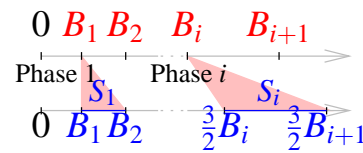


## Vorlesung

# Fortgeschrittene Algorithmen und Datenstrukturen

Wintersemester 2002/2003

Dr. Sven O. Krumke (krumke@zib.de)



## INHALT

In der Kombinatorischen Optimierung treten beim Design von Algorithmen viele „elementare Probleme“ auf. Mit Hilfe von geeigneten Datenstrukturen für diese „elementaren Probleme“ läßt sich der gesamte Algorithmus oft (theoretisch *und* auch in der Praxis) deutlich beschleunigen.

Die Vorlesung bietet anhand von ausgewählten Themen einen Einblick in moderne Datenstrukturen und Algorithmen sowie ihre Analyse. Dabei werden die einzelnen Datenstrukturen nicht isoliert behandelt, sondern stets im Zusammenhang mit konkreten Fragestellungen aus der Kombinatorischen Optimierung vorgestellt. Unter anderem werden wir folgende Themen in der Vorlesung behandeln:

**Amortisierte Analyse.** Informell gesprochen wird bei der amortisierten Analyse wird die Laufzeit eines Algorithmus für eine Folge von Operationen nicht separat für jede einzelne Operation, sondern für die gesamte Folge analysiert. Mit Hilfe der amortisierten Analyse und Potentialfunktionen sind oft verbesserte und aussagekräftigere Laufzeitabschätzungen für Algorithmen möglich. Die amortisierte Analyse wird eines der grundlegenden Hilfsmittel für die Analyse der „fortgeschrittenen Datenstrukturen und Algorithmen“ sein.

**Fibonacci-Heaps und Binomial-Heaps.** Fibonacci-Heaps senken etwa bei kürzeste-Wege-Algorithmen (Dijkstra) und Algorithmen für Minimale Aufspannende Bäume (Prim) die Laufzeit drastisch. Mit Hilfe von Fibonacci-Heaps lassen sich die Algorithmen von Dijkstra und Prim so implementieren, daß sie in  $O(m + n \log n)$  Zeit auf Graphen mit  $n$  Ecken und  $m$  Kanten laufen. Wir werden zeigen, wie man dies noch weiter steigern kann, indem wir einen Minimalbaum-Algorithmus mit Laufzeit  $O(n + m\beta(m, n))$  vorstellen, wobei  $\beta(m, n) = \min\{i : \log^{(i)} n \leq m/n\}$ .

**Union-Find Strukturen.** Union-Find Strukturen kommen dann in Spiel, wenn man effizient Partitionen einer Menge verwalten möchte, beispielsweise die Zusammenhangskomponenten eines Graphen im Algorithmus von Kruskal. Die Anwendung von geeigneten (gar nicht so komplizierten) Strukturen ermöglicht es, Kruskals Algorithmus in Zeit  $O(m \log m + m\alpha(m, n))$

laufen zu lassen. Hier bezeichnet  $\alpha(m, n)$  die inverse Ackermann Funktion, welche für alle Zahlen, die kleiner als die Anzahl der Atome im Universum sind, nach oben durch fünf beschränkt ist.

**Splay-Trees.** Zur Konstruktion von optimalen statischen Suchbäumen (etwa zur Datenkompression) benutzt man den Algorithmus von Huffman. Wenn die zu organisierenden Daten jedoch erst im Verlauf des Algorithmus (online) verfügbar werden, ist die statische Methode nicht anwendbar. Splay-Trees ermöglichen es mit einfachen Ideen (aber einer nichttrivialen Analyse), bis auf einen konstanten Strafterm die selben Schranken wie in statischen Fall auch für den dynamischen Fall zu bekommen. Anwendungen findet man vor allem in der Datenkompression.

**Dynamic Trees.** Dynamische Bäume (Dynamic Trees) sind ein Kniff, um etwa aus schnellen Maximalfluß-Algorithmen noch das Letzte an Laufzeit herauszuholen.

**Randomisierte Algorithmen.** Randomisierte Algorithmen entschärfen viele *Worst-Case*-Szenarien. Oft kann man einen randomisierten Algorithmus finden, der viel einfacher und zudem (im Erwartungswert) schneller ist als der bestmögliche deterministische Algorithmus.

#### FORMALITÄTEN

*Anrechenbarkeit:* 2 Semesterwochenstunden.

*Übungen:* keine.

*Schein:* Teilnahmebescheinigungen können ausgestellt werden.

#### ZIELGRUPPE UND VORAUSSETZUNGEN

Die Veranstaltung richtet sich an Studenten der Mathematik und Informatik im Grund- und Hauptstudium. Grundkenntnisse in der kombinatorischen Optimierung sind hilfreich, aber nicht erforderlich.

#### FORTSETZUNG

Bei ausreichendem Interesse kann ein Seminar im Sommersemester 2002 angeboten werden.

#### LITERATUR

1. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Networks flows*, Prentice Hall, Englewood Cliffs, New Jersey, 1993.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, 2 ed., MIT Press, 2001.