Prof. Dr. Volker Kaibel
M.Sc. Jonas Frede
Wintersemester 2018/2019

Kombinatorische Optimierung – Blatt 6

www.math.uni-magdeburg.de/institute/imo/teaching/wise18/kombopt/

Präsentation in der Übung am 22.11.2018

Aufgabe 1

Wir betrachten das Problem, in einem bipartiten ungerichteten Graphen $G = (V \cup W, E)$ mit Kantenkosten $c \in \mathbb{Q}^E$ ein gewichtsminimales perfektes Matching zu finden. Formuliere das Problem als Min-Cost-Flow Problem (d.h. als das Problem, eine kostenminimale Zirkulation in einem geeigneten Netzwerk zu finden)!

Aufgabe 2

Bei einem *Terminal Assignment Problem* ist eine zentrale CPU gegeben, eine Menge von Konzentratoren $\{K_1, \dots, K_p\}$ und eine Menge von Terminals $\{T_1, \dots, T_q\}$. Ein Terminal T_j kann direkt an die CPU zu Kosten $w_{0,j}$ angeschlossen werden oder zu Kosten $w_{i,j}$ an den Konzentrator K_i, von dem aus die weitere Verbindung zur CPU erfolgt (wir nehmen an, dass diese Verbindung zwischen Konzentrator und CPU bereits hergestellt ist und keine neuen Kosten verursacht). Ein Konzentrator kann höchstens $C \in \mathbb{N}$ viele Terminals bedienen.
Formuliere das Problem, alle Terminals an die CPU zu minimalen Kosten anzuschließen, als ein Problem, eine minimal gewichtete Zirkulation in einem geeigneten Netzwerk mit ganzzahligen Kapazitäten zu finden.

Aufgabe 3

Für $b \in \mathbb{Z}^V$ ist ein *b-Fluss* in einem Netzwerk $(D = (V, A), u)$ mit $u \in \mathbb{Z}_+^A$ ein Fluss $f \in \mathbb{Q}^A$ mit $\mathbb{O}_A \le f \le u$ und $\mathrm{ex}_f(v) = b_v$ für alle $v \in V$.
Wir betrachten folgenden Algorithmus für das *Min-Cost-Flow Problem*, für konservative Bogengewichte $c \in \mathbb{Q}^A$ einen b-Fluss minimaler c-Kosten $\sum_{a \in A} c_a f_a$ zu finden.

Algorithm: SuccessiveShortestPath

Eingabe: $D = (V, A)$, $u \in \mathbb{Z}_+^A$, $b \in \mathbb{Z}^V$ mit $\sum_{v \in V} b_v = 0$, $c \in \mathbb{Q}^A$ konservativ.

Ausgabe: c-minimaler b-Fluss in (D, u).

$b^0 \longleftarrow b$, $f^0 \longleftarrow \mathbb{O}_A$, $k \longleftarrow 0$
while $b^k \neq \mathbb{O}_V$ **do**
  Wähle Knoten $s \in V$ mit $b^k(s) < 0$.
  if *$\exists t \in V$ mit $b^k(t) > 0$, t von s aus in D_{f^k} erreichbar* **then**
    Finde c-kürzesten s-t-Weg P in D_{f^k}.
    Sei $\gamma := \min\left\{-b^k(s), b^k(t), \min_{a \in P} \overline{u}_a\right\}$.   `// ` $\overline{u}$ `sind Residualkapazitäten`
    $f^{k+1} \longleftarrow f^k$ augmentiert entlang von P um Flusswert γ
    $b^{k+1} \longleftarrow b^k$, $b^{k+1}(s) \longleftarrow b^{k+1}(s) + \gamma$, $b^{k+1}(t) \longleftarrow b^{k+1}(t) - \gamma$, $k \longleftarrow k + 1$
  end
  else return *“Es gibt keinen b-Fluss”.*
end
return f^k

Beweise folgende Aussagen:

(a) Der Algorithmus terminiert nach höchstens $B := \frac{1}{2} \sum_{v \in V} |b_v|$ Augmentierungen.

(b) Der Algorithmus terminiert genau dann in Zeile 10, wenn es keinen b-Fluss in (D, u) gibt. Setze dazu f^k mit $b - b^k$ in Relation.

(c) Sei f ein c-minimaler b-Fluss und P ein c-minimaler s-t-Weg in D_f und sei f' durch Augmentieren von f entlang von P um die kleinste auf P auftretende Residualkapazität γ entstanden. Dann ist f' ein c-minimaler b'-Fluss.

Hinweis: Wenn f' nicht minimal ist, gibt es einen negativen Kreis. Betrachte diesen und die Bögen von P, um einen kürzeren s-t-Weg zu konstruieren.

(d) Wenn der Algorithmus in Zeile 12 terminiert, dann berechnet er einen c-minimalen b-Fluss.