Universität Paderborn
Prof. Dr. Johannes Blömer

Gruppe 27:
Eli Kogan-Wang (7251030)
David Noah Stamm (7249709)
Bogdan Rerich (7248483)
Jan Schreiber(7253698)

Berechenbarkeit und Komplexität – WS 2022/2023

Heimübung 6

Abgabe: 28. November 2022 – 13:00 Uhr

(Dieser Übungszettel besteht aus 4 Aufgaben mit insgesamt 24 Punkten)

Aufgabe 1 ($\mathcal{O}$-Kalkül) (6 Punkte)

Welche der folgenden Aussagen sind korrekt? Beweisen Sie Ihre Antwort!

a) $10^{-1000}n^{1.01} \in \Theta(n)$

b) $n^2 \in \mathcal{O}(n\log(n))$

c) $n\log(n) \in o(n^2)$

d) $n \in \Omega\left(2^{\sqrt{\log(n)}}\right)$

e) $n! \in 2^{\mathcal{O}(n)}$, das heiSSt, es existiert $c > 0$, sodass $n! \leq 2^{cn}$.

Lösung:

a) $\lim\limits_{n\to\infty} \frac{10^{-1000}n^{1.01}}{n} = \lim\limits_{n\to\infty} 10^{-1000}n^{0.01} = \infty \Rightarrow 10^{-1000}n^{1.01} \notin \Theta(n)$

b) $\lim\limits_{n\to\infty} \frac{n^2}{nlog(n)} = \lim\limits_{n\to\infty} \frac{n}{log(n)} \xrightarrow[Krankenhaus]{\text{Regel vom}} \lim\limits_{n\to\infty} \frac{\frac{d}{dx}n}{\frac{d}{dx}log(n)} = \lim\limits_{n\to\infty} \frac{1}{\frac{1}{n}} = \lim\limits_{n\to\infty} n = \infty$
$\Rightarrow n^2 \notin \mathcal{O}(n\log(n))$

c) $\lim\limits_{n\to\infty} \frac{nlog(n)}{n^2} = \lim\limits_{n\to\infty} \frac{log(n)}{n} \xrightarrow[Krankenhaus]{\text{Regel vom}} \lim\limits_{n\to\infty} \frac{\frac{d}{dx}log(n)}{\frac{d}{dx}n} = \lim\limits_{n\to\infty} \frac{\frac{1}{n}}{1} = \lim\limits_{n\to\infty} \frac{1}{n} = 0$
$\Rightarrow nlog(n) \in o(n^2)$

d) $2^{\sqrt{\log(n)}} = 2^{(\log(n)^{\frac{1}{2}}} = (2^{log(n)})^{\frac{1}{2}} = \sqrt{n}$
$\lim\limits_{n\to\infty} \frac{n}{\sqrt{n}} = \lim\limits_{n\to\infty} \sqrt{n} = \infty \Rightarrow \sqrt{n} \in \mathcal{O}(n) \iff n \in \Omega(\sqrt{n}) = \Omega\left(2^{\sqrt{\log(n)}}\right)$

e) Angenommen: $n! \in 2^{\mathcal{O}(n)} \iff log(n!) \in \mathcal{O}(n) \iff nlog(n) \in \mathcal{O}(n)$, aber $nlog(n) \notin \mathcal{O}(n)\lightning \Rightarrow n! \notin 2^{\mathcal{O}(n)}$

Aufgabe 2 (Sprachen in P) (6 Punkte)

Sei

$$\textsc{Connected} = \{\langle G\rangle \mid G \text{ ist ein ungerichteter zusammenhängender Graph}\} \text{ und}$$
$$\textsc{Triangle} = \{\langle G\rangle \mid G \text{ ist ein ungerichteter Graph mit einer 3-Clique}\}\ .$$

Zeigen Sie, dass beide Sprachen in P liegen.

Lösung: Für CONNECTED:
Wir zeigen eine Turing Maschine $M_{\text{CONNECTED}}$, wobei CONNECTED $= L(M_{\text{CONNECTED}})$.
$M_{\text{CONNECTED}}$ lehnt alle Eingaben nicht der Form $\langle G \rangle$ mit G ist ein ungerichteter Graph ab (Form-Check).
$M_{\text{CONNECTED}}$ bei Eingabe $\langle G \rangle$ mit $G = (V, E)$ ist ein ungerichteter Graph:

1. Markiere den ersten Knoten s_1
2. Wiederhole die Folgenden Schritte bis keine Knoten mehr markiert werden können:
   a) Durchlaufe alle Kanten $\{a, b\}$ des Graphen G.
   b) Ist a markiert, aber b nicht so markiere b.
   c) Ist b markiert, aber a nicht so markiere a.
3. Überprüfe ob alle Knoten markiert sind. Wenn ja, akzeptiere, sonst lehne ab.

Mit der Länge eines Weges bezeichnen wir die Anzahl der Knoten auf einem Weg. Ein ungerichteter zusammenhängender Graph ist ein Graph, in dem es von jeden Knoten zu jedem anderen einen Weg gibt.
Insbesondere genügt es zu zeigen, dass es von einem bestimmten Knoten s zu jedem anderen einen Weg gibt, da daraus alle anderen Wege über s abgeleitet werden können.
Die Schritte in 2 markieren im k-ten durchlauf genau die Knoten, die in einem Weg von k, aber nicht früher erreicht werden können.
Nach den Ausführungen von 2 sind damit alle Knoten markiert, die von s aus erreicht werden können.
Die Ausgabe in Schritt 3 ist damit korrekt und die DTM $M_{\text{CONNECTED}}$ entscheided Connected.

Weiter analysieren wir die Laufzeit von $M_{\text{CONNECTED}}$.
Schritt 1 und 3 werden nur einmal ausgeführt. Bei jedem Durchlauf bis auf den letzten der Schritte in 2 wird mindestenz ein Knoten markiert. Maximal $|V| + 1$ Durchläufe können demnach durchgeführt werden.
Schritt 1 ist polynomiell ausführbar. Genau so wie mit 3. Auch 2 ist polynomiell, da es maximal $|E| \in \mathcal{O}(|V|^2)$ Schritten, $|V| + 1$ mal ausführt.
Damit ist die Laufzeit von $M_{\text{CONNECTED}}$ polynomiell und Connected in P.

□

Für TRIANGLE.
Wir zeigen eine Turing Maschine M_{TRIANGLE}, wobei TRIANGLE $= L(M_{\text{TRIANGLE}})$.
M_{TRIANGLE} lehnt alle Eingaben nicht der Form $\langle G \rangle$ mit G ist ein ungerichteter Graph ab (Form-Check).
$M_{\text{CONNECTED}}$ bei Eingabe $\langle G \rangle$ mit $G = (V, E)$ ist ein ungerichteter Graph:

1. Wiederhole die folgenden Schritte für jedes 3-Tupel von Knoten aus V: $(s_a, s_b, s_c) \in V \times V \times V$
   a) Wenn nicht $s_a \neq s_b$, $s_b \neq s_c$, $s_c \neq s_a$, gehe zum nächsten Durchlauf.

b) Prüfe, ob $\{s_a, s_b\}$, $\{s_b, s_c\}$, $\{s_c, s_a\}$ in E. Falls alle in E, akzeptiere, sonst gehe zum nächsten Durchlauf.

2. Falls wir nicht schon akzeptiert haben, hat der Graph G keine 3-Clique und wir lehnen ab.

Wir betrachten alle $|V|^3$, 3-Tupel von Knoten in V. Für die Tupel mit 3 paarweise ungleichen Elementen, prüfen wir, ob sie eine 3-Clique in G bilden.
Besitzt G eine solche 3-Clique so finden wir diese in einem Durchlauf von 1 und akzeptieren in 1b.
Besitzt G keine 3-Clique, so erreichen wir Schritt 2 und lehnen ab.
Damit ist die Ausgabe in Schritt 1b oder 2 korrekt und M_{TRIANGLE} entscheidet TRIANGLE.

Weiter analysieren wir die Laufzeit von M_{TRIANGLE}:
Die Schritte in 1 werden $|V|^3$ mal ausgeführt und sowohl 1a als auch 1b sind polynomiell über $|V|$.
Der letzte Schritt 2 auch polynomiell.
Damit ist die Laufzeit von M_{TRIANGLE} polynomiell und TRIANGLE in P.

□

Aufgabe 3 (Klasse P) (6 Punkte)
Sei $L \in$ P und $L^* = \{\epsilon\} \cup \bigcup_{k\in\mathbb{N}} L^k$. Die formale Definition von L^k, der Sprache aller Verkettungen von k Worten aus L, finden Sie auf Präsenzübung 3 Aufgabe 2. Zeigen Sie, dass $L^* \in$ P.

Lösung:
Nach Präsenzübung 3 Aufgabe 2 entscheidet die folgende Turingmaschine $\langle M_{L^k}\rangle$ die Sprache L^k:
$\langle M_{L^k}\rangle$ bei Eingabe w $\in \{0,1\}$

1. Für alle Aufteilungen $v_1, ..., v_k \in L$ von w mache
   a) Für i = 1...k wiederhole Simulation von $\langle M_L\rangle$ mit Eingabe v_i
   b) Falls $\langle M_L\rangle$ in jedem Durchgang akzeptiert hat, akzeptiere
2. Verwerfe.

Laufzeitanalyse von $\langle M_{L^k}\rangle$:
Da L $\in P$ ist die Laufzeit von $\langle M_{L^k}\rangle$ bei einer Eingabe u höchstens $\mathcal{O}(|u|^m)$ für ein festes m $\in \mathbf{N}$. Da jedes v_i höchstens so lang wie w, ist die Laufzeit von jeder Simulation in b) höchstens $\mathcal{O}(|w|^m)$.

Da es insgesamt k Simulation in a) durchgeführt werden und k eine von w unabhängige Konstante kann a) durch $\mathcal{O}(|w|^m)$ abgeschätzt werden.

b) kann trivialerweise in $\mathcal{O}(1)$ entschieden werden.

Insgesamt führen wir in 1) maximal $\binom{|w|}{k}$ Schleifendurchläufe durch und 2) kann durch $\mathcal{O}(1)$ abgeschätzt werden.

Insgesamt hat $\langle M_{L^k} \rangle$ also eine Laufzeit von $\binom{|w|}{k} * \mathcal{O}(|w|^k) + \mathcal{O}(1) \in \mathcal{O}(|w|^k)$ und ist damit also in polynomoieller Zeit entscheidbar, da $\binom{|w|}{k} = \frac{|w|!}{k!*(|w|-k)!}$ eine Konstante bezüglich $\mathcal{O}(|w|^k)$ ist.

Behauptung: Falls L_1 und $L_2 \in P$, so gilt: $L_1 \cup L_2 \in P$

Beweis: Nach Aufgabe 3 aus Präsenzübung 6 gilt, dass da L_1 und $L_2 \in P$ auch $(L_1)^C \text{und} (L_2)^C \in P$ liegen.

Also nach selbiger Aufgabe auch $(L_1)^C \cap (L_2)^C = L_1 \cup L_2 \in P \quad \square$

Dies gilt dann auch dem entsprechend für abzählbare Vereinigungen von Sprachen aus P.

Folglich liegt $L^* \in P$, da $L^* = \{\epsilon\} \cup \bigcup_{k \in \mathbb{N}} L^k$ und damit eine abzählbare Vereinigung von Sprachen aus P ist, da $\{\epsilon\}$ durch die Turingmaschiene: $\langle M_{\{\epsilon\}} \rangle$, welche nur $\{\epsilon\}$ akzeptiert, trivialerweise in polynomieller Zeit entschieden werden kann.
Folglich ist $L^* \in$ P $\quad \square$

Aufgabe 4 (Reduktion mit $\overline{H}$) (6 Punkte)
Beweisen Sie, dass die im Folgenden definierte Sprache des Äquivalenzproblems nicht rekursiv aufzählbar ist. Nutzen Sie in ihrer Reduktion das Komplement des Halteproblems.

$$L = \{(\langle M \rangle, \langle M' \rangle) \mid L(M) = L(M')\}$$

Lösung: Behauptung $\overline{H} \leq L$ Beweis:
Wir definieren die Reduktionsfunktion $f : \{0,1\}^* \to \{0,1\}^*$:

$$f(w) = \begin{cases} \langle M^{(reject)} \rangle & \text{wenn w} \neq \langle M \rangle x \\ (\langle M^{(x)} \rangle, \langle M_{(x)} \rangle) & \text{wenn w} = \langle M \rangle x \end{cases}$$

Sei $\langle M^{(reject)} \rangle$ die Turingmaschine die jede Eingabe ablehnt.

$\langle M^{(x)} \rangle$ sei eine leicht abgeänderte Turingmaschine aus dem Satz 2.10.1, welche wie folgt definiert ist:

$\langle M^{(x)} \rangle$ bei Eingabe z $\in \{0,1\}^*$

1. Berechne |z|.

2. Simuliere M bei Eingabe x für |z| Schritte.

3. Falls M in |z| Schritten hält, so akzeptiere z.

4. Sonst akzeptiere z

Folglich ist $f(w)$ nach diesem Satz 2.10.1 auch berechbar.

Darüber hinaus gilt die Implikation 2.2:

M hält bei Eingabe x nicht $\Rightarrow$ $M^{(x)}$ hält bei jeder Eingabe z $\{0,1\}^*$ (1)

$\langle M_{(x)} \rangle$ sei eine leicht abgeänderte Turingmaschine aus dem Satz 2.10.1, welche wie folgt definiert ist:

$\langle M_{(x)} \rangle$ bei Eingabe z $\in \{0,1\}^*$

1. Berechne |z|.

2. Simuliere M bei Eingabe x für |z| Schritte.

3. Falls M in |z| Schritten hält, so lehne z ab.

4. Sonst akzeptiere z

Folglich ist $f(w)$ nach diesem Satz 2.10.1 auch berechbar.
Darüber hinaus gilt die Implikation 2.2:
M hält bei Eingabe x nicht $\Rightarrow$ $M_{(x)}$ hält bei jeder Eingabe z $\{0,1\}^*$ (2)

Z.z. $(w \in \overline{H} \iff f(w) \in L)$

Richtung $\Longrightarrow$: Angenommen $w \in \overline{H}$.

Das heiSSt, $w = \langle M \rangle x$ mit M hält bei Eingabe x nicht.

Also wird w unter f nach $\langle M^{(x)} \rangle$, $\langle M_{(x)} \rangle$ abgebildet.

Da M bei keiner Eingabe hält, gilt nach (1) und (2) (alternativ folgt dies aus Zeile 3,4 der jeweiligen DTMS), dass sowohl $\langle M^{(x)} \rangle$als auch$\langle M_{(x)} \rangle$ jede Eingabe akzeptiert.

Dementsprechend ist $L((\langle M^{(x)} \rangle) = L((\langle M_{(x)} \rangle)))$und es gilt w $\in f(w)$

Richtung $\Leftarrow$: Angenommen $w \notin \overline{H}$.

Also ist entweder w von falscher Form oder M hält bei Eingabe x.

Ist w von falscher Form, dann ist $f(w) = \langle M^{(reject)} \rangle \notin L$, da $\langle M^{(reject)} \rangle$ nicht von der Form $(\langle M \rangle, \langle M' \rangle)$

Ist $w = \langle M \rangle x$ und M hält bei Eingabe x, dann:

Es existiert ein n mit M hält bei Eingabe x nach n Schritten.

Und für alle $n' > n$ gilt: M hält bei Eingabe x nach n' Schritten.

Damit werden $\langle M^{(x)} \rangle$ und $\langle M_{(x)} \rangle$ bei jeder Eingabe der Länge $n' > n$ halten und die jeweilige Eingabe akzeptieren, bzw ablehnen.

Also gilt $L(\langle M^{(x)} \rangle) = \{z | z \in \{0,1\}^* \text{mit} |n| \leq |z|\}$

und $L(\langle M_{(x)} \rangle) = \{z | z \notin \{0,1\}^* \text{mit} |n| \leq |z|\}$

Folglich gilt: $L(\langle M^{(x)} \rangle) \neq L(\langle M_{(x)} \rangle)$

Damit ist $f(w) = \langle M^{(x)} \rangle, \langle M_{(x)} \rangle \notin L$.

$\Rightarrow \overline{H} \leq L \quad \square$