Optimierung für Studierende der Informatik

Wintersemester 2019/20
Blatt 10

A: Präsenzaufgaben am 8./9. Januar 2018

1. Beim Entwurf von Approximationsalgorithmen spielen neben LP-basierten Methoden auch *Greedy-Verfahren* eine wichtige Rolle. Im Folgenden wird ein Beispiel betrachtet, das dies illustriert.

Ein Schiff mit n Containern $1, 2, \ldots, n$ erreicht den Hafen. Die Abmessungen der Container spielen keine Rolle, es geht ums Gewicht: Container i habe das Gewicht $w_i > 0$ ($i = 1, \ldots, n$). Zum Weitertransport stehen Lastwagen bereit, von denen jeder einen oder mehrere Container aufnehmen kann. *Einzige Restriktion*: Es gibt eine Gewichtsschranke K, die für jeden Laster gilt, d.h., kein Lastwagen darf Container von einem Gesamtgewicht größer K aufnehmen. Die Schranke K und auch die Gewichte w_i sollen als ganzzahlig angenommen werden. Es gelte $K \geq w_i$ für $i = 1, \ldots, n$. Zu minimieren ist die Anzahl der Lastwagen, die zum Weitertransport aller Container benötigt werden. (Anmerkung: Das beschriebene Problem ist ein NP-schweres Optimierungsproblem.)
Ein Greedy-Algorithmus: Die Container werden in der Reihenfolge $1, 2, \ldots, n$ verladen, wobei immer nur ein Lastwagen zur Zeit beladen wird. Immer, wenn der nächste Container nicht mehr aufgeladen werden kann (wegen Überschreitung von K), wird ein Lastwagen für „voll“ erklärt und auf die Reise geschickt.

Mit $m^\star$ sei das optimale Ergebnis bezeichnet, d.h., $m^\star$ ist die minimale Anzahl der benötigten Lastwagen. Das Ergebnis, das der Greedy-Algorithmus liefert, werde mit m bezeichnet.

a) Belegen Sie anhand eines Beispiels, dass der Greedy-Algorithmus nicht immer das bestmögliche Ergebnis liefert. Mit anderen Worten: $m > m^\star$ ist möglich.

b) **Behauptung**: Es gilt immer $m < 2m^\star$. (Dies bedeutet, dass unser Greedy-Algorithmus gar nicht so schlecht ist: Es handelt sich um einen *2-Approximationsalgorithmus*.)

Zeigen Sie die Richtigkeit dieser Behauptung für den Fall, dass m ungerade ist.
Hinweis: L_i sei das Gewicht, das der Greedy-Algorithmus auf den i-ten Lastwagen packt ($i = 1, \ldots, m$). Welche naheliegende Feststellung lässt sich für die Summe $L_1 + L_2$ und die Schranke K treffen?

2. Im Skript wurde auf Seite 166 ein 2-Approximationsalgorithmus für das Knotenüberdeckungsproblem vorgestellt.

a) Beschreiben Sie kurz in eigenen Worten, worum es beim Knotenüberdeckungsproblem geht.

b) Beschreiben Sie kurz (ebenfalls in eigenen Worten), wie der erwähnte 2-Approximationsalgorithmus funktioniert.

c) Begründen Sie kurz, weshalb für die vom Algorithmus gelieferte Knotenüberdeckung U gilt: $|U| \leq 2c(G)$.

d) Was versteht man unter dem vollständig bipartiten Graphen $K_{n,n}$?

B: Hausaufgaben zum 15./16. Januar 2018

1. Wir betrachten das Lastwagenproblem aus Präsenzaufgabe 1 und verwenden die dort eingeführten Bezeichnungen.

a) Zeigen Sie die Richtigkeit der Behauptung aus Präsenzaufgabe 1b) für den Fall, dass m gerade ist.

b) Es sei $k \geq 2$ eine ganze Zahl. Zeigen Sie, dass es für jedes derartige k ein Beispiel gibt, für das $m^\star = k$ und $m = 2k - 1$ gilt.
Hinweis: Lösen Sie zunächst die Fälle $k = 2, 3, 4$ und orientieren Sie sich an diesen Fällen.

c) Begründen Sie kurz, weshalb aus b) folgt, dass unser Algorithmus *kein* γ-Approximationsalgorithmus für $\gamma < 2$ ist.

2. a) Auf Seite 166 des Skripts wurde im Zusammenhang mit dem Knotenüberdeckungsproblem die folgende Frage gestellt: *Ist es möglich, den gefundenen Faktor 2 zu verbessern, indem man den Algorithmus unverändert lässt, aber eine raffiniertere Analyse des Algorithmus vornimmt?*

Zeigen Sie mithilfe des vollständig bipartiten Graphen $K_{n,n}$, dass dies nicht möglich ist.

b) Es sei $k \geq 2$ eine fest gewählte ganze Zahl. Gegeben sei eine Menge S und eine Kollektion $T_1, \ldots, T_m$ von k-elementigen Teilmengen von S. Es gelte also

$$T_i \subseteq S \text{ und } |T_i| = k \quad (i = 1, \ldots, m).$$

Eine Teilmenge $H \subseteq S$ wird ein *Hitting Set* genannt, falls alle Mengen T_i von H getroffen werden, d.h., falls $H \cap T_i \neq \emptyset$ für alle $i = 1, \ldots, m$ gilt. Gesucht ist ein Hitting Set mit einer minimalen Anzahl von Elementen. (Mitteilung: Dies ist ein NP-schweres Optimierungsproblem.) Wir nennen das beschriebene Problem k-HITTING SET. Das Problem lässt sich also wie folgt beschreiben:

k-HITTING SET

Eingabe: eine Menge S sowie eine Kollektion von k-elementigen Teilmengen von S.

Gesucht: ein Hitting Set mit einer minimalen Anzahl von Elementen.

Man beachte: k ist *nicht* Teil der Eingabe, sondern eine *Konstante*.

(i) Beschreiben Sie einen k-Approximationsalgorithmus für k-HITTING SET.

(ii) Weisen Sie nach, dass der von Ihnen unter (i) vorgeschlagene Algorithmus tatsächlich ein k-Approximationsalgorithmus ist.

Hinweis zu (i) und (ii): Denken Sie zunächst an den Spezialfall $k = 2$. Sie kennen bereits einen 2-Approximationsalgorithmus für diesen Spezialfall: siehe Abschnitt 11.7.1 im Skript.