RWTH Aachen | **SS 2010**
Lehrgebiet Theoretische Informatik | Übungsblatt 3
Kneis—Langer—Rossmanith | 5.05.2010

Übung zur Vorlesung Effiziente Algorithmen

Tutoraufgabe 7

Ein Gemischtwarenladen will durch einen besonderen Rabatt die Umsätze steigern: Wer an der Kasse zwei Artikel zum Kauf vorlegt, deren Gesamtpreis auf 11, 33, 55, 77 oder 99 Cent endet, erhält zusätzlich einen Gutschein in Höhe des erreichten Centbetrages.

Entwerfen Sei einen Algorithmus, der zu einer Menge von Preisen eine optimale Einkaufsstrategie angibt.

Lösung:

Seien $X = \{x_1, \ldots, x_n\}$ die Preise der Gegenstände. Das Problem läßt sich leicht als Matchingproblem darstellen. Wir konstruieren einen bipartiten Graphen G mit den folgenden Knoten $U = \{x_i \in X \mid x_i \text{ ist ungerade}\}$ und $G = \{x_i \in X \mid x_i \text{ ist gerade}\}$. Wir verbinden $x_i \in U$ mit $x_j \in G$, falls der Centbetrag c von $x_i + x_j$ entweder in 11, 33, 55, 77 oder in 99 ist. Die Kapazität der entsprechende Kante setzen wir auf c.

Es ist leicht einzusehen, daß ein maximales gewichtetes Matching in G genau eine Lösung unseres Problems darstellt. Formal wäre zu zeigen, daß jedes Matching eine Lösung unseres Problems darstellt und anders herum. Auf Grund der Konstruktion ist dies jedoch offensichtlich.

Tutoraufgabe 8

Wir wollen zehn Kilogramm Schokolade mit 70% Kakaoanteil fabrizieren. Zur Verfügung stehen die vier Sorten Kneis, Langer, Rossmanith und Sikdar. Sie haben die folgenden Daten:

Sorte	Kakaoanteil	Preis pro kg	Verfügbarkeit
Kneis	24%	EUR 3,70	beliebig
Langer	40%	EUR 4,00	beliebig
Rossmanith	73%	EUR 5,00	4 kg
Sikdar	80%	EUR 6,00	beliebig

Finden Sie jeweils eine Lösung mit minimalen bzw. maximalen Kosten. Formulieren Sie diese Aufgabe als lineares Programm.

Ansatz:

Wir verwenden eine Syntax, wie sie z.B. auf der Webseite `http://algos.inesc.pt/lp/` genutzt wird:

```
min: 3.7x1+4x2+5x3+6x4;
x1+x2+x3+x4=10;
x1>=0;
x2>=0;
x3>=0;
x3<=4;
x4>=0;
24x1+40x2+73x3+80x4=700;
```

Für die Maximierung sind nur leichte Änderungen notwendig.

Tutoraufgabe 9
Entwerfen Sie eine Datenstruktur, die es dem Preflow-Push-Algorithmus erlaubt

- in konstanter Zeit einen überfließenden Knoten zu finden,
- in konstanter Zeit zu bestimmen, ob eine Push- oder Lift-Operation, anwendbar ist
- eine Lift-Operation in $O(|V|)$ Schritten durchzuführen und
- eine Push-Operation in konstanter Zeit durchzuführen.

Lösung:
Wir können davon ausgehen, daß die Nachbarn eines Knotens als Adjazensliste gespeichert sind. Diese Listen wiederum sind in einem Array gespeichert.
Desweiteren nutzen wir eine

- Liste aller überfließender Knoten
- zu jedem Knoten v eine Liste L(v) aller Nachbarn u von v mit $h(v) = h(u) + 1$
- Liste mit überfließenenden Knoten, mit L(v) ¿ 0
- Liste mit Knoten die liftbar sind
- Liste mit Knoten die liftbar wären, wenn sie nicht überfließend
- zu jedem Knoten Anzahl der Kanten nach unten

Hausaufgabe 5 (10 Punkte)
Entwerfen Sie einen Algorithmus, der das folgende Problem in polynomieller Zeit löst:

Eingabe: Ein gewichteter Graph $G = (V, E, c)$ mit $c\colon E \to \mathbb{R}$‹$_0$
Frage: Ein Pfad von s nach t mit geringstem Gewicht w, sodaß es keinen Pfad von s nach t mit Gewicht w gibt, der kürzer ist.

Beweisen Sie sowohl die Korrektheit als auch eine möglichst gute Laufzeitschranke ihrer Lösung.

Lösung:
Wir definieren eine neue Kostenfunktion $c'\colon E \to (\mathbb{R}_{\geq 0} \times \mathbb{N})$ mit $c'(e) = (c(e), 1)$ und erweitern dies auf die natürliche Weise für Pfade indem wir komponentenweise addieren.
Desweiteren setzen $(a, b) <_{c'} (a', b')$ falls $a < a'$ oder $a = a'$ und $b < b'$.
Mit diesem Maß können wir nun leicht den Abstand bezüglich c' zweier Knoten s, t definieren, als die Kosten des Pfades P von s nach t mit gerinstem $c'(P)$.
Offensichtlich ist eine Lösung der Aufgabe genau ein Pfad P von s nach t mit geringstem $c'(P)$. Er kann nun leicht durch Anwenden von Dijstra's Algorithmus mit Kostenfunktion c' und Vergleichsoperation $<_{c'}$ gefunden werden.

Hausaufgabe 6 (10 Punkte)
Gegeben ist folgendes Problem:
Flat-Rate
Eingabe: Ein s-t-Netzwerk G, eine Kostenfunktion $w\colon E \to \mathbb{R}_{\geq 0}$, $a, b \in \mathbb{R}$
Frage: Gibt es einen Fluss von s nach t der Größe a mit "Flat-Rate-Kosten" höchstens b?
Beweisen Sie, daß dieses Problem in P ist oder daß es NP-schwer ist.
Die "Flat-Rate-Kosten" eines Flusses f sind $\sum_{e \in E, f(e) > 0} w(e)$. Bei Bedarf findet sich auf der Homepage ein Hinweis zur Lösung.

Lösung:
Wir zeigen mittels einer Reduktion von Vertex Cover, daß dieses Problem NP-schwer ist.
Vertex Cover $\leq$ Flat-Rate
Sei $(G = (V, E), k)$ eine Eingabe für Vertex Cover. Wir konstruieren eine Eingabe $G', a = |E|, b = k$ für Flat-Rate wie folgt:
Als Knotenmenge V' wählen wir $\{s, t\} \cup V \cup E$. Dann fügen wir folgende Kanten ein

- (s, v) mit Kapazität $|V|$ und Kosten 1 für alle $v \in V$
- (v, e) mit $v \in V$, $e \in E$, e inzident zu v mit Kapazität 1 und Kosten 0
- (e, t) mit Kapazität 1 und Kosten 0 für alle $e \in E$.

Wir zeigen nun daß G genau dann ein Vertex Cover der Größe k hat, wenn es in G' einen Fluss der Größe $|E|$ mit Kosten höchstens k gibt.
Sei $V' \subseteq V$ ein Vertex Cover. Sei $E_v \subseteq E$ die Menge der Kanten in E, die durch v gecovert werden. Falls eine Kante durch zwei Knoten abgedeckt wird, wird sie nur zu einem E_v hinzugefügt.
Dann konstruieren wir den Fluss f in G' indem von s zu jedem $v \in V'$ genau $|E_v|$ fließen lassen. Weiter von jedem v zu allen Knoten in E_v jeweils 1 mit Kosten 0 und von jedem e zu t ebenfalls 1 mit Kosten 0. Offensichtlich ist dies eine gültige Lösung der Größe $|E|$ mit Kosten k.
Sei nun f ein Fluss der Größe $|E|$ mit Kosten höchstens k in G'. Offensichtlich fließt in jeden Knoten $e \in E$ genau 1, da sonst kein Fluss der Größe $|E|$ möglich wäre. Sei nun V' die Menge aller Knoten, über die ein Fluß zu Knoten in E fließt. Offensichtlich ist jeder Knoten in E adjazent zu einem Knoten in V', also ist V' ein Vertex Cover. Da die Kosten des Flusses höchstens k betragen, folgt außerdem $|V'| \leq k$.