

Frank Hoffmann
Klaus Kriegel
Romain Grunert
Ludmila Scharf
André Schulz

Ws 2007/08

25. Januar 2008
Abgabe: 4. Februar 2008
vor der Vorlesung

13. Übung zur Vorlesung Höhere Algorithmik

1. Aufgabe (5 Punkte).

Im Load Balancing Problem steht für die Bearbeitung einer Menge von Jobs festgelegter Dauer eine vorgegebene Anzahl von Maschinen zur Verfügung. Im bisher betrachteten Problem hatte jede dieser Maschinen die gleiche Bearbeitungsgeschwindigkeit. Wie löst man dieses Problem, wenn die Maschinen unterschiedlich schnell sein können?

Betrachten Sie dazu folgende Situation: Es gibt m langsame und k schnelle Maschinen. Die schnellen Maschinen sind doppelt so schnell wie die langsamen Maschinen, d. h. wenn Job i auf einer langsamen Maschine Zeit t_i benötigt, so wird er von einer schnellen Maschine in Zeit $\frac{1}{2}t_i$ abgearbeitet. Gesucht ist wieder eine Verteilung der Jobs auf die Maschinen, die die Bearbeitungszeit der am längsten arbeitenden Maschine minimiert.

Passen Sie den aus der Vorlesung bekannten Approximationsalgorithmus für das ursprüngliche Problem so an das abgewandelte Problem an, dass er in polynomieller Zeit eine Lösung berechnet, die höchstens dreimal so groß ist wie die optimale Lösung.

2. Aufgabe (9 Punkte).

Eine Studentengruppe soll folgendes Problem lösen: Gegeben sind zwei aufsteigend sortierte n -elementige Mengen $A, B \subseteq \mathbb{R}$, wobei A aus $a_1 < a_2 < \dots < a_n$ und B aus $b_1 < b_2 < \dots < b_n$ besteht. Die Menge A soll so um einen Wert $r \in \mathbb{R}$ verschoben werden, dass die paarweisen Abstände zu B minimiert werden, d. h. die Zahl r soll so gewählt werden, dass der Wert $D(r) = \max\{|a_i + r - b_i| \mid 1 \leq i \leq n\}$ minimiert wird.

- André hat die Minimum-Heuristik implementiert, bei der man $r = b_1 - a_1$ setzt. Zeigen Sie, dass dies eine 2-Approximation ist. Ergänzend bemerkt André, dass man nicht unbedingt die Differenz der ersten beiden Werte nehmen muss. Jede Wahl $r_i = b_i - a_i$ ergibt eine 2-Approximation.
- Romain hat die Schwerpunkt-Heuristik mit $r = \frac{\sum_{i=1}^n b_i}{n} - \frac{\sum_{i=1}^n a_i}{n}$ implementiert und hat in Testläufen festgestellt, dass sie besser als die Minimum-Heuristik arbeitet. Stimmt das? Welchen Approximationsfaktor hat diese Heuristik? Beweisen Sie, dass der von Ihnen angegebene Approximationsfaktor exakt ist.
- Ludmila kritisiert, dass die Schwerpunkt-Heuristik lineare Laufzeit benötigt und man in dieser Zeit bereits die optimale Lösung berechnen kann. Zeigen Sie, dass Ludmila Recht hat, indem Sie für r den Wert $\frac{1}{2}(\max_{1 \leq i \leq n} r_i + \min_{1 \leq i \leq n} r_i)$ betrachten.

3. Aufgabe (6 Punkte).

In der Vorlesung wurde der Algorithmus *Vertex-Cover-Approx* zur Approximation eines optimalen Vertex-Covers vorgestellt.

- (a) In der while-Schleife dieses Algorithmus werden Kanten $\{i, j\}$ gesucht, für die weder i noch j fest sind, und der Preis so weit angehoben, bis einer der beiden Knoten fest ist. Beschreiben Sie genauer, wie man diese Auswahl und die Anhebung der Preise organisieren kann, um eine möglichst gute Laufzeit zu bekommen. Ihre Lösung sollte eine Laufzeitschranke von $O(n^2 \log m)$ nicht überschreiten, wobei n die Anzahl der Knoten und m die Anzahl der Kanten im Graphen ist.
- (b) Verallgemeinern Sie die Idee dieses Algorithmus für Set-Cover Probleme mit der Eigenschaft, dass jedes $s \in U$ von höchstens drei Mengen des Systems S_1, \dots, S_m überdeckt wird. Welcher Approximationsfaktor lässt sich dadurch erreichen?