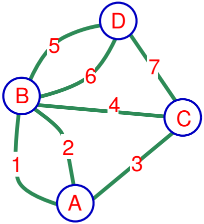


[Image credit for background image: Wikipedia.]

Seven Bridges of Königsberg

- Early 18th century: Does there exist a trail (or even a tour) through the city of Königsberg that crosses every of its seven bridges exactly once? (Of course, every bridge had to be crossed fully, and no other means to get across the river Pregel were allowed.)



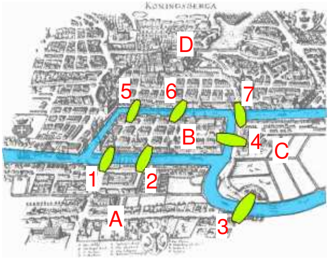


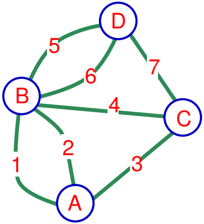
[Image credit for background image: Wikipedia.]

- In 1736, Leonhard Euler (1707–1783) treated this problem as a graph problem and proved, using a parity argument, that such a trail or tour does not exist.

Seven Bridges of Königsberg

- Early 18th century: Does there exist a trail (or even a tour) through the city of Königsberg that crosses every of its seven bridges exactly once? (Of course, every bridge had to be crossed fully, and no other means to get across the river Pregel were allowed.)





[Image credit for background image: Wikipedia.]

- In 1736, Leonhard Euler (1707–1783) treated this problem as a graph problem and proved, using a parity argument, that such a trail or tour does not exist.
- His solution is generally regarded as the first theorem of graph theory.

Definition 240 (Euler trail, Dt.: Eulerscher Weg)

An *Euler trail* is a trail that contains all edges of a graph exactly once.

Definition 240 (Euler trail, Dt.: Eulerscher Weg)

An *Euler trail* is a trail that contains all edges of a graph exactly once.

Definition 241 (Euler tour, Dt.: Eulersche Tour)

An *Euler tour* is a tour that contains all edges of a graph exactly once. A graph is an *Eulerian graph* if it has an Euler tour.

Euler Tour and Hamilton Cycle

Definition 240 (Euler trail, Dt.: Eulerscher Weg)

An *Euler trail* is a trail that contains all edges of a graph exactly once.

Definition 241 (Euler tour, Dt.: Eulersche Tour)

An *Euler tour* is a tour that contains all edges of a graph exactly once. A graph is an *Eulerian graph* if it has an Euler tour.

Definition 242 (Hamilton path, Dt.: Hamiltonscher Pfad)

A *Hamilton path* is a path that passes through all vertices of a graph exactly once.

Definition 240 (Euler trail, Dt.: Eulerscher Weg)

An *Euler trail* is a trail that contains all edges of a graph exactly once.

Definition 241 (Euler tour, Dt.: Eulersche Tour)

An *Euler tour* is a tour that contains all edges of a graph exactly once. A graph is an *Eulerian graph* if it has an Euler tour.

Definition 242 (Hamilton path, Dt.: Hamiltonscher Pfad)

A *Hamilton path* is a path that passes through all vertices of a graph exactly once.

Definition 243 (Hamilton cycle, Dt.: Hamiltonscher Kreis)

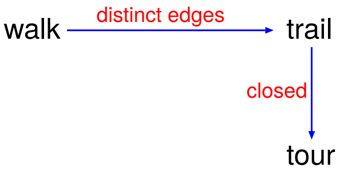
A *Hamilton cycle* is a cycle that passes through all vertices of a graph exactly once.

walk

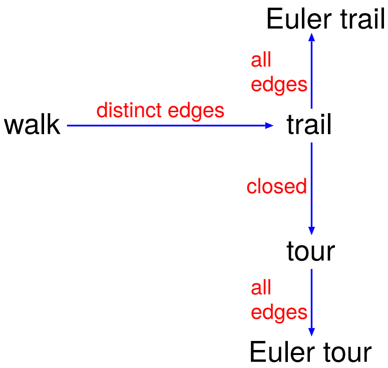
Euler Tour and Hamilton Cycle

walk —(distinct edges)→ trail

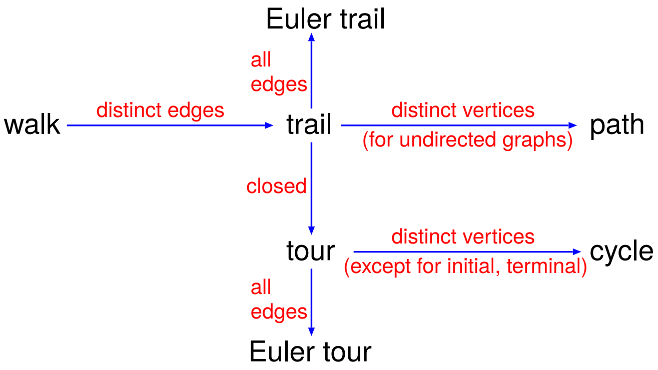
Euler Tour and Hamilton Cycle

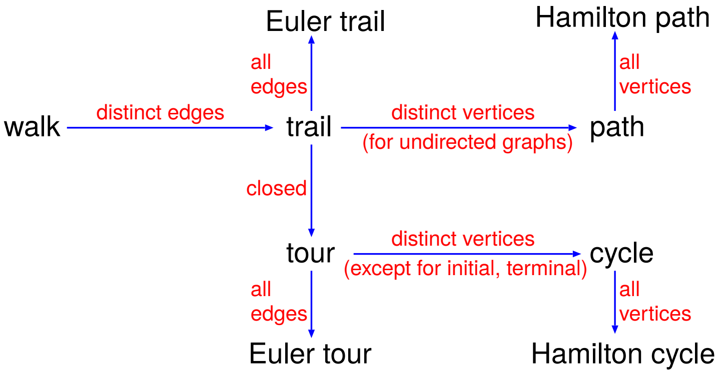


Euler Tour and Hamilton Cycle



Euler Tour and Hamilton Cycle




Euler trail
Hamilton path
all
edges
all
vertices
walk
distinct edges
trail
distinct vertices
(for undirected graphs)
path
closed
tour
distinct vertices
(except for initial, terminal)
cycle
all
edges
all
vertices
Euler tour
Hamilton cycle

Euler Tour

Theorem 244

Suppose that every node of a graph $\mathcal{G}$ has degree at least one. Then $\mathcal{G}$ has an Euler tour if and only if $\mathcal{G}$ is connected and every vertex of $\mathcal{G}$ has even degree.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Euler Tour

Theorem 244

Suppose that every node of a graph $\mathcal{G}$ has degree at least one. Then $\mathcal{G}$ has an Euler tour if and only if $\mathcal{G}$ is connected and every vertex of $\mathcal{G}$ has even degree.

Theorem 245

Suppose that every node of a graph $\mathcal{G}$ has degree at least one. Then $\mathcal{G}$ has an Euler trail (but no Euler tour) if and only if $\mathcal{G}$ is connected and exactly two vertices of $\mathcal{G}$ have odd degrees.

Euler Tour

Theorem 244

Suppose that every node of a graph $\mathcal{G}$ has degree at least one. Then $\mathcal{G}$ has an Euler tour if and only if $\mathcal{G}$ is connected and every vertex of $\mathcal{G}$ has even degree.

Theorem 245

Suppose that every node of a graph $\mathcal{G}$ has degree at least one. Then $\mathcal{G}$ has an Euler trail (but no Euler tour) if and only if $\mathcal{G}$ is connected and exactly two vertices of $\mathcal{G}$ have odd degrees.

Corollary 246

An Euler tour or trail in a graph $\mathcal{G} := (V, E)$ can be determined in $O(|E|)$ time, if it exists. Otherwise, again in $O(|E|)$ time, we can determine that neither an Euler tour nor an Euler trail exists in $\mathcal{G}$.

Constructive Proof of Theorem 244

Sketch of proof of Theorem 244 : Let $\mathcal{G} := (V, E)$ be a graph such that every node of a graph $\mathcal{G}$ has degree at least one.

Constructive Proof of Theorem 244

Sketch of proof of Theorem 244 : Let $\mathcal{G} := (V, E)$ be a graph such that every node of a graph $\mathcal{G}$ has degree at least one.

Suppose that $\mathcal{G}$ has an Euler tour T. It is obvious that $\mathcal{G}$ is connected.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Constructive Proof of Theorem 244

Sketch of proof of Theorem 244 : Let $\mathcal{G} := (V, E)$ be a graph such that every node of a graph $\mathcal{G}$ has degree at least one.

Suppose that $\mathcal{G}$ has an Euler tour T. It is obvious that $\mathcal{G}$ is connected. Every occurrence of a vertex $v \in V$ in T is preceded and followed by an edge. Thus, each time T passes through v, two of the edges incident to v are consumed. Since T does neither start nor end in v, it is necessary that $\deg(v)$ is even.

Constructive Proof of Theorem 244

Sketch of proof of Theorem 244 : Let $\mathcal{G} := (V, E)$ be a graph such that every node of a graph $\mathcal{G}$ has degree at least one.

Suppose that $\mathcal{G}$ has an Euler tour T. It is obvious that $\mathcal{G}$ is connected. Every occurrence of a vertex $v \in V$ in T is preceded and followed by an edge. Thus, each time T passes through v, two of the edges incident to v are consumed. Since T does neither start nor end in v, it is necessary that $\deg(v)$ is even.

Now suppose that every vertex of $\mathcal{G}$ has even degree, and, of course, that $\mathcal{G}$ is connected. We give a constructive proof that $\mathcal{G}$ admits an Euler tour.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Constructive Proof of Theorem 244

Sketch of proof of Theorem 244 : Let $\mathcal{G} := (V, E)$ be a graph such that every node of a graph $\mathcal{G}$ has degree at least one.

Suppose that $\mathcal{G}$ has an Euler tour T. It is obvious that $\mathcal{G}$ is connected. Every occurrence of a vertex $v \in V$ in T is preceded and followed by an edge. Thus, each time T passes through v, two of the edges incident to v are consumed. Since T does neither start nor end in v, it is necessary that $\deg(v)$ is even.

Now suppose that every vertex of $\mathcal{G}$ has even degree, and, of course, that $\mathcal{G}$ is connected. We give a constructive proof that $\mathcal{G}$ admits an Euler tour. Pick any vertex v to start with and trace out a trail T. Every edge that is being traversed is marked. As above, we observe that passing through a vertex that is neither the start nor the end vertex of T consumes two edges.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Sketch of proof of Theorem 244 : Let $\mathcal{G} := (V, E)$ be a graph such that every node of a graph $\mathcal{G}$ has degree at least one.

Suppose that $\mathcal{G}$ has an Euler tour T. It is obvious that $\mathcal{G}$ is connected. Every occurrence of a vertex $v \in V$ in T is preceded and followed by an edge. Thus, each time T passes through v, two of the edges incident to v are consumed. Since T does neither start nor end in v, it is necessary that $\deg(v)$ is even.

Now suppose that every vertex of $\mathcal{G}$ has even degree, and, of course, that $\mathcal{G}$ is connected. We give a constructive proof that $\mathcal{G}$ admits an Euler tour. Pick any vertex v to start with and trace out a trail T. Every edge that is being traversed is marked. As above, we observe that passing through a vertex that is neither the start nor the end vertex of T consumes two edges.
We realize that, eventually, T will get us back to v. (We cannot be stuck in some other vertex w since w has even degree.)

Constructive Proof of Theorem 244

Sketch of proof of Theorem 244: Let $\mathcal{G} := (V, E)$ be a graph such that every node of a graph $\mathcal{G}$ has degree at least one.

Suppose that $\mathcal{G}$ has an Euler tour T. It is obvious that $\mathcal{G}$ is connected. Every occurrence of a vertex $v \in V$ in T is preceded and followed by an edge. Thus, each time T passes through v, two of the edges incident to v are consumed. Since T does neither start nor end in v, it is necessary that $\deg(v)$ is even.

Now suppose that every vertex of $\mathcal{G}$ has even degree, and, of course, that $\mathcal{G}$ is connected. We give a constructive proof that $\mathcal{G}$ admits an Euler tour. Pick any vertex v to start with and trace out a trail T. Every edge that is being traversed is marked. As above, we observe that passing through a vertex that is neither the start nor the end vertex of T consumes two edges.
We realize that, eventually, T will get us back to v. (We cannot be stuck in some other vertex w since w has even degree.) If at the time when we are back at v every vertex of T has no unmarked incident edge then we are done. Otherwise, we start a new trail T' at a vertex w of T which has an unmarked incident edge and follow it until we get back to w.

Constructive Proof of Theorem 244

Sketch of proof of Theorem 244 : Let $\mathcal{G} := (V, E)$ be a graph such that every node of a graph $\mathcal{G}$ has degree at least one.

Suppose that $\mathcal{G}$ has an Euler tour T. It is obvious that $\mathcal{G}$ is connected. Every occurrence of a vertex $v \in V$ in T is preceded and followed by an edge. Thus, each time T passes through v, two of the edges incident to v are consumed. Since T does neither start nor end in v, it is necessary that $\deg(v)$ is even.

Now suppose that every vertex of $\mathcal{G}$ has even degree, and, of course, that $\mathcal{G}$ is connected. We give a constructive proof that $\mathcal{G}$ admits an Euler tour. Pick any vertex v to start with and trace out a trail T. Every edge that is being traversed is marked. As above, we observe that passing through a vertex that is neither the start nor the end vertex of T consumes two edges.
We realize that, eventually, T will get us back to v. (We cannot be stuck in some other vertex w since w has even degree.) If at the time when we are back at v every vertex of T has no unmarked incident edge then we are done. Otherwise, we start a new trail T' at a vertex w of T which has an unmarked incident edge and follow it until we get back to w.
This process continues until no unmarked edges remain. At the end the trails are spliced together appropriately.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Hamilton Cycle

Theorem 247

It is $\mathcal{NP}$-complete to determine whether a Hamilton cycle or Hamilton path exists in a general graph.

- Informally, Theorem 247 says that no (deterministic sequential) algorithm is known which determines the existence of a Hamilton cycle or path in an n-vertex graph in a time that is a polynomial function of n.
- Even worse, an efficient (polynomial-time) algorithm will never be found unless $\mathcal{P} = \mathcal{NP}$ holds, which seems rather unlikely.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Hamilton Cycle

Theorem 247

It is $\mathcal{NP}$-complete to determine whether a Hamilton cycle or Hamilton path exists in a general graph.

- Informally, Theorem 247 says that no (deterministic sequential) algorithm is known which determines the existence of a Hamilton cycle or path in an n-vertex graph in a time that is a polynomial function of n.
- Even worse, an efficient (polynomial-time) algorithm will never be found unless $\mathcal{P} = \mathcal{NP}$ holds, which seems rather unlikely.

Theorem 248 (Dirac, 1952)

If the degree of every vertex of an n-vertex graph $\mathcal{G}$, with $n \geqslant 3$, is at least $\lceil \frac{n}{2} \rceil$ then $\mathcal{G}$ has a Hamilton cycle.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 247

It is $\mathcal{NP}$-complete to determine whether a Hamilton cycle or Hamilton path exists in a general graph.

- Informally, Theorem 247 says that no (deterministic sequential) algorithm is known which determines the existence of a Hamilton cycle or path in an n-vertex graph in a time that is a polynomial function of n.
- Even worse, an efficient (polynomial-time) algorithm will never be found unless $\mathcal{P} = \mathcal{NP}$ holds, which seems rather unlikely.

Theorem 248 (Dirac, 1952)

If the degree of every vertex of an n-vertex graph $\mathcal{G}$, with $n \geqslant 3$, is at least $\lceil \frac{n}{2} \rceil$ then $\mathcal{G}$ has a Hamilton cycle.

Theorem 249 (Ore, 1960)

If the sum of the degrees of every pair of non-adjacent vertices of an n-vertex graph $\mathcal{G}$, with $n \geqslant 3$, is at least n then $\mathcal{G}$ has a Hamilton cycle.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

7 Graph Theory

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 250 (Acyclic, Dt.: zyklenfrei)

A graph is called *acyclic* if it contains no cycles.

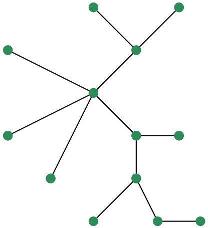
Trees

Definition 250 (Acyclic, Dt.: zyklenfrei)

A graph is called *acyclic* if it contains no cycles.

Definition 251 (Tree, Dt.: Baum)

A *tree* is an undirected graph that is acyclic and connected.



UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

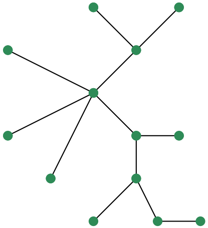
Definition 250 (Acyclic, Dt.: zyklenfrei)

A graph is called *acyclic* if it contains no cycles.

Definition 251 (Tree, Dt.: Baum)

A *tree* is an undirected graph that is acyclic and connected.

- For trees most authors prefer to speak about *nodes* rather than vertices.



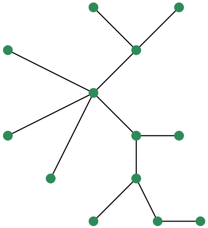
Definition 250 (Acyclic, Dt.: zyklenfrei)

A graph is called *acyclic* if it contains no cycles.

Definition 251 (Tree, Dt.: Baum)

A *tree* is an undirected graph that is acyclic and connected.

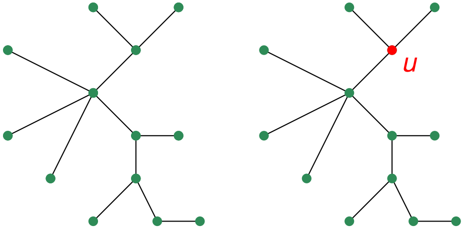
- For trees most authors prefer to speak about *nodes* rather than vertices.
- Unless explicitly stated otherwise, we will only deal with trees that have at least one node. (Some authors call a tree with $V = E = \emptyset$ a *null tree*.)



Definition 252 (Rooted tree, Dt.: Baum mit Wurzel, Wurzelbaum)

A *rooted tree* is a directed graph with a node u such that

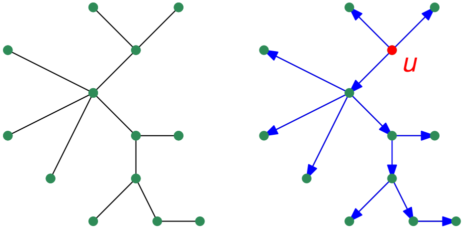
1. the graph contains u as node ("*root*"),



Definition 252 (Rooted tree, Dt.: Baum mit Wurzel, Wurzelbaum)

A *rooted tree* is a directed graph with a node *u* such that

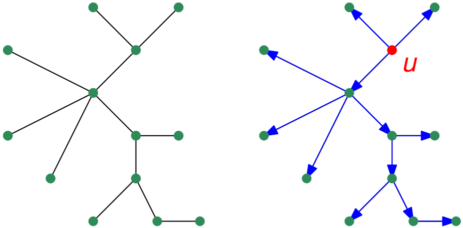
1. the graph contains *u* as node ("*root*"),
2. paths from *u* to all other nodes of the graph exist,



Definition 252 (Rooted tree, Dt.: Baum mit Wurzel, Wurzelbaum)

A *rooted tree* is a directed graph with a node u such that

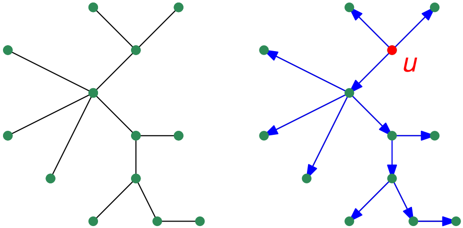
1. the graph contains u as node ("*root*"),
2. paths from u to all other nodes of the graph exist,
3. the in-degree of u is zero,



Definition 252 (Rooted tree, Dt.: Baum mit Wurzel, Wurzelbaum)

A *rooted tree* is a directed graph with a node u such that

1. the graph contains u as node ("*root*"),
2. paths from u to all other nodes of the graph exist,
3. the in-degree of u is zero,
4. the in-degree of every other node of the graph is one.

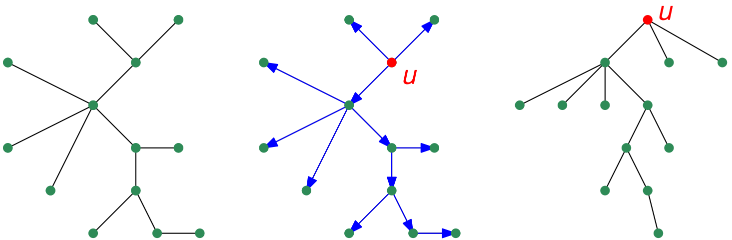


Definition 252 (Rooted tree, Dt.: Baum mit Wurzel, Wurzelbaum)

A *rooted tree* is a directed graph with a node u such that

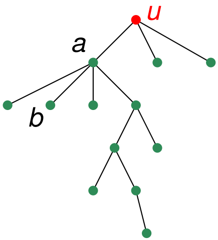
1. the graph contains u as node ("*root*"),
2. paths from u to all other nodes of the graph exist,
3. the in-degree of u is zero,
4. the in-degree of every other node of the graph is one.

- It is common practice to draw rooted trees from the root downwards such that the (downwards) orientations of the edges are implied by the positions of the nodes.



Definition 253 (Child and parent, Dt.: Kind und Eltern)

For a rooted tree $\mathcal{T} := (V, E)$ and nodes $a, b \in V$, the node b is a *child* of the node a, and a is the *parent* of b, if the edge ab belongs to E. *Siblings* are nodes which share the same parent.

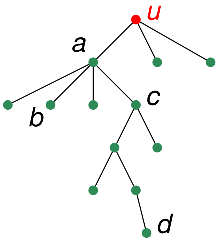


Definition 253 (Child and parent, Dt.: Kind und Eltern)

For a rooted tree $\mathcal{T} := (V, E)$ and nodes $a, b \in V$, the node b is a *child* of the node a, and a is the *parent* of b, if the edge ab belongs to E. *Siblings* are nodes which share the same parent.

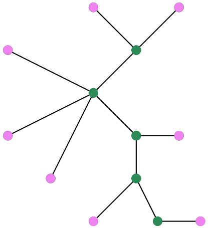
Definition 254 (Descendant and ancestor, Dt.: Nachfahre und Vorfahre)

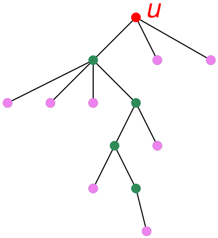
In a rooted tree $\mathcal{T} := (V, E)$, with $c, d \in V$, a node d is a *descendant* of a node c, and c is an *ancestor* of d, if $c \neq d$ and if the path from the root to d contains c.



Definition 255 (Leaf, Dt.: Blatt)

A *leaf* of a rooted tree is a node without children. For a tree (that is not rooted) a leaf is a node with degree 1. All non-leaf nodes of a (rooted) tree are called *inner nodes*.

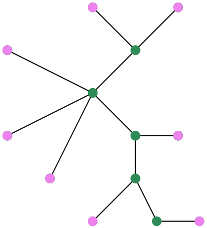


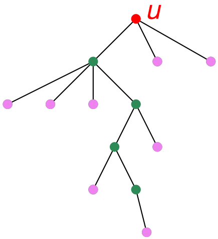


Definition 255 (Leaf, Dt.: Blatt)

A *leaf* of a rooted tree is a node without children. For a tree (that is not rooted) a leaf is a node with degree 1. All non-leaf nodes of a (rooted) tree are called *inner nodes*.

- Of course, the root of a rooted tree $\mathcal{T}$ may also be the (only) leaf of $\mathcal{T}$.

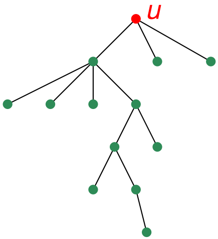




Definition 256 (Subtree, Dt.: Teilbaum)

A tree $\mathcal{T}' := (V', E')$ is a *subtree* of a tree $\mathcal{T} := (V, E)$ rooted at the node u if

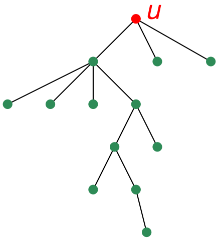
1. $\mathcal{T}'$ is a subgraph of $\mathcal{T}$,



Definition 256 (Subtree, Dt.: Teilbaum)

A tree $\mathcal{T}' := (V', E')$ is a *subtree* of a tree $\mathcal{T} := (V, E)$ rooted at the node u if

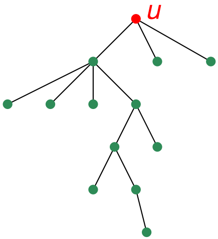
1. $\mathcal{T}'$ is a subgraph of $\mathcal{T}$,
2. $\mathcal{T}'$ is rooted at a node v that is a descendant of u,



Definition 256 (Subtree, Dt.: Teilbaum)

A tree $\mathcal{T}' := (V', E')$ is a *subtree* of a tree $\mathcal{T} := (V, E)$ rooted at the node u if

1. $\mathcal{T}'$ is a subgraph of $\mathcal{T}$,
2. $\mathcal{T}'$ is rooted at a node v that is a descendant of u, and
3. $\mathcal{T}'$ contains all descendants of v in $\mathcal{T}$, together with the appropriate edges of E.

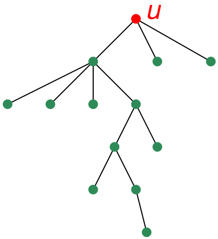


Definition 256 (Subtree, Dt.: Teilbaum)

A tree $\mathcal{T}' := (V', E')$ is a *subtree* of a tree $\mathcal{T} := (V, E)$ rooted at the node u if

1. $\mathcal{T}'$ is a subgraph of $\mathcal{T}$,
2. $\mathcal{T}'$ is rooted at a node v that is a descendant of u, and
3. $\mathcal{T}'$ contains all descendants of v in $\mathcal{T}$, together with the appropriate edges of E.

A subtree rooted at v is called a *proper subtree* if v is a child of u.



Definition 256 (Subtree, Dt.: Teilbaum)

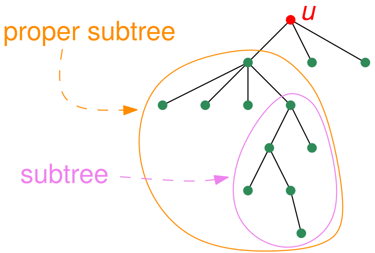
A tree $\mathcal{T}' := (V', E')$ is a *subtree* of a tree $\mathcal{T} := (V, E)$ rooted at the node u if

1. $\mathcal{T}'$ is a subgraph of $\mathcal{T}$,
2. $\mathcal{T}'$ is rooted at a node v that is a descendant of u, and
3. $\mathcal{T}'$ contains all descendants of v in $\mathcal{T}$, together with the appropriate edges of E.

A subtree rooted at v is called a *proper subtree* if v is a child of u.

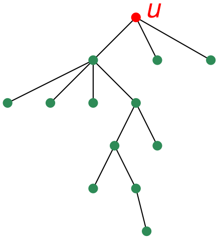
Warning

Some authors do not make the distinction between the node v being a child of u or some arbitrary descendant of u.



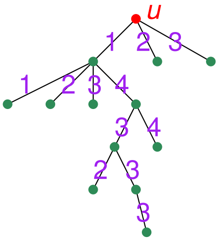
Definition 257 (Ordered tree, Dt.: geordneter Baum)

An *ordered tree* is a rooted tree $\mathcal{T}$ such that the children of every node of $\mathcal{T}$ are arranged in some specific order, e.g., by means of a numbering scheme.



Definition 257 (Ordered tree, Dt.: geordneter Baum)

An *ordered tree* is a rooted tree $\mathcal{T}$ such that the children of every node of $\mathcal{T}$ are arranged in some specific order, e.g., by means of a numbering scheme.

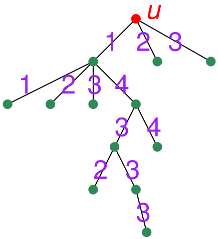


Definition 257 (Ordered tree, Dt.: geordneter Baum)

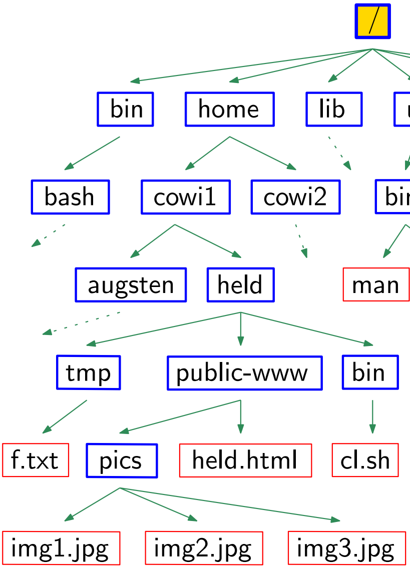
An *ordered tree* is a rooted tree $\mathcal{T}$ such that the children of every node of $\mathcal{T}$ are arranged in some specific order, e.g., by means of a numbering scheme.

Definition 258 (Forest, Dt.: Wald)

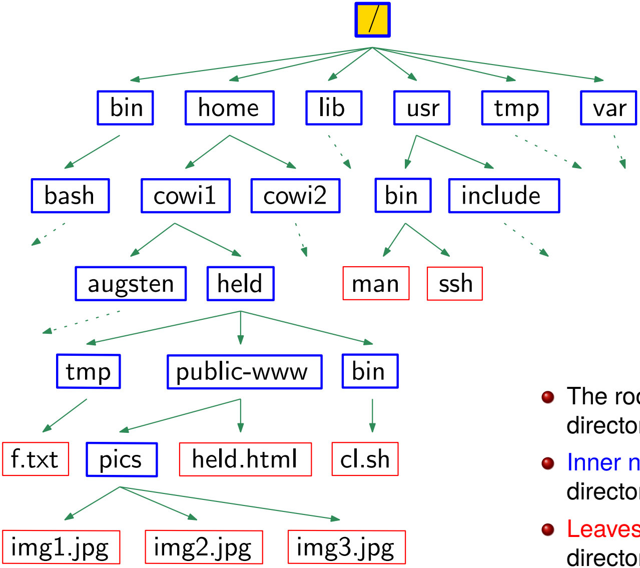
A *forest* is a graph such that all its connected components are trees.



Real-World Application: File System as a Rooted Tree



Real-World Application: File System as a Rooted Tree



- The root of the tree is the root directory /.
- Inner nodes are (non-empty) directories.
- Leaves are files (or empty directories).

Theorem 259

Every pair of nodes in a tree is connected by exactly one path.

Trees: Elementary Properties

Theorem 259

Every pair of nodes in a tree is connected by exactly one path.

Theorem 260

In a rooted tree there exists exactly one path from the root to any node.

Theorem 259

Every pair of nodes in a tree is connected by exactly one path.

Theorem 260

In a rooted tree there exists exactly one path from the root to any node.

Lemma 261

Removing an edge from a (rooted) tree results in a graph with two connected components, each of which is a (rooted) tree.

Trees: Elementary Properties

Theorem 262

For every (rooted) tree $\mathcal{T} := (V, E)$ we get $|E| = |V| - 1$.

Theorem 262

For every (rooted) tree $\mathcal{T} := (V, E)$ we get $|E| = |V| - 1$.

Proof of Theorem 262 for rooted trees : We use structural induction relative to proper subtrees.

Theorem 262

For every (rooted) tree $\mathcal{T} := (V, E)$ we get $|E| = |V| - 1$.

Proof of Theorem 262 for rooted trees : We use structural induction relative to proper subtrees. Obviously, the claim holds for the minimal elements, i.e., for trees that contain no proper subtrees and, thus, have only a root and no edges.

Theorem 262

For every (rooted) tree $\mathcal{T} := (V, E)$ we get $|E| = |V| - 1$.

Proof of Theorem 262 for rooted trees : We use structural induction relative to proper subtrees. Obviously, the claim holds for the minimal elements, i.e., for trees that contain no proper subtrees and, thus, have only a root and no edges.
Now consider an arbitrary but fixed rooted tree $\mathcal{T} := (V, E)$ and suppose that the equality claimed holds for all its $k > 0$ proper subtrees $(V_1, E_1), \ldots, (V_k, E_k)$. (We do not need to assume explicitly that it holds for all subtrees of $\mathcal{T}$.)

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 262

For every (rooted) tree $\mathcal{T} := (V, E)$ we get $|E| = |V| - 1$.

Proof of Theorem 262 for rooted trees : We use structural induction relative to proper subtrees. Obviously, the claim holds for the minimal elements, i.e., for trees that contain no proper subtrees and, thus, have only a root and no edges.
Now consider an arbitrary but fixed rooted tree $\mathcal{T} := (V, E)$ and suppose that the equality claimed holds for all its $k > 0$ proper subtrees $(V_1, E_1), \ldots, (V_k, E_k)$. (We do not need to assume explicitly that it holds for all subtrees of $\mathcal{T}$.) We get

$$|E| = k + \sum_{i=1}^{k} |E_i|$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 262

For every (rooted) tree $\mathcal{T} := (V, E)$ we get $|E| = |V| - 1$.

Proof of Theorem 262 for rooted trees : We use structural induction relative to proper subtrees. Obviously, the claim holds for the minimal elements, i.e., for trees that contain no proper subtrees and, thus, have only a root and no edges.
Now consider an arbitrary but fixed rooted tree $\mathcal{T} := (V, E)$ and suppose that the equality claimed holds for all its $k > 0$ proper subtrees $(V_1, E_1), \ldots, (V_k, E_k)$. (We do not need to assume explicitly that it holds for all subtrees of $\mathcal{T}$.) We get

$$|E| = k + \sum_{i=1}^{k} |E_i| = k + \sum_{i=1}^{k} (|V_i| - 1) = k + (-k) + \sum_{i=1}^{k} |V_i| = \sum_{i=1}^{k} |V_i|$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 262

For every (rooted) tree $\mathcal{T} := (V, E)$ we get $|E| = |V| - 1$.

Proof of Theorem 262 for rooted trees : We use structural induction relative to proper subtrees. Obviously, the claim holds for the minimal elements, i.e., for trees that contain no proper subtrees and, thus, have only a root and no edges.
Now consider an arbitrary but fixed rooted tree $\mathcal{T} := (V, E)$ and suppose that the equality claimed holds for all its $k > 0$ proper subtrees $(V_1, E_1), \ldots, (V_k, E_k)$. (We do not need to assume explicitly that it holds for all subtrees of $\mathcal{T}$.) We get

$$|E| = k + \sum_{i=1}^{k} |E_i| = k + \sum_{i=1}^{k} (|V_i| - 1) = k + (-k) + \sum_{i=1}^{k} |V_i| = \sum_{i=1}^{k} |V_i|$$
$$= |V| - 1,$$

thus establishing the claim also for $\mathcal{T} = (V, E)$. □

Trees: Elementary Properties

Theorem 262

For every (rooted) tree $\mathcal{T} := (V, E)$ we get $|E| = |V| - 1$.

Proof of Theorem 262 for rooted trees : We use structural induction relative to proper subtrees. Obviously, the claim holds for the minimal elements, i.e., for trees that contain no proper subtrees and, thus, have only a root and no edges.
Now consider an arbitrary but fixed rooted tree $\mathcal{T} := (V, E)$ and suppose that the equality claimed holds for all its $k > 0$ proper subtrees $(V_1, E_1), \ldots, (V_k, E_k)$. (We do not need to assume explicitly that it holds for all subtrees of $\mathcal{T}$.) We get

$$|E| = k + \sum_{i=1}^{k} |E_i| = k + \sum_{i=1}^{k} (|V_i| - 1) = k + (-k) + \sum_{i=1}^{k} |V_i| = \sum_{i=1}^{k} |V_i|$$
$$= |V| - 1,$$

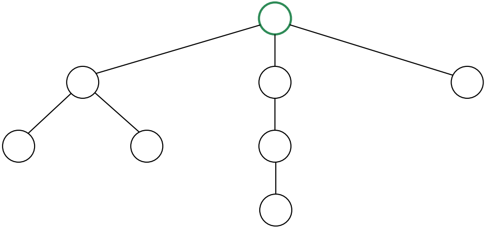
thus establishing the claim also for $\mathcal{T} = (V, E)$. □

Corollary 263

If $|V| > 1$ holds for a (rooted) tree $\mathcal{T} := (V, E)$, then $\mathcal{T}$ has at least one leaf.

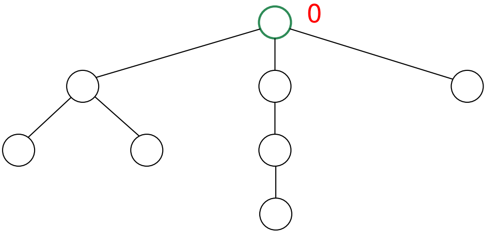
Definition 264 (Depth, Dt.: Tiefe)

The *depth* of the root u of a rooted tree $\mathcal{T} := (V, E)$ is 0, and the depth of a node $v \neq u$ of $\mathcal{T}$ is k if the depth of the parent of v is $k - 1$, for all $v \in V$.



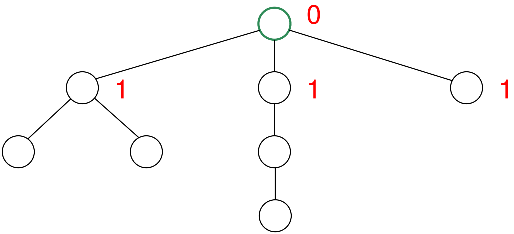
Definition 264 (Depth, Dt.: Tiefe)

The *depth* of the root u of a rooted tree $\mathcal{T} := (V, E)$ is 0, and the depth of a node $v \neq u$ of $\mathcal{T}$ is k if the depth of the parent of v is $k - 1$, for all $v \in V$.



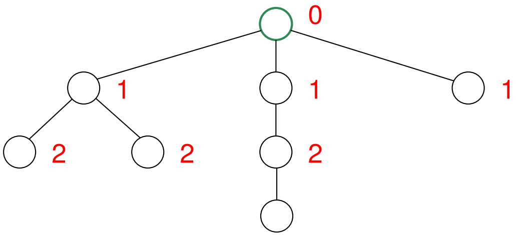
Definition 264 (Depth, Dt.: Tiefe)

The *depth* of the root u of a rooted tree $\mathcal{T} := (V, E)$ is 0, and the depth of a node $v \neq u$ of $\mathcal{T}$ is k if the depth of the parent of v is $k - 1$, for all $v \in V$.



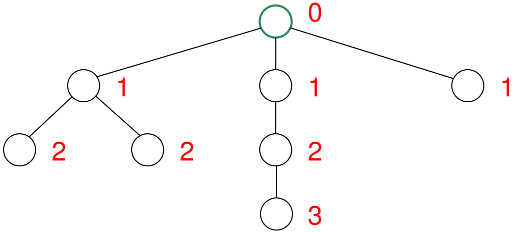
Definition 264 (Depth, Dt.: Tiefe)

The *depth* of the root u of a rooted tree $\mathcal{T} := (V, E)$ is 0, and the depth of a node $v \neq u$ of $\mathcal{T}$ is k if the depth of the parent of v is $k - 1$, for all $v \in V$.



Definition 264 (Depth, Dt.: Tiefe)

The *depth* of the root u of a rooted tree $\mathcal{T} := (V, E)$ is 0, and the depth of a node $v \neq u$ of $\mathcal{T}$ is k if the depth of the parent of v is $k - 1$, for all $v \in V$.

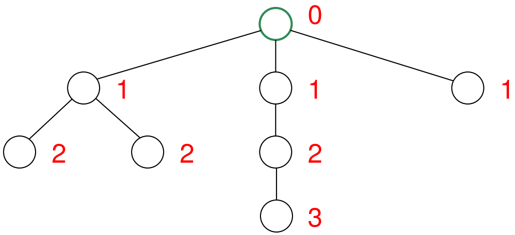


Definition 264 (Depth, Dt.: Tiefe)

The *depth* of the root u of a rooted tree $\mathcal{T} := (V, E)$ is 0, and the depth of a node $v \neq u$ of $\mathcal{T}$ is k if the depth of the parent of v is $k - 1$, for all $v \in V$.

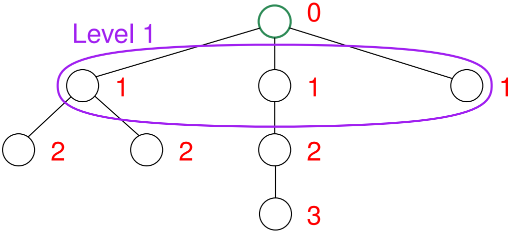
Warning

Some authors prefer to regard the root as a node at depth 1. Hence, make sure to check how depth is defined in a textbook prior to using the results stated!



Definition 265 (Level, Dt.: Niveau)

A *level* of a rooted tree $\mathcal{T}$ comprises all nodes of $\mathcal{T}$ which have the same depth.

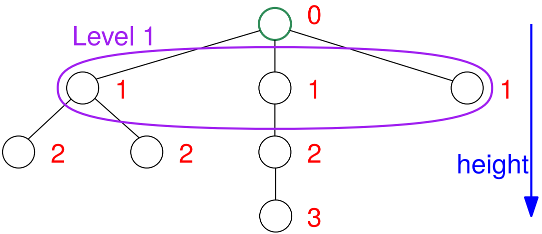


Definition 265 (Level, Dt.: Niveau)

A *level* of a rooted tree $\mathcal{T}$ comprises all nodes of $\mathcal{T}$ which have the same depth.

Definition 266 (Height, Dt.: Höhe)

The *height* of a rooted tree $\mathcal{T}$ is the maximum depth of nodes of $\mathcal{T}$.



Binary Tree

Definition 267 (Binary tree, Dt.: Binärbaum)

A *binary tree* is an ordered tree $\mathcal{T}$ with a root node u and at most two proper subtrees that are called *left subtree*, L, and *right subtree*, R. If $\mathcal{T}$ has a left (right, resp.) subtree then L (R, resp.) is in turn a binary tree rooted in the left (right, resp.) child of u.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Binary Tree

Definition 267 (Binary tree, Dt.: Binärbaum)

A *binary tree* is an ordered tree $\mathcal{T}$ with a root node u and at most two proper subtrees that are called *left subtree*, L, and *right subtree*, R. If $\mathcal{T}$ has a left (right, resp.) subtree then L (R, resp.) is in turn a binary tree rooted in the left (right, resp.) child of u.

Definition 268 (Complete binary tree, Dt.: vollständiger Binärbaum)

A *complete binary tree* is a binary tree in which every level, except possibly the last level, is completely filled, and the last level is filled from left to right.

Binary Tree

Definition 267 (Binary tree, Dt.: Binärbaum)

A *binary tree* is an ordered tree $\mathcal{T}$ with a root node u and at most two proper subtrees that are called *left subtree*, L, and *right subtree*, R. If $\mathcal{T}$ has a left (right, resp.) subtree then L (R, resp.) is in turn a binary tree rooted in the left (right, resp.) child of u.

Definition 268 (Complete binary tree, Dt.: vollständiger Binärbaum)

A *complete binary tree* is a binary tree in which every level, except possibly the last level, is completely filled, and the last level is filled from left to right.

- E.g., a (binary) heap is a complete binary tree.

Binary Tree

Definition 267 (Binary tree, Dt.: Binärbaum)

A *binary tree* is an ordered tree $\mathcal{T}$ with a root node u and at most two proper subtrees that are called *left subtree*, L, and *right subtree*, R. If $\mathcal{T}$ has a left (right, resp.) subtree then L (R, resp.) is in turn a binary tree rooted in the left (right, resp.) child of u.

Definition 268 (Complete binary tree, Dt.: vollständiger Binärbaum)

A *complete binary tree* is a binary tree in which every level, except possibly the last level, is completely filled, and the last level is filled from left to right.

- E.g., a (binary) heap is a complete binary tree.

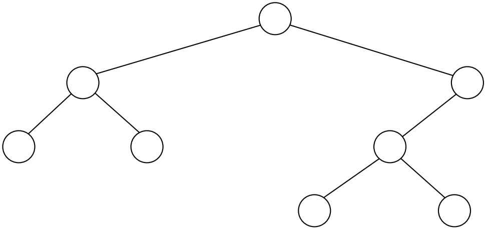
Definition 269 (Perfect binary tree, Dt.: perfekter Binärbaum)

A *perfect binary tree* is a binary tree that has the maximum number of nodes (relative to its height).

Binary Search Tree

Definition 270 (Binary search tree, Dt.: binärer Suchbaum)

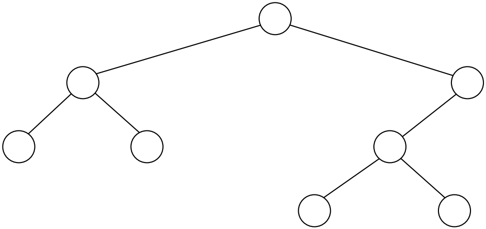
A *binary search tree* is a binary tree $\mathcal{T}$ which has distinct values associated with its nodes such that (relative to some total order)



Definition 270 (Binary search tree, Dt.: binärer Suchbaum)

A *binary search tree* is a binary tree $\mathcal{T}$ which has distinct values associated with its nodes such that (relative to some total order)

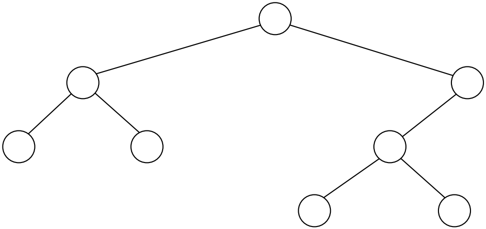
- if it has a proper left subtree L then
  1. all values of nodes in L are less than the root value,



Definition 270 (Binary search tree, Dt.: binärer Suchbaum)

A *binary search tree* is a binary tree $\mathcal{T}$ which has distinct values associated with its nodes such that (relative to some total order)

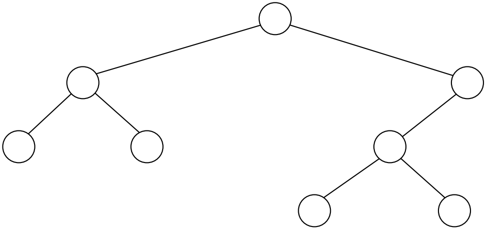
- if it has a proper left subtree L then
  1. all values of nodes in L are less than the root value,
  2. L is a binary search tree itself,



Definition 270 (Binary search tree, Dt.: binärer Suchbaum)

A *binary search tree* is a binary tree $\mathcal{T}$ which has distinct values associated with its nodes such that (relative to some total order)

- if it has a proper left subtree L then
  1. all values of nodes in L are less than the root value,
  2. L is a binary search tree itself,
- if it has a proper right subtree R then
  3. all values of nodes in R are greater than the root value,
  4. R is a binary search tree itself.

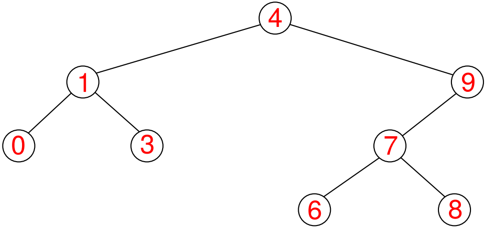


Binary Search Tree

Definition 270 (Binary search tree, Dt.: binärer Suchbaum)

A *binary search tree* is a binary tree $\mathcal{T}$ which has distinct values associated with its nodes such that (relative to some total order)

- if it has a proper left subtree L then
  1. all values of nodes in L are less than the root value,
  2. L is a binary search tree itself,
- if it has a proper right subtree R then
  3. all values of nodes in R are greater than the root value,
  4. R is a binary search tree itself.



Definition 271 (k-balanced tree, Dt.: k-balanzierter Baum)

A binary tree is *height-balanced* with balance factor k if it either has no proper subtrees

Definition 271 (k-balanced tree, Dt.: k-balanzierter Baum)

A binary tree is *height-balanced* with balance factor k if it either has no proper subtrees or if

1. it has two proper subtrees and the heights of both subtrees differ by not more than k,

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 271 (k-balanced tree, Dt.: k-balanzierter Baum)

A binary tree is *height-balanced* with balance factor k if it either has no proper subtrees or if

1. it has two proper subtrees and the heights of both subtrees differ by not more than k, or if
2. it has one proper subtree of height at most $k - 1$,

Balanced Binary Trees

Definition 271 (k-balanced tree, Dt.: k-balanzierter Baum)

A binary tree is *height-balanced* with balance factor k if it either has no proper subtrees or if

1. it has two proper subtrees and the heights of both subtrees differ by not more than k, or if
2. it has one proper subtree of height at most $k - 1$,

and if

3. all proper subtrees are height-balanced with balance factor k.

Balanced Binary Trees

Definition 271 (k-balanced tree, Dt.: k-balanzierter Baum)

A binary tree is *height-balanced* with balance factor k if it either has no proper subtrees or if

1. it has two proper subtrees and the heights of both subtrees differ by not more than k, or if
2. it has one proper subtree of height at most $k - 1$,

and if

3. all proper subtrees are height-balanced with balance factor k.

- E.g., for $k := 1$: AVL tree.
- Trees with balance factor 1 are simply called *balanced* or *self-balancing*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 272 (Perfectly balanced binary tree, Dt.: perfekt balanz. Binärbaum)

A binary tree $\mathcal{T}$ is *perfectly balanced* if all inner nodes of $\mathcal{T}$, except possibly on the second-last level, have exactly two children.

Definition 272 (Perfectly balanced binary tree, Dt.: perfekt balanz. Binärbaum)

A binary tree $\mathcal{T}$ is *perfectly balanced* if all inner nodes of $\mathcal{T}$, except possibly on the second-last level, have exactly two children.

- E.g., a (binary) heap is a perfectly balanced binary tree.

Lemma 273

A complete binary tree is perfectly balanced.

Balanced Binary Trees

Definition 272 (Perfectly balanced binary tree, Dt.: perfekt balanz. Binärbaum)

A binary tree $\mathcal{T}$ is *perfectly balanced* if all inner nodes of $\mathcal{T}$, except possibly on the second-last level, have exactly two children.

- E.g., a (binary) heap is a perfectly balanced binary tree.

Lemma 273

A complete binary tree is perfectly balanced.

Lemma 274

A perfectly balanced binary tree has leaves only at its two bottom-most levels.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Height-Related Properties of Binary Trees

Lemma 275

For $i \in \mathbb{N}_0$, level i of a binary tree contains at most 2^i nodes.

Lemma 275

For $i \in \mathbb{N}_0$, level i of a binary tree contains at most 2^i nodes.

Sketch of proof by induction : The claim holds for $i := 0$. If we have at most 2^k nodes on level k then we have at most $2 \cdot 2^k = 2^{k+1}$ nodes on level $k + 1$. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Height-Related Properties of Binary Trees

Lemma 275

For $i \in \mathbb{N}_0$, level i of a binary tree contains at most 2^i nodes.

Sketch of proof by induction : The claim holds for $i := 0$. If we have at most 2^k nodes on level k then we have at most $2 \cdot 2^k = 2^{k+1}$ nodes on level $k + 1$. □

Lemma 276

Let h be the height and n be the number of nodes of a binary tree. Then $h \geqslant \lceil \log(n+1) \rceil - 1$, i.e., $h \in \Omega(\log n)$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 275

For $i \in \mathbb{N}_0$, level i of a binary tree contains at most 2^i nodes.

Sketch of proof by induction : The claim holds for $i := 0$. If we have at most 2^k nodes on level k then we have at most $2 \cdot 2^k = 2^{k+1}$ nodes on level $k+1$. □

Lemma 276

Let h be the height and n be the number of nodes of a binary tree. Then $h \geqslant \lceil \log(n+1) \rceil - 1$, i.e., $h \in \Omega(\log n)$.

Proof : Lemma 275 implies that a binary tree with height h contains at most

$$\sum_{i=0}^{h} 2^i = 2^{h+1} - 1$$

nodes.

Height-Related Properties of Binary Trees

Lemma 275

For $i \in \mathbb{N}_0$, level i of a binary tree contains at most 2^i nodes.

Sketch of proof by induction: The claim holds for $i := 0$. If we have at most 2^k nodes on level k then we have at most $2 \cdot 2^k = 2^{k+1}$ nodes on level $k + 1$. □

Lemma 276

Let h be the height and n be the number of nodes of a binary tree. Then $h \geqslant \lceil \log(n+1) \rceil - 1$, i.e., $h \in \Omega(\log n)$.

Proof: Lemma 275 implies that a binary tree with height h contains at most

$$\sum_{i=0}^{h} 2^i = 2^{h+1} - 1$$

nodes. Hence, $n \leqslant 2^{h+1} - 1$ and, thus, $h \geqslant \lceil \log_2(n+1) \rceil - 1$. □

Height-Related Properties of Binary Trees

Lemma 275

For $i \in \mathbb{N}_0$, level i of a binary tree contains at most 2^i nodes.

Sketch of proof by induction: The claim holds for $i := 0$. If we have at most 2^k nodes on level k then we have at most $2 \cdot 2^k = 2^{k+1}$ nodes on level $k+1$. □

Lemma 276

Let h be the height and n be the number of nodes of a binary tree. Then $h \geqslant \lceil \log(n+1) \rceil - 1$, i.e., $h \in \Omega(\log n)$.

Proof: Lemma 275 implies that a binary tree with height h contains at most

$$\sum_{i=0}^{h} 2^i = 2^{h+1} - 1$$

nodes. Hence, $n \leqslant 2^{h+1} - 1$ and, thus, $h \geqslant \lceil \log_2(n+1) \rceil - 1$. □

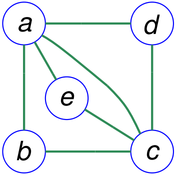
Theorem 277

If $\mathcal{T}$ is a balanced binary tree with n nodes and height h then $h \in \Theta(\log n)$.

Definition 278 (Spanning tree, Dt.: spannender Baum)

A *spanning tree* of a connected graph $\mathcal{G}$ is a subgraph of $\mathcal{G}$ that

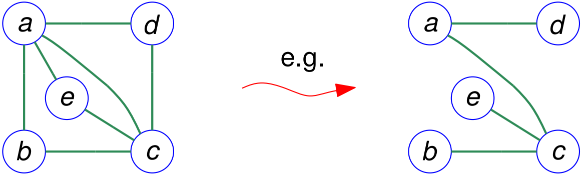
1. is a tree,
2. includes all vertices of $\mathcal{G}$.



Definition 278 (Spanning tree, Dt.: spannender Baum)

A *spanning tree* of a connected graph $\mathcal{G}$ is a subgraph of $\mathcal{G}$ that

1. is a tree,
2. includes all vertices of $\mathcal{G}$.



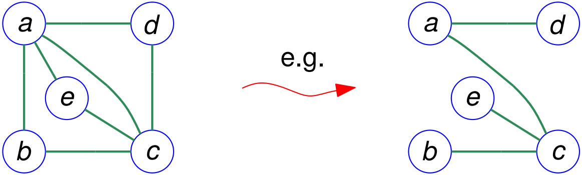
Definition 278 (Spanning tree, Dt.: spannender Baum)

A *spanning tree* of a connected graph $\mathcal{G}$ is a subgraph of $\mathcal{G}$ that

1. is a tree,
2. includes all vertices of $\mathcal{G}$.

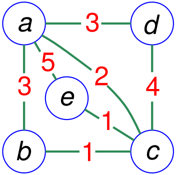
Theorem 279

Every connected graph $\mathcal{G}$ contains a spanning tree.



Definition 280 (Weighted graph, Dt.: gewichteter Graph)

An *(edge-)weighted graph* is a graph in which every edge is assigned a (non-negative) real number, the so-called *weight* or *cost*.

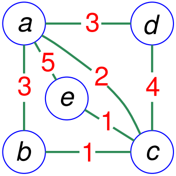


Definition 280 (Weighted graph, Dt.: gewichteter Graph)

An *(edge-)weighted graph* is a graph in which every edge is assigned a (non-negative) real number, the so-called *weight* or *cost*.

Definition 281 (Minimum spanning tree, Dt.: minimal spannender Baum)

A *minimum spanning tree* (MST) of a weighted connected graph $\mathcal{G}$ is a spanning tree $\mathcal{T}$ of $\mathcal{G}$ such that the sum of the weights of the edges of $\mathcal{T}$ is minimum over all spanning trees of $\mathcal{G}$.

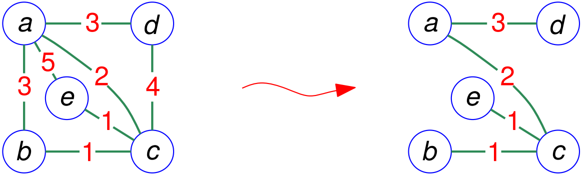


Definition 280 (Weighted graph, Dt.: gewichteter Graph)

An *(edge-)weighted graph* is a graph in which every edge is assigned a (non-negative) real number, the so-called *weight* or *cost*.

Definition 281 (Minimum spanning tree, Dt.: minimal spannender Baum)

A *minimum spanning tree* (MST) of a weighted connected graph $\mathcal{G}$ is a spanning tree $\mathcal{T}$ of $\mathcal{G}$ such that the sum of the weights of the edges of $\mathcal{T}$ is minimum over all spanning trees of $\mathcal{G}$.



Recursion Tree

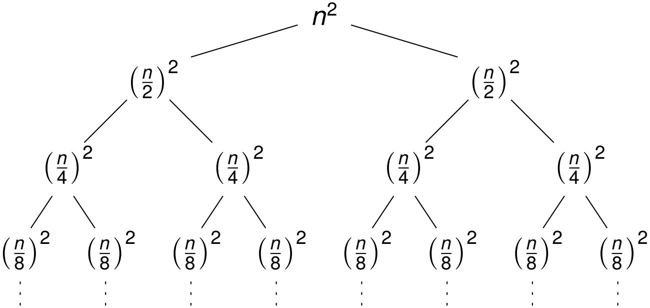
- A *recursion tree* visualizes the recursive calls of a function and the work done at each call, as given by a recurrence relation.

Recursion Tree

- A *recursion tree* visualizes the recursive calls of a function and the work done at each call, as given by a recurrence relation.
- E.g., consider $T(n) = 2T(n/2) + n^2$.

Recursion Tree

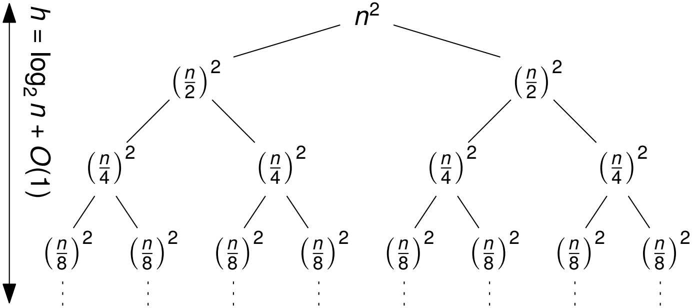
- A *recursion tree* visualizes the recursive calls of a function and the work done at each call, as given by a recurrence relation.
- E.g., consider $T(n) = 2T(n/2) + n^2$. We get the following recursion tree



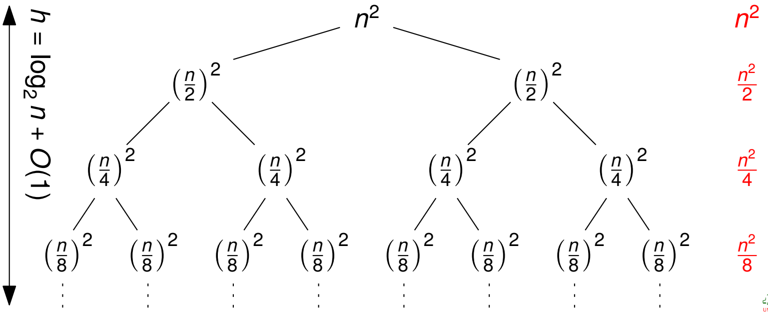
UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Recursion Tree

- A *recursion tree* visualizes the recursive calls of a function and the work done at each call, as given by a recurrence relation.
- E.g., consider $T(n) = 2T(n/2) + n^2$. We get the following recursion tree with height $h = \log_2 n + O(1)$.

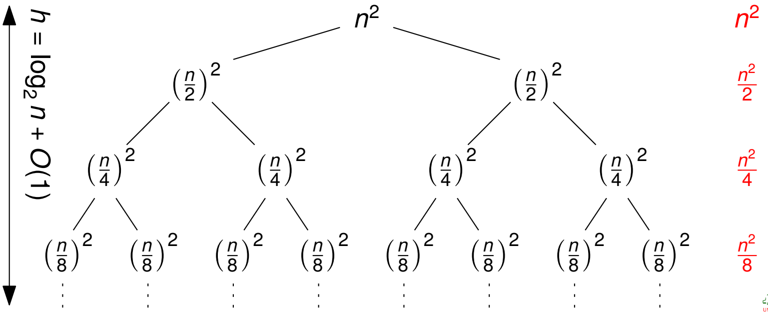


- A *recursion tree* visualizes the recursive calls of a function and the work done at each call, as given by a recurrence relation.
- E.g., consider $T(n) = 2T(n/2) + n^2$. We get the following recursion tree with height $h = \log_2 n + O(1)$.
- Summing across every level gives the total work done per level.



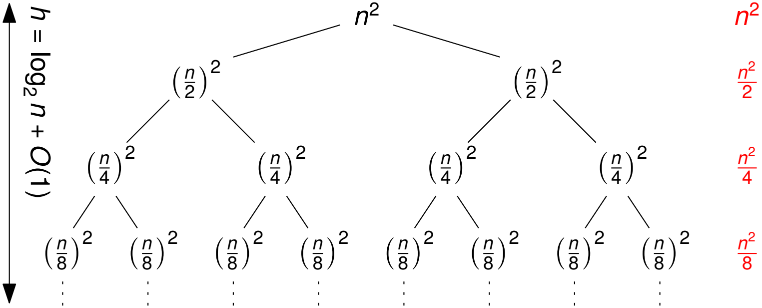
Recursion Tree

- A *recursion tree* visualizes the recursive calls of a function and the work done at each call, as given by a recurrence relation.
- E.g., consider $T(n) = 2T(n/2) + n^2$. We get the following recursion tree with height $h = \log_2 n + O(1)$.
- Summing across every level gives the total work done per level.
- Summing over all levels yields $T(n)$: This is a geometric series, with $T \in \Theta(n^2)$.



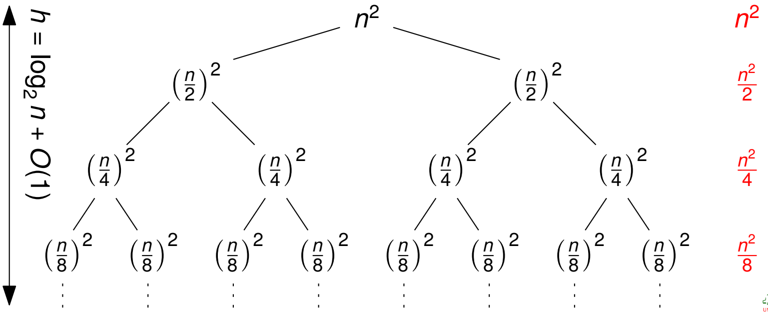
Recursion Tree

- A *recursion tree* visualizes the recursive calls of a function and the work done at each call, as given by a recurrence relation.
- E.g., consider $T(n) = 2T(n/2) + n^2$. We get the following recursion tree with height $h = \log_2 n + O(1)$.
- Summing across every level gives the total work done per level.
- Summing over all levels yields $T(n)$: This is a geometric series, with $T \in \Theta(n^2)$.
- Master Theorem 215: We have $a = b = k = 2$ and, thus, $a < b^k$.



Recursion Tree

- Note that in this case the height of the tree does not really matter: The amount of work done at every level decreases so quickly that the total work is only a constant factor more than the work done at the root.

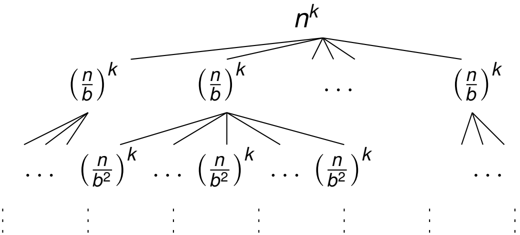


Recursion Tree

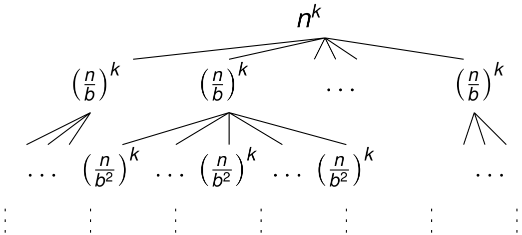
- For the recurrence relation $T(n) = a \cdot T\left(\frac{n}{b}\right) + n^k$ we get an a-ary recursion tree:

Recursion Tree

- For the recurrence relation $T(n) = a \cdot T\left(\frac{n}{b}\right) + n^k$ we get an a-ary recursion tree:
  - The problem size at level i is n/b^i.

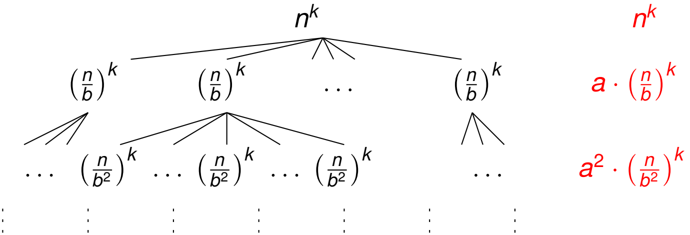


- For the recurrence relation $T(n) = a \cdot T\left(\frac{n}{b}\right) + n^k$ we get an a-ary recursion tree:
  - The problem size at level i is n/b^i.
  - The work done at every node at level i is $(n/b^i)^k$.

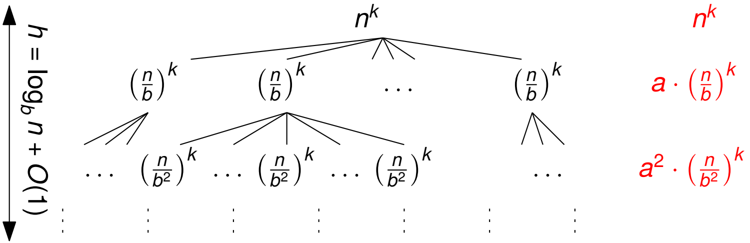


Recursion Tree

- For the recurrence relation $T(n) = a \cdot T\left(\frac{n}{b}\right) + n^k$ we get an a-ary recursion tree:
  - The problem size at level i is n/b^i.
  - The work done at every node at level i is $(n/b^i)^k$.
  - The total work done at level i is $a^i \cdot (n/b^i)^k$.

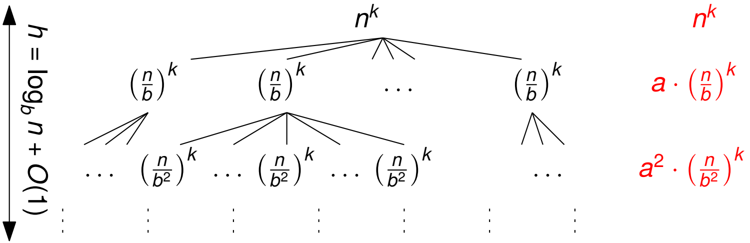


- For the recurrence relation $T(n) = a \cdot T\left(\frac{n}{b}\right) + n^k$ we get an a-ary recursion tree:
  - The problem size at level i is n/b^i.
  - The work done at every node at level i is $(n/b^i)^k$.
  - The total work done at level i is $a^i \cdot (n/b^i)^k$.
  - The tree has $\log_b n + O(1)$ levels, i.e., a height of $O(\log n)$.

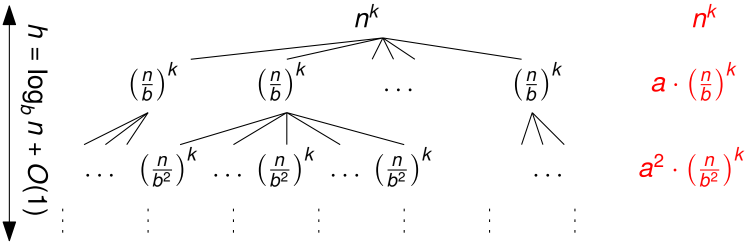


Recursion Tree

- For the recurrence relation $T(n) = a \cdot T\left(\frac{n}{b}\right) + n^k$ we get an a-ary recursion tree:
  - The problem size at level i is n/b^i.
  - The work done at every node at level i is $(n/b^i)^k$.
  - The total work done at level i is $a^i \cdot (n/b^i)^k$.
  - The tree has $\log_b n + O(1)$ levels, i.e., a height of $O(\log n)$.
  - The total number of leaves is $a^{\log_b n} = n^{\log_b a}$. (Recall $\log_b x = \log_a x / \log_a b$.)

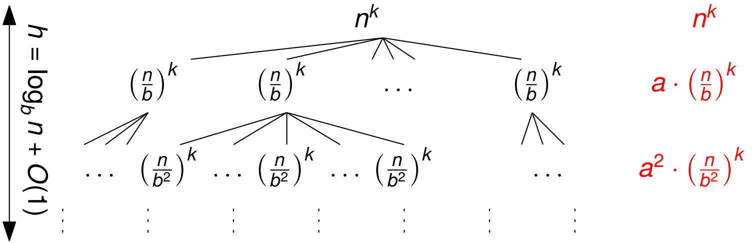


- For the recurrence relation $T(n) = a \cdot T\left(\frac{n}{b}\right) + n^k$ we get an a-ary recursion tree:
  - The problem size at level i is n/b^i.
  - The work done at every node at level i is $(n/b^i)^k$.
  - The total work done at level i is $a^i \cdot (n/b^i)^k$.
  - The tree has $\log_b n + O(1)$ levels, i.e., a height of $O(\log n)$.
  - The total number of leaves is $a^{\log_b n} = n^{\log_b a}$. (Recall $\log_b x = \log_a x / \log_a b$.)
  - The work done is constant per leaf.



- For the recurrence relation $T(n) = a \cdot T\left(\frac{n}{b}\right) + n^k$ we get an a-ary recursion tree:
  - The problem size at level i is n/b^i.
  - The work done at every node at level i is $(n/b^i)^k$.
  - The total work done at level i is $a^i \cdot (n/b^i)^k$.
  - The tree has $\log_b n + O(1)$ levels, i.e., a height of $O(\log n)$.
  - The total number of leaves is $a^{\log_b n} = n^{\log_b a}$. (Recall $\log_b x = \log_a x / \log_a b$.)
  - The work done is constant per leaf.
  - Total work:

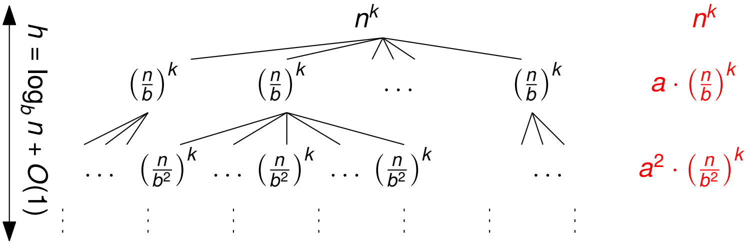
$$T(n) = \sum_{0 \leqslant i < \log_b n} a^i \cdot \left(\frac{n}{b^i}\right)^k + O(n^{\log_b a}) = \sum_{0 \leqslant i < \log_b n} n^k \cdot \left(\frac{a}{b^k}\right)^i + O(n^{\log_b a}).$$



Recursion Tree and Master Theorem

- Total work:

$$T(n) = \sum_{0 \leqslant i < \log_b n} n^k \cdot \left(\frac{a}{b^k}\right)^i + O(n^{\log_b a}).$$

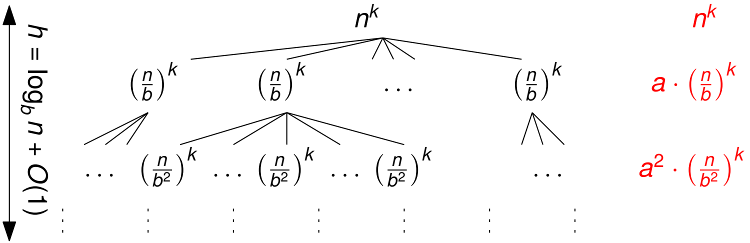


- Total work:

$$T(n) = \sum_{0 \leqslant i < \log_b n} n^k \cdot \left(\frac{a}{b^k}\right)^i + O(n^{\log_b a}).$$

- If $a = b^k$, i.e., if $k = \log_b a$, then

$$n^k \cdot \left(\frac{a}{b^k}\right)^i = n^k = n^{\log_b a}.$$



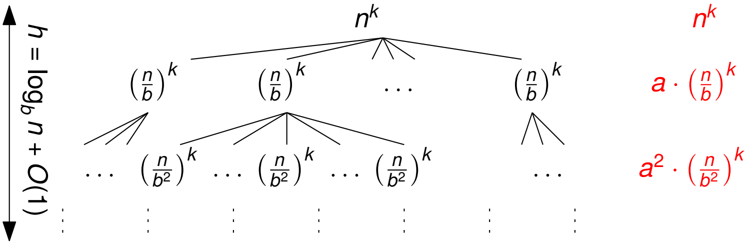
- Total work:

$$T(n) = \sum_{0 \leqslant i < \log_b n} n^k \cdot \left(\frac{a}{b^k}\right)^i + O(n^{\log_b a}).$$

- If $a = b^k$, i.e., if $k = \log_b a$, then

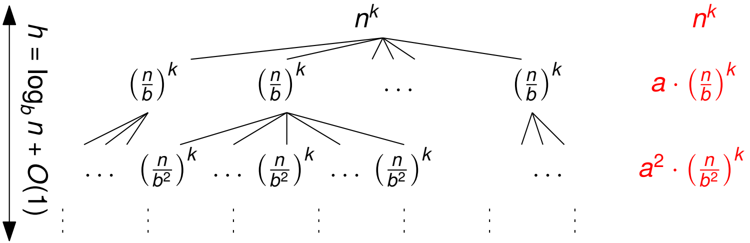
$$n^k \cdot \left(\frac{a}{b^k}\right)^i = n^k = n^{\log_b a}.$$

- Hence, the same order of work is done on every level, and since the tree has $O(\log n)$ levels, we get $T \in \Theta(n^{\log_b a} \log n)$; recall Thm. 215.



- Total work:

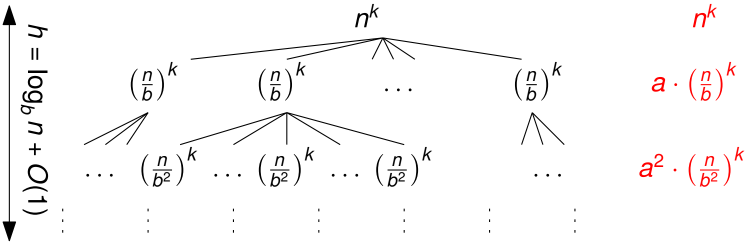
$$T(n) = \sum_{0 \leqslant i < \log_b n} n^k \cdot \left(\frac{a}{b^k}\right)^i + O(n^{\log_b a}).$$



- Total work:

$$T(n) = \sum_{0 \leqslant i < \log_b n} n^k \cdot \left(\frac{a}{b^k}\right)^i + O(n^{\log_b a}).$$

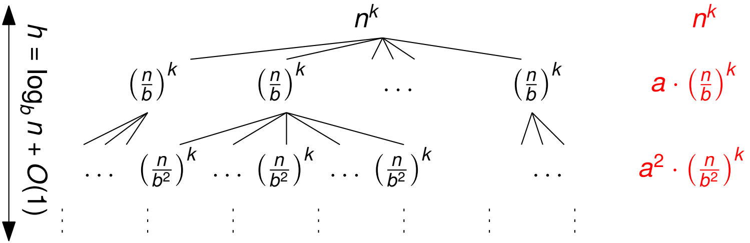
- If $a < b^k$, i.e., if $k > \log_b a$, then n^k grows asymptotically faster than the number of leaves. Hence, asymptotically the total work is dominated by the work done at the root, and we get $T \in \Theta(n^k)$; recall Thm. 215.



Recursion Tree and Master Theorem

- Total work:

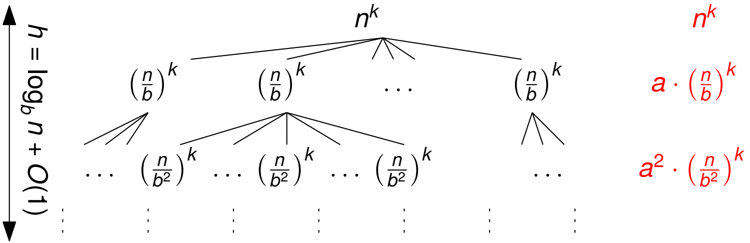
$$T(n) = \sum_{0 \leqslant i < \log_b n} n^k \cdot \left(\frac{a}{b^k}\right)^i + O(n^{\log_b a}).$$



- Total work:

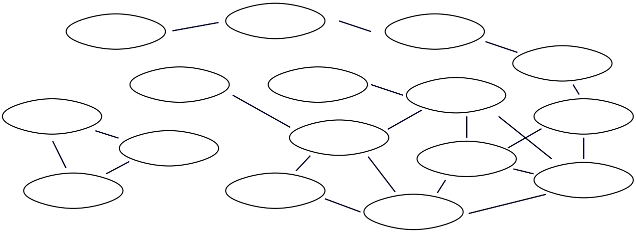
$$T(n) = \sum_{0 \leqslant i < \log_b n} n^k \cdot \left(\frac{a}{b^k}\right)^i + O(n^{\log_b a}).$$

- If $a > b^k$, i.e., if $k < \log_b a$, then n^k grows asymptotically slower than the number of leaves. Hence, asymptotically the total work is dominated by the work done at the leaves, and we get $T \in \Theta(n^{\log_b a})$; recall Thm. 215.



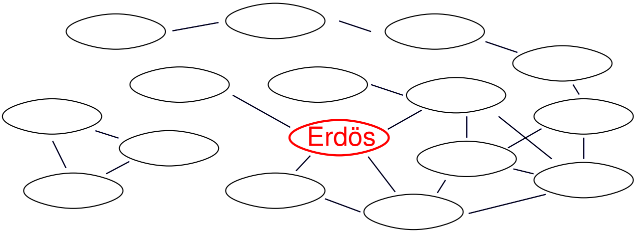
Real-World Application: Collaboration Graph and the Erdös Number

- A *collaboration graph* for a set of n scientists is a graph with n vertices such that two vertices are connected by an edge if the corresponding scientists have at least one joint publication.



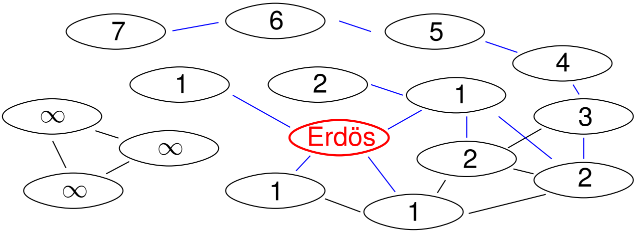
Real-World Application: Collaboration Graph and the Erdös Number

- A *collaboration graph* for a set of n scientists is a graph with n vertices such that two vertices are connected by an edge if the corresponding scientists have at least one joint publication.
- The *Erdös number* of a scientist is the “collaborative distance” of a scientist to the extremely prolific Hungarian mathematician Paul Erdös (1913–1996, more than 500 co-authors and more than 1500 publications): Erdös has 0, and a scientist has Erdös number $k + 1$ if k is the lowest Erdös number of his/her co-authors.

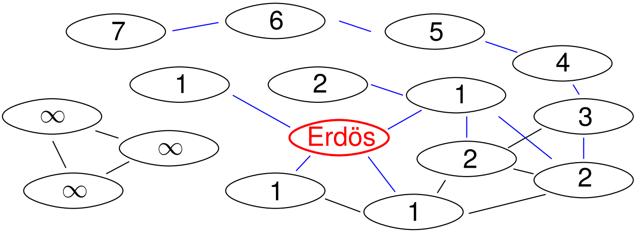


Real-World Application: Collaboration Graph and the Erdös Number

- A *collaboration graph* for a set of n scientists is a graph with n vertices such that two vertices are connected by an edge if the corresponding scientists have at least one joint publication.
- The *Erdös number* of a scientist is the “collaborative distance” of a scientist to the extremely prolific Hungarian mathematician Paul Erdös (1913–1996, more than 500 co-authors and more than 1500 publications): Erdös has 0, and a scientist has Erdös number $k + 1$ if k is the lowest Erdös number of his/her co-authors.

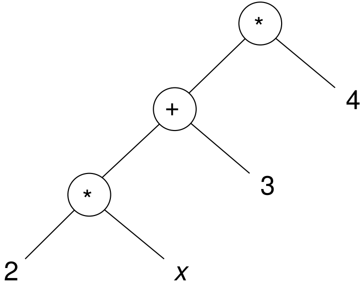


- A *collaboration graph* for a set of n scientists is a graph with n vertices such that two vertices are connected by an edge if the corresponding scientists have at least one joint publication.
- The *Erdös number* of a scientist is the “collaborative distance” of a scientist to the extremely prolific Hungarian mathematician Paul Erdös (1913–1996, more than 500 co-authors and more than 1500 publications): Erdös has 0, and a scientist has Erdös number $k + 1$ if k is the lowest Erdös number of his/her co-authors.
- One’s Erdös number can be obtained by computing minimum-weight paths on a collaboration graph.



Real-World Application: Algebraic Expression Trees

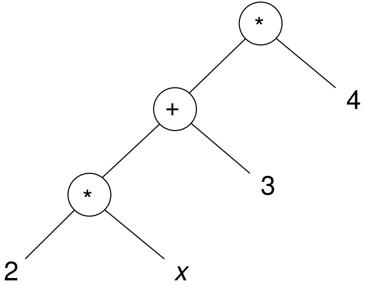
- An algebraic expression tree is a rooted tree that corresponds to an expression.
- E.g., an in-order traversal of the tree



produces the standard (infix) expression $(2x + 3) \cdot 4$.

Real-World Application: Algebraic Expression Trees

- An algebraic expression tree is a rooted tree that corresponds to an expression.
- E.g., an in-order traversal of the tree

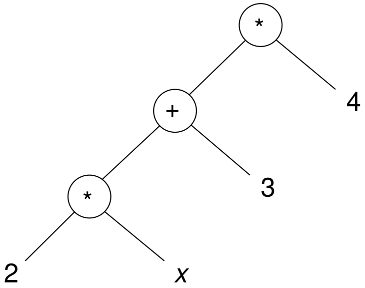


produces the standard (infix) expression $(2x + 3) \cdot 4$.

- A post-order traversal yields the postfix expression $2\ x \cdot 3 + 4 \cdot$, while a pre-order traversal yields the prefix expression $\cdot\,(+(\cdot\,(2\ x)\,3)\,4)$.

Real-World Application: Algebraic Expression Trees

- An algebraic expression tree is a rooted tree that corresponds to an expression.
- E.g., an in-order traversal of the tree



produces the standard (infix) expression $(2x + 3) \cdot 4$.

- A post-order traversal yields the postfix expression $2\ x \cdot 3 + 4 \cdot$, while a pre-order traversal yields the prefix expression $\cdot\,(+(\cdot\,(2\ x)\,3)\,4)$.
- The analysis of expression trees is a central task for the simplification and parallel evaluation of an expression.

7 Graph Theory

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

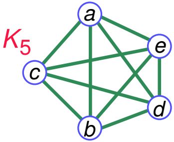
Definition 282 (Complete graph, Dt.: vollständiger Graph)

For $n \in \mathbb{N}$, the *complete graph* on n vertices, commonly denoted by K_n, is an undirected graph with n vertices in which every pair of vertices is adjacent.

Complete and Bipartite Graphs

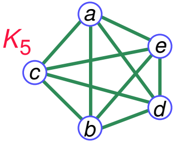
Definition 282 (Complete graph, Dt.: vollständiger Graph)

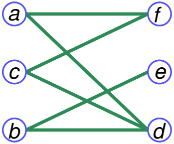
For $n \in \mathbb{N}$, the *complete graph* on n vertices, commonly denoted by K_n, is an undirected graph with n vertices in which every pair of vertices is adjacent.



Definition 282 (Complete graph, Dt.: vollständiger Graph)

For $n \in \mathbb{N}$, the *complete graph* on n vertices, commonly denoted by K_n, is an undirected graph with n vertices in which every pair of vertices is adjacent.





Definition 283 (Bipartite graph, Dt.: bipartiter Graph)

An undirected graph $\mathcal{G} := (V, E)$ is a *bipartite graph* if V can be partitioned into two (non-empty) subsets V_1, V_2 such that $E \subseteq \{\{v_1, v_2\} : v_1 \in V_1, v_2 \in V_2\}$.

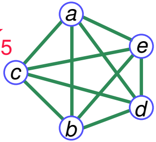
UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

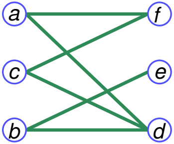
Complete and Bipartite Graphs

Definition 282 (Complete graph, Dt.: vollständiger Graph)

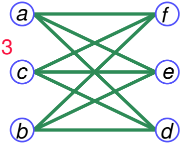
For $n \in \mathbb{N}$, the *complete graph* on n vertices, commonly denoted by K_n, is an undirected graph with n vertices in which every pair of vertices is adjacent.

K_5





$K_{3,3}$



Definition 283 (Bipartite graph, Dt.: bipartiter Graph)

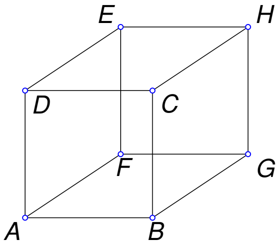
An undirected graph $\mathcal{G} := (V, E)$ is a *bipartite graph* if V can be partitioned into two (non-empty) subsets V_1, V_2 such that $E \subseteq \{\{v_1, v_2\} : v_1 \in V_1, v_2 \in V_2\}$.

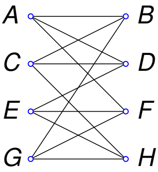
Definition 284 (Complete bipartite graph, Dt.: vollständig-bipartiter Graph)

An undirected graph $\mathcal{G} := (V, E)$ is a *complete bipartite graph* if V can be partitioned into two (non-empty) subsets V_1, V_2 such that $E = \{\{v_1, v_2\} : v_1 \in V_1, v_2 \in V_2\}$. If $n := |V_1|$ and $m := |V_2|$ then $\mathcal{G}$ is denoted by $K_{n,m}$.

Complete and Bipartite Graphs

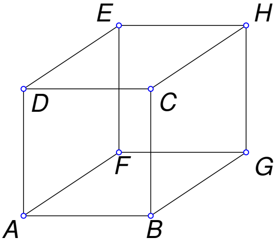
- The edges and corners of a cube can be interpreted as a bipartite graph.

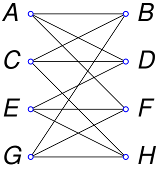




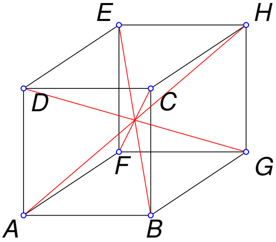
Complete and Bipartite Graphs

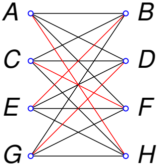
- The edges and corners of a cube can be interpreted as a bipartite graph.





- If we add all diagonals that cross the cube then we get $K_{4,4}$.





UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 285

Let $\mathcal{G} := (V, E)$ be a bipartite graph and let V_1, V_2 be the partition of V according to Def. 283. Then

$$\sum_{v_1 \in V_1} \deg(v_1) = \sum_{v_2 \in V_2} \deg(v_2) = |E|.$$

Lemma 285

Let $\mathcal{G} := (V, E)$ be a bipartite graph and let V_1, V_2 be the partition of V according to Def. 283. Then

$$\sum_{v_1 \in V_1} \deg(v_1) = \sum_{v_2 \in V_2} \deg(v_2) = |E|.$$

Proof :

- As each edge has exactly one vertex from V_1, we can write

$$\sum_{v_1 \in V_1} \deg(v_1) = |E|.$$

- Similarly,

$$\sum_{v_2 \in V_2} \deg(v_2) = |E|.$$

□

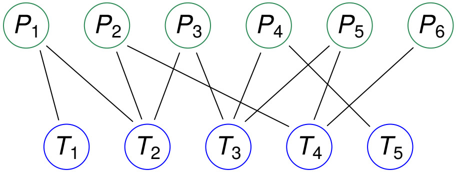
Real-World Application: Task Assignment and Matchings

- Suppose that we are given a set of tasks and a set of processors. We know which processor can carry out which tasks.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

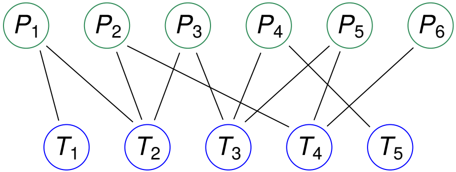
Real-World Application: Task Assignment and Matchings

- Suppose that we are given a set of tasks and a set of processors. We know which processor can carry out which tasks.
- These relations can be represented as a bipartite graph.



Real-World Application: Task Assignment and Matchings

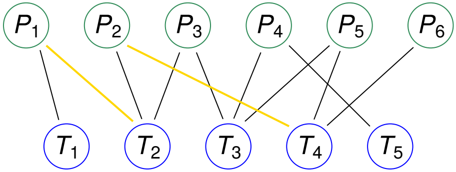
- Suppose that we are given a set of tasks and a set of processors. We know which processor can carry out which tasks.
- These relations can be represented as a bipartite graph.
- How can we get the maximum number of tasks processed concurrently?



- Suppose that we are given a set of tasks and a set of processors. We know which processor can carry out which tasks.
- These relations can be represented as a bipartite graph.
- How can we get the maximum number of tasks processed concurrently?

Definition 286 (Matching, Dt.: Paarung)

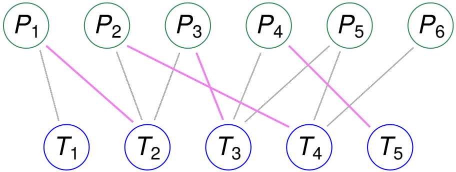
- A *matching* in a simple graph $\mathcal{G} := (V, E)$ is a subset E' of E such that no two edges of E' are incident upon the same vertex of V.



- Suppose that we are given a set of tasks and a set of processors. We know which processor can carry out which tasks.
- These relations can be represented as a bipartite graph.
- How can we get the maximum number of tasks processed concurrently?

Definition 286 (Matching, Dt.: Paarung)

- A *matching* in a simple graph $\mathcal{G} := (V, E)$ is a subset E' of E such that no two edges of E' are incident upon the same vertex of V.
- A *maximal matching* is a matching that does not allow to add an additional edge.

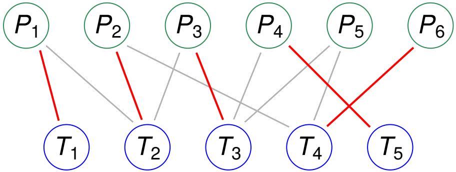


Real-World Application: Task Assignment and Matchings

- Suppose that we are given a set of tasks and a set of processors. We know which processor can carry out which tasks.
- These relations can be represented as a bipartite graph.
- How can we get the maximum number of tasks processed concurrently?

Definition 286 (Matching, Dt.: Paarung)

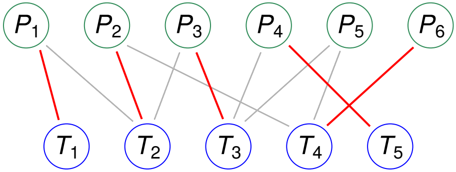
- A *matching* in a simple graph $\mathcal{G} := (V, E)$ is a subset E' of E such that no two edges of E' are incident upon the same vertex of V.
- A *maximal matching* is a matching that does not allow to add an additional edge.
- A *maximum matching* is a matching with the largest-possible number of edges.



- Suppose that we are given a set of tasks and a set of processors. We know which processor can carry out which tasks.
- These relations can be represented as a bipartite graph.
- How can we get the maximum number of tasks processed concurrently?

Definition 286 (Matching, Dt.: Paarung)

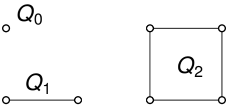
- A *matching* in a simple graph $\mathcal{G} := (V, E)$ is a subset E' of E such that no two edges of E' are incident upon the same vertex of V.
- A *maximal matching* is a matching that does not allow to add an additional edge.
- A *maximum matching* is a matching with the largest-possible number of edges.
- A *perfect matching* is a matching that leaves no vertex unmatched.



Definition 287 (Hypercube)

For $n \in \mathbb{N}_0$, the hypercube Q_n is defined recursively as follows:

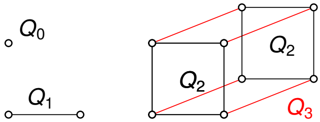
1. Q_0 is a single vertex;
2. Q_{n+1} is obtained by taking two disjoint copies of Q_n and linking each vertex in one copy of Q_n to the corresponding vertex in the other copy of Q_n.



Definition 287 (Hypercube)

For $n \in \mathbb{N}_0$, the hypercube Q_n is defined recursively as follows:

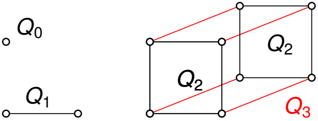
1. Q_0 is a single vertex;
2. Q_{n+1} is obtained by taking two disjoint copies of Q_n and linking each vertex in one copy of Q_n to the corresponding vertex in the other copy of Q_n.



Definition 287 (Hypercube)

For $n \in \mathbb{N}_0$, the hypercube Q_n is defined recursively as follows:

1. Q_0 is a single vertex;
2. Q_{n+1} is obtained by taking two disjoint copies of Q_n and linking each vertex in one copy of Q_n to the corresponding vertex in the other copy of Q_n.



Lemma 288

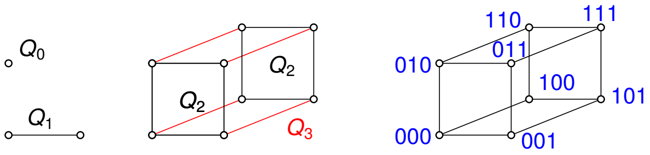
For $n \in \mathbb{N}_0$, the hypercube Q_n is a regular graph of degree n with 2^n vertices and $n \cdot 2^{n-1}$ edges; it is bipartite for $n \geqslant 1$.

Hypercube

Definition 287 (Hypercube)

For $n \in \mathbb{N}_0$, the hypercube Q_n is defined recursively as follows:

1. Q_0 is a single vertex;
2. Q_{n+1} is obtained by taking two disjoint copies of Q_n and linking each vertex in one copy of Q_n to the corresponding vertex in the other copy of Q_n.



Lemma 288

For $n \in \mathbb{N}_0$, the hypercube Q_n is a regular graph of degree n with 2^n vertices and $n \cdot 2^{n-1}$ edges; it is bipartite for $n \geqslant 1$.

- We could also obtain Q_n by labeling 2^n vertices with distinct n-bit binary strings, and by connecting those vertices by edges whose strings differ in exactly one bit.

Definition 289 (Gray code)

A *(cyclic) Gray code* of an ordered sequence of 2^n entities, for $n \in \mathbb{N}$, is a sequence of n-bit binary strings such that the encodings of two neighboring entities have Hamming distance one, i.e., differ by exactly one bit.

- Gray codes are widely used in position encoders and for error detection and correction in digital communication.

Real-World Application: Hamilton Cycles in Q_n Yield Gray Codes

Definition 289 (Gray code)

A *(cyclic) Gray code* of an ordered sequence of 2^n entities, for $n \in \mathbb{N}$, is a sequence of n-bit binary strings such that the encodings of two neighboring entities have Hamming distance one, i.e., differ by exactly one bit.

- Gray codes are widely used in position encoders and for error detection and correction in digital communication.

Lemma 290

For $n \in \mathbb{N}$ with $n \geqslant 2$, the number of different n-bit cyclic Gray codes equals the number of different Hamilton cycles in Q_n.

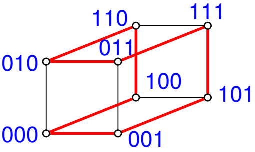
Definition 289 (Gray code)

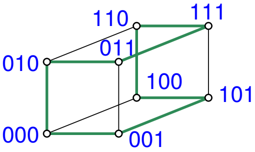
A *(cyclic) Gray code* of an ordered sequence of 2^n entities, for $n \in \mathbb{N}$, is a sequence of n-bit binary strings such that the encodings of two neighboring entities have Hamming distance one, i.e., differ by exactly one bit.

- Gray codes are widely used in position encoders and for error detection and correction in digital communication.

Lemma 290

For $n \in \mathbb{N}$ with $n \geqslant 2$, the number of different n-bit cyclic Gray codes equals the number of different Hamilton cycles in Q_n.





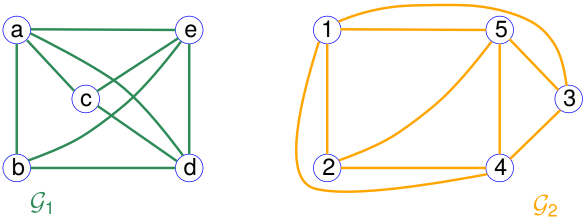
Definition 291 (Isomorphic, Dt.: isomorph)

Two (directed) graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ are *isomorphic*, denoted by $\mathcal{G}_1 \simeq \mathcal{G}_2$, if there exists a one-to-one mapping f between V_1 and V_2 that preserves adjacency; i.e., $uv \in E_1 \Leftrightarrow f(u)f(v) \in E_2$ for all $u, v \in V_1$. Such a suitable function f is called *graph isomorphism*.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

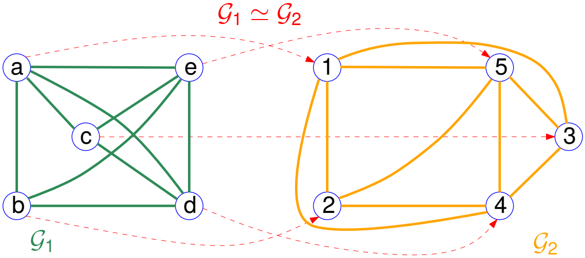
Definition 291 (Isomorphic, Dt.: isomorph)

Two (directed) graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ are *isomorphic*, denoted by $\mathcal{G}_1 \simeq \mathcal{G}_2$, if there exists a one-to-one mapping f between V_1 and V_2 that preserves adjacency; i.e., $uv \in E_1 \Leftrightarrow f(u)f(v) \in E_2$ for all $u, v \in V_1$. Such a suitable function f is called *graph isomorphism*.



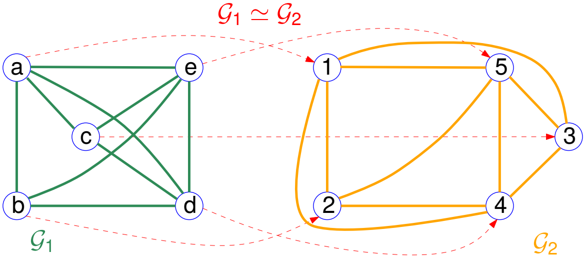
Definition 291 (Isomorphic, Dt.: isomorph)

Two (directed) graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ are *isomorphic*, denoted by $\mathcal{G}_1 \simeq \mathcal{G}_2$, if there exists a one-to-one mapping f between V_1 and V_2 that preserves adjacency; i.e., $uv \in E_1 \Leftrightarrow f(u)f(v) \in E_2$ for all $u, v \in V_1$. Such a suitable function f is called *graph isomorphism*.



Definition 291 (Isomorphic, Dt.: isomorph)

Two (directed) graphs $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ are *isomorphic*, denoted by $\mathcal{G}_1 \simeq \mathcal{G}_2$, if there exists a one-to-one mapping f between V_1 and V_2 that preserves adjacency; i.e., $uv \in E_1 \Leftrightarrow f(u)f(v) \in E_2$ for all $u, v \in V_1$. Such a suitable function f is called *graph isomorphism*.

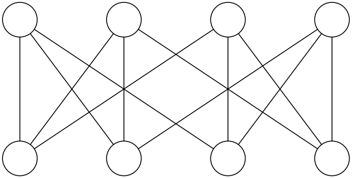


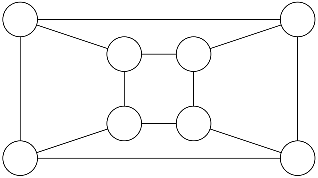
Lemma 292

The relation $\simeq$ is an equivalence relation on graphs.

Isomorphic Graphs

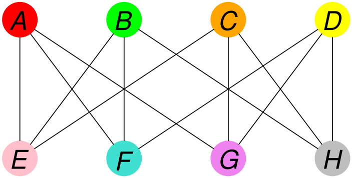
- Don't be fooled by drawings! Two graphs may be isomorphic even if their drawings look strikingly different.

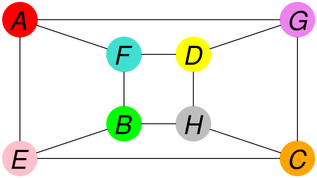




Isomorphic Graphs

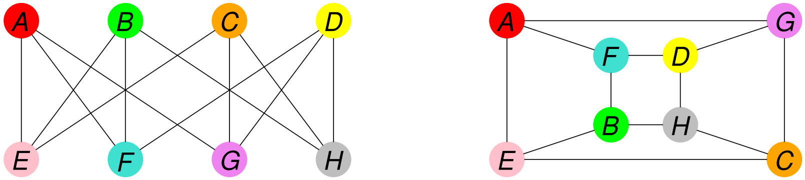
- Don't be fooled by drawings! Two graphs may be isomorphic even if their drawings look strikingly different.





Isomorphic Graphs

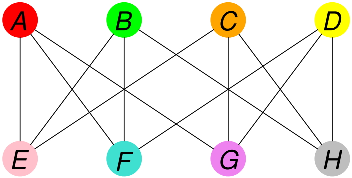
- Don't be fooled by drawings! Two graphs may be isomorphic even if their drawings look strikingly different.

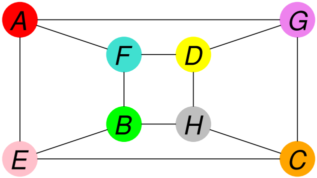


- Necessary (but not sufficient) conditions for two graphs to be isomorphic: same numbers of vertices and edges, same degrees.

Isomorphic Graphs

- Don't be fooled by drawings! Two graphs may be isomorphic even if their drawings look strikingly different.





- Necessary (but not sufficient) conditions for two graphs to be isomorphic: same numbers of vertices and edges, same degrees.
- The complexity of the graph isomorphism problem for general n-vertex graphs is unknown. No polynomial-time algorithm is known, but the problem is also not known to be $\mathcal{NP}$-complete. In December 2015, Babai announced a deterministic algorithm that runs in time $2^{O(\log^c n)}$ time for some positive constant c, i.e., in quasi-polynomial time. In 2017, Helfgott claimed that one can take $c := 3$.
- Practically efficient algorithms for graph canonical labeling are known, though.

Real-World Application: Non-Isomorphic Trees Represent Molecules

- [Cayley 1857]: Molecules can be represented as graphs, where atoms are represented by vertices and bonds are represented by edges.
- Saturated hydrocarbons, C_nH_{2n+2}, are given by trees where each carbon atom is represented by a degree-four vertex and each hydrogen atom is a leaf.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

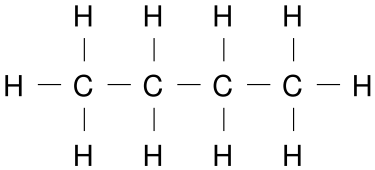
Real-World Application: Non-Isomorphic Trees Represent Molecules

- [Cayley 1857]: Molecules can be represented as graphs, where atoms are represented by vertices and bonds are represented by edges.
- Saturated hydrocarbons, C_nH_{2n+2}, are given by trees where each carbon atom is represented by a degree-four vertex and each hydrogen atom is a leaf.
- How many different isomers can exist for $n := 4$?

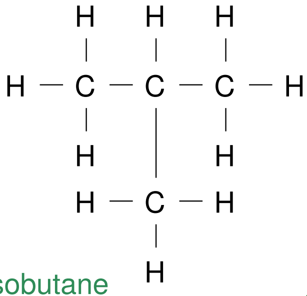
UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Non-Isomorphic Trees Represent Molecules

- [Cayley 1857]: Molecules can be represented as graphs, where atoms are represented by vertices and bonds are represented by edges.
- Saturated hydrocarbons, C_nH_{2n+2}, are given by trees where each carbon atom is represented by a degree-four vertex and each hydrogen atom is a leaf.
- How many different isomers can exist for $n := 4$?
- We have exactly two non-isomorphic trees of this type and, thus, two different isomers of C_4H_{10}, namely butane and isobutane.



Butane



Isobutane

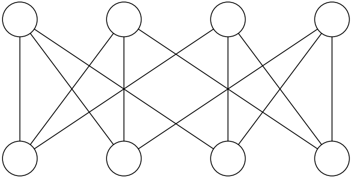
Definition 293 (Planar graph, Dt.: planarer oder plättbarer Graph)

A *planar graph* is a graph which can be drawn in the plane without edge crossings. A suitable drawing is called a *(planar) embedding* (Dt.: planare Einbettung).

Definition 293 (Planar graph, Dt.: planarer oder plättbarer Graph)

A *planar graph* is a graph which can be drawn in the plane without edge crossings. A suitable drawing is called a *(planar) embedding* (Dt.: planare Einbettung).

- Note: A graph may be planar even if a non-planar embedding is seen!

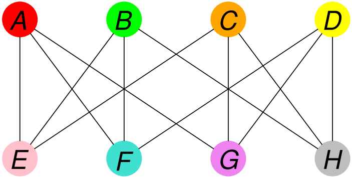


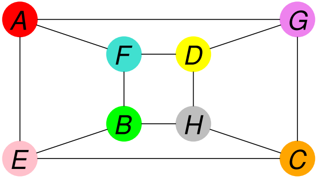
Planar Graphs

Definition 293 (Planar graph, Dt.: planarer oder plättbarer Graph)

A *planar graph* is a graph which can be drawn in the plane without edge crossings. A suitable drawing is called a *(planar) embedding* (Dt.: planare Einbettung).

- Note: A graph may be planar even if a non-planar embedding is seen!



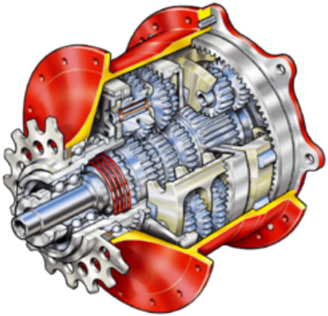


Real-World Applications of Planar Graphs

- Graphs that can be drawn in the plane without edge crossings should be drawn without edge crossings if a human is to interprete such a drawing: e.g., bus or subway map, drawing of a molecule, social network.

Real-World Applications of Planar Graphs

- Graphs that can be drawn in the plane without edge crossings should be drawn without edge crossings if a human is to interprete such a drawing: e.g., bus or subway map, drawing of a molecule, social network.
- VLSI circuits are easier/cheaper to manufacture if their connections live in fewer layers.
- A scheme for a planetary gearset is compatible if and only if a suitably designed graph is planar.



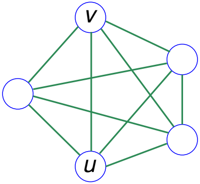
[Image credit: Rohloff AG, http://www.rohloff.de/]

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Subdivision of a Graph

Definition 294 (Subdivision, Dt.: Unterteilung)

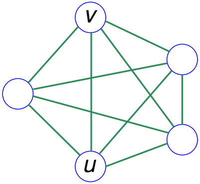
An *edge subdivision* of the edge $uv \in E$ by means of the vertex $w \notin V$ transforms the graph $\mathcal{G} := (V, E)$ into the graph $\mathcal{G}' = (V', E')$, where $V' = V \cup \{w\}$ and $E' = (E \backslash \{uv\}) \cup \{uw, wv\}$.

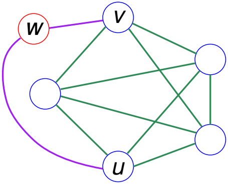


Subdivision of a Graph

Definition 294 (Subdivision, Dt.: Unterteilung)

An *edge subdivision* of the edge $uv \in E$ by means of the vertex $w \notin V$ transforms the graph $\mathcal{G} := (V, E)$ into the graph $\mathcal{G}' = (V', E')$, where $V' = V \cup \{w\}$ and $E' = (E \backslash \{uv\}) \cup \{uw, wv\}$.





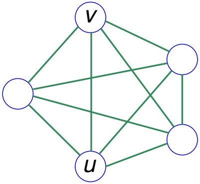
Subdivision of a Graph

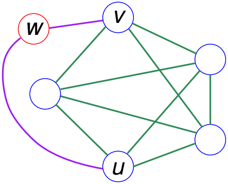
Definition 294 (Subdivision, Dt.: Unterteilung)

An *edge subdivision* of the edge $uv \in E$ by means of the vertex $w \notin V$ transforms the graph $\mathcal{G} := (V, E)$ into the graph $\mathcal{G}' = (V', E')$, where $V' = V \cup \{w\}$ and $E' = (E \backslash \{uv\}) \cup \{uw, wv\}$.

Definition 295 (Subdivision graph, Dt.: Unterteilungsgraph)

A graph $\mathcal{G}'$ is a *subdivision graph* of $\mathcal{G}$ if $\mathcal{G}'$ is obtained from $\mathcal{G}$ via a finite sequence of edge subdivisions.





Subdivision of a Graph: Planarity

Theorem 296 (Kuratowski (1930))

A graph is planar if and only if it does not contain a subgraph that is isomorphic to a subdivision graph of K_5 or $K_{3,3}$.

Theorem 296 (Kuratowski (1930))

A graph is planar if and only if it does not contain a subgraph that is isomorphic to a subdivision graph of K_5 or $K_{3,3}$.

Corollary 297

If a graph contains K_5 or $K_{3,3}$ as a subgraph then it is not planar.

Subdivision of a Graph: Planarity

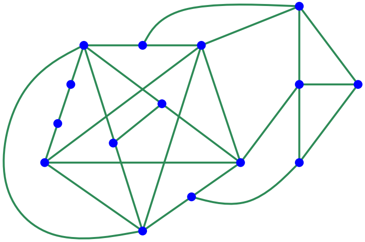
Theorem 296 (Kuratowski (1930))

A graph is planar if and only if it does not contain a subgraph that is isomorphic to a subdivision graph of K_5 or $K_{3,3}$.

Corollary 297

If a graph contains K_5 or $K_{3,3}$ as a subgraph then it is not planar.

- Is the following graph planar?



Subdivision of a Graph: Planarity

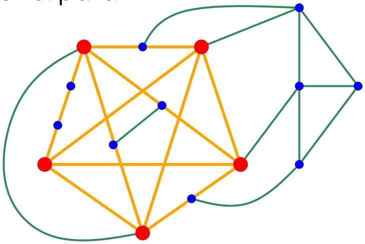
Theorem 296 (Kuratowski (1930))

A graph is planar if and only if it does not contain a subgraph that is isomorphic to a subdivision graph of K_5 or $K_{3,3}$.

Corollary 297

If a graph contains K_5 or $K_{3,3}$ as a subgraph then it is not planar.

- Is the following graph planar? No: It contains a subdivision graph of K_5 as a subgraph. Hence, it is not planar.

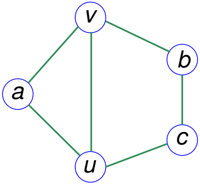


UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Edge Contraction

Definition 298 (Edge contraction, Dt.: Kantenkontraktion)

In a graph $\mathcal{G} := (V, E)$, the *contraction* of an edge $e \in E$, with $e = uv$ for some $u, v \in V$, replaces u and v by a new vertex $w \notin V$ such that edges incident to w are all edges other than e that were incident with u or v. All other nodes and edges are preserved.



Edge Contraction

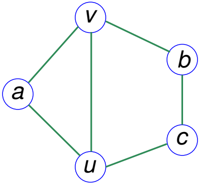
Definition 298 (Edge contraction, Dt.: Kantenkontraktion)

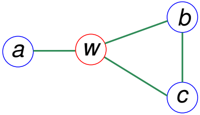
In a graph $\mathcal{G} := (V, E)$, the *contraction* of an edge $e \in E$, with $e = uv$ for some $u, v \in V$, replaces u and v by a new vertex $w \notin V$ such that edges incident to w are all edges other than e that were incident with u or v. All other nodes and edges are preserved.

Edge Contraction

Definition 298 (Edge contraction, Dt.: Kantenkontraktion)

In a graph $\mathcal{G} := (V, E)$, the *contraction* of an edge $e \in E$, with $e = uv$ for some $u, v \in V$, replaces u and v by a new vertex $w \notin V$ such that edges incident to w are all edges other than e that were incident with u or v. All other nodes and edges are preserved. Parallel edges may be unified to get a simple graph rather than a multigraph with loops.





Edge Contraction

Definition 298 (Edge contraction, Dt.: Kantenkontraktion)

In a graph $\mathcal{G} := (V, E)$, the *contraction* of an edge $e \in E$, with $e = uv$ for some $u, v \in V$, replaces u and v by a new vertex $w \notin V$ such that edges incident to w are all edges other than e that were incident with u or v. All other nodes and edges are preserved. Parallel edges may be unified to get a simple graph rather than a multigraph with loops.

Theorem 299 (Wagner (1937))

A graph is planar if and only if it does not contain a subgraph that can be contracted to K_5 or $K_{3,3}$ via a finite sequence of edge contractions.

Theorem 300 (Hopcroft&Tarjan (1974))

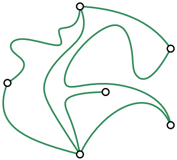
Testing whether a given graph with n vertices is planar can be done in $O(n)$ time.

Theorem 300 (Hopcroft&Tarjan (1974))

Testing whether a given graph with n vertices is planar can be done in $O(n)$ time.

Theorem 301 (Wagner (1936), Fáry (1948), Stein (1951))

Any planar graph can be embedded into the plane without edge crossings such that all its edges are straight-line segments:

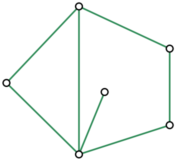


Theorem 300 (Hopcroft&Tarjan (1974))

Testing whether a given graph with n vertices is planar can be done in $O(n)$ time.

Theorem 301 (Wagner (1936), Fáry (1948), Stein (1951))

Any planar graph can be embedded into the plane without edge crossings such that all its edges are straight-line segments: *planar straight-line graph* (PSLG).



Theorem 300 (Hopcroft&Tarjan (1974))

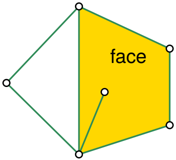
Testing whether a given graph with n vertices is planar can be done in $O(n)$ time.

Theorem 301 (Wagner (1936), Fáry (1948), Stein (1951))

Any planar graph can be embedded into the plane without edge crossings such that all its edges are straight-line segments: *planar straight-line graph* (PSLG).

Definition 302 (Planar subdivision, Dt.: planare Unterteilung)

A *face* of a PSLG embedding of a planar graph is a maximal connected region of the plane that is disjoint from all edges. The embedding of the graph together with the collection of faces induced is called *planar subdivision*.



Theorem 300 (Hopcroft&Tarjan (1974))

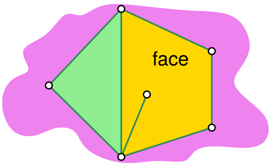
Testing whether a given graph with n vertices is planar can be done in $O(n)$ time.

Theorem 301 (Wagner (1936), Fáry (1948), Stein (1951))

Any planar graph can be embedded into the plane without edge crossings such that all its edges are straight-line segments: *planar straight-line graph* (PSLG).

Definition 302 (Planar subdivision, Dt.: planare Unterteilung)

A *face* of a PSLG embedding of a planar graph is a maximal connected region of the plane that is disjoint from all edges. The embedding of the graph together with the collection of faces induced is called *planar subdivision*.



- Note that one of the faces of a planar subdivision is unbounded: *outer face*.

Theorem 303 (Euler, Dt.: Eulerscher Polyedersatz)

Consider a planar subdivision induced by a connected planar graph $\mathcal{G}$. We denote

- the number of its vertices by v,
- the number of its edges by e,
- the number of its faces by f.

Then

$$v - e + f = 2.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Theorem 303 (Euler, Dt.: Eulerscher Polyedersatz)

Consider a planar subdivision induced by a connected planar graph $\mathcal{G}$. We denote

- the number of its vertices by v,
- the number of its edges by e,
- the number of its faces by f.

Then

$$v - e + f = 2.$$

Proof: Suppose that $\mathcal{G}$ is connected but no tree. Therefore $\mathcal{G}$ contains a cycle, and we may remove an edge from $\mathcal{G}$ without destroying its connectivity. The removal of one edge of a cycle decreases both e and f by one, implying that the value of $v - e + f$ does not change.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Euler's Formula for Planar Graphs

Theorem 303 (Euler, Dt.: Eulerscher Polyedersatz)

Consider a planar subdivision induced by a connected planar graph $\mathcal{G}$. We denote

- the number of its vertices by v,
- the number of its edges by e,
- the number of its faces by f.

Then

$$v - e + f = 2.$$

Proof: Suppose that $\mathcal{G}$ is connected but no tree. Therefore $\mathcal{G}$ contains a cycle, and we may remove an edge from $\mathcal{G}$ without destroying its connectivity. The removal of one edge of a cycle decreases both e and f by one, implying that the value of $v - e + f$ does not change. By using induction we can prove that a series of such edge removals (for breaking up cycles) does not change the value of $v - e + f$, while allowing us to transform $\mathcal{G}$ into a tree.

Euler's Formula for Planar Graphs

Theorem 303 (Euler, Dt.: Eulerscher Polyedersatz)

Consider a planar subdivision induced by a connected planar graph $\mathcal{G}$. We denote

- the number of its vertices by v,
- the number of its edges by e,
- the number of its faces by f.

Then

$$v - e + f = 2.$$

Proof: Suppose that $\mathcal{G}$ is connected but no tree. Therefore $\mathcal{G}$ contains a cycle, and we may remove an edge from $\mathcal{G}$ without destroying its connectivity. The removal of one edge of a cycle decreases both e and f by one, implying that the value of $v - e + f$ does not change. By using induction we can prove that a series of such edge removals (for breaking up cycles) does not change the value of $v - e + f$, while allowing us to transform $\mathcal{G}$ into a tree.
If, however, $\mathcal{G}$ is a tree then Thm. 262 tells us that $1 = v - e$. Since $f = 1$, we get $2 = v - e + f$, thus establishing the claim. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Euler's Formula for Planar Graphs

Theorem 303 (Euler, Dt.: Eulerscher Polyedersatz)

Consider a planar subdivision induced by a connected planar graph $\mathcal{G}$. We denote

- the number of its vertices by v,
- the number of its edges by e,
- the number of its faces by f.

Then

$$v - e + f = 2.$$

Proof: Suppose that $\mathcal{G}$ is connected but no tree. Therefore $\mathcal{G}$ contains a cycle, and we may remove an edge from $\mathcal{G}$ without destroying its connectivity. The removal of one edge of a cycle decreases both e and f by one, implying that the value of $v - e + f$ does not change. By using induction we can prove that a series of such edge removals (for breaking up cycles) does not change the value of $v - e + f$, while allowing us to transform $\mathcal{G}$ into a tree.
If, however, $\mathcal{G}$ is a tree then Thm. 262 tells us that $1 = v - e$. Since $f = 1$, we get $2 = v - e + f$, thus establishing the claim. □

- Euler's Formula generalizes to $v - e + f = 1 + c$ for a planar graph with c connected components.

Corollary 304

Let v, e, f for a connected planar graph $\mathcal{G}$ as defined in Theorem 303. If $v \geqslant 3$ then

$$e \leqslant 3v - 6 \quad \text{and} \quad f \leqslant 2v - 4 \quad \text{and} \quad f \leqslant \frac{2}{3}e.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Corollary 304

Let v, e, f for a connected planar graph $\mathcal{G}$ as defined in Theorem 303. If $v \geqslant 3$ then

$$e \leqslant 3v - 6 \quad \text{and} \quad f \leqslant 2v - 4 \quad \text{and} \quad f \leqslant \frac{2}{3}e.$$

If every vertex of $\mathcal{G}$ has a degree of three or greater then we get

$$v \leqslant \frac{2}{3}e \quad \text{and} \quad e \leqslant 3f - 6 \quad \text{and} \quad v \leqslant 2f - 4.$$

Corollary 304

Let v, e, f for a connected planar graph $\mathcal{G}$ as defined in Theorem 303. If $v \geqslant 3$ then

$$e \leqslant 3v - 6 \quad \text{and} \quad f \leqslant 2v - 4 \quad \text{and} \quad f \leqslant \frac{2}{3}e.$$

If every vertex of $\mathcal{G}$ has a degree of three or greater then we get

$$v \leqslant \frac{2}{3}e \quad \text{and} \quad e \leqslant 3f - 6 \quad \text{and} \quad v \leqslant 2f - 4.$$

Furthermore, every planar graph contains one node with degree at most five.

Corollary 304

Let v, e, f for a connected planar graph $\mathcal{G}$ as defined in Theorem 303. If $v \geqslant 3$ then

$$e \leqslant 3v - 6 \quad \text{and} \quad f \leqslant 2v - 4 \quad \text{and} \quad f \leqslant \frac{2}{3}e.$$

If every vertex of $\mathcal{G}$ has a degree of three or greater then we get

$$v \leqslant \frac{2}{3}e \quad \text{and} \quad e \leqslant 3f - 6 \quad \text{and} \quad v \leqslant 2f - 4.$$

Furthermore, every planar graph contains one node with degree at most five.

Proof: We prove that $3f \leqslant 2e$, which is obvious if $f = 1$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Corollary 304

Let v, e, f for a connected planar graph $\mathcal{G}$ as defined in Theorem 303. If $v \geqslant 3$ then

$$e \leqslant 3v - 6 \quad \text{and} \quad f \leqslant 2v - 4 \quad \text{and} \quad f \leqslant \frac{2}{3}e.$$

If every vertex of $\mathcal{G}$ has a degree of three or greater then we get

$$v \leqslant \frac{2}{3}e \quad \text{and} \quad e \leqslant 3f - 6 \quad \text{and} \quad v \leqslant 2f - 4.$$

Furthermore, every planar graph contains one node with degree at most five.

Proof: We prove that $3f \leqslant 2e$, which is obvious if $f = 1$. We call an edge a "side" of a face if the edge is in the boundary of the face. Let k denote the total number of sides.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Corollary 304

Let v, e, f for a connected planar graph $\mathcal{G}$ as defined in Theorem 303. If $v \geqslant 3$ then

$$e \leqslant 3v - 6 \quad \text{and} \quad f \leqslant 2v - 4 \quad \text{and} \quad f \leqslant \frac{2}{3}e.$$

If every vertex of $\mathcal{G}$ has a degree of three or greater then we get

$$v \leqslant \frac{2}{3}e \quad \text{and} \quad e \leqslant 3f - 6 \quad \text{and} \quad v \leqslant 2f - 4.$$

Furthermore, every planar graph contains one node with degree at most five.

Proof: We prove that $3f \leqslant 2e$, which is obvious if $f = 1$. We call an edge a "side" of a face if the edge is in the boundary of the face. Let k denote the total number of sides. If $f > 1$ then each face is bounded by at least three sides, so $k \geqslant 3f$.

Corollary 304

Let v, e, f for a connected planar graph $\mathcal{G}$ as defined in Theorem 303. If $v \geqslant 3$ then

$$e \leqslant 3v - 6 \quad \text{and} \quad f \leqslant 2v - 4 \quad \text{and} \quad f \leqslant \frac{2}{3}e.$$

If every vertex of $\mathcal{G}$ has a degree of three or greater then we get

$$v \leqslant \frac{2}{3}e \quad \text{and} \quad e \leqslant 3f - 6 \quad \text{and} \quad v \leqslant 2f - 4.$$

Furthermore, every planar graph contains one node with degree at most five.

Proof: We prove that $3f \leqslant 2e$, which is obvious if $f = 1$. We call an edge a “side” of a face if the edge is in the boundary of the face. Let k denote the total number of sides.
If $f > 1$ then each face is bounded by at least three sides, so $k \geqslant 3f$.
But each edge has at most two different sides, so $k \leqslant 2e$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Corollary 304

Let v, e, f for a connected planar graph $\mathcal{G}$ as defined in Theorem 303. If $v \geqslant 3$ then

$$e \leqslant 3v - 6 \quad \text{and} \quad f \leqslant 2v - 4 \quad \text{and} \quad f \leqslant \frac{2}{3}e.$$

If every vertex of $\mathcal{G}$ has a degree of three or greater then we get

$$v \leqslant \frac{2}{3}e \quad \text{and} \quad e \leqslant 3f - 6 \quad \text{and} \quad v \leqslant 2f - 4.$$

Furthermore, every planar graph contains one node with degree at most five.

Proof : We prove that $3f \leqslant 2e$, which is obvious if $f = 1$. We call an edge a "side" of a face if the edge is in the boundary of the face. Let k denote the total number of sides. If $f > 1$ then each face is bounded by at least three sides, so $k \geqslant 3f$.
But each edge has at most two different sides, so $k \leqslant 2e$.
We conclude $3f \leqslant 2e$. □

Euler's Formula for Planar Graphs

Corollary 305

K_5 is not planar.

Corollary 305

K_5 is not planar.

Proof: We get $v = 5$ and $e = \binom{5}{2} = 10$. So, $e \leqslant 3v - 6$ (Cor. 304) does not hold. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Corollary 305

K_5 is not planar.

Proof: We get $v = 5$ and $e = \binom{5}{2} = 10$. So, $e \leqslant 3v - 6$ (Cor. 304) does not hold. □

Definition 306 (Triangle-free, Dt.: dreiecksfrei)

A *triangle-free graph* is a graph which does not contain a cycle of length three, i.e., in which no three vertices form a triangle of edges.

Corollary 305

K_5 is not planar.

Proof: We get $v = 5$ and $e = \binom{5}{2} = 10$. So, $e \leqslant 3v - 6$ (Cor. 304) does not hold. □

Definition 306 (Triangle-free, Dt.: dreiecksfrei)

A *triangle-free graph* is a graph which does not contain a cycle of length three, i.e., in which no three vertices form a triangle of edges.

Corollary 307

A triangle-free planar graph has one node of degree at most three and $e \leqslant 2v - 4$ holds (if $v \geqslant 3$).

Corollary 305

K_5 is not planar.

Proof: We get $v = 5$ and $e = \binom{5}{2} = 10$. So, $e \leqslant 3v - 6$ (Cor. 304) does not hold. □

Definition 306 (Triangle-free, Dt.: dreiecksfrei)

A *triangle-free graph* is a graph which does not contain a cycle of length three, i.e., in which no three vertices form a triangle of edges.

Corollary 307

A triangle-free planar graph has one node of degree at most three and $e \leqslant 2v - 4$ holds (if $v \geqslant 3$).

Corollary 308

$K_{3,3}$ is not planar.

Euler's Formula for Planar Graphs

Corollary 305

K_5 is not planar.

Proof: We get $v = 5$ and $e = \binom{5}{2} = 10$. So, $e \leqslant 3v - 6$ (Cor. 304) does not hold. □

Definition 306 (Triangle-free, Dt.: dreiecksfrei)

A *triangle-free graph* is a graph which does not contain a cycle of length three, i.e., in which no three vertices form a triangle of edges.

Corollary 307

A triangle-free planar graph has one node of degree at most three and $e \leqslant 2v - 4$ holds (if $v \geqslant 3$).

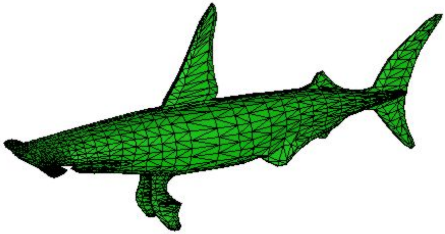
Corollary 308

$K_{3,3}$ is not planar.

Proof: $K_{3,3}$ is triangle-free and has six vertices and nine edges. If it were planar then, by Cor. 307, it could have at most $2 \cdot 6 - 4 = 8$ edges. Thus, $K_{3,3}$ is non-planar.

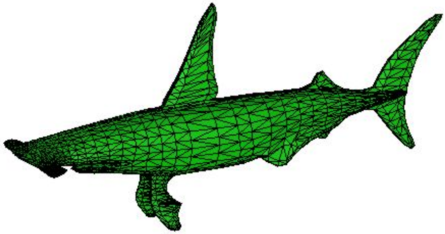
Real-World Application: Number of Edges of a Polyhedron

- Suppose that a polyhedral model has n vertices. How many edges and faces can it have at most? What is the storage complexity relative to n?



Real-World Application: Number of Edges of a Polyhedron

- Suppose that a polyhedral model has n vertices. How many edges and faces can it have at most? What is the storage complexity relative to n?

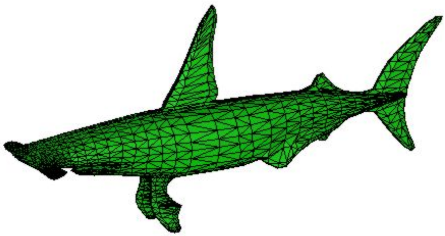


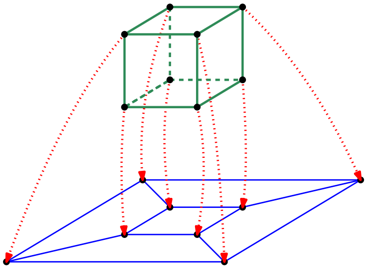
Theorem 309

The vertices and edges of a simple (bounded) polyhedron form a planar graph.

Real-World Application: Number of Edges of a Polyhedron

- Suppose that a polyhedral model has n vertices. How many edges and faces can it have at most? What is the storage complexity relative to n?



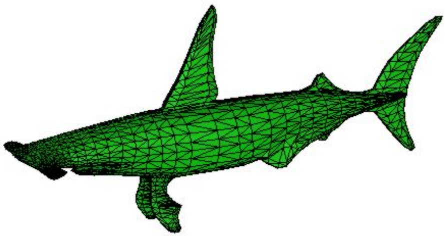


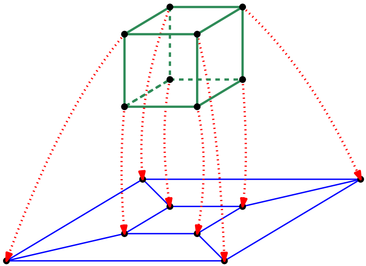
Theorem 309

The vertices and edges of a simple (bounded) polyhedron form a planar graph.

Real-World Application: Number of Edges of a Polyhedron

- Suppose that a polyhedral model has n vertices. How many edges and faces can it have at most? What is the storage complexity relative to n?





Theorem 309

The vertices and edges of a simple (bounded) polyhedron form a planar graph.

Corollary 310

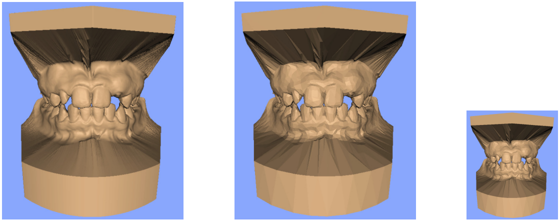
A simple (bounded) polyhedron with n vertices has at most $3n - 6$ edges and $2n - 4$ faces.

Real-World Application: Reducing the Face Count

- Recent improvements in laser rangefinder technology allow the digitization of the shapes of physical objects at extremely high resolutions.
- The resulting polyhedral models are huge: E.g., a 0.25 mm model of Michelangelo's 5-meter statue of David contains about 1 billion polygonal faces!

Real-World Application: Reducing the Face Count

- Recent improvements in laser rangefinder technology allow the digitization of the shapes of physical objects at extremely high resolutions.
- The resulting polyhedral models are huge: E.g., a 0.25 mm model of Michelangelo's 5-meter statue of David contains about 1 billion polygonal faces!
- Goal of multi-resolution modeling and level-of-detail modeling: Reduce the face count without sacrificing the visual appearance.
- E.g., the left dental model has 424 376 faces, while the other two models have only a few thousand faces.

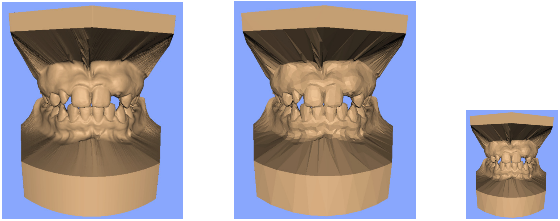


[Image credit: Michael Garland, Eurographics'99 STAR]

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Reducing the Face Count

- Recent improvements in laser rangefinder technology allow the digitization of the shapes of physical objects at extremely high resolutions.
- The resulting polyhedral models are huge: E.g., a 0.25 mm model of Michelangelo's 5-meter statue of David contains about 1 billion polygonal faces!
- Goal of multi-resolution modeling and level-of-detail modeling: Reduce the face count without sacrificing the visual appearance.
- E.g., the left dental model has 424 376 faces, while the other two models have only a few thousand faces.
- Edge contraction is one of the techniques used for reducing the face count.



[Image credit: Michael Garland, Eurographics'99 STAR]

7 Graph Theory

Definition 311 (Coloring, Dt.: Färbung)

An assignment of colors to all vertices of a graph $\mathcal{G}$ is called a *(vertex) coloring* if adjacent vertices are assigned different colors.

Graph Coloring

Definition 311 (Coloring, Dt.: Färbung)

An assignment of colors to all vertices of a graph $\mathcal{G}$ is called a *(vertex) coloring* if adjacent vertices are assigned different colors.

Definition 312 (k-colorable, Dt.: k-färbbar)

A graph $\mathcal{G}$ is *k-colorable* if k colors suffice to establish a coloring of $\mathcal{G}$.

Graph Coloring

Definition 311 (Coloring, Dt.: Färbung)

An assignment of colors to all vertices of a graph $\mathcal{G}$ is called a *(vertex) coloring* if adjacent vertices are assigned different colors.

Definition 312 (k-colorable, Dt.: k-färbbar)

A graph $\mathcal{G}$ is *k-colorable* if k colors suffice to establish a coloring of $\mathcal{G}$.

Definition 313 (Chromatic number, Dt.: chromatische Zahl)

The *chromatic number* of a graph $\mathcal{G}$, written as $\chi(\mathcal{G})$, is the least number of colors required to color $\mathcal{G}$.

Graph Coloring

Definition 311 (Coloring, Dt.: Färbung)

An assignment of colors to all vertices of a graph $\mathcal{G}$ is called a *(vertex) coloring* if adjacent vertices are assigned different colors.

Definition 312 (k-colorable, Dt.: k-färbbar)

A graph $\mathcal{G}$ is *k-colorable* if k colors suffice to establish a coloring of $\mathcal{G}$.

Definition 313 (Chromatic number, Dt.: chromatische Zahl)

The *chromatic number* of a graph $\mathcal{G}$, written as $\chi(\mathcal{G})$, is the least number of colors required to color $\mathcal{G}$.

- $\chi(K_n) = n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Definition 311 (Coloring, Dt.: Färbung)

An assignment of colors to all vertices of a graph $\mathcal{G}$ is called a *(vertex) coloring* if adjacent vertices are assigned different colors.

Definition 312 (k-colorable, Dt.: k-färbbar)

A graph $\mathcal{G}$ is *k-colorable* if k colors suffice to establish a coloring of $\mathcal{G}$.

Definition 313 (Chromatic number, Dt.: chromatische Zahl)

The *chromatic number* of a graph $\mathcal{G}$, written as $\chi(\mathcal{G})$, is the least number of colors required to color $\mathcal{G}$.

- $\chi(K_n) = n$.

Lemma 314

The chromatic number of a graph $\mathcal{G}$ is two if and only if $\mathcal{G}$ is bipartite.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Graph Coloring

- It is straightforward that every planar graph can be colored by six colors and that every triangle-free planar graph can be colored by four colors.

Graph Coloring

- It is straightforward that every planar graph can be colored by six colors and that every triangle-free planar graph can be colored by four colors.
- Still easy to see: Every planar graph can be colored by five colors.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- It is straightforward that every planar graph can be colored by six colors and that every triangle-free planar graph can be colored by four colors.
- Still easy to see: Every planar graph can be colored by five colors.

Theorem 315 (Four Color Theorem, Haken and Appel (1976))

Every planar graph can be colored using no more than four colors.

- It is straightforward that every planar graph can be colored by six colors and that every triangle-free planar graph can be colored by four colors.
- Still easy to see: Every planar graph can be colored by five colors.

Theorem 315 (Four Color Theorem, Haken and Appel (1976))

Every planar graph can be colored using no more than four colors.

- Haken and Appel used a super-computer at the University of Illinois to check 1936 “reducible” configurations. The proof is not accepted by all mathematicians as it has two parts, one of which can only be solved using computers. (And the second part that is solveable by hand is also very tedious.)

Graph Coloring

- It is straightforward that every planar graph can be colored by six colors and that every triangle-free planar graph can be colored by four colors.
- Still easy to see: Every planar graph can be colored by five colors.

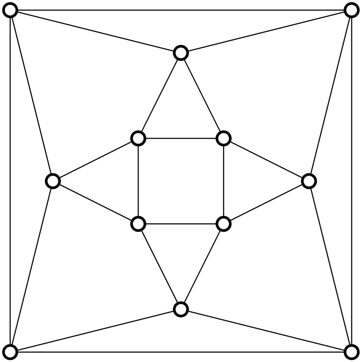
Theorem 315 (Four Color Theorem, Haken and Appel (1976))

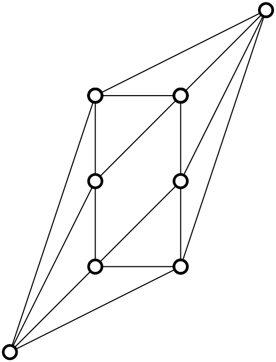
Every planar graph can be colored using no more than four colors.

- Haken and Appel used a super-computer at the University of Illinois to check 1936 "reducible" configurations. The proof is not accepted by all mathematicians as it has two parts, one of which can only be solved using computers. (And the second part that is solveable by hand is also very tedious.)
- In 1996, Robertson et alii reduced the number of computer-checked cases to 633.
- In 2005, Werner and Gonthier used a general-purpose proof assistant ("Coq") to prove the theorem.

Graph Coloring of 4-Regular Planar Graphs

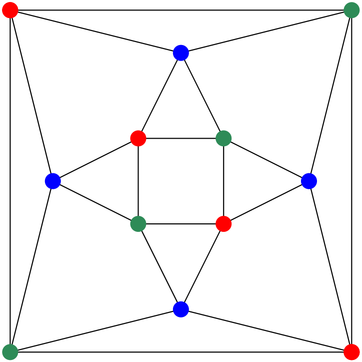
- Determining $\chi(\mathcal{G})$ is $\mathcal{NP}$-hard even if $\mathcal{G}$ is a planar 4-regular graph!

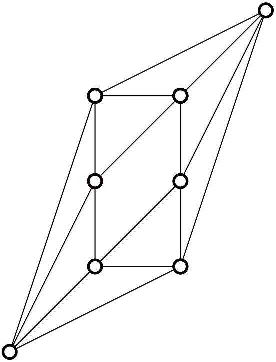




Graph Coloring of 4-Regular Planar Graphs

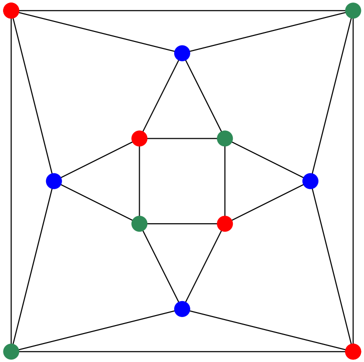
- Determining $\chi(\mathcal{G})$ is $\mathcal{NP}$-hard even if $\mathcal{G}$ is a planar 4-regular graph!

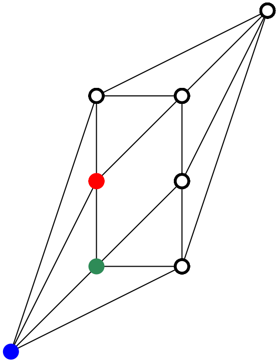




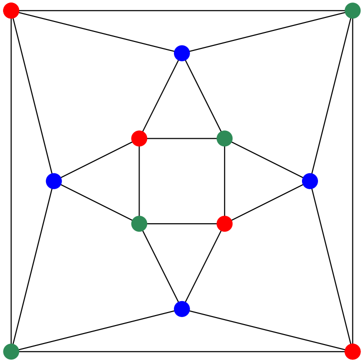
Graph Coloring of 4-Regular Planar Graphs

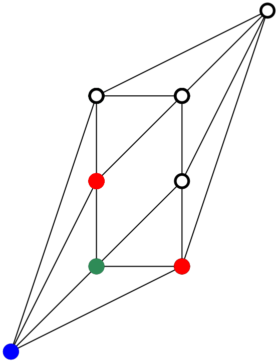
- Determining $\chi(\mathcal{G})$ is $\mathcal{NP}$-hard even if $\mathcal{G}$ is a planar 4-regular graph!





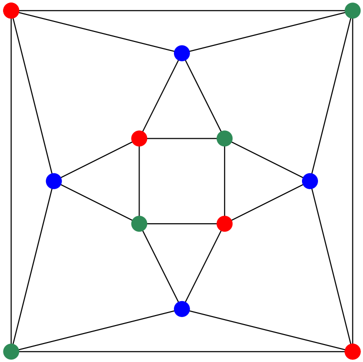
- Determining $\chi(\mathcal{G})$ is $\mathcal{NP}$-hard even if $\mathcal{G}$ is a planar 4-regular graph!

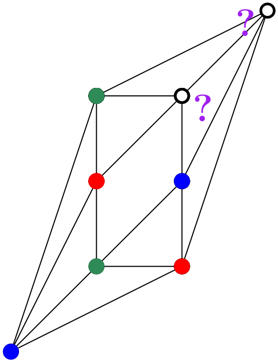




Graph Coloring of 4-Regular Planar Graphs

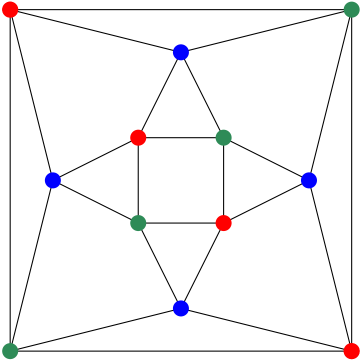
- Determining $\chi(\mathcal{G})$ is $\mathcal{NP}$-hard even if $\mathcal{G}$ is a planar 4-regular graph!

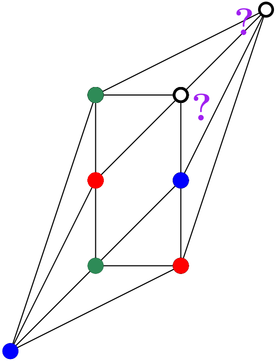




Graph Coloring of 4-Regular Planar Graphs

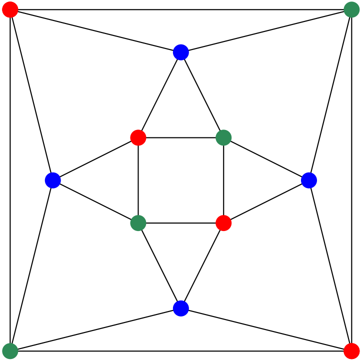
- Determining $\chi(\mathcal{G})$ is $\mathcal{NP}$-hard even if $\mathcal{G}$ is a planar 4-regular graph!
- Thus, it is rather unlikely that a polynomial-time algorithm will ever be found for determining $\chi(\mathcal{G})$.

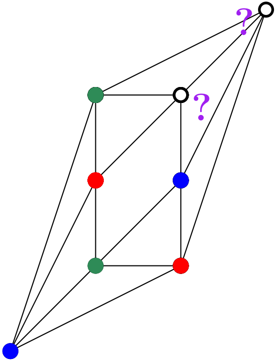




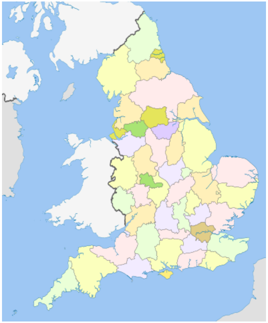
Graph Coloring of 4-Regular Planar Graphs

- Determining $\chi(\mathcal{G})$ is $\mathcal{NP}$-hard even if $\mathcal{G}$ is a planar 4-regular graph!
- Thus, it is rather unlikely that a polynomial-time algorithm will ever be found for determining $\chi(\mathcal{G})$.
- However, fairly efficient heuristics exist for approximate graph coloring.





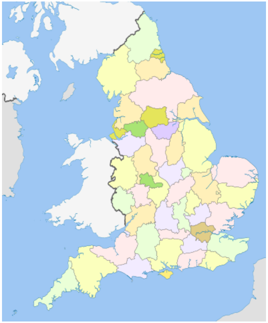
Graph Coloring and Topographic Maps in a Plane



[Image credit: Wikipedia]

- In 1852, Francis Guthrie noticed that the map of the English counties could be colored with only four colors.

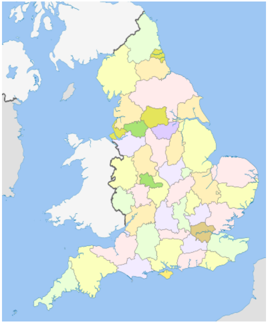
Graph Coloring and Topographic Maps in a Plane



[Image credit: Wikipedia]

- In 1852, Francis Guthrie noticed that the map of the English counties could be colored with only four colors.
- Subsequent attempts for prove the Four Color Theorem by de Morgan and Cayley.
- In 1879, Kempe released an alleged proof that was understood to be incorrect eleven years later. But his work provided the ideas for Haken and Appel.

Graph Coloring and Topographic Maps in a Plane



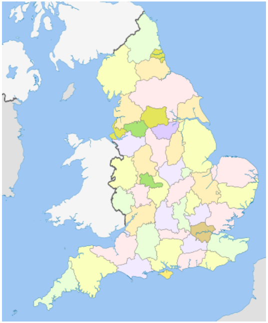
[Image credit: Wikipedia]

- In 1852, Francis Guthrie noticed that the map of the English counties could be colored with only four colors.
- Subsequent attempts for prove the Four Color Theorem by de Morgan and Cayley.
- In 1879, Kempe released an alleged proof that was understood to be incorrect eleven years later. But his work provided the ideas for Haken and Appel.

Corollary 316

If every entity of a topographic map is a connected area then four colors suffice to color the map such that no two entities that share a common border (other than a common point) are colored with the same color.

Graph Coloring and Topographic Maps in a Plane



[Image credit: Wikipedia]

- In 1852, Francis Guthrie noticed that the map of the English counties could be colored with only four colors.
- Subsequent attempts for prove the Four Color Theorem by de Morgan and Cayley.
- In 1879, Kempe released an alleged proof that was understood to be incorrect eleven years later. But his work provided the ideas for Haken and Appel.

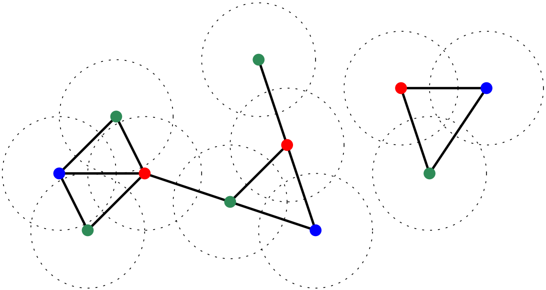
Corollary 316

If every entity of a topographic map is a connected area then four colors suffice to color the map such that no two entities that share a common border (other than a common point) are colored with the same color.

- Note that this result holds only in the plane! E.g., on the surface of a torus seven colors are sufficient and may be necessary.

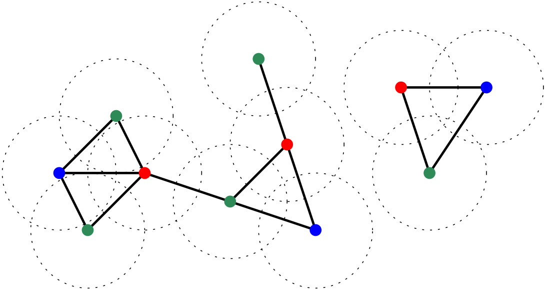
Real-World Application: Channel Assignment

- We can solve the channel assignment problem by considering its so-called unit-disk graph, where
  - the vertices are given by the broadcast stations,
  - two vertices are connected by an edge if their service areas overlap.



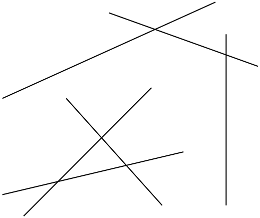
Real-World Application: Channel Assignment

- We can solve the channel assignment problem by considering its so-called unit-disk graph, where
  - the vertices are given by the broadcast stations,
  - two vertices are connected by an edge if their service areas overlap.
- Obviously, the chromatic number of that graph equals the minimum number of frequencies needed.



Real-World Application: Minimum Plane Partition

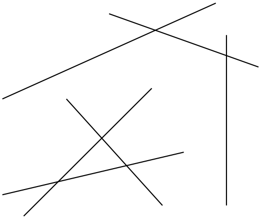
- CG:SHOP Geometric Optimization Challenge 2022: Given is a set S of line segments in the plane.



UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

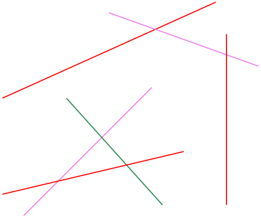
Real-World Application: Minimum Plane Partition

- CG:SHOP Geometric Optimization Challenge 2022: Given is a set S of line segments in the plane.
- We seek a partitioning of S into a minimum number of k subsets $S_1, \ldots, S_k$ such that, for all $1 \leqslant i \leqslant k$, the line segments of S_i do not intersect pairwise.



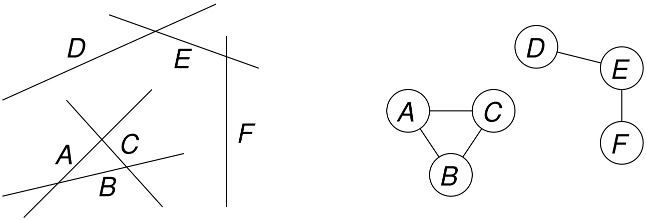
Real-World Application: Minimum Plane Partition

- CG:SHOP Geometric Optimization Challenge 2022: Given is a set S of line segments in the plane.
- We seek a partitioning of S into a minimum number of k subsets $S_1, \ldots, S_k$ such that, for all $1 \leqslant i \leqslant k$, the line segments of S_i do not intersect pairwise.



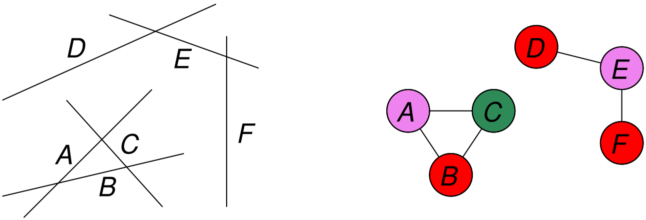
Real-World Application: Minimum Plane Partition

- CG:SHOP Geometric Optimization Challenge 2022: Given is a set S of line segments in the plane.
- We seek a partitioning of S into a minimum number of k subsets $S_1, \dots, S_k$ such that, for all $1 \leqslant i \leqslant k$, the line segments of S_i do not intersect pairwise.
- An obvious attempt to solve this problem is to construct the conflict graph $\mathcal{G}$ for S and then apply graph coloring to $\mathcal{G}$.

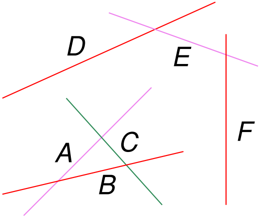


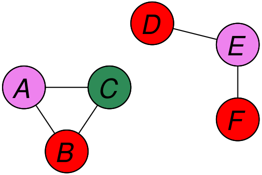
Real-World Application: Minimum Plane Partition

- CG:SHOP Geometric Optimization Challenge 2022: Given is a set S of line segments in the plane.
- We seek a partitioning of S into a minimum number of k subsets $S_1, \ldots, S_k$ such that, for all $1 \leqslant i \leqslant k$, the line segments of S_i do not intersect pairwise.
- An obvious attempt to solve this problem is to construct the conflict graph $\mathcal{G}$ for S and then apply graph coloring to $\mathcal{G}$.



- CG:SHOP Geometric Optimization Challenge 2022: Given is a set S of line segments in the plane.
- We seek a partitioning of S into a minimum number of k subsets $S_1, \ldots, S_k$ such that, for all $1 \leqslant i \leqslant k$, the line segments of S_i do not intersect pairwise.
- An obvious attempt to solve this problem is to construct the conflict graph $\mathcal{G}$ for S and then apply graph coloring to $\mathcal{G}$.





Real-World Application: Index Registers

- Optimizing compilers try to store frequently used variables of the body of a loop in index registers of the CPU (rather than in regular memory).
- How many index registers are needed for a given loop?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Real-World Application: Index Registers

- Optimizing compilers try to store frequently used variables of the body of a loop in index registers of the CPU (rather than in regular memory).
- How many index registers are needed for a given loop?
- We set up a graph whose vertices are given by the variables, and where two vertices are connected by an edge if the corresponding variables ought to be kept in registers at the same time.

Real-World Application: Index Registers

- Optimizing compilers try to store frequently used variables of the body of a loop in index registers of the CPU (rather than in regular memory).
- How many index registers are needed for a given loop?
- We set up a graph whose vertices are given by the variables, and where two vertices are connected by an edge if the corresponding variables ought to be kept in registers at the same time.
- Then the chromatic number of that graph gives the minimum number of registers needed.

Real-World Application: Index Registers

- Optimizing compilers try to store frequently used variables of the body of a loop in index registers of the CPU (rather than in regular memory).
- How many index registers are needed for a given loop?
- We set up a graph whose vertices are given by the variables, and where two vertices are connected by an edge if the corresponding variables ought to be kept in registers at the same time.
- Then the chromatic number of that graph gives the minimum number of registers needed.

- Other applications of graph coloring:
  - Scheduling consumer-producer interactions to allow concurrency.
  - Sudoku puzzles.

8 **Cryptography**

- Introduction
- Symmetric-Key Cryptography
- Public-Key Cryptography

8 Cryptography

- Introduction
- Symmetric-Key Cryptography
- Public-Key Cryptography

Introduction — What is Cryptography?

- Cryptography is the science of sending and receiving messages in secret code.
- A *sender A* (“Alice”) sends an encoded message to a *receiver B* (“Bob”).
- The goal is to keep the transmission of the message secure (from others to read it) and to ensure successful communication of the information.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Introduction — What is Cryptography?

- Cryptography is the science of sending and receiving messages in secret code.
- A *sender A* (“Alice”) sends an encoded message to a *receiver B* (“Bob”).
- The goal is to keep the transmission of the message secure (from others to read it) and to ensure successful communication of the information.
- Cryptography has been used for at least 2500 years:
  - The use of invisible ink and, more generally, steganography can be traced back to 440 BCE, due to writings by Aeneas Tacticus and Herodotos.
  - Caesar used an encryption scheme for military communication.

Introduction — What is Cryptography?

- Cryptography is the science of sending and receiving messages in secret code.
- A *sender A* ("Alice") sends an encoded message to a *receiver B* ("Bob").
- The goal is to keep the transmission of the message secure (from others to read it) and to ensure successful communication of the information.
- Cryptography has been used for at least 2500 years:
  - The use of invisible ink and, more generally, steganography can be traced back to 440 BCE, due to writings by Aeneas Tacticus and Herodotos.
  - Caesar used an encryption scheme for military communication.

Two main schemes in use nowadays

Symmetric-Key Cryptography (SKC): The same secret key is used for both encryption and decryption; aka secret-key cryptography.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Cryptography is the science of sending and receiving messages in secret code.
- A *sender A* (“Alice”) sends an encoded message to a *receiver B* (“Bob”).
- The goal is to keep the transmission of the message secure (from others to read it) and to ensure successful communication of the information.
- Cryptography has been used for at least 2500 years:
    - The use of invisible ink and, more generally, steganography can be traced back to 440 BCE, due to writings by Aeneas Tacticus and Herodotos.
    - Caesar used an encryption scheme for military communication.

Two main schemes in use nowadays

Symmetric-Key Cryptography (SKC): The same secret key is used for both encryption and decryption; aka secret-key cryptography.

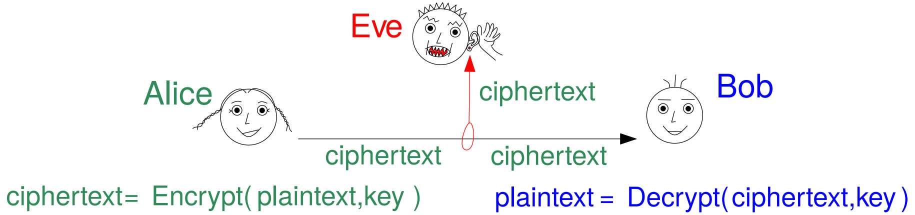
Public-Key Cryptography (PKC): Different keys are used for encryption and decryption, with some keys being known publicly; aka asymmetric-key cryptography.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

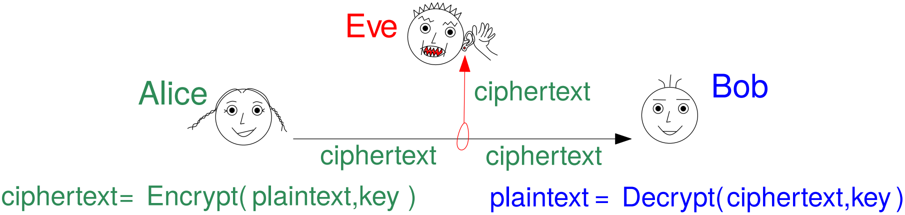
Basic Terms

- *Plaintext* — original message.
- *Ciphertext* — encoded/encrypted message.
- *Encryption* — generating ciphertext from plaintext.
- *Decryption / Deciphering* — generating plaintext from ciphertext.
- *Cryptanalysis* — trying to break the encryption by applying various methods.
- *Adversary, Spy* — the message thief.
- *Eavesdropper* — a secret listener who listens to private conversations.
- *Authentication* — the process of proving one's identity.
- *Privacy* — ensuring that the message is read only by the intended receiver. (GnuPG: "Privacy is not a crime!")

Eavesdropper Attacks



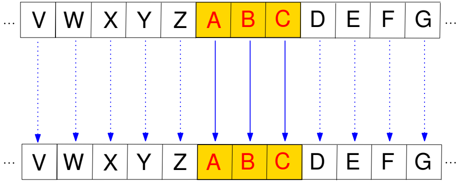
Eavesdropper Attacks



- Eve might attempt to
  - break the encryption,
  - replay the encrypted message (e.g., login) without breaking the encryption,
  - modify the message,
  - block the message,
  - fabricate a new message.

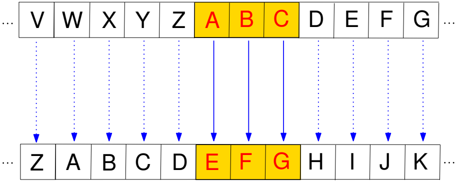
Classical Cryptography: Caesar's Cipher

- According to Suetonius, Caesar (100–44 BCE) used an encryption scheme (for communication with his generals) that shifted the alphabet of the plaintext by some fixed position value *n*.



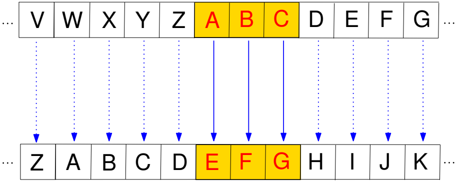
Classical Cryptography: Caesar's Cipher

- According to Suetonius, Caesar (100–44 BCE) used an encryption scheme (for communication with his generals) that shifted the alphabet of the plaintext by some fixed position value *n*.



Classical Cryptography: Caesar's Cipher

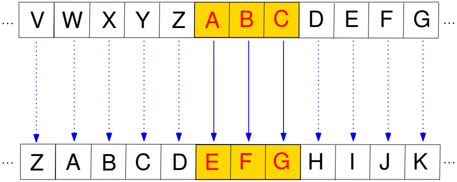
- According to Suetonius, Caesar (100–44 BCE) used an encryption scheme (for communication with his generals) that shifted the alphabet of the plaintext by some fixed position value n.



- With $n := 4$:
  Plaintext: `alea iacta est`
  Ciphertext: `epie megxe iwx`

Classical Cryptography: Caesar's Cipher

- According to Suetonius, Caesar (100–44 BCE) used an encryption scheme (for communication with his generals) that shifted the alphabet of the plaintext by some fixed position value n.



- With $n := 4$:
  Plaintext: `alea iacta est`
  Ciphertext: `epie megxe iwx`
- Suppose that the (Roman) letters are mapped to the numbers $0, 1, \ldots, 25$.
- Then Caesar's encryption and decryption with shift n can be computed as follows:

$$\begin{aligned} \textit{ciphertext} &:= \text{Encrypt}_n(\textit{plaintext}) &= (\textit{plaintext} + n) \bmod 26 \\ \textit{plaintext} &:= \text{Decrypt}_n(\textit{ciphertext}) &= (\textit{ciphertext} - n) \bmod 26 \end{aligned}$$

Classical Cryptography: Caesar's Cipher

- Likely, Caesar's cipher was reasonably secure at the time when it was used.

Classical Cryptography: Caesar's Cipher

- Likely, Caesar's cipher was reasonably secure at the time when it was used.
- It is broken easily by means of frequency analysis and brute-force attacks — it offers no security by today's standards!
- Nevertheless, Caesar's cipher with $n := 13$, aka ROT13, has been (mis-)used for serious applications even rather recently.
- However, it is used within more complex systems, e.g., the Vigenère cipher.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Classical Cryptography: Caesar's Cipher

- Likely, Caesar's cipher was reasonably secure at the time when it was used.
- It is broken easily by means of frequency analysis and brute-force attacks — it offers no security by today's standards!
- Nevertheless, Caesar's cipher with $n := 13$, aka ROT13, has been (mis-)used for serious applications even rather recently.
- However, it is used within more complex systems, e.g., the Vigenère cipher.

- On a Unix machine, the `tr` utility can be used for carrying out Caesar's cipher. E.g.,

  ```
  echo "alea iacta est" | tr 'A-Za-z' 'E-ZA-De-za-d'
  ```

  yields

  ```
  epie megxe iwx,
  ```

  and

  ```
  echo "epie megxe iwx" | tr 'E-ZA-De-za-d' 'A-Za-z'
  ```

  recovers the original text

  ```
  alea iacta est.
  ```

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Symmetric-Key Cryptography

- A single secret key is used for both encryption and decryption (aka "*secret-key algorithms*").

- A single secret key is used for both encryption and decryption (aka “*secret-key algorithms*”).

plaintext —key→ ciphertext —key→ plaintext

- Simple example: Suppose that Alice wants to encrypt a bit string A. Then Alice and Bob could choose a secret key B and apply a bit-wise XOR (exclusive OR, $\oplus$) — an output bit is 1 if exactly one of the two input bits is 1 — in order to transmit $A \oplus B$. Then Bob would compute $(A \oplus B) \oplus B$ and, thus, retrieve A.

Symmetric-Key Cryptography

- A single secret key is used for both encryption and decryption (aka "*secret-key algorithms*").

plaintext —key→ ciphertext —key→ plaintext

- Simple example: Suppose that Alice wants to encrypt a bit string A. Then Alice and Bob could choose a secret key B and apply a bit-wise XOR (exclusive OR, $\oplus$) — an output bit is 1 if exactly one of the two input bits is 1 — in order to transmit $A \oplus B$. Then Bob would compute $(A \oplus B) \oplus B$ and, thus, retrieve A.

A	B	$A \oplus B$	$(A \oplus B) \oplus B$
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

Symmetric-Key Cryptography

- A single secret key is used for both encryption and decryption (aka "*secret-key algorithms*").

plaintext $\xrightarrow{\text{key}}$ ciphertext $\xrightarrow{\text{key}}$ plaintext

- Simple example: Suppose that Alice wants to encrypt a bit string A. Then Alice and Bob could choose a secret key B and apply a bit-wise XOR (exclusive OR, $\oplus$) — an output bit is 1 if exactly one of the two input bits is 1 — in order to transmit $A \oplus B$. Then Bob would compute $(A \oplus B) \oplus B$ and, thus, retrieve A.

A	B	$A \oplus B$	$(A \oplus B) \oplus B$
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

- Of course, the key B must be known to both Alice and Bob, and, in fact, it must not be known to anybody else.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Key Distribution Problem

- The key B must be known to both Alice and Bob, and, in fact, it must not be known to anybody else.
- That is, Alice and Bob need to share the secret key in order to be able to encrypt and decrypt their messages!
- What is a secure mechanism for them to exchange a key??

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Key Distribution Problem

- The key B must be known to both Alice and Bob, and, in fact, it must not be known to anybody else.
- That is, Alice and Bob need to share the secret key in order to be able to encrypt and decrypt their messages!
- What is a secure mechanism for them to exchange a key??
  - Meet in person at a secret place and share the key?!
  - Share in parts?!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Key Distribution Problem

- The key B must be known to both Alice and Bob, and, in fact, it must not be known to anybody else.
- That is, Alice and Bob need to share the secret key in order to be able to encrypt and decrypt their messages!
- What is a secure mechanism for them to exchange a key??
  - Meet in person at a secret place and share the key?!
  - Share in parts?!
- The key distribution problem is a major roadblock on the road to secure communication among folks who do not meet regularly.

Key Distribution Problem

- The key B must be known to both Alice and Bob, and, in fact, it must not be known to anybody else.
- That is, Alice and Bob need to share the secret key in order to be able to encrypt and decrypt their messages!
- What is a secure mechanism for them to exchange a key??
  - Meet in person at a secret place and share the key?!
  - Share in parts?!
- The key distribution problem is a major roadblock on the road to secure communication among folks who do not meet regularly.
- A second big disadvantage is the need for multiple keys in order to encrypt messages intended for different receivers.

8 Cryptography

- Introduction
- Symmetric-Key Cryptography
- Public-Key Cryptography
  - Diffie-Hellman Algorithm
  - RSA

Public-Key Cryptography

- A pair of keys is used to encrypt and decrypt the messages, with one key being public.

Public-Key Cryptography

- A pair of keys is used to encrypt and decrypt the messages, with one key being public.

plaintext —key1→ ciphertext —key2→ plaintext

- PKC schemes make use of so-called *one-way functions* which are “easy” to compute for every input but extremely “hard” to invert for an output given ([Jevons 1874]).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Public-Key Cryptography

- A pair of keys is used to encrypt and decrypt the messages, with one key being public.

$$\text{plaintext} \xrightarrow{\text{key1}} \text{ciphertext} \xrightarrow{\text{key2}} \text{plaintext}$$

- PKC schemes make use of so-called *one-way functions* which are "easy" to compute for every input but extremely "hard" to invert for an output given ([Jevons 1874]). For example, consider
  - multiplication versus factorization ("factorization problem"):
    - If $f(a,b) := a \cdot b$, then
      $f(a,b) = 533$ for $a := 13, b := 41$;

Public-Key Cryptography

- A pair of keys is used to encrypt and decrypt the messages, with one key being public.

plaintext $\xrightarrow{\text{key1}}$ ciphertext $\xrightarrow{\text{key2}}$ plaintext

- PKC schemes make use of so-called *one-way functions* which are "easy" to compute for every input but extremely "hard" to invert for an output given ([Jevons 1874]). For example, consider
  - multiplication versus factorization ("factorization problem"):
    - If $f(a, b) := a \cdot b$, then $f(a, b) = 533$ for $a := 13, b := 41$;
    - While you can calculate $f(13, 41)$ in your head, it is less trivial to obtain a, b such that $f(a, b) = 533$;

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Public-Key Cryptography

- A pair of keys is used to encrypt and decrypt the messages, with one key being public.

$$\text{plaintext} \xrightarrow{\text{key1}} \text{ciphertext} \xrightarrow{\text{key2}} \text{plaintext}$$

- PKC schemes make use of so-called *one-way functions* which are "easy" to compute for every input but extremely "hard" to invert for an output given ([Jevons 1874]). For example, consider
  - multiplication versus factorization ("factorization problem"):
    - If $f(a,b) := a \cdot b$, then $f(a,b) = 533$ for $a := 13, b := 41$;
    - While you can calculate $f(13,41)$ in your head, it is less trivial to obtain a, b such that $f(a,b) = 533$;
  - exponentiation versus logarithms ("discrete log problem"):
    - If $f(a,b) := a^b$, then $f(a,b) = 243$ for $a := 3, b := 5$;
    - Again, finding a and b such that $\log_a 243 = b$ is considerably more difficult.

Public-Key Cryptography

- A pair of keys is used to encrypt and decrypt the messages, with one key being public.

$$\text{plaintext} \xrightarrow{\text{key1}} \text{ciphertext} \xrightarrow{\text{key2}} \text{plaintext}$$

- PKC schemes make use of so-called *one-way functions* which are "easy" to compute for every input but extremely "hard" to invert for an output given ([Jevons 1874]). For example, consider
  - multiplication versus factorization ("factorization problem"):
    - If $f(a,b) := a \cdot b$, then $f(a,b) = 533$ for $a := 13, b := 41$;
    - While you can calculate $f(13,41)$ in your head, it is less trivial to obtain a, b such that $f(a,b) = 533$;
  - exponentiation versus logarithms ("discrete log problem"):
    - If $f(a,b) := a^b$, then $f(a,b) = 243$ for $a := 3, b := 5$;
    - Again, finding a and b such that $\log_a 243 = b$ is considerably more difficult.
- Diffie and his advisor Hellman were the first to *publish* a PKC scheme in 1976. (They were the recipients of the 2015 ACM Turing Award.)

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \mod p$.

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \mod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \mod p$ to Bob	sends $T := g^t \mod p$ to Alice

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- We have

$$T^s = (g^t \bmod p)^s$$

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- We have

$$T^s = (g^t \bmod p)^s \equiv_p (g^t)^s$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- We have

$$T^s = (g^t \bmod p)^s \equiv_p (g^t)^s = g^{ts} = (g^s)^t$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- We have

$$T^s = (g^t \bmod p)^s \equiv_p (g^t)^s = g^{ts} = (g^s)^t \equiv_p S^t.$$

Hence, $k := T^s \bmod p = S^t \bmod p$ can be used as a common key by Alice and Bob.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- We have

$$T^s = (g^t \bmod p)^s \equiv_p (g^t)^s = g^{ts} = (g^s)^t \equiv_p S^t.$$

Hence, $k := T^s \bmod p = S^t \bmod p$ can be used as a common key by Alice and Bob.

- In general, the public information is p, g, S and T, while s and t are secret.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- We have

$$T^s = (g^t \bmod p)^s \equiv_p (g^t)^s = g^{ts} = (g^s)^t \equiv_p S^t.$$

  Hence, $k := T^s \bmod p = S^t \bmod p$ can be used as a common key by Alice and Bob.

- In general, the public information is p, g, S and T, while s and t are secret.
- To find s, Eve could attempt to solve the discrete log problem $S = g^s \bmod p$. Same for t. At present, nobody knows how to solve this problem efficiently.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange

- Alice and Bob share two public numbers: a (large) prime number $p \in \mathbb{P}$ and a so-called generator $g \in \{2, 3, \ldots, p-1\}$ such that for every $n \in \{1, 2, \ldots, p-1\}$ there exists a $k \in \mathbb{N}$ with $n = g^k \bmod p$.

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- We have

$$T^s = (g^t \bmod p)^s \equiv_p (g^t)^s = g^{ts} = (g^s)^t \equiv_p S^t.$$

  Hence, $k := T^s \bmod p = S^t \bmod p$ can be used as a common key by Alice and Bob.
- In general, the public information is p, g, S and T, while s and t are secret.
- To find s, Eve could attempt to solve the discrete log problem $S = g^s \bmod p$. Same for t. At present, nobody knows how to solve this problem efficiently.
- Diffie-Hellman key exchange is used by the Tor system to set-up secure communication links with onion routers.
- The Diffie-Hellman key exchange is vulnerable to man-in-the-middle attacks.

Diffie-Hellman Symmetric Key Exchange: Sample

	Alice	Bob
(1)	selects s with $1 < s < p - 1$	selects t with $1 < t < p - 1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- Alice and Bob make $p := 13$ and $g := 2$ public.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange: Sample

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- Alice and Bob make $p := 13$ and $g := 2$ public. The number 2 is indeed a generator modulo 13 because the following powers of two taken modulo 13 yield the integers $1, 2, \ldots, 12$: $2^{12}, 2^1, 2^4, 2^2, 2^9, 2^5, 2^{11}, 2^{15}, 2^8, 2^{10}, 2^7, 2^6$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- Alice and Bob make $p := 13$ and $g := 2$ public. The number 2 is indeed a generator modulo 13 because the following powers of two taken modulo 13 yield the integers $1, 2, \ldots, 12$: $2^{12}, 2^1, 2^4, 2^2, 2^9, 2^5, 2^{11}, 2^{15}, 2^8, 2^{10}, 2^7, 2^6$.
- Alice chooses the private value $s := 5$, while Bob chooses $t := 6$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange: Sample

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- Alice and Bob make $p := 13$ and $g := 2$ public. The number 2 is indeed a generator modulo 13 because the following powers of two taken modulo 13 yield the integers $1, 2, \ldots, 12$: $2^{12}, 2^1, 2^4, 2^2, 2^9, 2^5, 2^{11}, 2^{15}, 2^8, 2^{10}, 2^7, 2^6$.
- Alice chooses the private value $s := 5$, while Bob chooses $t := 6$.
- We get $S = g^s \bmod p = 2^5 \bmod 13 = 32 \bmod 13 = 6$, and $T = g^t \bmod p = 2^6 \bmod 13 = 12$, which can be exchanged publicly.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange: Sample

	Alice	Bob
(1)	selects s with $1 < s < p-1$	selects t with $1 < t < p-1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- Alice and Bob make $p := 13$ and $g := 2$ public. The number 2 is indeed a generator modulo 13 because the following powers of two taken modulo 13 yield the integers $1, 2, \ldots, 12$: $\quad 2^{12}, 2^1, 2^4, 2^2, 2^9, 2^5, 2^{11}, 2^{15}, 2^8, 2^{10}, 2^7, 2^6$.
- Alice chooses the private value $s := 5$, while Bob chooses $t := 6$.
- We get $S = g^s \bmod p = 2^5 \bmod 13 = 32 \bmod 13 = 6$, and $T = g^t \bmod p = 2^6 \bmod 13 = 12$, which can be exchanged publicly.
- Finally, $T^s \bmod p = 12^5 \bmod 13 = (12140 \cdot 13 + 12) \bmod 13 = 12$, and $S^t \bmod p = 6^6 \bmod 13 = (3588 \cdot 13 + 12) \bmod 13 = 12$.
- Hence, Alice and Bob have managed to exchange 12 as a master key for their future communication.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Diffie-Hellman Symmetric Key Exchange: Sample

	Alice	Bob
(1)	selects s with $1 < s < p - 1$	selects t with $1 < t < p - 1$
(2)	sends $S := g^s \bmod p$ to Bob	sends $T := g^t \bmod p$ to Alice
(3)	calculates $T^s \bmod p$	calculates $S^t \bmod p$

- Alice and Bob make $p := 13$ and $g := 2$ public. The number 2 is indeed a generator modulo 13 because the following powers of two taken modulo 13 yield the integers $1, 2, \ldots, 12$: $2^{12}, 2^1, 2^4, 2^2, 2^9, 2^5, 2^{11}, 2^{15}, 2^8, 2^{10}, 2^7, 2^6$.
- Alice chooses the private value $s := 5$, while Bob chooses $t := 6$.
- We get $S = g^s \bmod p = 2^5 \bmod 13 = 32 \bmod 13 = 6$, and $T = g^t \bmod p = 2^6 \bmod 13 = 12$, which can be exchanged publicly.
- Finally, $T^s \bmod p = 12^5 \bmod 13 = (12140 \cdot 13 + 12) \bmod 13 = 12$, and $S^t \bmod p = 6^6 \bmod 13 = (3588 \cdot 13 + 12) \bmod 13 = 12$.
- Hence, Alice and Bob have managed to exchange 12 as a master key for their future communication.

No toy numbers!

Of course, in practice considerably larger values are chosen for p!!

Lemma 317

Let $a, b \in \mathbb{N}$ such that $\gcd(a, b) = 1$. Then there exists $x \in \mathbb{Z}$ such that $a \cdot x \equiv_b 1$.

Lemma 317

Let $a, b \in \mathbb{N}$ such that $\gcd(a, b) = 1$. Then there exists $x \in \mathbb{Z}$ such that $a \cdot x \equiv_b 1$.

Proof: Since $\gcd(a, b) = 1$, Cor. 125 tells us that there exist $x, y \in \mathbb{Z}$ such that $a \cdot x + b \cdot y = 1$. Hence, $a \cdot x = 1 - b \cdot y \equiv_b 1$. □

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 317

Let $a, b \in \mathbb{N}$ such that $\gcd(a,b) = 1$. Then there exists $x \in \mathbb{Z}$ such that $a \cdot x \equiv_b 1$.

Proof: Since $\gcd(a,b) = 1$, Cor. 125 tells us that there exist $x, y \in \mathbb{Z}$ such that $a \cdot x + b \cdot y = 1$. Hence, $a \cdot x = 1 - b \cdot y \equiv_b 1$. □

Definition 318 (Euler's Totient Function, Dt.: Eulersche φ-Funktion)

Euler's totient function $\varphi : \mathbb{N} \to \mathbb{N}$ is defined as

$$\varphi(n) := |U_n|, \quad \text{with } U_n := \{x \in \mathbb{N} : 1 \leqslant x \leqslant n \ \wedge \ \gcd(x,n) = 1\}.$$

The set U_n is called the *group of units* of n.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

Lemma 317

Let $a, b \in \mathbb{N}$ such that $\gcd(a, b) = 1$. Then there exists $x \in \mathbb{Z}$ such that $a \cdot x \equiv_b 1$.

Proof: Since $\gcd(a, b) = 1$, Cor. 125 tells us that there exist $x, y \in \mathbb{Z}$ such that $a \cdot x + b \cdot y = 1$. Hence, $a \cdot x = 1 - b \cdot y \equiv_b 1$. □

Definition 318 (Euler's Totient Function, Dt.: Eulersche φ-Funktion)

Euler's totient function $\varphi : \mathbb{N} \to \mathbb{N}$ is defined as

$$\varphi(n) := |U_n|, \quad \text{with } U_n := \{x \in \mathbb{N} : 1 \leqslant x \leqslant n \wedge \gcd(x, n) = 1\}.$$

The set U_n is called the *group of units* of n.

- Hence, $\varphi(n)$ is the number of integers among $1, 2, \ldots, n$ that are coprime to n.
- We have $\varphi(4) = 2$, $\varphi(5) = 4$, $\varphi(6) = 2$.
- More generally, $\varphi(p) = p - 1$ for every $p \in \mathbb{P}$.

Lemma 319

Let $p, q \in \mathbb{P}$. If $p \neq q$ then $\varphi(pq) = (p-1)(q-1)$.

Lemma 319

Let $p, q \in \mathbb{P}$. If $p \neq q$ then $\varphi(pq) = (p-1)(q-1)$.

Proof: There are q multiples of p and p multiples of q within $\{1, 2, \ldots, pq\}$, and the only common multiple of both p and q is pq. Hence, by the Inclusion-Exclusion Principle (Thm. 167), $\varphi(pq) = pq - p - q + 1 = (p-1)(q-1)$. □

Lemma 319

Let $p, q \in \mathbb{P}$. If $p \neq q$ then $\varphi(pq) = (p-1)(q-1)$.

Proof: There are q multiples of p and p multiples of q within $\{1, 2, \ldots, pq\}$, and the only common multiple of both p and q is pq. Hence, by the Inclusion-Exclusion Principle (Thm. 167), $\varphi(pq) = pq - p - q + 1 = (p-1)(q-1)$. □

Lemma 320 (Fermat/Euler)

Let $n \in \mathbb{N}$ and $x \in U_n$. Then $x^{\varphi(n)} \equiv_n 1$.

Lemma 319

Let $p, q \in \mathbb{P}$. If $p \neq q$ then $\varphi(pq) = (p-1)(q-1)$.

Proof: There are q multiples of p and p multiples of q within $\{1, 2, \ldots, pq\}$, and the only common multiple of both p and q is pq. Hence, by the Inclusion-Exclusion Principle (Thm. 167), $\varphi(pq) = pq - p - q + 1 = (p-1)(q-1)$. □

Lemma 320 (Fermat/Euler)

Let $n \in \mathbb{N}$ and $x \in U_n$. Then $x^{\varphi(n)} \equiv_n 1$.

Corollary 321

Let $n \in \mathbb{N}$ and $x \in U_n$. If $n = pq$, with $p, q \in \mathbb{P}$ and $p \neq q$, then $x^{(p-1)(q-1)} \equiv_n 1$.

RSA Encryption

- The RSA system [Rivest, Shamir, Adleman 1977] makes use of Lemma 320 and of the fact that state-of-the-art factorization methods take far too long for products of numbers with several hundred digits each.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

RSA Encryption

- The RSA system [Rivest, Shamir, Adleman 1977] makes use of Lemma 320 and of the fact that state-of-the-art factorization methods take far too long for products of numbers with several hundred digits each.
- The basic idea is very simple:
  - Select two distinct prime numbers p and q (each of which, in practice, has at least 150 digits) and compute $n = p \cdot q$.
  - Lemma 319 tells us that $\varphi(n) = (p-1) \cdot (q-1)$.

RSA Encryption

- The RSA system [Rivest, Shamir, Adleman 1977] makes use of Lemma 320 and of the fact that state-of-the-art factorization methods take far too long for products of numbers with several hundred digits each.
- The basic idea is very simple:
  - Select two distinct prime numbers p and q (each of which, in practice, has at least 150 digits) and compute $n = p \cdot q$.
  - Lemma 319 tells us that $\varphi(n) = (p-1)\cdot(q-1)$.
  - Select an integer $e \in \mathbb{N}\backslash\{1\}$ such that $\gcd(e, \varphi(n)) = 1$.
  - The numbers n and e are published (Bob's *public key*).

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

RSA Encryption

- The RSA system [Rivest, Shamir, Adleman 1977] makes use of Lemma 320 and of the fact that state-of-the-art factorization methods take far too long for products of numbers with several hundred digits each.
- The basic idea is very simple:
  - Select two distinct prime numbers p and q (each of which, in practice, has at least 150 digits) and compute $n = p \cdot q$.
  - Lemma 319 tells us that $\varphi(n) = (p-1) \cdot (q-1)$.
  - Select an integer $e \in \mathbb{N} \backslash \{1\}$ such that $\gcd(e, \varphi(n)) = 1$.
  - The numbers n and e are published (Bob's *public key*).
  - Compute a number d which is the inverse of e in $\mathbb{Z}_{\varphi(n)}$, i.e., such that $d \cdot e \equiv_{\varphi(n)} 1$. (Such a number exists due to Lem. 317.)
  - The number d is called Bob's *private key* and is kept secret.

RSA Encryption

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1) \cdot (q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.

RSA Encryption

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1) \cdot (q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
    - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n:
      $y := x^e \bmod n$ with $0 < y < n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

RSA Encryption

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1) \cdot (q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
  - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n:
    $y := x^e \bmod n$ with $0 < y < n$.
- *Decoding the ciphertext:*
  - Bob computes $z := y^d \bmod n$ with $0 < z < n$.

RSA Encryption

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1) \cdot (q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
  - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n:
    $y := x^e \bmod n$ with $0 < y < n$.
- *Decoding the ciphertext:*
  - Bob computes $z := y^d \bmod n$ with $0 < z < n$.

- Why does $z = x \bmod n$ hold?

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

RSA Encryption

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1) \cdot (q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
  - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n:
    $y := x^e \bmod n$ with $0 < y < n$.
- *Decoding the ciphertext:*
  - Bob computes $z := y^d \bmod n$ with $0 < z < n$.

- Why does $z = x \bmod n$ hold?
  - The condition $d \cdot e \equiv_{\varphi(n)} 1$ ensures that there exists $k \in \mathbb{Z}$ such that

$$d \cdot e = k \cdot \varphi(n) + 1.$$

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1)\cdot(q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
  - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n:
    $y := x^e \bmod n$ with $0 < y < n$.
- *Decoding the ciphertext:*
  - Bob computes $z := y^d \bmod n$ with $0 < z < n$.

- Why does $z = x \bmod n$ hold?
  - The condition $d \cdot e \equiv_{\varphi(n)} 1$ ensures that there exists $k \in \mathbb{Z}$ such that

    $$d \cdot e = k \cdot \varphi(n) + 1.$$

  - It follows that

    $$z \equiv_n y^d$$

RSA Encryption

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1)\cdot(q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
  - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n: $y := x^e \bmod n$ with $0 < y < n$.
- *Decoding the ciphertext:*
  - Bob computes $z := y^d \bmod n$ with $0 < z < n$.
- Why does $z = x \bmod n$ hold?
  - The condition $d \cdot e \equiv_{\varphi(n)} 1$ ensures that there exists $k \in \mathbb{Z}$ such that

$$d \cdot e = k \cdot \varphi(n) + 1.$$

  - It follows that

$$z \equiv_n y^d \equiv_n x^{de}$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1) \cdot (q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
  - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n: $y := x^e \bmod n$ with $0 < y < n$.
- *Decoding the ciphertext:*
  - Bob computes $z := y^d \bmod n$ with $0 < z < n$.

- Why does $z = x \bmod n$ hold?
  - The condition $d \cdot e \equiv_{\varphi(n)} 1$ ensures that there exists $k \in \mathbb{Z}$ such that

$$d \cdot e = k \cdot \varphi(n) + 1.$$

  - It follows that

$$z \equiv_n y^d \equiv_n x^{de} = x^{k\varphi(n)+1}$$

RSA Encryption

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1)\cdot(q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
  - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n: $y := x^e \bmod n$ with $0 < y < n$.
- *Decoding the ciphertext:*
  - Bob computes $z := y^d \bmod n$ with $0 < z < n$.

- Why does $z = x \bmod n$ hold?
  - The condition $d \cdot e \equiv_{\varphi(n)} 1$ ensures that there exists $k \in \mathbb{Z}$ such that

$$d \cdot e = k \cdot \varphi(n) + 1.$$

  - It follows that

$$z \equiv_n y^d \equiv_n x^{de} = x^{k\varphi(n)+1} = (x^{\varphi(n)})^k x$$

RSA Encryption

- Hence, we have $n = p \cdot q$ and, thus, $\varphi(n) = (p-1)\cdot(q-1)$. Furthermore, $\gcd(e, \varphi(n)) = 1$ and $d \cdot e \equiv_{\varphi(n)} 1$.
- *Encoding the ciphertext:*
  - Alice encodes a message $x \in \mathbb{N}$, with $x < n$ to keep it in $\mathbb{Z}_n$ and with $\gcd(x, n) = 1$, by using Bob's public key e and n: $y := x^e \bmod n$ with $0 < y < n$.
- *Decoding the ciphertext:*
  - Bob computes $z := y^d \bmod n$ with $0 < z < n$.

- Why does $z = x \bmod n$ hold?
  - The condition $d \cdot e \equiv_{\varphi(n)} 1$ ensures that there exists $k \in \mathbb{Z}$ such that

$$d \cdot e = k \cdot \varphi(n) + 1.$$

  - It follows that

$$z \equiv_n y^d \equiv_n x^{de} = x^{k\varphi(n)+1} = (x^{\varphi(n)})^k x \overset{L.\ 320}{\equiv_n} (1)^k x = x.$$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

RSA Encryption: Sample

- Suppose that $p := 5$ and $q := 11$. Hence $n = 55$ and $\varphi(n) = 40$. Suppose further that three users chose the following keys:

	e	d
Alice	23	7
Bob	37	13
Caesar	9	9

RSA Encryption: Sample

- Suppose that $p := 5$ and $q := 11$. Hence $n = 55$ and $\varphi(n) = 40$. Suppose further that three users chose the following keys:

	e	d
Alice	23	7
Bob	37	13
Caesar	9	9

- We have $23 \cdot 7 \equiv_{40} 1$ and $37 \cdot 13 \equiv_{40} 1$ and $9 \cdot 9 \equiv_{40} 1$.

RSA Encryption: Sample

- Suppose that $p := 5$ and $q := 11$. Hence $n = 55$ and $\varphi(n) = 40$. Suppose further that three users chose the following keys:

	e	d
Alice	23	7
Bob	37	13
Caesar	9	9

- We have $23 \cdot 7 \equiv_{40} 1$ and $37 \cdot 13 \equiv_{40} 1$ and $9 \cdot 9 \equiv_{40} 1$.
- Let $x := 2$ (and use Mathematica to do the arithmetic).

Alice: $2^{23} = 8388608 = 152520 \cdot 55 + 8 \equiv_{55} 8 =: y$

$8^7 = 2097152 = 38130 \cdot 55 + 2 \equiv_{55} 2 =: z$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

RSA Encryption: Sample

- Suppose that $p := 5$ and $q := 11$. Hence $n = 55$ and $\varphi(n) = 40$. Suppose further that three users chose the following keys:

	e	d
Alice	23	7
Bob	37	13
Caesar	9	9

- We have $23 \cdot 7 \equiv_{40} 1$ and $37 \cdot 13 \equiv_{40} 1$ and $9 \cdot 9 \equiv_{40} 1$.
- Let $x := 2$ (and use Mathematica to do the arithmetic).

Alice: $2^{23} = 8388608 = 152520 \cdot 55 + 8 \equiv_{55} 8 =: y$

$8^{7} = 2097152 = 38130 \cdot 55 + 2 \equiv_{55} 2 =: z$

Bob: $2^{37} = 137438953472 = 2498890063 \cdot 55 + 7 \equiv_{55} 7 =: y$

$7^{13} = 96889010407 = 1761618371 \cdot 55 + 2 \equiv_{55} 2 =: z$

Caesar: $2^{9} = 512 = 9 \cdot 55 + 17 \equiv_{55} 17 =: y$

$17^{9} = 118587876497 = 2156143209 \cdot 55 + 2 \equiv_{55} 2 =: z$

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

RSA Encryption: Analysis

- Note that there are $\varphi(n)$ many messages that can be sent for n given.
- Since

$$\frac{\varphi(n)}{n} = \frac{(p-1)(q-1)}{pq} = (1-\frac{1}{p})(1-\frac{1}{q})$$

is close to 1 for large p, q, the probability of selecting a message that is coprime to n is the larger the larger p, q are.

RSA Encryption: Analysis

- Note that there are $\varphi(n)$ many messages that can be sent for n given.
- Since

$$\frac{\varphi(n)}{n} = \frac{(p-1)(q-1)}{pq} = (1-\frac{1}{p})(1-\frac{1}{q})$$

  is close to 1 for large p, q, the probability of selecting a message that is coprime to n is the larger the larger p, q are.
- Although n is publicly known, it is important to keep $\varphi(n)$ and, thus, also p, q secret!

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

- Note that there are $\varphi(n)$ many messages that can be sent for n given.
- Since

$$\frac{\varphi(n)}{n} = \frac{(p-1)(q-1)}{pq} = (1-\frac{1}{p})(1-\frac{1}{q})$$

  is close to 1 for large p, q, the probability of selecting a message that is coprime to n is the larger the larger p, q are.
- Although n is publicly known, it is important to keep $\varphi(n)$ and, thus, also p, q secret!
- An eavesdropper who only knows n, e, and y cannot do much with this information. In particular, no efficient algorithm is known to factor n into p, q as a simple means to obtain $\varphi(n)$.

RSA Encryption: Analysis

- Note that there are $\varphi(n)$ many messages that can be sent for n given.
- Since
$$\frac{\varphi(n)}{n} = \frac{(p-1)(q-1)}{pq} = (1-\frac{1}{p})(1-\frac{1}{q})$$
is close to 1 for large p, q, the probability of selecting a message that is coprime to n is the larger the larger p, q are.
- Although n is publicly known, it is important to keep $\varphi(n)$ and, thus, also p, q secret!
- An eavesdropper who only knows n, e, and y cannot do much with this information. In particular, no efficient algorithm is known to factor n into p, q as a simple means to obtain $\varphi(n)$.
- It is also important to ensure that $x^e > n$, i.e., that y is obtained by exponentiation and then by a reduction modulo n.
  - If $x^e < n$ then one could simply recover x by taking the e-th root of y. (After all, e is known publicly!)
  - Hence, it is wise to select e such that $2^e > n$.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

The End!

I hope that you enjoyed this course, and I wish you all the best for your future studies.

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab

UNIVERSITÄT SALZBURG
Computational Geometry and Applications Lab