Lehrstuhl für Informatik 1
Prof. Dr. Gerhard Woeginger
Jan Böker, Tim Hartmann

WS 2018/19

Übung zur Vorlesung
Berechenbarkeit und Komplexität

Lösung Blatt 12

Hausaufgabe 12.1 **(4 Punkte + 3 Bonuspunkte)**

Partition-Into-Three-Sets ist das folgende Entscheidungsproblem:

> Partition-Into-Three-Sets
>
> **Eingabe: Positive ganze Zahlen $a_1, \ldots, a_n$.**
>
> **Frage: Gibt es paarweise disjunkte Mengen $I, J, K \subseteq \{1, \ldots, n\}$ mit $I \cup J \cup K = \{1, \ldots, n\}$, so dass**
>
> $$\sum_{i \in I} a_i = \sum_{j \in J} a_j = \sum_{k \in K} a_k \text{ ?}$$

Zeigen Sie, dass Partition-Into-Three-Sets NP-vollständig ist. Um die Bonuspunkte zu erhalten, müssen Sie zudem zeigen, dass Partition-Into-Three-Sets in pseudopolynomieller Zeit gelöst werden kann.

Wir gehen nach dem "Kochrezept" in Foliensatz 15 vor.

(a) Zunächst zeigen wir durch Angabe eines Zertifikats und eines Polynomialzeitverifizierers, dass Partition-Into-Three-Sets in NP liegt.

Zertifikat: Ein String y der Länge $2n$, wobei $y_{2i-1}y_{2i}$ die Binärkodierung einer Zahl $s_i \in \{1, 2, 3\}$ ist. Dabei bedeutet $s_i = 1$ ($s_i = 2$, $s_i = 3$), dass a_i in die Menge K (I, J) gehört.

Verifizierer: Prüfe zunächst, ob der String y die richtige Länge und das richtige Format hat. Dann berechne für $j \in \{1, 2, 3\}$ die Summe A_j aller a_i, sodass $y_{2i-1}y_{2i}$ die Binärkodierung der Zahl j ist. Falls $A_1 = A_2 = A_3$, dann akzeptiere, sonst verwerfe.

Laufzeit: Die Länge des Strings y kann offensichtlich in polynomieller Zeit geprüft werden. Für die Prüfung des Formats genügt ein Lauf über y, also werden hier nur $2n$ Schritte benötigt. Das Aufsummieren von Zahlen kann genauso wie das Vergleichen zweier Zahlen ebenfalls in polynomieller Zeit durchgeführt werden.

(b) Wir geben eine polynomielle Reduktion von Partition auf Partition-Into-Three-Sets an.

(c) Sei $a_1, \ldots, a_n$ mit $\sum_{i=1}^{n} a_i = 2A$ eine Eingabe für Partition. Daraus wird die Eingabe $a_1, \ldots, a_n, a_{n+1} = A$ für Partition-Into-Three-Sets konstruiert.

(d) Offensichtlich können wir die Summe von n Binärzahlen in polynomieller Zeit berechnen. Dann müssen wir das Ergebnis nur noch durch zwei dividieren und an die Eingabe anhängen, was ebenfalls in polynomieller Zeit möglich ist.

(e) Für die Korrektheit sei zunächst $I \subseteq \{1, \ldots, n\}$ eine Menge mit $\sum_{i \in I} a_i = A$. Setze $J := \{1, \ldots, n\} \setminus I$ und $K := \{n+1\}$. Dann gilt

$$\sum_{i \in I} a_i = A = 2A - A = \sum_{i=1}^{n} a_i - \sum_{i \in I} a_i = \sum_{j \in J} a_j$$

und

$$\sum_{i \in I} a_i = A = \sum_{k \in K} a_k.$$

Also ist I, J, K eine korrekte Partition für $a_1, \ldots, a_n, a_{n+1} = A$.

Für die Rückrichtung sei nun I, J, K eine korrekte Partition für $a_1, \ldots, a_n, a_{n+1} = A$, d. h., $\sum_{i \in I} a_i = \sum_{j \in J} a_j = \sum_{k \in K} a_k = \frac{1}{3} \cdot \sum_{i=1}^{n+1} a_i$. Es gilt $\frac{1}{3} \cdot \sum_{i=1}^{n+1} a_i = \frac{1}{3} \cdot 3A = A = a_{n+1}$, weshalb eine der Mengen nur a_{n+1} enthalten muss. Ohne Einschränkung der Allgemeinheit sei dies K. Dann ist die Menge I mit $I \subseteq \{1, \ldots, n\}$ und $\sum_{i \in I} a_i = A$ eine Lösung für die Partition-Instanz.

Partition-Into-Three-Sets kann in pseudopolynomieller Zeit gelöst werden: Sei $a_1, \ldots, a_n$ die Eingabe für Partition-Into-Three-Sets und sei A so, dass $3A = \sum_{i=1}^{n} a_i$; falls die Summe nicht durch drei teilbar ist, kann man direkt verwerfen.

Wir nutzen dynamische Programmierung. Für $k \in \{0, \ldots, n\}$ und $c_1, c_2, c_3 \in \{0, \ldots, A\}$ sei

$$F[k, c_1, c_2, c_3] = \text{TRUE}$$

genau dann, wenn es eine Aufteilung I_1, I_2, I_3 der Indizes $1, \ldots, k$ gibt sodass $\sum_{i \in I_1} a_i = c_1$, $\sum_{i \in I_2} a_i = c_2$ und $\sum_{i \in I_3} a_i = c_3$.

Wir initialisieren $F[0, c_1, c_2, c_3] = \text{TRUE}$, wenn $c_1 = c_2 = c_3 = 0$, und $F[0, c_1, c_2, c_3] = \text{FALSE}$ sonst. Dann setze nacheinander

$$F[k, c_1, c_2, c_3] = F[k-1, c_1 - a_k, c_2, c_3] \vee F[k-1, c_1, c_2 - a_k, c_3] \vee F[k-1, c_1, c_2, c_3 - a_k].$$

Am Ende befindet sich das Ergebnis in $F[n, A, A, A]$. Die Laufzeit ist beschränkt durch $\mathcal{O}(n \cdot A^3) = \mathcal{O}(n^4 \cdot (\max_{i \in \{1,\ldots,n\}} a_i)^3)$.

Hausaufgabe 12.2 (5 Punkte)

Zeigen Sie, dass das folgende Problem stark NP-schwer ist:

Triple-Partition

Eingabe: **Positive ganze Zahlen $d_1, \ldots, d_{3n}$ mit $\sum_{i=1}^{3n} d_i = nD$.**

Frage: **Können diese Zahlen in n Tripel partitioniert werden, sodass sich in jedem Tripel die Elemente zur Summe D aufaddieren?**

Hinweis: Konstruieren Sie eine Reduktion von THREE-PARTITION. Ersetzen Sie dabei die Zahlen $\boldsymbol{a_i}$ durch $\boldsymbol{a_i + X}$, die Zahlen $\boldsymbol{b_i}$ durch $\boldsymbol{b_i + Y}$ und die Zahlen $\boldsymbol{c_i}$ durch $\boldsymbol{c_i + Z}$, wobei die Zahlen $\boldsymbol{X, Y, Z}$ geeignet von $\boldsymbol{S}$ abhängen.

Wir zeigen THREE-PARTITION $\leq_p$ TRIPLE-PARTITION, indem wir dem Hinweis folgen und $X \coloneqq 4S$, $Y \coloneqq 2S$ und $Z \coloneqq S$ wählen. Dadurch ergibt sich dann $D = 8S$:

$$\begin{aligned}
&\sum_{i=1}^{n}(a_i + X) + \sum_{i=1}^{n}(b_i + Y) + \sum_{i=1}^{n}(c_i + Z) \\
=&\sum_{i=1}^{n}(a_i + b_i + c_i) + n \cdot (X + Y + Z) \\
=&n \cdot S + n \cdot 7S \\
=&n \cdot 8S
\end{aligned}$$

Beachte, dass sich die maximalen Werte der Zahlen nur um einen polynomiellen Faktor vergrößern: Es sei q ein Polynom, sodass THREE-PARTITION nach Einschränkung der Zahlenwerte einer Instanz I auf höchstens $q(|I|)$ noch NP-schwer ist. Für eine solche Instanz I gilt $\sum_{i=1}^{n}(a_i + b_i + c_i) = n \cdot S$ und somit

$$S = \frac{1}{n} \cdot \sum_{i=1}^{n}(a_i + b_i + c_i) \leq \frac{1}{n} \cdot \sum_{i=1}^{n} 3 \cdot q(|I|) = 3 \cdot q(|I|).$$

Folglich gilt für die Zahlenwerte der resultierenden TRIPLE-PARTITION-Instanz I', dass diese maximal $q(|I|) + 4S \leq q(|I|) + 12 \cdot q(|I|) = 13 \cdot q(|I|) \leq 13 \cdot q(|I'|)$ groß ist, wobei wir $|I| \leq |I'|$ genutzt und Monotonie von q angenommen haben.

Für die Korrektheit zeigen wir, dass Permutationen α und β mit $a_{\alpha(i)} + b_{\beta(i)} + c_i = S$ für jedes i genau dann existieren, wenn eine Partition der Werte $a_i + X$, $b_i + Y$ und $c_i + Z$ $(i \in \{1, \dots, n\})$ in Tripel der Summe D existiert.

Die Hinrichtung ist einfach: Aus den Permutationen α und β erhält man direkt eine Partition in die Tripel $(a_{\alpha(i)} + X, b_{\beta(i)} + Y, c_i + Z)$ mit $a_{\alpha(i)} + X + b_{\beta(i)} + Y + c_i + Z = S + X + Y + Z = 8S = D$.

Für die Rückrichtung nehmen wir an, dass wir eine Partition der Zahlen in n Tripel $t_1, \dots, t_n$ der Summe $D = 8S$ haben. Wir verteilen nun nacheinander die a-, b- und c-Werte auf diese Tripel (dabei jeweils genau einen a-, b- und c-Wert pro Tripel) und erhalten so die gesuchten Permutationen.

Angenommen, in einem der Tripel kommen zwei Werte $a_i + X$ und $a_j + X$ vor. Dann ist die Summe der Tripelwerte mindestens $a_i + X + a_j + X = 2X + a_i + a_j = 8S + a_i + a_j > 8S = D$, was ein Widerspruch ist. Also kommen in keinem der Tripel $t_1, \dots, t_n$ zwei a-Werte vor, und da es genau n a-Werte gibt, folgt, dass jedes Tripel genau einen Wert der Form $a_i + X$ enthält. Damit erhalten wir eine bijektive Verteilung der Werte $a_1, \dots, a_n$ auf die Tripel $t_1, \dots, t_n$.

Betrachten wir nun die b-Werte. Angenommen, in einem der Tripel $t_1, \dots, t_n$ sind zwei der übrigen Werte solche b-Werte, d. h. das Tripel hat die Form $(a_i + X, b_j + Y, b_k + Y)$. Dann ist die Summe der Tripelwerte $a_i + X + b_j + Y + b_k + Y = 8S + a_i + b_j + b_k > 8S = D$, was ein Widerspruch ist. Also kann höchstens einer der übrigen Werte der Tupel $t_1, \dots, t_n$

ein b-Wert sein, und da es genau n b-Werte gibt, folgt, dass einer der übrigen Werte ein solcher b-Wert sein muss. Damit erhalten wir also auch eine bijektive Verteilung der Werte $b_1, \ldots, b_n$ auf die Tripel $t_1, \ldots, t_n$.

Damit haben wir die a- und b-Werte bijektiv auf die Tripel $t_1, \ldots, t_n$ verteilt. Es bleiben noch die Werte $c_1, \ldots, c_n$ und jeweils ein unzugewiesener Wert pro Tripel; damit erhalten wir direkt eine bijektive Verteilung der Werte $c_1, \ldots, c_n$ auf die Tripel $t_1, \ldots, t_n$. Insgesamt haben wir also eine Bijektion zwischen den a-, b- und c-Werten und den Komponenten der Tripel $t_1, \ldots, t_n$, die jedem Tripel t_i genau einen a-, b- und c-Wert zuordnet. Für ein solches Tripel $t_\ell = (a_i + X, b_j + Y, c_k + Z)$ gilt nach Annahme $a_i + X + b_j + Y + c_k + Z = D = 8S$ und damit $a_i + b_j + c_k = S$; damit liefert die Bijektion auch die gesuchten Permutationen (ohne diese hier genau zu definieren).

Hausaufgabe 12.3 **(4 Punkte)**

Beweisen Sie, dass das folgende Problem coNP-vollständig ist:

> AT-MOST-THREE-SAT
>
> **Eingabe: Boole'sche Formel φ in CNF über den Variablen $x_1, \ldots, x_n$.**
>
> **Frage: Besitzt φ höchstens drei verschiedene erfüllende Variablenbelegungen?**

Zeige zunächst, dass AT-MOST-THREE-SAT $\in$ coNP gilt, indem ein Nein-Zertifikat und ein Polynomialzeitverifizierer angegeben werden.

Nein-Zertifikat: Vier paarweise verschiedene Variablenbelegungen, die φ erfüllen. Die Länge des Zertifikates ist $4n$, wobei n die Anzahl der Variablen in φ ist.

Verifizierer: Prüfe zunächst, ob die Eingabe das richtige Format hat. Stelle dann sicher, dass alle vier Variablenbelegungen paarweise verschieden sind und die Formel φ erfüllen.

Laufzeit: Der Test, dass die Belegungen paarweise verschieden sind, kann in quadratischer Zeit durchgeführt werden. Der Test, dass die Variablenbelegungen φ erfüllen, kann auch in quadratischer Zeit durchgeführt werden.

Korrektheit: Wenn φ nicht höchstens drei verschiedene erfüllende Belegungen besitzt, dann existieren mindestens vier verschiedene Belegungen. Also existiert ein Zertifikat, dass den Verifizierer zum akzeptieren bringt. Akzeptiert umgekehrt der Verifizierer, so existieren mindestens vier verschiedene Belegungen, die φ erfüllen. Also ist φ eine Nein-Instanz.

Damit folgt, dass AT-MOST-THREE-SAT in coNP liegt.

Um zu zeigen, dass AT-MOST-THREE-SAT coNP-schwer ist, wird UNSAT $\leq_p$ AT-MOST-THREE-SAT gezeigt. Definiere für eine Formel φ die Formel $\varphi' := \varphi \wedge (a \vee b \vee c)$, wobei a, b und c neue Variablen sind; dann ist φ' nur konstant größer als φ und in polynomieller Zeit aus φ berechenbar.

Zeige für die Korrektheit, dass φ unerfüllbar ist genau dann, wenn φ' höchstens drei verschiedene erfüllende Belegungen hat. Ist φ unerfüllbar, so ist auch φ' unerfüllbar und hat höchstens drei verschiedene erfüllende Belegungen und liegt in AT-MOST-THREE-SAT. Ist φ erfüllbar, so hat φ' sogar mindestens sieben verschiedene erfüllende Belegungen (für jede Belegung von a, b und c, die $a \vee b \vee c$ erfüllt, erhält man mindestens eine) und liegt damit nicht in At-Most-Three-SAT.

Indem man nun ungültige Eingaben auf ungültige Eingaben abbildet, erhält man die gewünschte Reduktion.