Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Fachbereich Informatik
Sommersemester 2024
Prof. Dr. Peter Becker

Kombinatorische Optimierung

Lösungen zu Aufgabenblatt 5

Aufgabe 1 (Packungsprobleme)

(a) Gegeben ist eine Menge $A = \{a_1, \ldots, a_n\} \subseteq \mathbb{N}$.

Das Problem Subset Sum lautet:

- Existiert eine Teilmenge $B \subseteq A$ für die $\sum_{a \in B} a = \sum_{a \in A \setminus B} a$ gilt?

Zeigen Sie: Subset Sum ist $\mathcal{NP}$-vollständig.

(b) Gegeben ist eine Menge $A = \{a_1, \ldots, a_n\} \subseteq \mathbb{N}$ sowie Zahlen $B \in \mathbb{N}$ (Behältergröße) und $k \in \mathbb{N}$ (Behälteranzahl). Das Bin Packing Problem (Bpp) lautet:

- Existiert eine Abbildung $f : \{1, \ldots, n\} \to \{1, \ldots, k\}$, so dass $\sum_{\{1 \leq i \leq n | f(i) = j\}} a_i \leq B$ für alle $j = 1, \ldots, k$ gilt?
  Anschaulich: Kann man n Gegenstände der Größe $a_1, \ldots, a_n$ so auf k Behälter der Größe B verteilen, dass alle Gegenstände in einen Behälter passen?

Zeigen Sie: Bpp ist $\mathcal{NP}$-vollständig.

(c) Zeigen Sie: Für $w_1 = \cdots = w_n = 1$ ist das Rucksackproblem in $\mathcal{P}$.

Lösung:

(a)
1. Für einen Lösungsvorschlag $B \subseteq A$ können wir in polynomieller Zeit prüfen, ob $\sum_{a \in B} a = \sum_{a \in A \setminus B} a$ gilt.
2. Wir zeigen Sum $\leq_p$ Subset Sum.
   Sei eine Probleminstanz von Sum gegeben, mit einer Menge $A = \{a_1, a_2, \ldots, a_n\} \subseteq \mathbb{N}_0$ und $S \in \mathbb{N}_0$.
   Dann nehmen wir für Subset Sum die Menge $A' = \{a_1, a_2, \ldots, a_n, a'\}$ mit $a' = 2\left|S - \frac{1}{2}\sum_{i=1}^{n} a_i\right|$.
   „$\Rightarrow$“: Sei $B \subseteq A$ eine Lösung für Sum. Wir machen eine Fallunterscheidung.
   (a) Es gelte $S \geq \frac{1}{2}\sum_{i=1}^{n} a_i$.
   Dann ist B auch eine Lösung für Subset Sum, denn die Summe der Elemente in A' ist $\sum_{i=1}^{n} a_i + a' = \sum_{i=1}^{n} a_i + 2\left(S - \frac{1}{2}\sum_{i=1}^{n} a_i\right) = 2S$ und die Summe der Elemente in B ist als Lösung von Sum gleich S.
   (b) Es gelte $S < \frac{1}{2}\sum_{i=1}^{n} a_i$.
   Dann beträgt die Summe der Elemente in A' gleich $\sum_{i=1}^{n} a_i + a' = \sum_{i=1}^{n} a_i + 2\left(\frac{1}{2}\sum_{i=1}^{n} a_i - S\right) = 2\left(\sum_{i=1}^{n} a_i - S\right)$.
   Damit ist $B \cup \{a'\}$ eine Lösung für Subset Sum, denn das Gewicht dieser Menge ist $S + 2\left(\frac{1}{2}\sum_{i=1}^{n} a_i - S\right) = \sum_{i=1}^{n} a_i - S$.
   „$\Leftarrow$“: Es sei $B' \subseteq A'$ eine Lösung für Subset Sum. Wir machen eine Fallunterscheidung.

(a) Es gelte $S \geq \frac{1}{2}\sum_{i=1}^{n} a_i$.
Gilt $a' \notin B'$, dann wählen wir $B = B'$ als Lösung für SUM, ansonsten $B = A' \setminus B'$. Damit ist sichergestellt, dass $B \subseteq A$ gilt. Das Gewicht von B ist dann

$$\frac{1}{2}\left(\sum_{i=1}^{n} a_i + a'\right) = \frac{1}{2}\left(\sum_{i=1}^{n} a_i + 2\left(S - \frac{1}{2}\sum_{i=1}^{n} a_i\right)\right) = S.$$

(b) Es gelte $S < \frac{1}{2}\sum_{i=1}^{n} a_i$.
Gilt $a' \in B'$, dann wählen wir $B = B' \setminus \{a'\}$, ansonsten $B = (A' \setminus B') \setminus \{a'\}$. Damit ist sichergestellt, dass $B \subseteq A$ gilt. Das Gewicht von B ist dann

$$\frac{1}{2}\left(\sum_{i=1}^{n} a_i + a'\right) - a' = \frac{1}{2}\left(\sum_{i=1}^{n} a_i - a'\right) = \frac{1}{2}\left(\sum_{i=1}^{n} a_i - 2\left(\frac{1}{2}\sum_{i=1}^{n} a_i - S\right)\right) = S.$$

(b) 1. Für einen Lösungsvorschlag $f : \{1, \ldots, n\} \to \{1, \ldots, k\}$ können wir in polynomieller Zeit prüfen, ob kein Behälter überläuft.

2. Wir zeigen SUBSET SUM $\leq_p$ BPP.
Sei eine Menge $A = \{a_1, \ldots, a_n\}$ als Probleminstanz von SUBSET SUM gegeben. Für die BPP-Instanz setzen wir $k = 2$ und $B = \frac{1}{2}\sum_{i=1}^{n} a_i$.
„$\Rightarrow$“: Ist $B \subseteq A$ eine Lösung der SUBSET SUM-Instanz, dann legen wir die Elemente aus B in den ersten Behälter und die Elemente aus $A \setminus B$ in den zweiten.
„$\Leftarrow$“: Es sei $f : \{1, \ldots, n\} \to \{1, 2\}$ eine Lösung von BPP. Wir setzen $B = \{a_i \in A | f(i) = 1\}$, also die Elemente im ersten Behälter. Da die Gesamtkapazität der Behälter gleich $\sum_{i=1}^{n} a_i$ ist, müssen beide Behälter voll ausgelastet sein. Somit folgt $\sum_{a \in B} a = \sum_{a \in A \setminus B} a$.

(c) Für $w_i = 1$ mit $1 \leq i \leq n$ bedeutet die Bedingung $\sum_{i \in I} w_i \leq W$ des Rucksackproblems, dass höchstens W Gegenstände ausgewählt werden dürfen.

Damit können wir das Rucksackproblem leicht lösen: Wir wählen die W Gegenstände mit dem größten Profit aus und prüfen für die entsprechende Indexmenge I, ob $\sum_{i \in I} p_i \geq P$ gilt.

Aufgabe 2 (Logische Inferenz mittels 0-1-Programmierung)

Fünf Häuser stehen nebeneinander. In ihnen wohnen Menschen von fünf unterschiedlichen Nationalitäten, die fünf unterschiedliche Getränke trinken, fünf unterschiedliche Zigarettenmarken rauchen und fünf unterschiedliche Haustiere haben.

1. Der Engländer lebt im roten Haus.
2. Der Spanier hat einen Hund.
3. Der Ukrainer trinkt gern Tee.
4. Das grüne Haus ist (direkt) links vom weißen Haus.
5. Im grünen Haus wird Kaffee getrunken.
6. Die Person, die Old-Gold raucht, hat eine Schnecke.
7. Der Bewohner des mittleren Hauses trinkt Milch.
8. Der Bewohner des gelben Hauses raucht Kools.
9. Der Norweger wohnt im ersten Haus.

10. Der Chesterfields-Raucher wohnt neben der Person mit der Fuchs.
11. Der Mann mit dem Pferd lebt neben der Person, die Kools raucht.
12. Der Lucky-Strike-Raucher trinkt Orangensaft.
13. Der Norweger wohnt neben dem blauen Haus.
14. Der Japaner raucht Parliaments.
15. Der Chesterfields-Raucher hat einen Nachbarn, der Wasser trinkt.

Finden Sie heraus, wem das Zebra gehört! Erstellen Sie hierzu ein 0-1-Programm und lösen Sie dieses mit dem GLPK oder Gurobi.

Anmerkung: „Links“ und „Rechts“ sind von einem Betrachter aus zu verstehen, der vor den Häusern steht. Das erste Haus ist aus dieser Blickrichtung das Haus auf der linken Seite.

Hinweise:

- Formulieren Sie die Punkte 1. bis 15. als logische Formeln und überführen Sie diese wie in Beispiel 2.43 in Gleichungen oder Ungleichungen.
- Darüber hinaus benötigen Sie noch weitere Gleichungen zur Modellierung der allgemeinen Bedingungen (pro Haus genau eine Farbe, Haustier, Nationalität, Zigarettenmarke und Getränk).
- Natürlich ist dieses Problem kein Optimierungsproblem, sondern ein Constraint-Satisfaction-Problem (CSP). Nutzen Sie eine Dummy-Zielfunktion, um hieraus ein Optimierungsproblem zu machen.

Lösung: siehe Homepage

Aufgabe 3 (Sudoku) 8 Punkte

Gegeben ist das folgende Sudoku:

				5	4			3
			9			8	7	
2	1							9
8	7			6			2	
9	4			1	3			5
6			4					
				4			8	1
			2	8	5	9		
							3	

Lösen Sie dieses Sudoku mit einem ganzzahligen LP.

Lösung: siehe Homepage

Aufgabe 4 ($\mathcal{NP}$-Äquivalenz, kombinatorische Auktion)

Wir betrachten die Situation bei einer *kombinatorischen Auktion*. Gegeben ist eine Gütermenge $G = \{g_1, \ldots, g_n\}$ von n Gütern, die versteigert werden sollen. Bei kombinatorischen Auktionen sind dies typischerweise Güter, die von gleicher Art aber trotzdem verschieden sind, z. B. Zeitslots für Start- und Landungen an einem Flughafen, Zimmerbelegungen in einem Hotel, Transportaufträge oder Strecken im öffentlichen Nahverkehr.

Für die Güter von G liegen insgesamt m Gebote vor. Jedes Gebot besteht aus einer Menge $B_i \subseteq G$ mit $B_i \neq \emptyset$ und einem Preis p_i (für $1 \leq i \leq m$). Die Bieter bieten also auf nichtleere Teilmengen von G. Ziel der Bieter ist es, mehrere Güter auf einmal zu ersteigern, um Synergieeffekte nutzen zu können, beispielsweise mehrere Transportaufträge zu nahe beieinanderliegenden Zielen, für die nur eine LKW-Fahrt erforderlich ist.

Welche Gebote sollen den Zuschlag erhalten? Dabei möchten wir natürlich, dass der Gesamterlös maximal ist und jedes Gut g_i höchstens einmal versteigert wird, also höchstens in einem Gebot, das den Zuschlag erhält, enthalten ist. Wir bezeichnen dieses Problem als Cap (Combinatorial Auction Problem).

Zeigen Sie, dass Cap $\mathcal{NP}$-äquivalent ist.

Hinweis: Zum Beweis für $\mathcal{NP}$-schwer: Formulieren Sie ein entsprechendes Entscheidungsproblem und zeigen Sie für dieses, dass es $\mathcal{NP}$-vollständig ist. Nutzen Sie hierfür Independent Set.

Lösung: Die Entscheidungsvariante von Cap lautet:

> Gegeben ist eine Gütermenge $\mathcal{G} = \{g_1, \ldots, g_n\}$, eine Menge $\mathcal{B} = \{B_1, \ldots, B_m\}$ von Geboten mit $B_i \subseteq \mathcal{G}$ und $B_i \neq \emptyset$ für $i = 1, \ldots, m$, sowie Zahlen $p_1, \ldots, p_m \in \mathbb{N}$ und $P \in \mathbb{N}$.
>
> Existiert eine Indexmenge $I \subseteq \{1, \ldots, m\}$ mit $B_i \cap B_j = \emptyset$ für alle $i, j \in I$ und $i \neq j$, so dass $\sum_{i \in I} p_i \geq P$ gilt?

Wir zeigen zunächst, dass die Entscheidungsvariante von Cap $\mathcal{NP}$-vollständig ist.

1. Für einen Lösungsvorschlag I können wir in polynomieller Zeit prüfen, ob $B_i \cap B_j = \emptyset$ für alle $i, j \in I$ mit $i \neq j$ und $\sum_{i \in I} p_i \geq P$ gilt.

2. Wir zeigen Independent Set $\leq_p$ Cap.

   Seien $G = (V, E)$ ein Graph und $k \in \mathbb{N}$ eine Probleminstanz für Independent Set. Wir konstruieren wie folgt eine Probleminstanz für Cap:

   - $\mathcal{G} = E = \{e_1, \ldots, e_n\}$
   - Sei $V = \{v_1, \ldots, v_m\}$ die Knotenmenge von G. Dann setzen wir $B_i = \{e \in E | e \text{ ist indizent mit } v_i\}$ für $i = 1, \ldots, m$. Das Gebot B_i besteht also aus allen Kanten, die inzident mit Knoten v_i sind.
   - $p_1, \ldots, p_m = 1$
   - $P = k$

   „$\Rightarrow$“: Sei $U \subseteq V$ mit $|U| \geq k$ eine Lösung für Independent Set. Dann gilt für $v_i, v_j \in U$ mit $i \neq j$, dass v_i und v_j nicht adjazent sind. Daraus folgt $B_i \cap B_j = \emptyset$. Somit bildet die Indexmenge $I = \{1 \leq i \leq m | v_i \in U\}$ eine Lösung für Cap mit $|I| = |U| \geq k$. Wegen $p_1, \ldots, p_m = 1$ folgt $\sum_{i \in I} p_i = |I| \geq k = P$.

   „$\Leftarrow$“: Sei I eine Lösung für Cap mit $\sum_{i \in I} p_i \geq P$. Aus $p_1, \ldots, p_m = 1$ folgt $|I| \geq P = k$. Sei $U = \{v_i | i \in I\}$. Aus $B_i \cap B_j = \emptyset$ folgt, dass v_i und v_j nicht adjazent sein können. Also ist U ein Independent Set mit $|U| = |I| \geq k$.

Damit ist gezeigt, dass die Entscheidungsvariante von CAP $\mathcal{NP}$-vollständig ist. Gemäß Definition 2.36 ist damit das Optimierungsproblem CAP $\mathcal{NP}$-schwer.

Jetzt müssen wir noch zeigen, dass CAP $\mathcal{NP}$-leicht ist. Hierzu betrachten wir

$$p := \max_{i=1}^{m} \frac{p_i}{|B_i|}.$$

Dies entspricht dem höchsten Preis pro Gut in einem Gebot. Für den optimalen Zielfunktionswert P von CAP muss dann

$$0 \leq P \leq n \cdot p$$

gelten. Damit können wir CAP als Optimierugsproblem durch nicht mehr als $\lceil \log_2(np) \rceil + 1$ Aufrufe eines Algorithmus für die Entscheidungsvariante von CAP lösen (vgl. Folien 133 f.).