Technische Universität München, Zentrum Mathematik
Lehrstuhl für Angewandte Geometrie und Diskrete Mathematik

Algorithmische Diskrete Mathematik (MA 2501)

Prof. Dr. P. Gritzmann | Dr. M. Ritter | Dipl.-Math. M. Silbernagl

Aufgabenblatt 1

Tutoraufgabe 1.1 (Baum-Äquivalenzen, Teil 1) [7 Punkte]
Sei $G = (V, E)$ ein Graph und $n := |V|, m := |E|$. Zeigen Sie die Äquivalenz der folgenden Aussagen:

(1) G ist ein Baum.

(2) G ist zusammenhängend und $m = n - 1$.

(3) G ist kreisfrei und $m = n - 1$.

Aufgabe 1.2 (Baum-Äquivalenzen, Teil 2) [4 Punkte]
Sei $G := (V, E)$ ein Graph. Zeigen Sie die Äquivalenz der folgenden Aussagen:

(1) G ist ein Baum.

(2) Für je zwei Knoten v,$w \in V$ existiert genau ein v-w-Weg in G.

(3) G ist zusammenhängend, aber für kein $e \in E$ ist $(V, E \setminus \{e\})$ zusammenhängend.

(4) G ist kreisfrei, aber für alle v,$w \in V$ mit $v \neq w$ und $e := \{v, w\} \notin E$ enthält $(V, E \cup \{e\})$ einen Kreis.

Aufgabe 1.3 (Greedy-Algorithmus für das Knapsack-Problem) [5 Punkte]
Ein Lastwagen soll mit n Gütern beladen werden. Gut i hat Gewicht $w_i \geq 0$ und Wert $v_i > 0$, $i = 1, \ldots, n$. Das *Knapsack*-Problem besteht darin, den Wagen so zu beladen, dass der Wert der Ladung maximiert wird und das Gesamtgewicht der Ladung höchstens $C \geq 0$ Gewichtseinheiten beträgt (es gelte $w_i \leq C$). Dabei wird unterschieden, ob von jedem Gut

(1) genau ein Exemplar vorliegt (*0-1-Knapsack Problem*),

(2) unbegrenzt viele Exemplare vorliegen (*Integer Knapsack Problem*).

Betrachten Sie für diese Fälle nun folgende Greedy-Heuristik:

Die Güter seien so sortiert, dass $v_1 \geq \ldots \geq v_n$ gilt. Die Beladung erfolgt gemäß der Vorschrift:

0-1-Knapsack Problem:

for $i = 1$ *to* n **do**
  if $w_i \leq C$ **then**
    Lade Gut i und setze $C \leftarrow C - w_i$.
  else
    Lade Gut i nicht.

Integer Knapsack Problem:

for $i = 1$ *to* n **do**
  Lade $\left\lfloor \frac{C}{w_i} \right\rfloor$ Stück von Gut i.
  Setze $C \leftarrow C - \left\lfloor \frac{C}{w_i} \right\rfloor w_i$.

Zeigen Sie, dass in beiden Fällen diese Greedy-Heuristik beliebig schlecht arbeitet, d. h. für jedes $\varepsilon > 0$ existiert ein Beispiel, so dass für das Optimum z_{Opt} und den Zielfunktionswert z_{Greedy} aus dem Algorithmus $z_{Greedy}/z_{Opt} \leq \varepsilon$ gilt.

Bitte wenden!

Aufgabe 1.4 (Verbesserter Greedy-Algorithmus für das Knapsack-Problem) [10 Punkte]
Wir betrachten wieder das Knapsack-Problem mit den Bezeichnungen aus der vorigen Aufgabe.

a) Nehmen Sie an, die zu ladenden Güter aus der letzten Aufgabe könnten auch nur teilweise geladen werden – beispielsweise Säcke mit verschiedenen Getreidesorten. Ein Gut i, welches zu einem Anteil $\lambda \in [0,1]$ geladen wird, soll dabei λw_i zum geladenen Gewicht und λv_i zum geladenen Wert beitragen. Geben Sie eine geeignete Sortierung und einen einfachen Algorithmus an, der mit dieser zusätzlichen Annahme das 0-1-Knapsack-Problems optimal löst. Ein Beweis ist nicht nötig.

Im Folgenden seien die Güter *nicht* teilweise ladbar, dafür aber im Gegensatz zur vorigen Aufgabe so sortiert, dass $\frac{v_1}{w_1} \geq \ldots \geq \frac{v_n}{w_n}$ gilt. Zeigen Sie mit dieser Sortierung:

b) Die Greedy-Heuristik für das 0-1-Knapsack Problem arbeitet beliebig schlecht.

c) Sei z_{Greedy} der Gesamtwert einer Lösung, die mit der Greedy-Heuristik für das Integer Knapsack Problem ermittelt wurde und z_{opt} der Wert einer optimalen Lösung. Zeigen Sie:

$$\frac{z_{\text{Greedy}}}{z_{\text{opt}}} \geq \frac{1}{2}$$

d) Wir modifizieren die Greedy-Heuristik für das 0-1-Knapsack Problem folgendermaßen:

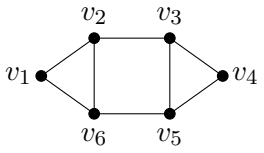
for $i = 1$ *to* n **do**
  if $w_i \leq C$ **then**
    Lade Gut i, setze $C \leftarrow C - w_i$ und $k \leftarrow i$.
  else
    Beende die Schleife.
if $\sum_{i=1}^{k} v_i < v_{k+1}$ **then**
  Entlade den Lastwagen komplett und lade Gut $k+1$.

Sei z'_{Greedy} der Gesamtwert der Lösung, die mit diesem Algorithmus berechnet wird. Zeigen Sie:

$$\frac{z'_{\text{Greedy}}}{z_{\text{opt}}} \geq \frac{1}{2}$$

Hinweis: Benutzen Sie Ihr Ergebnis aus Aufgabenteil a).

Aufgabe 1.5 (Eulertouren) [5 Punkte]
Es sei G folgender Graph:



a) Wie viele Kanten muss man zu G hinzufügen, damit es eine Eulertour gibt?

b) Kann man zu jedem zusammenhängenden Graphen Kanten hinzufügen, damit es eine Eulertour gibt? Geben Sie entweder einen Beweis oder ein Gegenbeispiel mit kurzer Begründung an.

c) Kann man zu jedem zusammenhängenden Graphen genau einen Knoten hinzufügen, und Kanten die den neuen Knoten als Endknoten haben, so dass es eine Eulertour gibt? Geben Sie entweder einen Beweis oder ein Gegenbeispiel mit kurzer Begründung an.

Die Abgabedaten von Blatt 1 für Ihre Gruppe finden Sie auf der Homepage zur Vorlesung unter `http://www-m9.ma.tum.de/WS2012/ADM`.