Karlsruher Institut für Technologie

Platzeffiziente Hash Tabellen - endlich richtig!

Präsentation · 07. Juli 2017

Tobias Maier

INSTITUT FÜR THEORETISCHE INFORMATIK · LEHRSTUHL ALGORITHMIK

Ziele der Datenstruktur

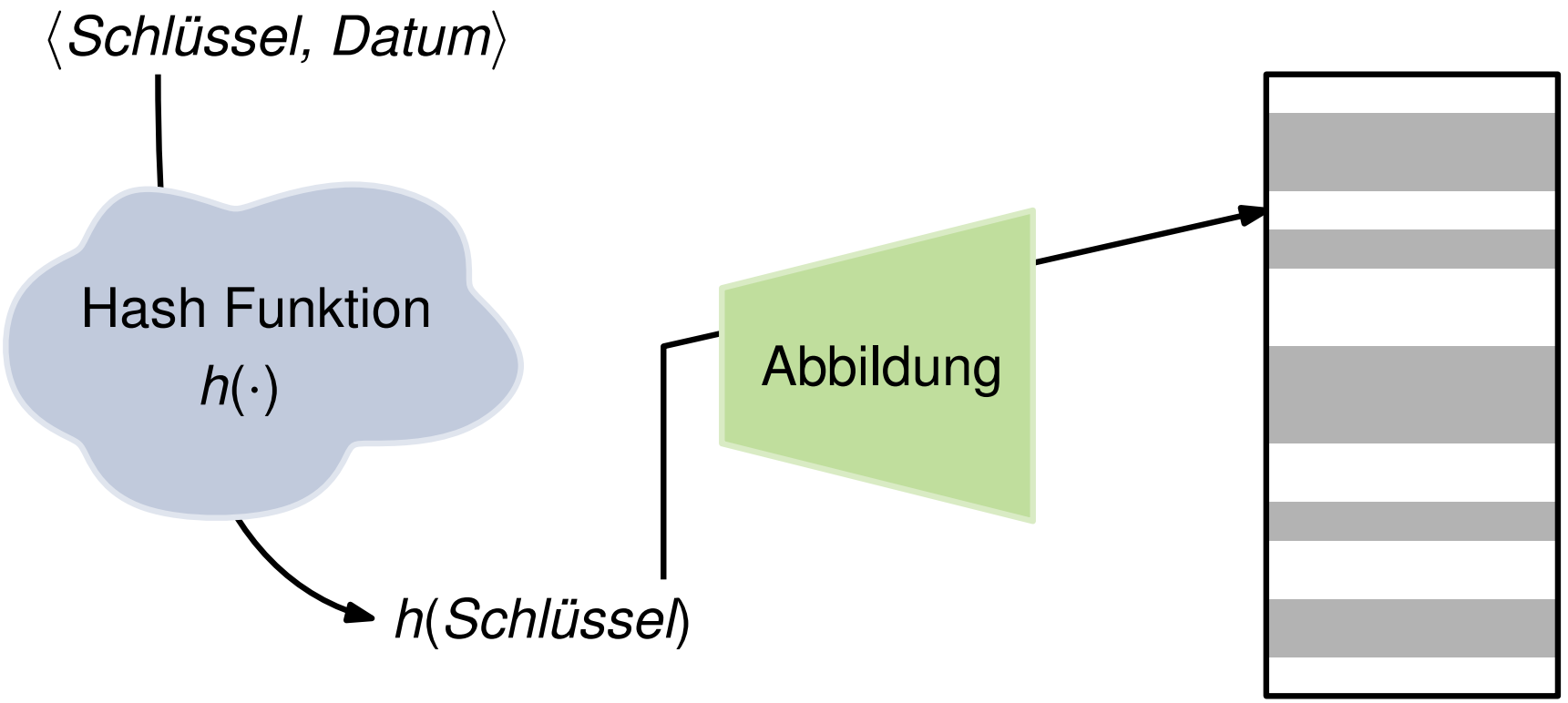
Matr. Nr.	Name
00001	Alice
00042	Bob
00256	Frank
00666	Grace
⋮	⋮
18769	Eve
⋮	⋮
58730	Carol
⋮	⋮
98746	Dan
⋮	⋮

- Dynamische Daten
- Jedes Datum hat einen Schlüssel
- Finde Daten zu geg. Schlüssel
- Erwartet konstante Zeit
- Wir opfern Speicherplatz

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

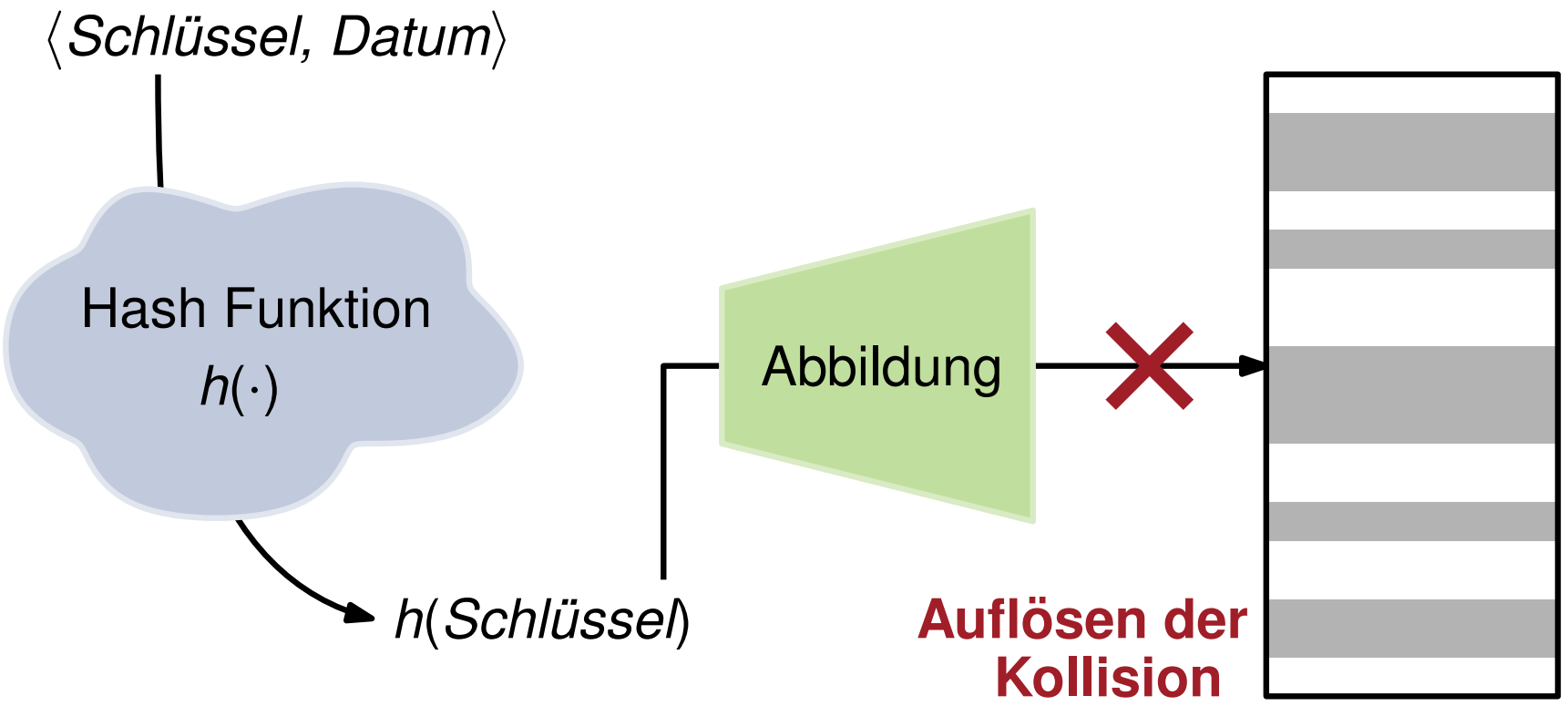
Grundlegendes Verfahren

- Position abhängig von Schlüssel
- Unabhängig vom Speicherzeitpunkt



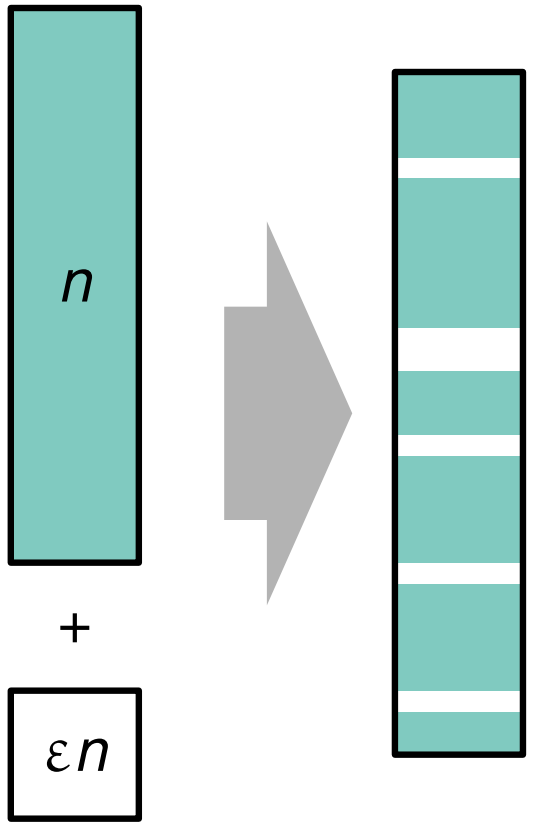
Grundlegendes Verfahren

- Position abhängig von Schlüssel
- Unabhängig vom Speicherzeitpunkt



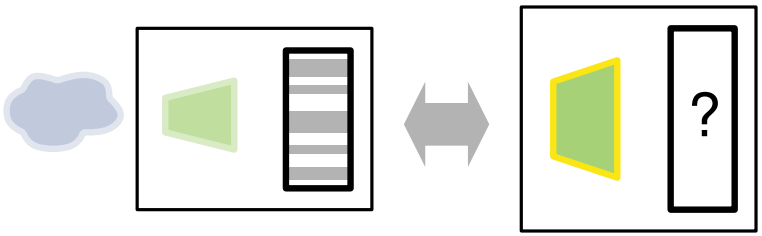
Platzeffizienz ohne zu Wachsen?

- Dicht gefüllte Tabelle
- Viele Kollisionen
  - Braucht gute Kollisionsauflösung
- Nach Konstruktion feste Größe
  - Feste Elementanzahl

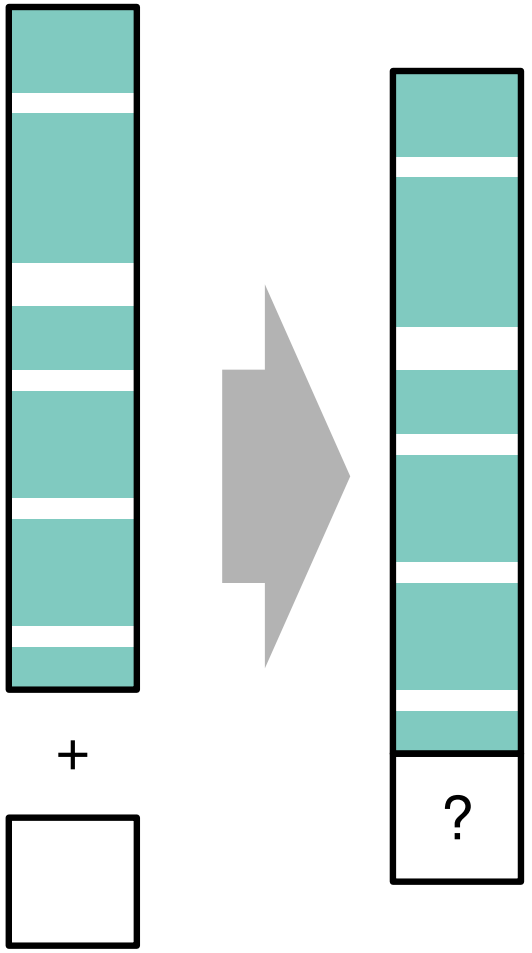


Vergrößern

- Abbildung Anpassen für neuen Speicher



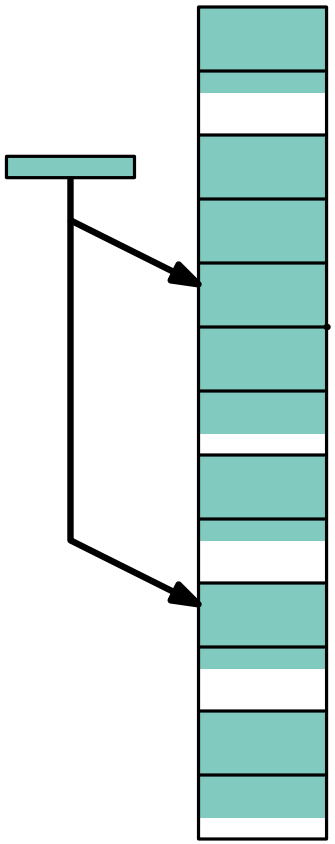
- Alte Elemente müssen bewegt werden
- Platzeffizienz benötigt viele kleine Schritte



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

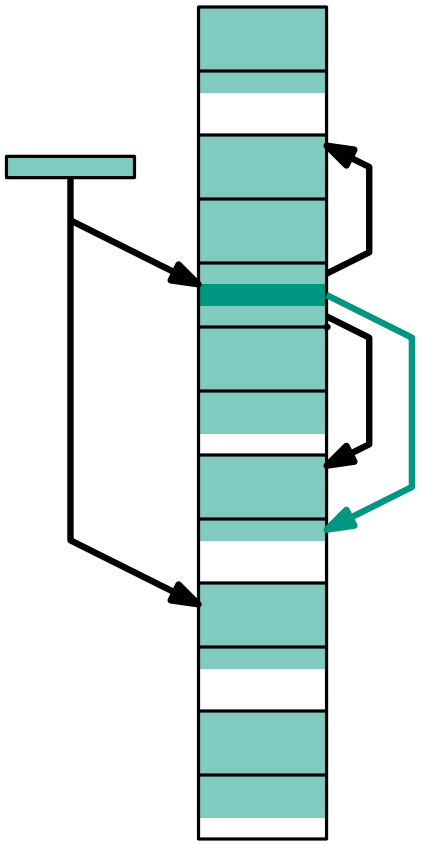
Kuckuck Verdrängung

- Zellen zu Buckets zusammengefasst
- Alternative Buckets pro Element
- Falls alle Buckets belegt sind, verdränge bestehende Elemente



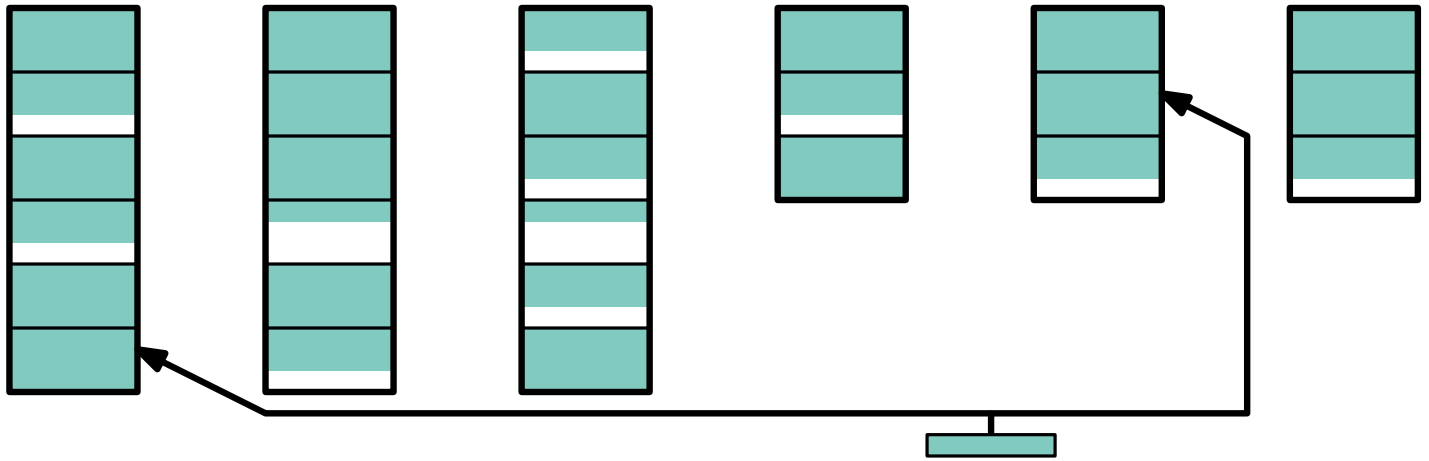
Kuckuck Verdrängung

- Zellen zu Buckets zusammengefasst
- Alternative Buckets pro Element
- Falls alle Buckets belegt sind, verdränge bestehende Elemente



DySECT (Dynamic Space Efficient Cuckoo Table)

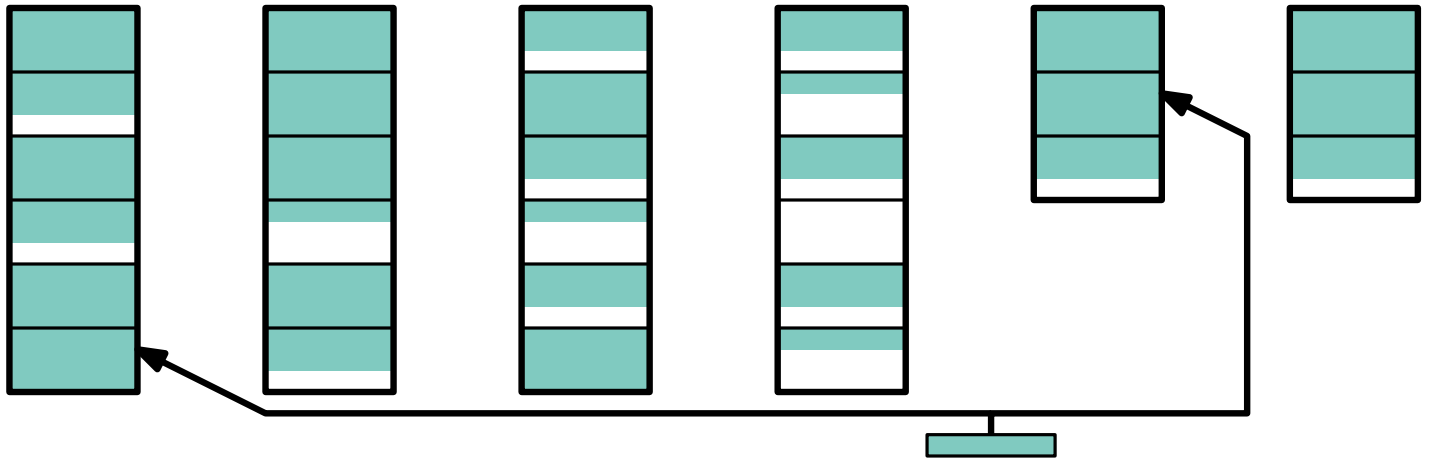
- Verwende Subtabellen ungleicher Größe
- Verwende die Verdrängungsstrategie um Ungleichgewichte auszugleichen



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

DySECT (Dynamic Space Efficient Cuckoo Table)

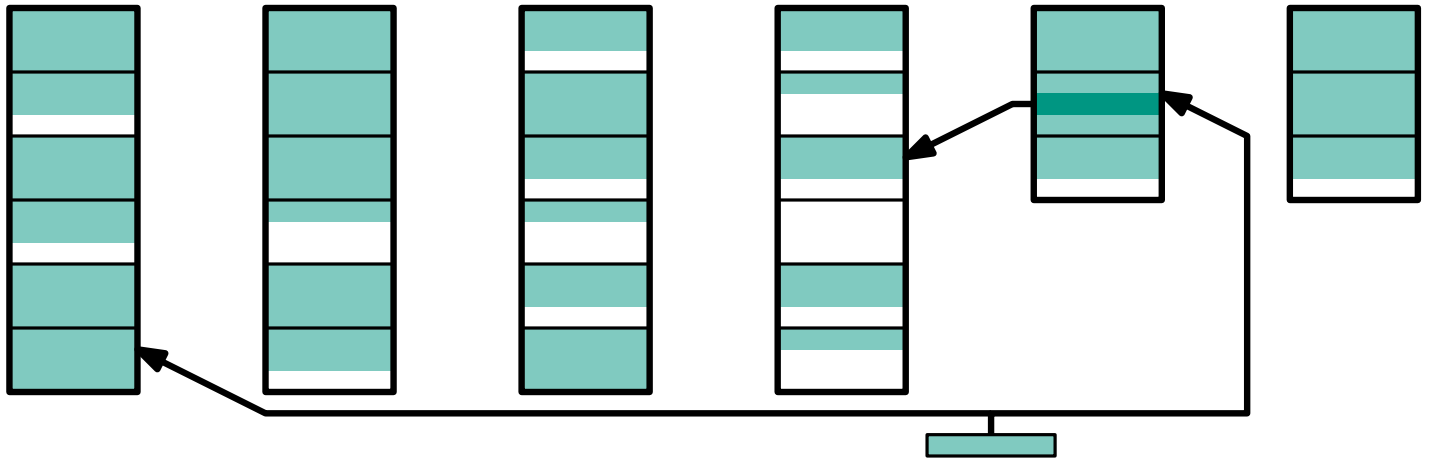
- Verwende Subtabellen ungleicher Größe
- Verwende die Verdrängungsstrategie um Ungleichgewichte auszugleichen



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

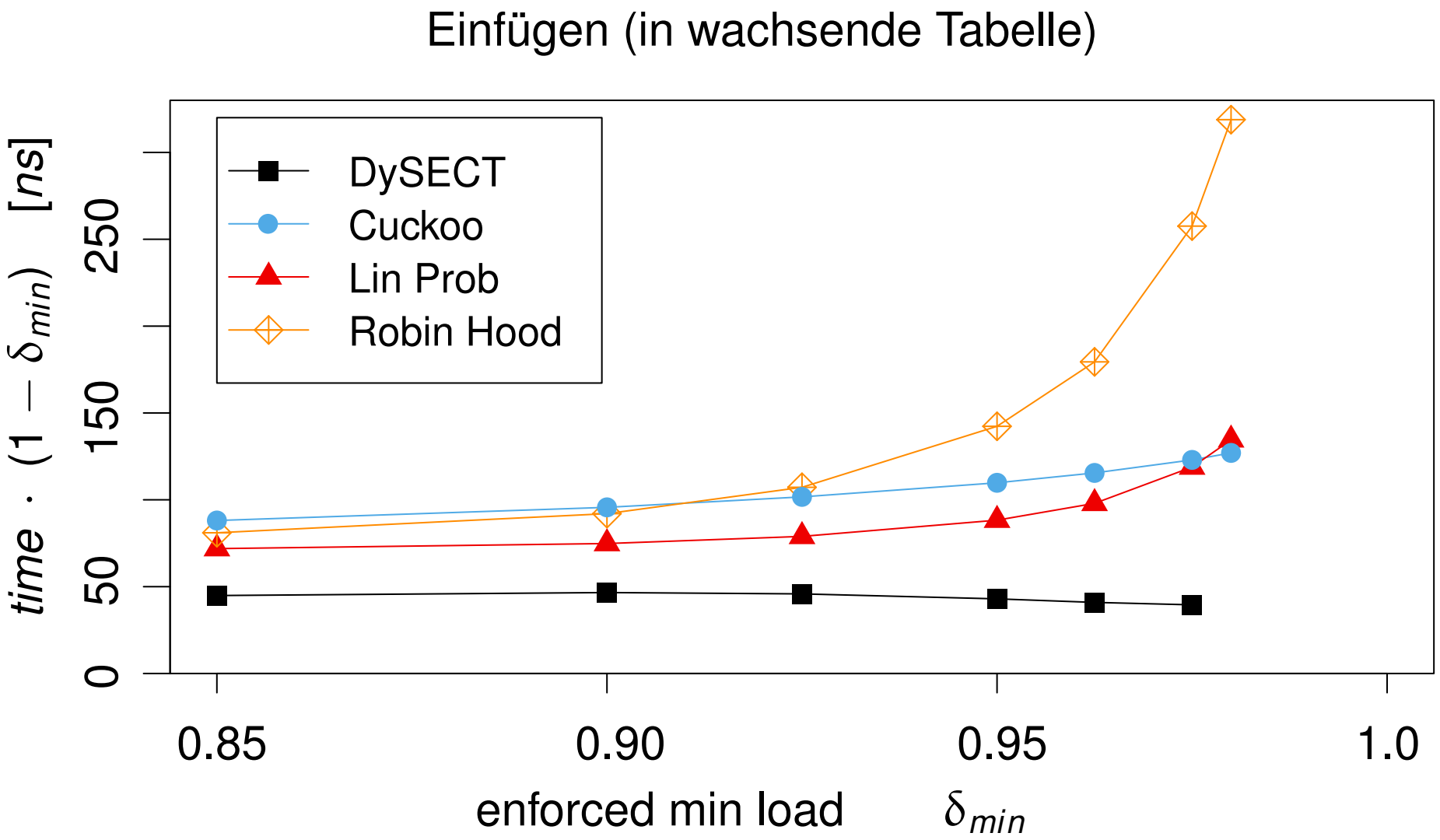
DySECT (Dynamic Space Efficient Cuckoo Table)

- Verwende Subtabellen ungleicher Größe
- Verwende die Verdrängungsstrategie um Ungleichgewichte auszugleichen



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

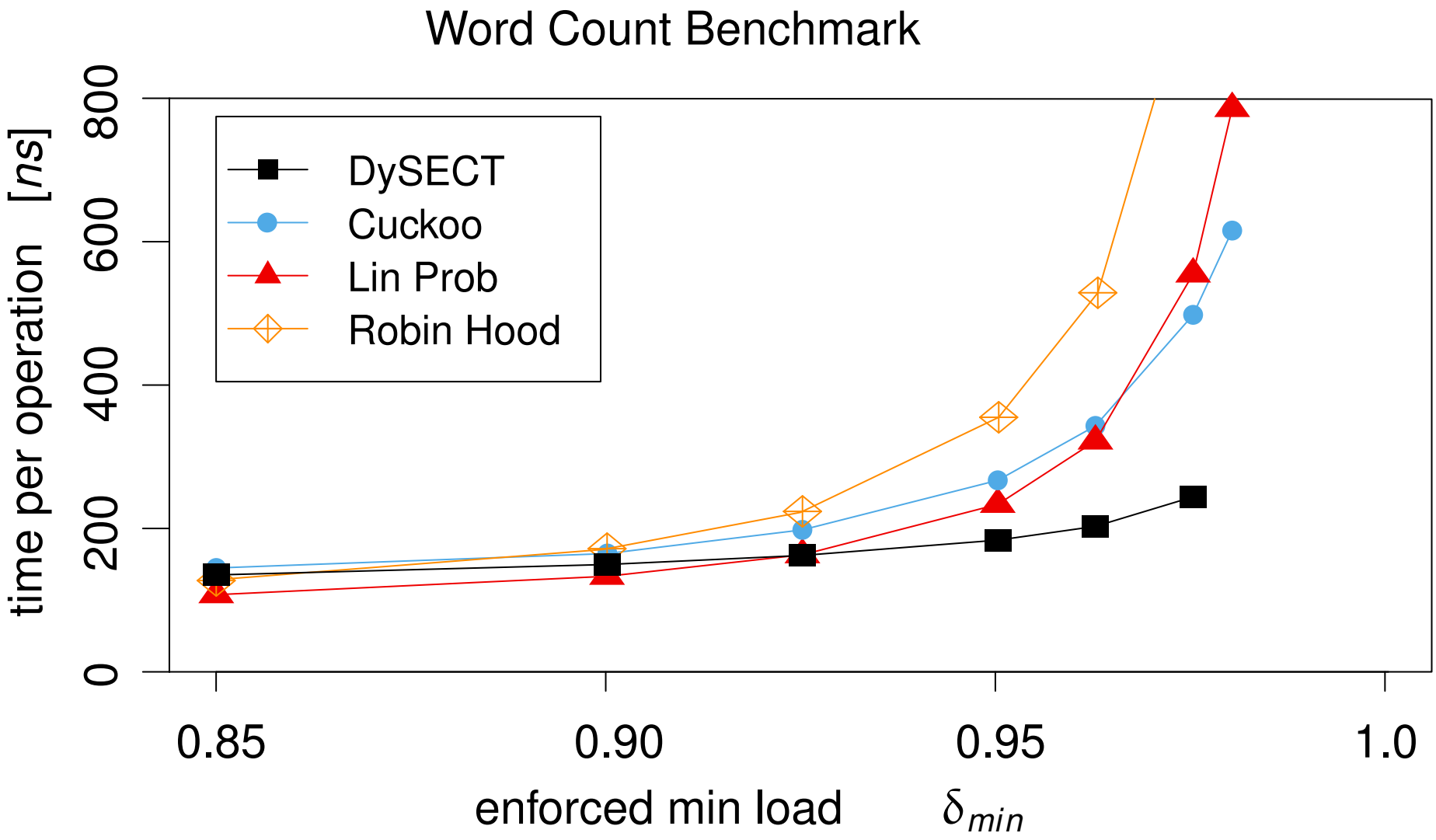
Einige Ergebnisse



Fazit

- Hash Tabellen gehören zu den häufigsten Datenstrukturen
- Gute Kollisionsauflösung erlaubt dicht gefüllte Tabellen
- Nur wachsende Tabellen ermöglichen Platzeffizienz bei unbekanntem n

Praktisch Motivierter Test



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Einige Gedanken zu Parllelisierung

- Parallele Datenstrukturen für Informationsaustausch
- Große Tabelle $\Rightarrow$ viele unabhängige Zugriffe
- Wachsende Tabellen benötigen Koordination
  - Häufiger Zugriff auf die selben Variablen
  - häufig vermeidbar durch lokale Duplizierung
- Beschränkt durch Speicherbandbreite