

Breaking the Sorting Barrier for Directed Single-Source Shortest Paths

B. Tascan, B. Durie, S. Lackner

FB Informatik
Universität Salzburg

January 15, 2026



Verzeichnis

- 1 Einführung
 - SSSP Algorithmen
 - Dijkstra
 - Bellman-Ford
- 2 Bounded Multi-Source Shortest Path
 - Der Algorithmus
 - Die Datenstruktur
 - Der Algorithmus
 - Laufzeit
- 3 Literatur







-
-
- Datenstrukturen



- 1 Einführung
 - SSSP Algorithmen
 - Dijkstra
 - Bellman-Ford
- 2 Bounded Multi-Source Shortest Path
 - Der Algorithmus
 - Die Datenstruktur
 - Der Algorithmus
 - Laufzeit
- 3 Literatur



SSSP Algorithmen

- Single-Source Shortest Path Algorithmen (SSSP's)



Dijkstra



Bellman-Ford

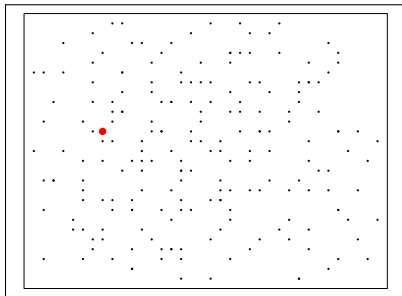


- 1 Einführung
 - SSSP Algorithmen
 - Dijkstra
 - Bellman-Ford
- 2 Bounded Multi-Source Shortest Path
 - Der Algorithmus
 - Die Datenstruktur
 - Der Algorithmus
 - Laufzeit
- 3 Literatur



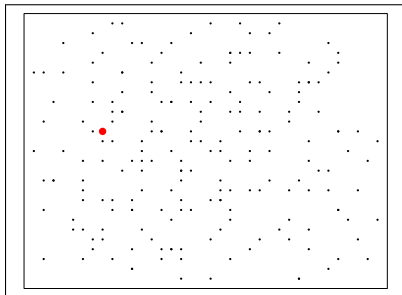
Der Algorithmus - Einleitung

- Dijkstra in $\mathcal{O}(m + n \log n)$



Der Algorithmus - Einleitung

- Dijkstra in $\mathcal{O}(m + n \log n)$
- Sortierbarriere $\Omega(n \log n)$



Der Algorithmus - Lösungsansatz

Sortierbarriere umgehen durch:

- **Eigene Datenstruktur**

Ermöglicht effizientes *Bucketing*, verhindert Sortierung



Der Algorithmus - Lösungsansatz

Sortierbarriere umgehen durch:

- **Eigene Datenstruktur**

Ermöglicht effizientes *Bucketing*, verhindert Sortierung

- **Pivoting**

Reduziert den Rechenaufwand



Der Algorithmus - Lösungsansatz

Sortierbarriere umgehen durch:

- **Eigene Datenstruktur**

Ermöglicht effizientes *Bucketing*, verhindert Sortierung

- **Pivoting**

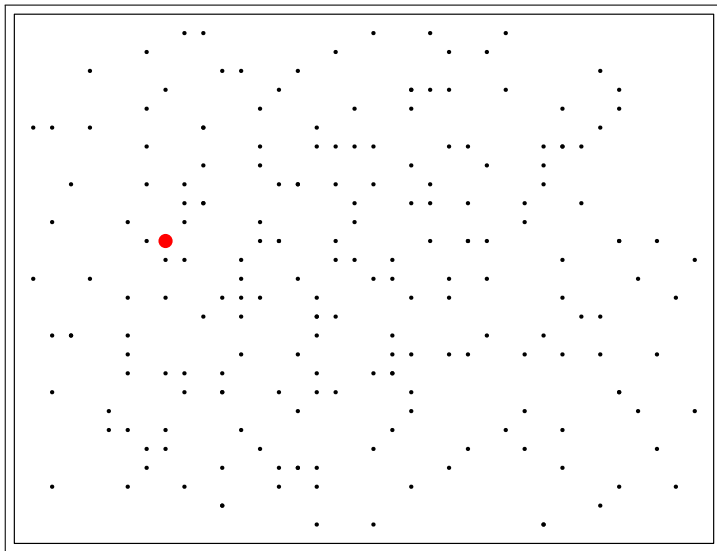
Reduziert den Rechenaufwand

- **Divide & Conquer**

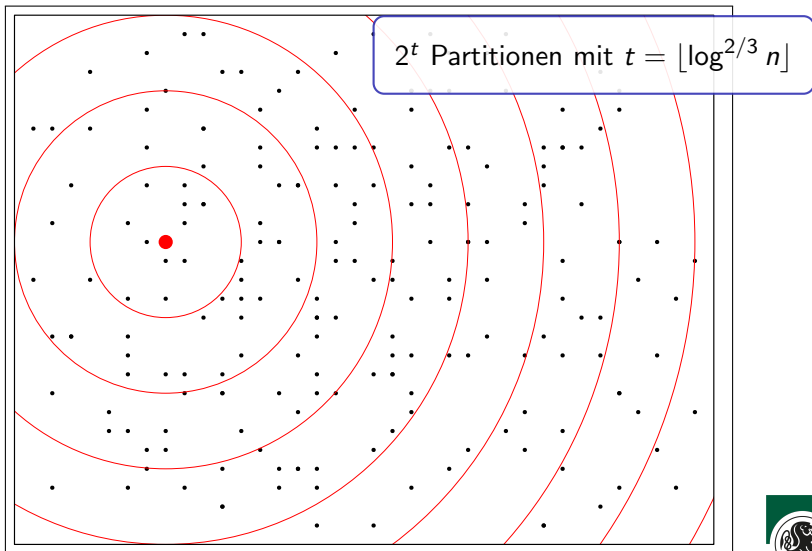
Mindert die Problemgröße durch Rekursion



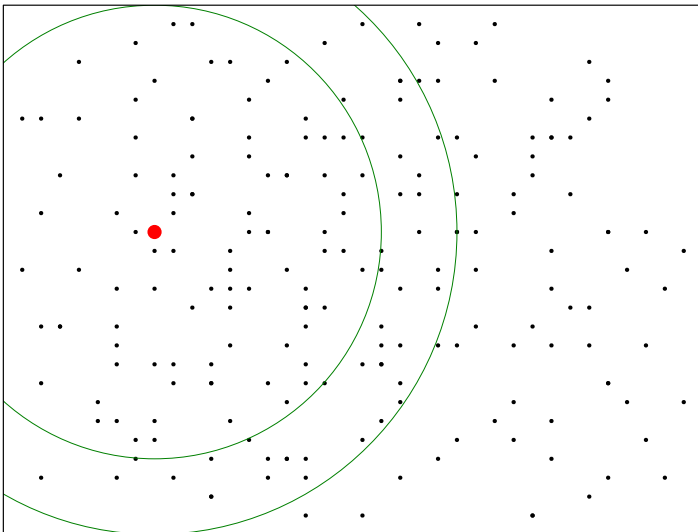
Der Algorithmus - Divide & Conquer



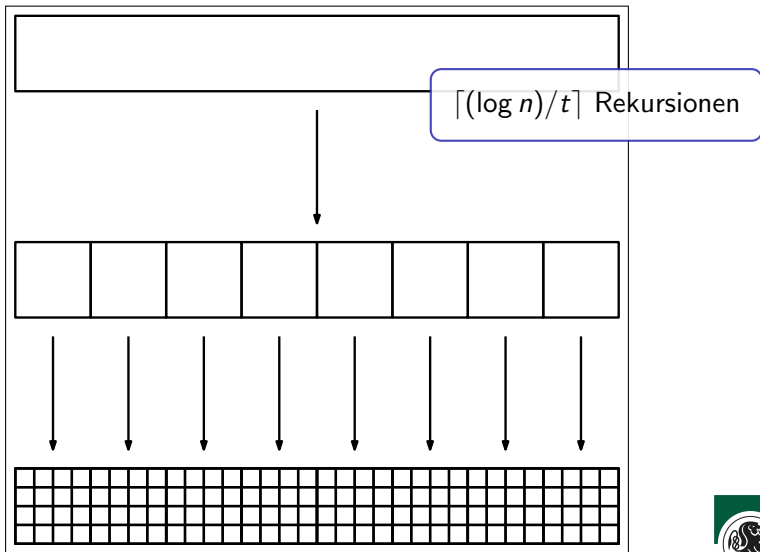
Der Algorithmus - Divide & Conquer



Der Algorithmus - Divide & Conquer



Der Algorithmus - Divide & Conquer



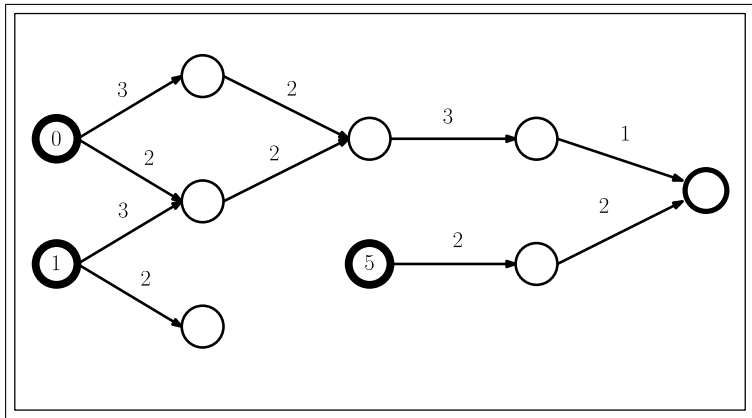
Der Algorithmus - Divide & Conquer

Wozu das ganze? ist notwendiges Hilfsmittel für:

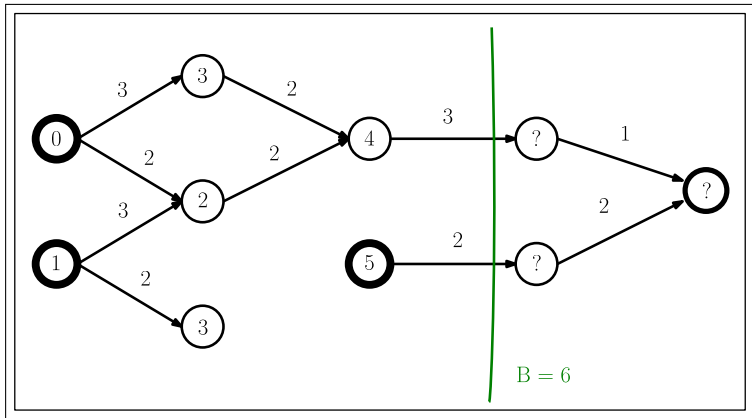
- **BMSSP**
funktioniert nur dank Abgrenzungen
- **Pivots**
erlaubt schnelle Auswahl von wichtigen Knoten



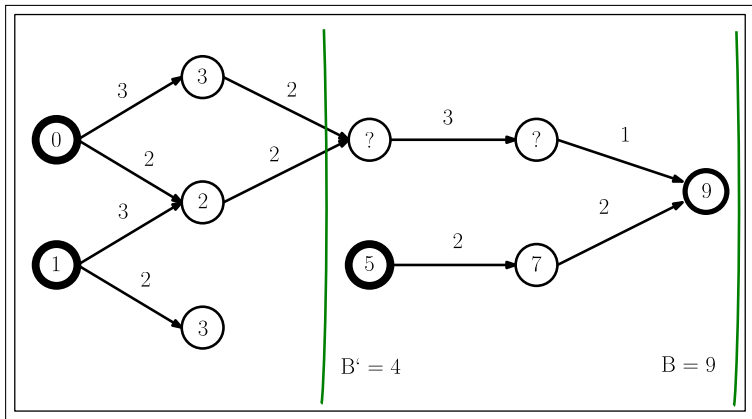
Der Algorithmus - BMSSP



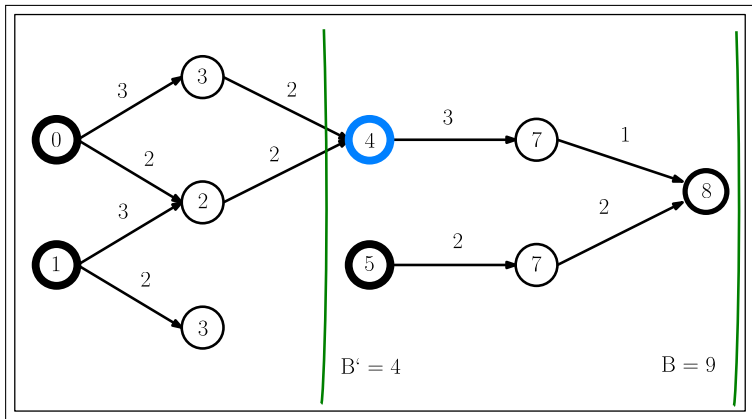
Der Algorithmus - BMSSP



Der Algorithmus - BMSSP

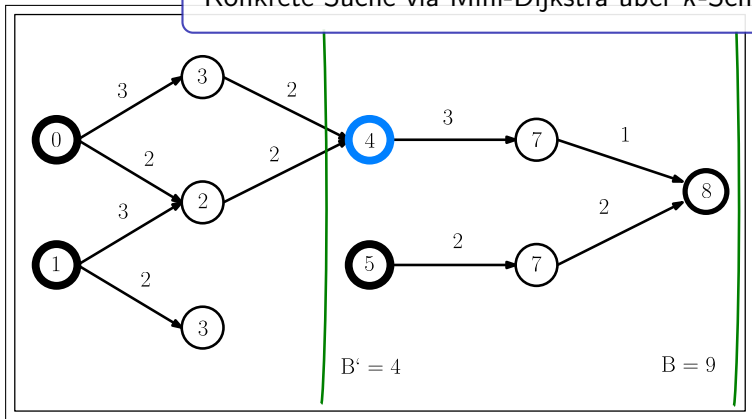


Der Algorithmus - BMSSP



Der Algorithmus - BMSSP

Konkrete Suche via Mini-Dijkstra über k -Schritte



Der Algorithmus - BMSSP

Wozu das ganze?

- **Sortierbarriere**

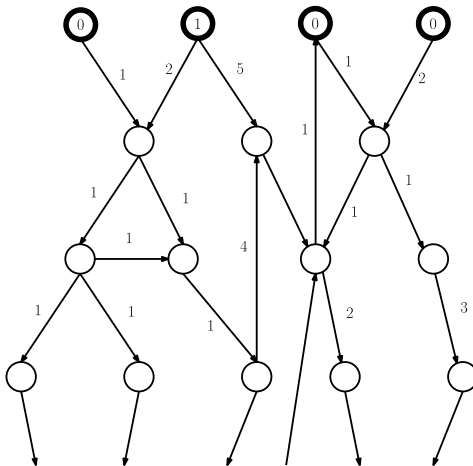
Ermöglicht Umgehung der $\Omega(n \log n)$ Schranke dank Bucketing



Der Algorithmus - Pivots

k Bellmann-Ford Schritte

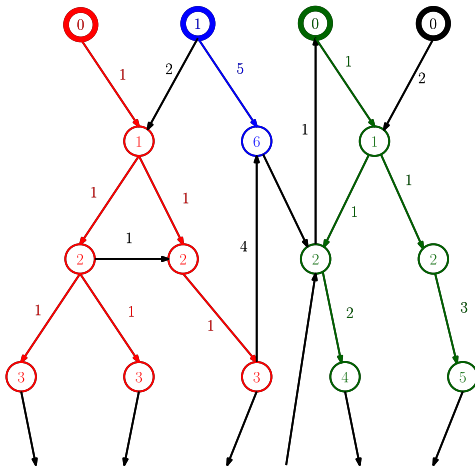
$k=3$



Der Algorithmus - Pivots

Markiere Knoten anhand nächster Quelle

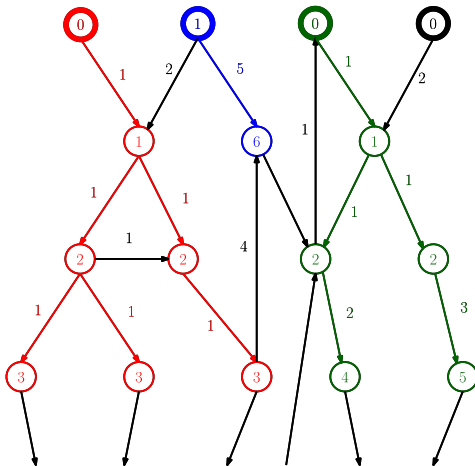
$k=3$



Der Algorithmus - Pivots

Bäume mit mindestens k Knoten $\Rightarrow$ Pivots

$k=3$

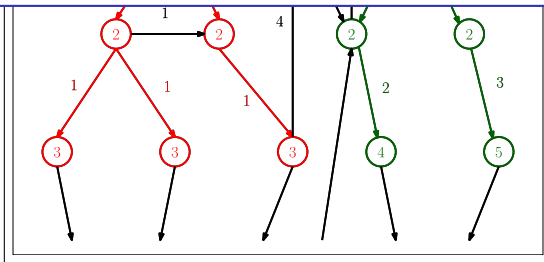


Der Algorithmus - Pivots

$k=3$

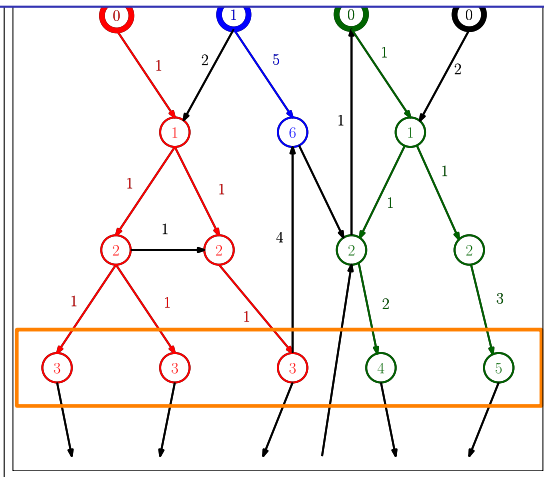
Bäume mit mindestens k Knoten $\Rightarrow$ Pivots

Maximal n/k Bäume mit mindestens k Knoten
 $\Rightarrow$ Maximal n/k Pivots

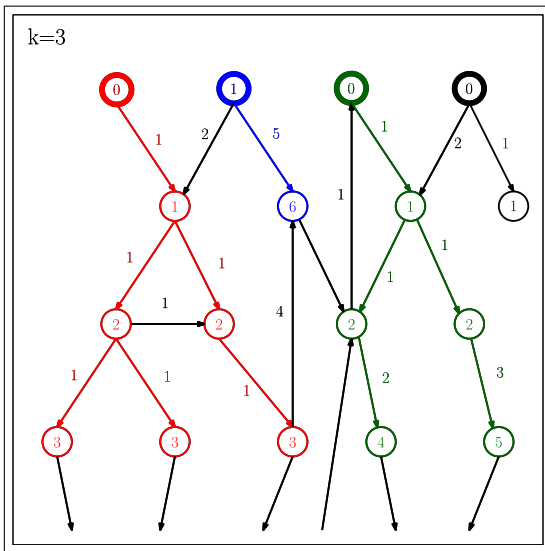


Der Algorithmus - Pivots

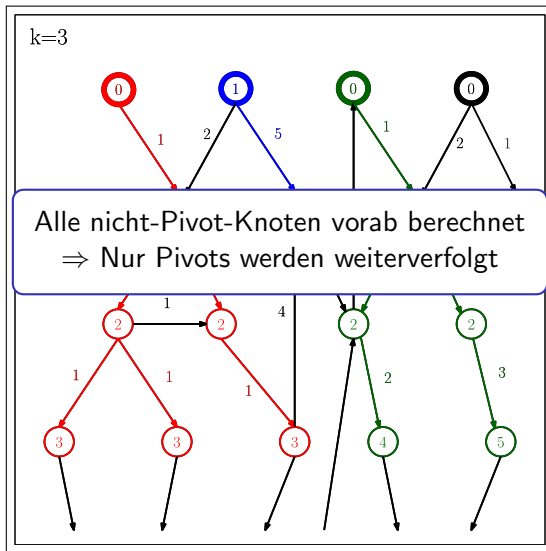
Weiterführende Bäume müssen unterste Schicht erreichen.
Nur Bäume mit mindestens k Knoten (=Pivots) können das.



Der Algorithmus - Pivots



Der Algorithmus - Pivots



Der Algorithmus - Pivots

Wozu das ganze? Hauptgrund für Geschwindigkeit:

- **Kostenreduktion**

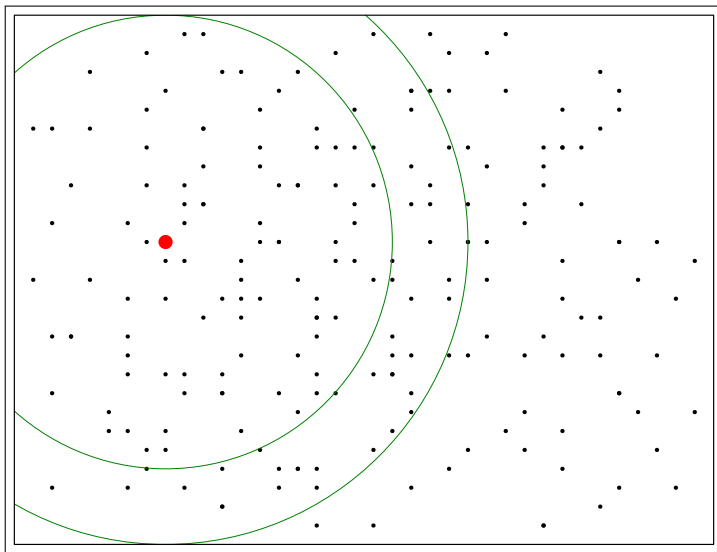
reduziert Aufwand durch gezielte Knotenwahl

- **Faktor**

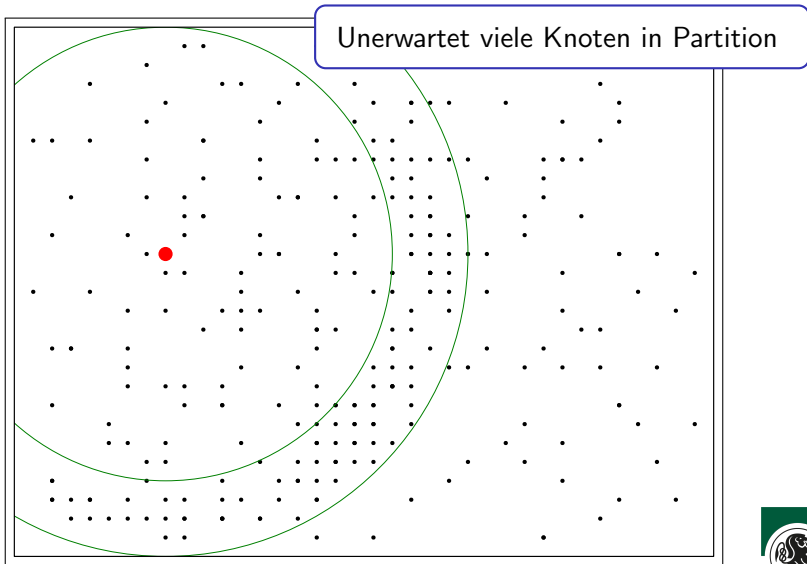
reduziert BMSSP Knotenanzahl um $\log^{1/3} n$



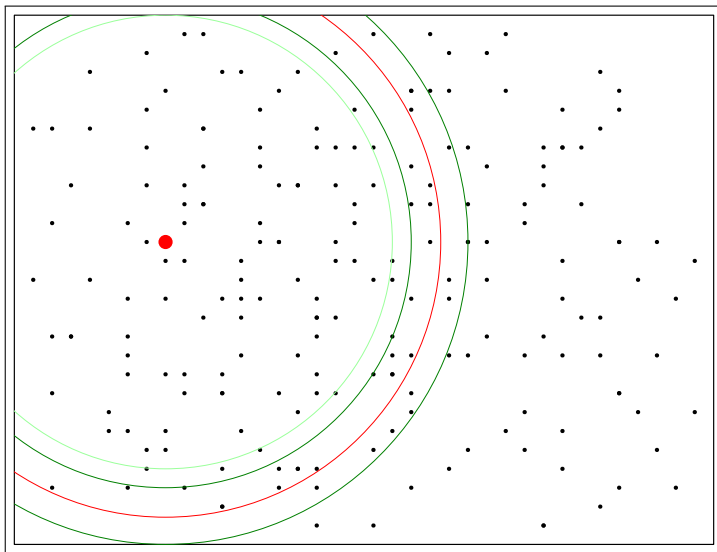
Der Algorithmus - Letzte Hürde



Der Algorithmus - Letzte Hürde



Der Algorithmus - Letzte Hürde



- 1 Einführung
 - SSSP Algorithmen
 - Dijkstra
 - Bellman-Ford
- 2 Bounded Multi-Source Shortest Path
 - Der Algorithmus
 - Die Datenstruktur
 - Der Algorithmus
 - Laufzeit
- 3 Literatur



Die Datenstruktur

- Dijkstra hat eine asymptotische Laufzeit von $\mathcal{O}(m \log n)$



Die Datenstruktur

- Dijkstra hat eine asymptotische Laufzeit von $\mathcal{O}(m \log n)$
- Um diese Laufzeit zu verbessern, wird eine spezielle Datenstruktur benötigt



Die Datenstruktur

- Dijkstra hat eine asymptotische Laufzeit von $\mathcal{O}(m \log n)$
- Um diese Laufzeit zu verbessern, wird eine spezielle Datenstruktur benötigt
- Diese Struktur ist eine sogenannte Block-based linked List



Die Datenstruktur

- Es gibt zwei Sequenzen an Blöcken, $\mathcal{D}_0$ und $\mathcal{D}_1$, welche beide Linked Lists sind mit maximal M Key/Value Paaren und einem Upperbound von B



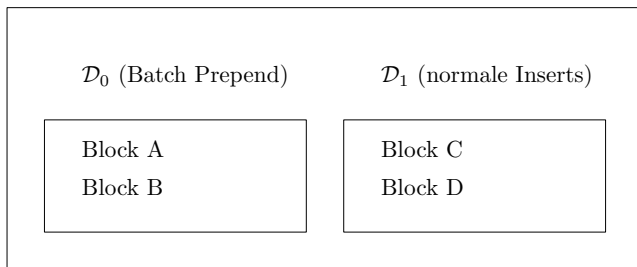
Die Datenstruktur

- Es gibt zwei Sequenzen an Blöcken, $\mathcal{D}_0$ und $\mathcal{D}_1$, welche beide Linked Lists sind mit maximal M Key/Value Paaren und einem Upperbound von B
- $\mathcal{D}_0$ enthält Batch Prepend Elemente, unbounded



Die Datenstruktur

- Es gibt zwei Sequenzen an Blöcken, $\mathcal{D}_0$ und $\mathcal{D}_1$, welche beide Linked Lists sind mit maximal M Key/Value Paaren und einem Upperbound von B
- $\mathcal{D}_0$ enthält Batch Prepend Elemente, unbounded
- $\mathcal{D}_1$ enthält mit Insert eingefügte Elemente, bounded mit $O(\max\{1, N/M\})$



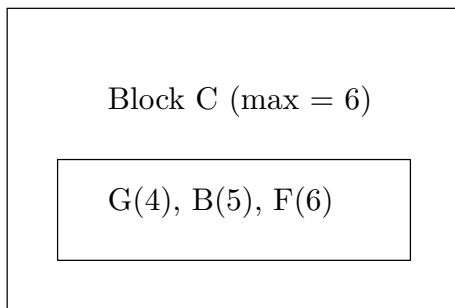
Die Datenstruktur

- Beide Sequenzen sind nach deren Werten sortiert, d.h. der Upperbound eines Blocks ist nie größer als alle Werte des darauffolgenden Blocks



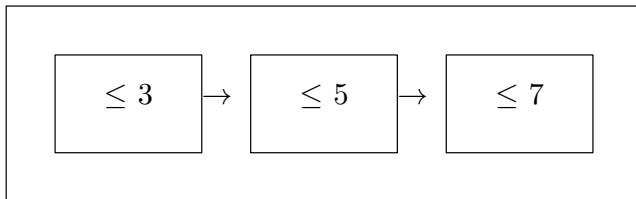
Die Datenstruktur

- Beide Sequenzen sind nach deren Werten sortiert, d.h. der Upperbound eines Blocks ist nie größer als alle Werte des darauffolgenden Blocks
- Die Blöcke werden von einem binären Suchbaum gebalanced



Die Datenstruktur

- Beide Sequenzen sind nach deren Werten sortiert, d.h. der Upperbound eines Blocks ist nie größer als alle Werte des darauffolgenden Blocks
- Die Blöcke werden von einem binären Suchbaum gebalanced



Die Datenstruktur

- Insert



Die Datenstruktur

- Insert
- Batch Prepend



Die Datenstruktur

- Insert
- Batch Prepend
- Pull



Die Datenstruktur

- $\text{Insert}(a, b)$:



Die Datenstruktur

- Insert(a, b):
- Sollten mehrere Key/Value Paare den selben Key haben, so wird das Paar mit dem kleineren Value bevorzugt



Die Datenstruktur

- $\text{Insert}(a, b)$:
- Sollten mehrere Key/Value Paare den selben Key haben, so wird das Paar mit dem kleineren Value bevorzugt
- Der entsprechende Block wird mithilfe des Binären Baums gesucht



Die Datenstruktur

- $\text{Insert}(a, b)$:
- Sollten mehrere Key/Value Paare den selben Key haben, so wird das Paar mit dem kleineren Value bevorzugt
- Der entsprechende Block wird mithilfe des Binären Baums gesucht
- Insert beim gefundenen Block in $\mathcal{D}_1$



Die Datenstruktur

- $\text{Insert}(a, b)$:
- Sollten mehrere Key/Value Paare den selben Key haben, so wird das Paar mit dem kleineren Value bevorzugt
- Der entsprechende Block wird mithilfe des Binären Baums gesucht
- Insert beim gefundenen Block in $\mathcal{D}_1$
- Laufzeit von Insert $O(\max\{1, \log(N/M)\})$



Die Datenstruktur

- Batch Prepend(L):



Die Datenstruktur

- Batch Prepend(L):
- L Key/value Paare werden so eingetragen dass keine kleineren Werte vorhanden sind



Die Datenstruktur

- Batch Prepend(L):
- L Key/value Paare werden so eingetragen dass keine kleineren Werte vorhanden sind
- Insert ist immer am Beginn von $\mathcal{D}_0$



Die Datenstruktur

- Batch Prepend(L):
- L Key/value Paare werden so eingetragen dass keine kleineren Werte vorhanden sind
- Insert ist immer am Beginn von $\mathcal{D}_0$
- Laufzeit von Batch Prepend $O(L \cdot \max\{1, \log(L/M)\})$



Die Datenstruktur

- Pull:



Die Datenstruktur

- Pull:
- Pull gibt eine Menge S' , mit einem Upper Bound x , an kleinsten Werten zurück



Die Datenstruktur

- Pull:
- Pull gibt eine Menge S' , mit einem Upper Bound x , an kleinsten Werten zurück
- Das führt zu einer Sortierung in Gruppen statt einer genauen Sortierung der Werte



Die Datenstruktur

- Pull:
- Pull gibt eine Menge S' , mit einem Upper Bound x , an kleinsten Werten zurück
- Das führt zu einer Sortierung in Gruppen statt einer genauen Sortierung der Werte
- Laufzeit von Pull $O(|S'|)$



- 1 Einführung
 - SSSP Algorithmen
 - Dijkstra
 - Bellman-Ford
- 2 Bounded Multi-Source Shortest Path
 - Der Algorithmus
 - Die Datenstruktur
 - Der Algorithmus
 - Laufzeit
- 3 Literatur



- 1 Einführung
 - SSSP Algorithmen
 - Dijkstra
 - Bellman-Ford
- 2 Bounded Multi-Source Shortest Path
 - Der Algorithmus
 - Die Datenstruktur
 - Der Algorithmus
 - Laufzeit
- 3 Literatur



Literatur

