Prof. Dr. Volker Kaibel
Dipl.-Comp.-Math. Matthias Walter
Wintersemester 2012/2013

2. Übung Kombinatorische Optimierung

www.math.uni-magdeburg.de/institute/imo/teaching/kombopt2012/

Präsentation in den Übungen am 25.10.2012

Aufgabe 1

Sei $D = (V, A)$ ein Digraph mit konservativen Bogengewichten c. In der Vorlesung wurde zur Berechnung kürzester Wege zwischen allen Knootenpaaren vorgeschlagen, ein Potential π zu berechnen und dann mit Hilfe von $n = |V|$ Anwendungen des Dijkstra-Algorithmus für veränderte Bogengewichte $\bar{c}_{u,v} = c_{u,v} + \pi_u - \pi_v$ das Problem in $\mathcal{O}(|V| \cdot |A| + |V|^2 \log |V|)$ Zeit zu lösen.
Das Problem lässt sich aber auch etwas einfacher mit dem Floyd-Warshall-Algorithmus lösen:

Algorithmus : Floyd-Warshall
Eingabe : $D = (V, A)$, $c \in \mathbb{Q}^A$
Ausgabe : Distanzen $d_{u,v}$ und letzte Bögen $(p_{u,v}, v)$ der kürzesten u-v-Wege

1 Initialisiere $d \in \mathbb{Q}^{V \times V}$ mit $d_{u,v} := \begin{cases} c_{u,v} & (u,v) \in A \\ 0 & u = v \\ \infty & \text{sonst} \end{cases}$.
2 Initialisiere $p \in V^{V \times V}$ mit $p_{u,v} \longleftarrow u$.
3 **for** $k := 1, \ldots n$ **do**
4 **for** $u := 1, \ldots n$ **do**
5 **for** $v := 1, \ldots n$ **do**
6 **if** $d_{u,v} > d_{u,k} + d_{k,v}$ **then**
7 $d_{u,v} \longleftarrow d_{u,k} + d_{k,v}$
8 $p_{u,v} \longleftarrow p_{k,v}$
9 **end**
10 **end**
11 **end**
12 **end**

1. Beweisen Sie die Korrektheit des Algorithmus, d.h., nach Ablauf enthält $d_{u,v}$ die Länge eines c-kürzesten u-v-Weges, sofern ein solcher existiert, und $(p_{u,v}, v)$ den letzten Bogen eines solchen Weges.

2. Geben Sie die Laufzeitfunktion an.

Bitte wenden!

Aufgabe 2

1. Betrachten Sie $\overline{\mathbb{Q}} := \mathbb{Q} \cup \{\infty\}$ als Halbring mit der Minimierungsfunktion als Addition und der normalen Addition als Multiplikation. Sei $M \in \overline{\mathbb{Q}}^{n \times n}$ eine Matrix über diesem Halbkörper. Beschreiben Sie die Einträge von M^2.

2. Angenommen, wir wollen Kürzeste-Wege-Distanzen zwischen allen Knotenpaaren mit Hilfe des Bellman-Ford Algorithmus lösen. Stellen Sie den dafür notwendigen Algorithmus auf (am besten in einer Form wie in Aufgabe 1). Betrachten Sie dabei den Bellman-Ford Algorithmus als Versuch, Kürzeste-Wege-Distanzen über höchstens t Kanten zu ermitteln, wobei schließlich über t iteriert wird.

3. Stellen Sie nun Ihren Algorithmus in Form von Matrixoperationen über $\overline{\mathbb{Q}}$ dar. Leiten Sie aus dieser Darstellung eine Beschleunigung her, die $\mathcal{O}(n^3 \log n)$ Zeit benötigt.

Aufgabe 3

Gegeben sei ein $0/1$-Rucksack-Problem mit Gewichten $a \in \mathbb{N}^n$, Gewichtsschranke β, und Profiten $c \in \mathbb{N}^n$. Konstruieren Sie einen azyklischen Graphen, mit dem sie das Problem als Kürzeste-Wege Problem lösen können. Die Laufzeit soll dabei polynomiell in n und $\sum_{i=1}^{n} c_i$ sein.
Hinweis: Ihr Algorithmus darf die Lösung des aufgestellten Kürzeste-Wege Problems noch untersuchen, bevor er seinerseits die Optimallösung des Rucksack-Problems ausgibt.

Aufgabe 4

Gegeben sei eine Zahl m von identischen Maschinen und eine Liste von n Aufgaben $(c_1, \ldots, c_n)$, die durch ihre Bearbeitungszeit $c_i \in \mathbb{N}$ gegeben sind. Eine Maschine kann genau eine Aufgabe gleichzeitig bearbeiten. Gesucht ist eine Zuordnung der Aufgaben auf die Maschinen, so dass die Gesamtbearbeitungszeit, also die Zeit, bis alle Aufgaben abgearbeitet sind, minimiert wird. Lösen Sie das Problem mit Hilfe von dynamischer Programmierung, d.h. stellen Sie einen azyklischen Graphen auf, in dem Sie das Problem durch eine Kürzeste-Wege Berechnung lösen können. Was ist die Laufzeit Ihres Algorithmus?