**Institut für Informatik** **Ludwig-Maximilians-Universität München**
Prof. Dr. Volker Heun

**Formale Sprachen und Komplexität**, SS 18
**Übungsblatt 11**

*Abgabe: bis Mo 16.07.2018 8 Uhr*

Nach Bearbeitung dieses Übungsblattes sollten Sie:

| | Check |
|---|---|
| Den Begriff der NP-Vollständigkeit von Problemen verstanden haben. | |
| Die polynomialen Reduktion verstanden haben. | |
| Mit der polynomialen Reduktion die NP-Vollständigkeit eines Problems nachweisen können. | |

Diese Ziele sind wichtige Hinweise für die Klausur!

**Aufgabe 11-1** schriftlich bearbeiten (ohne Bonuspunkte)
**Polynomielle Reduktion**

Im Buch ist das $BIN\text{-}PACKING$ Problem folgendermaßen definiert: :

$$BIN\text{-}PACKING = \{(b, k, a_1, \ldots, a_n) \in \mathbb{N}^{n+2} \mid a_1, \ldots, a_n \leq b \wedge \exists f : \{1, \ldots, n\} \to \{1, \ldots, k\} : \forall j \sum_{f(i)=j} a_i \leq b\}$$

Informell bedeutet $(b, k, a_1, \ldots, a_n) \in BIN\text{-}PACKING$ folgendes: Man kann die Gewichte $a_1, \ldots, a_n$ auf $k$ Behälter verteilen, ohne dass einer schwerer als $b$ wird.

Zeigen Sie, dass $BIN\text{-}PACKING$ NP-vollständig ist.

Hinweis: $PARTITION$ ist auf $BIN\text{-}PACKING$ reduzierbar. Wie viele Mengen sind in einer Partition? Wie viele Behälter sind dann wohl angebracht?

**Lösungsvorschlag:**

**1. Teil:** Wir zeigen, dass $BIN\text{-}PACKING \in \mathsf{NP}$ ist
durch einen nichtdeterministischen Entscheidungsalgorithmus für $BIN\text{-}PACKING$:

| Für Eingabe $(b, k, a_1, \ldots, a_n)$ | |
|---|---|
| 1. Ordne nichtdeterministisch jedem $i \in \{a, \ldots, n\}$ ein $j \in \{1, \ldots, k\}$ zu; | $O(n)$ |
| 2. Prüfe, ob $\sum_{f(i)=j} a_j \leq b$; | $O(n)$ |

Dieser informelle Algorithmus kann von einer nichtdeterministischen
(Mehrband-)Turingmaschine ausgeführt werden.

Sie akzeptiert jedes $(b, k, a_1, \ldots, a_n) \in BIN\text{-}PACKING$
mit einer Berechnung der Länge $O(2n)$.
Das ist proportional (und damit polynomial) zur Länge von $(b, k, a_1, \ldots, a_n)$.

Also ist $BIN\text{-}PACKING \in \mathsf{NP}$.

**2. Teil:** Seien $(a_1, \ldots, a_n)$ ein $PARTITION$-Problem und $b := \frac{1}{2} \sum_{i=1}^{n} a_i$. Definiere

$$\begin{array}{lll} \phi : \mathbb{N}^n & \to & \mathbb{N}^{n+2} \\ (a_1, \ldots, a_n) & \mapsto & (b, 2, a_1, \ldots, a_n) \end{array}$$

$\phi$ ist total und mit polynomialem Aufwand berechenbar ($O(n)$).

Es bleiben zu zeigen: $x \in PARTITION \Leftrightarrow \phi(x) \in BIN\text{-}PACKING$

'$\Rightarrow$' Sei $x = (a_1, \ldots, a_n)$ eine Lösung des $PARTITION$-Problems mit lösender Menge $J$. Dann betrachten wir jetzt $(b, 2, a_1, \ldots, a_n)$, und suchen eine Verteilungsfunktion der $n$ Elemente auf die beiden Behälter der Größe $b$.

$$\begin{array}{lll} f : \{1, \ldots, n\} & \to & \{1, 2\} \\ i & \mapsto & \begin{cases} 1 \; wenn \; i \; \in J \\ 2 \; sonst \end{cases} \end{array}$$

Die so gewählte Aufteilung leistet offensichtlich das Gewünschte.

'$\Leftarrow$' Sei $(b, 2, a_1, \ldots, a_n)$ mit passendem $f : \{1, \ldots, n\} \to \{1, 2\}$ gegeben. Dann ist $J := f^{-1}(1)$ eine Lösung des $PARTITION$-Problems auf $(a_1, \ldots, a_n)$

Insgesamt ist $BIN\text{-}PACKING$ $\mathsf{NP}$-vollständig.

**Aufgabe 11-2** schriftlich bearbeiten (ohne Bonuspunkte)
**Komplexitätsklassen**

Sei $L \in \mathit{TIME}(f(n))$ für eine Funktion $f : \mathbb{N} \to \mathbb{N}$. Welcher Zusammenhang besteht zwischen dieser Funktion und dem Speicherplatzbedarf, also der Anzahl der Bandfelder die zum Lösen von $L$ benötigt werden?

Begründen Sie Ihre Antwort.

**Lösungsvorschlag:**

Für ein Alphabet $\Sigma$ und eine formale Sprache $L \subseteq \Sigma^*$ gilt:

$L \in \mathit{TIME}(f(n))$ gdw. es gibt eine deterministische (Mehrband-)Turingmaschine $M$ mit $L = T(M)$ und $T_M(x) \leq f(|x|)$ für alle $x \in \Sigma^*$
gdw. es gibt eine deterministische (Mehrband-)Turingmaschine $M$ mit $L = T(M)$ und die Anzahl der Konfigurationen, die $M$ mit Eingabe $x$ durchläuft, ist $\leq f(|x|)$.

Eine Turingmaschine kann bei jedem Übergang von einer Konfiguration zur nächsten den Lese-/Schreibkopf um höchstens eine Position bewegen. Die Anzahl der besuchten Bandfelder ist deshalb immer kleiner-gleich der Anzahl der durchlaufenen Konfigurationen.

Zusammen ergibt das:
Wenn $L \in \mathit{TIME}(f(n))$, dann gibt es eine deterministische (Mehrband-)Turingmaschine $M$ mit $L = T(M)$, die für das Akzeptieren von $x \in \Sigma^*$ einen Speicherplatzbedarf $\leq f(|x|)$ hat.

Dieses Ergebnis wird oft verkürzt zu

$$\text{Speicherkomplexität} \leq \text{Zeitkomplexität}$$

**Aufgabe 11-3 Polynomielle Reduktion**

a) Seien $A \subseteq \Sigma_A^*$ und $B \subseteq \Sigma_B^*$ formale Sprachen. Beweisen Sie: $A \leq_p B$ gdw. $\overline{A} \leq_p \overline{B}$

**Lösungsvorschlag:**

$A \leq_p B$ gdw. es gibt eine totale und mit polynomialer Komplexität berechenbare Funktion $f : \Sigma_A^* \to \Sigma_B^*$, so dass für alle $x \in \Sigma_A^*$ gilt:

$w \in A$ gdw. $f(w) \in B$

gdw. es gibt ... gilt: $w \notin A$ gdw. $f(w) \notin B$

gdw. es gibt ... gilt: $w \in \overline{A}$ gdw. $f(w) \in \overline{B}$

gdw. es gibt ... gilt: $\overline{A} \leq_p \overline{B}$

b) **Zusatzaufgabe:** Die Komplexitätsklasse co-NP ist definiert durch co-NP $= \{L \mid \overline{L} \in \mathsf{NP}\}$. Beweisen Sie:

Wenn es eine Sprache $K \in$ co-NP gibt, die NP-vollständig ist, dann gilt co-NP = NP.

Hinweis: Neben dem Ergebnis der vorigen Teilaufgabe ist folgendes Lemma nützlich: Wenn $A \leq_p B$ und $B \in \mathsf{NP}$, dann $A \in \mathsf{NP}$.

**Lösungsvorschlag:**

**Beweis**

Sei $K \in$ co-NP, das heißt, $\overline{K} \in \mathsf{NP}$,
und sei $K$ NP-vollständig, das heißt, für alle $L \in \mathsf{NP}$ gilt $L \leq_p K$.

„NP $\subseteq$ co-NP“:

Sei $L \in \mathsf{NP}$

Dann gilt $L \leq_p K$ — da $K$ NP-vollständig ist
Dann gilt $\overline{L} \leq_p \overline{K}$ — wegen a)
Dann gilt $\overline{L} \in \mathsf{NP}$ — da $\overline{K} \in \mathsf{NP}$ und mit Lemma
Dann gilt $L \in$ co-NP — nach Definition von co-NP

„co-NP $\subseteq$ NP“:

Sei $L \in$ co-NP

Dann gilt $\overline{L} \in \mathsf{NP}$ — nach Definition von co-NP
Dann gilt $\overline{L} \leq_p K$ — da $K$ NP-vollständig ist
Dann gilt $L \leq_p \overline{K}$ — wegen a)
Dann gilt $L \in \mathsf{NP}$ — da $\overline{K} \in \mathsf{NP}$ und mit Lemma

**Bemerkung:**

Die Klasse P ist abgeschlossen unter Komplementbildung: $L \in \mathsf{P}$ gdw. $\overline{L} \in \mathsf{P}$.
Es gilt P = co-P

Die Frage, ob NP = co-NP ist, ist dagegen bisher offen.

**Aufgabe 11-4** NP-Vollständigkeit

| *MinSumDiff* | |
|---|---|
| *Gegeben*: | Endliche Mengen $A_1, \ldots, A_k \subseteq \mathbb{N}$ und eine Zahl $m \in \mathbb{N}$. |
| *Gefragt*: | $\exists\, I \subseteq \{1, \ldots, k\}$ mit $\sum_{i \in I} \min(A_i) - \sum_{i \notin I} \min(A_i) = m$ ? |

Beweisen Sie, dass das so definierte *MinSumDiff*-Problem NP-vollständig ist.

Hinweis: eines der in der Vorlesung behandelten NP-harten Probleme kann ziemlich einfach polynomial auf *MinSumDiff* reduziert werden.

**Lösungsvorschlag:**

**Idee:** *PARTITION* verwenden, also erst einmal beide Probleme veranschaulichen.

---

| *PARTITION* | |
|---|---|
| *Gegeben*: | Zahlen $a_1, \ldots, a_k \in \mathbb{N}$. |
| *Gefragt*: | $\exists\, I \subseteq \{1, \ldots, k\}$ mit $\sum_{i \in I} a_i = \sum_{i \notin I} a_i$ ? |

Beispiel-Instanzen:

| $a_1$ | $a_2$ | $a_3$ | $a_4$ |
|---|---|---|---|
| 2 | 4 | 6 | 8 |

$\in$ *PARTITION* mit $I = \{1, 4\}$

| $a_1$ | $a_2$ | $a_3$ | $a_4$ |
|---|---|---|---|
| 3 | 4 | 6 | 8 |

$\notin$ *PARTITION*

---

| *MinSumDiff* | |
|---|---|
| *Gegeben*: | Endliche Mengen $A_1, \ldots, A_k \subseteq \mathbb{N}$ und eine Zahl $m \in \mathbb{N}$. |
| *Gefragt*: | $\exists\, I \subseteq \{1, \ldots, k\}$ mit $\sum_{i \in I} \min(A_i) - \sum_{i \notin I} \min(A_i) = m$ ? |

Beispiel-Instanzen:

| $A_1$ | $A_2$ | $A_3$ | $A_4$ | $m$ |
|---|---|---|---|---|
| $\{2, 7\}$ | $\{4, 5\}$ | $\{6\}$ | $\{8, 9, 11\}$ | 0 |

$\in$ *MinSumDiff* mit $I = \{1, 4\}$

| $A_1$ | $A_2$ | $A_3$ | $A_4$ | $m$ |
|---|---|---|---|---|
| $\{3, 7\}$ | $\{4, 5\}$ | $\{6\}$ | $\{8, 9, 11\}$ | 0 |

$\notin$ *MinSumDiff*

---

**Lösungsvorschlag:**

---

**1. Teil:** wir zeigen, dass *MinSumDiff* NP-hart ist
durch polynomiale Reduktion von *PARTITION* auf *MinSumDiff*

Sei $f: \begin{array}{ccc} \Sigma^*_{PARTITION} & \to & \Sigma^*_{MinSumDiff} \\ \big[\, a_1, \ldots, a_k \,\big] & \mapsto & \big[\, \{a_1\}, \ldots, \{a_k\}, 0 \,\big] \end{array}$

**(1)** $f$ ist offensichtlich **total**.

**(2)** $f$ ist **berechenbar** mit Komplexität $O(k)$, also **mit polynomialer Komplexität**.

**(3)** $\big[\, a_1, \ldots, a_k \,\big] \in PARTITION$

$$\begin{aligned} &\text{gdw.} \quad \exists\, I \subseteq \{1, \ldots, k\} \text{ mit } \sum_{i \in I} a_i = \sum_{i \notin I} a_i \\ &\text{gdw.} \quad \exists\, I \subseteq \{1, \ldots, k\} \text{ mit } \sum_{i \in I} a_i - \sum_{i \notin I} a_i = 0 \\ &\text{gdw.} \quad \exists\, I \subseteq \{1, \ldots, k\} \text{ mit } \sum_{i \in I} \min(\{a_i\}) - \sum_{i \notin I} \min(\{a_i\}) = 0 \\ &\text{gdw.} \quad f(\big[\, a_1, \ldots, a_k \,\big]) = \big[\, \{a_1\}, \ldots, \{a_k\}, 0 \,\big] \in MinSumDiff \end{aligned}$$

Also $PARTITION \leq_p MinSumDiff$.
Da *PARTITION* NP-hart ist, ist auch *MinSumDiff* NP-hart.

---

**2. Teil:** wir zeigen, dass $MinSumDiff \in \mathsf{NP}$ ist
durch einen nichtdeterministischen Entscheidungsalgorithmus für *MinSumDiff*:

| Für Eingabe $\big[\, A_1, \ldots, A_k, m \,\big]$ | |
|---|---|
| 1. Für jede der Mengen $A_i$ berechne $\min(A_i)$; | $O(k \cdot \max\limits_{1 \leq i \leq k} \lvert A_i \rvert)$ |
| 2. Wähle nichtdeterministisch eine Teilmenge $I \subseteq \{1, \ldots, k\}$; | $O(k)$ |
| 3. Prüfe, ob $\sum\limits_{i \in I} \min(A_i) - \sum\limits_{i \notin I} \min(A_i) = m$; | $O(k)$ |

Dieser informelle Algorithmus kann von einer nichtdeterministischen
(Mehrband-)Turingmaschine ausgeführt werden.

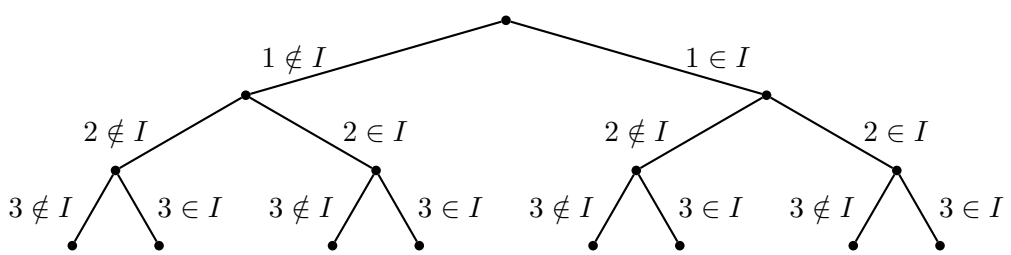
Sie akzeptiert jedes $\big[\, A_1, \ldots, A_k, m \,\big] \in MinSumDiff$
mit einer Berechnung der Länge $O\Big(k \cdot (2 + \max\limits_{1 \leq i \leq k} \lvert A_i \rvert)\Big)$.

Das ist proportional (und damit polynomial) zur Länge von $\big[\, A_1, \ldots, A_k, m \,\big]$.

Also ist $MinSumDiff \in \mathsf{NP}$.

---

**Veranschaulichung** von „2. Wähle nichtdeterministisch eine Teilmenge $I \subseteq \{1, \ldots, k\}$“




Es reicht eine Schleife für $i \in \{1, \ldots, k\}$.
Für jedes $i$ verzweigt die Berechnung in eine Fortsetzung mit $i \notin I$ und eine mit $i \in I$.
Jede dieser Berechnungen braucht $O(k)$ Schritte.