Review of Predicate Logic

Martin Held

FB Computerwissenschaften
Universität Salzburg
A-5020 Salzburg, Austria
held@cosy.sbg.ac.at

19. Jänner 2016

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

Legal Fine Print and Disclaimer

Table of Content

1 **Review of Propositional and Predicate Logic**

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

1. **Review of Propositional and Predicate Logic**
   - Propositional Logic
   - Predicate Logic
   - Special Quantifiers

- 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 connectives.

Language of Propositional Logic

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

A propositional formula is constructed inductively from a set of

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

based on the following rules:

- A propositional variable is a propositional formula.
- The constants $\bot$ 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.

$$\neg,\ \wedge,\ \vee,\ \begin{matrix}\Rightarrow\\ \Leftrightarrow\end{matrix}$$

- 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 connectives we also define another operator, NAND, denoted by $|$.

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

Truth Tables

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$	$p \mid q$
T	T	F	T	T	T	T	F
T	F	F	F	T	F	F	T
F	T	T	F	T	T	F	T
F	F	T	F	F	T	T	T

- Common names for the operators 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.

UNIV. SALZBURG COMPUTERWISSENSCHAFTEN

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 connectives, Dt.: vollständige Junktorenmenge)

A set S of connectives 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 connectives of S.

- Note: The set $\{|\}$ is a complete set of connectives.

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.:	$\phi_1 \Rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2$	

UNIV. SALZBURG COMPUTERWISSENSCHAFTEN

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!

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.

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

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

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

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

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 $k \geq 3$, 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.

1 Review of Propositional and Predicate Logic

- Propositional Logic
- Predicate Logic
- Special Quantifiers

Definition 16 (*n*-place relation, Dt.: *n*-stellige Relation)

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

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

Definition 16 (*n*-place relation, Dt.: *n*-stellige Relation)

Let $A_1, A_2, \ldots, A_n$ be sets, for some $n \in \mathbb{N}$. An *n-place 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*-place function, Dt.: *n*-stellige Funktion)

Let $A_1, A_2, \ldots, A_n, B$ be sets, for some $n \in \mathbb{N}$. An *n-place function* $\mathcal{F}$ from $A_1 \times A_2 \times \cdots \times A_n$ to B is an $(n+1)$-place 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 $(a_1, a_2, \ldots, a_n, b) \in \mathcal{F}$.

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

Let $A_1, A_2, \ldots, A_n$ be sets, for some $n \in \mathbb{N}$. An *n-place 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-place function, Dt.: n-stellige Funktion)

Let $A_1, A_2, \ldots, A_n, B$ be sets, for some $n \in \mathbb{N}$. An *n-place function* $\mathcal{F}$ from $A_1 \times A_2 \times \cdots \times A_n$ to B is an $(n+1)$-place 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 $\mathcal{F}$.

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

Definition 16 (*n*-place relation, Dt.: *n*-stellige Relation)

Let $A_1, A_2, \ldots, A_n$ be sets, for some $n \in \mathbb{N}$. An *n-place 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*-place function, Dt.: *n*-stellige Funktion)

Let $A_1, A_2, \ldots, A_n, B$ be sets, for some $n \in \mathbb{N}$. An *n-place function* $\mathcal{F}$ from $A_1 \times A_2 \times \cdots \times A_n$ to B is an $(n+1)$-place 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-place 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-ary relation/function.
- An 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 "$\to$" 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

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

- Dt.: Prädikatenlogik.

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

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) $\bot, \top$ or F, T,

together with

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

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 $(\forall x \quad \phi)$ and $(\exists x \quad \phi)$ 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.

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

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 $(\forall x \quad \phi)$ and $(\exists x \quad \phi)$.

Definition 24 (Universal quantifier, Dt.: Allquantor)

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

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

Definition 25 (Existential quantifier, Dt.: Existenzquantor)

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

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

- The notation $(\exists! x \quad P(x))$ 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.

UNIV. SALZBURG COMPUTERWISSENSCHAFTEN

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.

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 $f(t_1, \ldots, t_n)$: $FV(f(t_1, \ldots, t_n)) := FV(t_1) \cup \ldots \cup FV(t_n);$

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

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:

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

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:

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

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 $FV(\phi) = \{\}$.

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: $\phi \equiv (\exists y \in \mathbb{N} \quad y > 10 \quad \wedge \quad x < y)$ 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!
  $\phi \equiv (\forall z \in \mathbb{N} \quad z^2 > 0) \quad \vee \quad (\exists y \in \mathbb{N} \quad y > 10 \quad \wedge \quad x < y)$ and $t := 2z + 5$.

UNIV. SALZBURG COMPUTERWISSENSCHAFTEN

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: $(\neg(\forall x \quad \phi)) \equiv (\exists x \quad (\neg\phi))$

$(\neg(\exists x \quad \phi)) \equiv (\forall x \quad (\neg\phi))$

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

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

$(\forall x \quad (\phi \vee \psi)) \equiv ((\forall x \quad \phi) \vee \psi)$

Rules of Inference

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

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

- 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.

1. **Review of Propositional and Predicate Logic**
   - Propositional Logic
   - Predicate Logic
   - Special Quantifiers

Special Quantifiers

- What is the syntactical meaning of

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

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

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

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

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: $\cup_{i=1}^{n} A_i$.

UNIV. SALZBURG
COMPUTERWISSENSCHAFTEN

Special Quantifiers: Sets

- Standard notation for a set with a finite number of elements: $\{\quad,\quad,\ldots,\quad\}$; 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.

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:

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

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

- Other wide-spread notations include the following variations:

dropping the parentheses:	$\forall x \ \ P(x)$	instead of	$(\forall x \ \ P(x))$
colon instead of space as separator:	$\forall x\colon P(x)$	instead of	$\forall x \ \ P(x)$

UNIV. SALZBURG COMPUTERWISSENSCHAFTEN