Prof. Javier Esparza
Kevin Kappelmann, Simon Roßkopf, Lukas Stevens
Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2023 – Hausaufgabenblatt 12

- Bitte beachten Sie, dass in dieser Vorlesung generell Antworten mit Begründung gefordert werden, solange die Aufgabe nicht explizit das Gegenteil sagt.
- Zum Bestehen diese Blattes müssen Sie 50% der Punkte erreichen.
- Update: Es wird Aufgabe 2 korrigiert.

Aufgabe H12.1. (*Wir bieten ein* dynamisches *Umfeld, Kicker, Obstkorb,...*) 2 + 0 Punkte

Sei $A \subseteq \Sigma^*$ eine Sprache in P.

(a) Zeigen Sie, dass die Sprache $\bigcup_{i=1}^{k} A^i = \{w_1 \cdots w_l \mid w_1, \ldots, w_l \in A \wedge l \leq k\}$ für jedes konstante $k \in \mathbb{N}_+$ ebenfalls in P liegt.

(b) (Optional) Zeigen Sie, dass A^+ in P liegt. **Tipp:** Verwenden Sie dynamische Programmierung.

Lösungsskizze.

(a) Wir testen einfach für alle möglichen Zerlegungen von w in höchstens k Teilwörter, ob die jeweiligen Teilwörter alle in A liegen. Für die Zerlegung in *genau* $l \leq k$ Teilwörter gibt es $\binom{n-1}{l-1}$ Möglichkeiten, wobei $n \coloneqq |w|$. Dann gilt:

$$\binom{n-1}{l-1} \leq \binom{n}{l} = \frac{n * \cdots * (n-l+1)}{(n-l)!} \leq \frac{n^l}{1} = n^l$$

Somit müssen wir insgesamt $\mathcal{O}(\sum_{i=1}^{k} n^i) = \mathcal{O}(n^k)$ Zerlegungen betrachten. Da $A \in \mathsf{P}$, können wir für jedes Teilwort $v \in A$ in $\mathcal{O}(p(|v|))$ für ein Polynom p entscheiden. Der Algorithmus ist somit polynomiell in $\mathcal{O}(n^k p(n))$.

(b) Der Algorithmus aus Teil (a) würde eine Laufzeit von $\mathcal{O}(n^n p(n))$ liefern und ist nicht ausreichend. Wir verbessern den Ansatz mit Hilfe von dynamischer Programmierung. Für ein gegebenes Wort w ist dabei die Idee, sich für Präfixe von w zu merken, ob sie in A^+ liegen und in wie viele Teile man sie dazu mindestens zerlegen muss. Im folgenden Code speichern wir diese Information in der Liste L; in $L[n]$ steht also die minimale Anzahl an Teilworten in A, in die man $w_1 \ldots w_n$ zerlegen kann, bzw. ∞ falls dies nicht möglich ist.

In Pseudocode:

- **Eingabe**: Wort $w = w_1 \ldots w_n$
- Lege eine Liste L der Länge $n+1$ an, wobei anfangs $L[0] = 0$ und $L[i] = \infty$ für alle $i > 0$
- Für alle $i \in \{1, \ldots, n\}$
  - Für alle $j \in \{0, \ldots, n-1\}$
    - Wenn $L[j] < \infty \wedge (w_j \ldots w_{i-1}) \in A$, dann setze $L[i] = \min(A[j] + 1, A[i])$
- Gebe $A[n] \leq n$ zurück

Da $A \in \mathsf{P}$, können wir $w \in A$ in $\mathcal{O}(p(n))$ für ein gewisses Polynom p entscheiden. Damit läuft der obige Algorithmus in $\mathcal{O}(n^2 p(n))$, was wiederum polynomiell ist.

Aufgabe H12.2. (*NP Quizzfire*) 0.5*6 Punkte

Entscheiden Sie ob folgende Aussagen wahr sind. Begründen Sie ihre Antwort kurz:

(a) Es gibt keine reguläre Sprache in NP.

(b) Jede Sprache in NP ist unendlich.

(c) Sei $L \in P$. Dann ist $\overline{L}$ unendlich.

(d) Wenn $A \in NP$ dann ist $A^* \in NP$.

(e) Sei $B \in P$ und $A \leq B$. Dann ist $A \in P$.

(f) Sei $A \leq_p B$, $B \leq_p SAT$, $SAT \leq_p A$. Dann sind A, B NP-vollständig.

Lösungsskizze.

(a) Falsch: $\emptyset$ ist in NP und regulär.

(b) Falsch: $\emptyset$ ist in NP und endlich.

(c) Falsch: Σ^* ist in P und $\overline{\Sigma^*} = \emptyset$ ist endlich

(d) Wahr: rate nichtdeterministisch die Zerlegung des Wortes und überprüfe dann jedes Teilwort mit der NTM für A.

(e) Falsch: die Reduktion könnte superpolynomiell viel Zeit beanspruchen.

(f) Wahr: $\leq_P$ is transitiv, somit ist SAT reduzierbar auf A, B (damit sind A, B NP-schwer) und A, B reduzierbar auf SAT (damit sind A, B in NP).

Aufgabe H12.3. (*Total abge*SPACEd) 1 + 2 Punkte

Zeit ist nicht die einzig wertvolle Ressource der Komplexitätstheorie. Die Turingmaschinen der Praxis (aka Computer) besitzen leider nur endlich viel Speicher, weswegen auch der Speicherbedarf eines Programmes von Interesse ist. Wir werden uns daher in dieser Aufgabe auch ein wenig mit Speichergrenzen beschäftigen. Analog zu time_M und TIME aus der Vorlesung definieren wir hierzu:

- $\mathsf{space}_M(w)$: die Anzahl an verschiedenen, besuchten Zellen der DTM M bei Eingabe w.
- $\mathsf{SPACE}(f(n)) \coloneqq \{A \subseteq \Sigma^* \mid \exists \mathsf{DTM}\, M.\ L(M) = A \wedge \forall w \in \Sigma^*.\ \mathsf{space}_M(w) \leq f(|w|)\}$.

(a) Sei $f : \mathbb{N} \to \mathbb{N}$ total. Zeigen Sie: $\mathsf{TIME}(f(n)) \subseteq \mathsf{SPACE}(f(n))$.

(b) Sei PSPACE die Menge der Entscheidungsprobleme, die von deterministischen Turingmaschinen mit polynomiellem Platz entschieden werden können, d.h.

$$\text{PSPACE} \coloneqq \bigcup_{k \in \mathbb{N}, f \in \mathcal{O}(n^k)} \mathsf{SPACE}(f)$$

Sei EXPTIME die Menge der Entscheidungsprobleme, die von deterministischen Turingmaschinen in (einfach) exponentieller Zeit entschieden werden können, d.h.

$$\text{EXPTIME} \coloneqq \bigcup_{k \in \mathbb{N}, f \in \mathcal{O}(2^{n^k})} \mathsf{TIME}\,(f)\,.$$

Zeigen Sie: PSPACE $\subseteq$ EXPTIME.

Tipp: betrachten Sie die Anzahl der Konfigurationen, in der sich eine polynomiell speicherbegrenzte TM befinden kann.

Lösungsskizze.

(a) Sei $A \in \mathsf{TIME}(f(n))$. Dann existiert eine DTM M mit $L(M) = A$ und $\mathsf{time}_M(w) \leq f(|w|)$ für alle $w \in \Sigma^*$. Da M maximal $f(|w|)$ Schritte macht, kann M auch maximal $f(|w|)$ viele Zellen besuchen. Somit folgt $\mathsf{space}_M(w) \leq f(|w|)$ und damit $A \in \mathsf{SPACE}(f(n))$.

(b) Sei $A \in$ PSPACE und M eine DTM für A, die bei Eingabe der Länge n maximal $p(n)$ viel Speicher verwendet. Das Band von M kann dann in maximal $|\Gamma|^{p(n)}$ vielen Möglichkeiten beschrieben sein. Der Kopf kann sich dabei auf $p(n)$ vielen verschiedenen Positionen befinden. Die Maschine kann sich dabei in $|Q|$ vielen verschiedenen Zuständen befinden. Somit kann M maximal $\mathcal{O}(|\Gamma|^{p(n)} * p(n) * |Q|)$ viele Konfigurationen besuchen, bevor sich ihr Lauf wiederholen würde und M somit nie halten würde. Die Anzahl an Konfigurationen ist (einfach) exponentiell bezüglich n.

Um nun $A \in$ EXPTIME zu zeigen, simulieren wir also einfach M und überprüfen dabei, ob M eine Konfiguration doppelt besucht, in welchem Fall wir ablehnen können. Die Simulation hält somit in maximal (einfach) exponentiell vielen Schritten. Somit folgt $A \in$ EXPTIME.

Aufgabe H12.4. (*Unbezahltes Praktikum*) 1 + 4 + 1 Punkte

Die TUM plant eine innovative neues Lehrveranstaltung; eine Mischung aus Seminar und Praktikum. In den ersten v Wochen der Veranstaltung gibt es jeweils einen Vortrag eines Studierenden, der zweite Teil besteht dann aus p Praktika. Dabei gibt es jedoch potenzielle Terminkonflikte: Von den insgesamt n Studierenden ist in jeder Woche $k \in \{1, \ldots, v\}$ nur eine Teilmenge S_k verfügbar, um einen Vortrag zu halten. Außerdem hat jedes Projekt als Voraussetzung, dass ein gewisser Vortrag gehalten wurde. Für jedes Projekt $i \in \{1, \ldots, p\}$, gibt es eine Menge P_i an Studierenden, von denen mindestens eine:r einen Vortrag gehalten haben muss, damit das Projekt bearbeitet werden kann.

PRAKTIKUM:

Gegeben: Endliche Menge S (Studierende), Anzahl Vorträge v, Anzahl Praktika p. Teilmengen $S_k \subseteq S$ (Verfügbarkeit in Woche k), $P_i \subseteq S$ (Voraussetzungen für Praktika)

Problem: Gibt es eine Zuordnung von Studierenden zu Veranstaltungswochen, sodass jede Woche genau ein Studierender (der/die in dieser Woche verfügbar ist) vorträgt und dass jedes Projekt bearbeitet werden kann.

(a) Zeigen Sie, dass PRAKTIKUM in NP liegt.

(b) Zeigen Sie, dass PRAKTIKUM NP-schwer ist, indem sie eine geeignete polynomielle Reduktion von 3KNF-SAT angeben. Zeigen Sie dabei auch die Korrektheit Ihrer Reduktion.

(c) Wenden Sie Ihre Reduktionsfunktion auf die folgende Instanz von 3KNF-SAT an:

$$(x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee x_3)$$

Vergewissern Sie sich, dass Ihr Ergebnis Sinn macht, also dass eine erfüllende Belegung für die SAT Instanz eine Lösung für die PRAKTIKUM Instanz ergibt und umgekehrt.

Lösungsskizze.

(a) Als Zertifikat dient hier eine Reihung von Studierenden. Wir überprüfen dann, dass jede/r Student/in in der zugeordneten Woche verfügbar ist und dass für jedes Projekt mindestens ein Vortrag geplant wurde. Das ist in polynomieller Zeit möglich.

(b) Sei $F := \bigwedge_{i=1}^{k} C_i$ und $C_i := \bigvee_{j=1}^{3} L_{i,j}$ mit Variablen $x_i \dots x_n$ die Eingabe für das 3KNF-SAT Problem.

Wir definieren die Reduktion f, die eine Probleminstanz für Praktikum folgendermaßen konstruiert:

Erzeuge für jede Variable x_i zwei Studierende, s_i und s_i', die x_i und $\neg x_i$ repräsentieren. Es gibt dann n Veranstaltungswochen ($v := n$), in Woche i hat jeweils nur s_i und s_i' Zeit, d.h. $S_i := \{s_i, s_i'\}$.

Für jede der k Klauseln gibt es dann ein Praktikum ($p := k$), dass als Voraussetzungen genau die Studierenden hat, die zu den Literalen in der Klausel korrespondieren. Formal $P_i := \{g(L_{i,1}), g(L_{i,2}), g(L_{i,3})\}$, wobei $g(x_i) = s_i$ und $g(\neg x_i) = s_i'$. Die Reduktion f ist in polynomieller Zeit berechenbar.

Korrektheit:

Sei σ eine erfüllende Belegung für eine Formel F.

Sei $v_1, \dots, v_n$ eine Reihung mit

$$v_i = \begin{cases} s_i \text{ falls } \sigma(x_i) = 1 \\ s_i' \text{ falls } \sigma(x_i) = 0 \end{cases}$$

Dann gilt wegen der Definition von f dass $v_i \in S_i$ für alle $i \in \{1, \dots, n\}$.

Da σ eine erfüllende Belegung ist, gibt es mindestens ein Literal $L_{i,j}$ mit $\sigma(L_{i,j}) = 1$ für alle $i \in \{1, \dots, k\}$. Für ein solches Literal $L_{i,j}$ gilt dann per Konstruktion auch dass $g(L_{i,j}) \in P_i$ und dass $g(L_{i,j})$ in der Reihung $v_1, \dots, v_n$ enthalten ist. Damit erfüllt die Reihung die Anforderungen von Praktikum.

Sei umgekehrt $v_1, \dots, v_n$ eine valide Reihung von Studierenden. Konstruiere σ so, dass

$$\sigma(x_i) = \begin{cases} 1 \text{ falls } v_i = s_i \\ 0 \text{ falls } v_i = s_i' \end{cases}$$

Laut der Definition von Praktikum gilt, dass die Reihung für jedes $i \in \{1, \dots, k\}$ ein $v_j \in P_i$ enthält. Damit folgt per Konstruktion, dass es für jede Klausel C_i ein j gibt, so dass $\sigma(L_{i,j}) = 1$. Damit ist σ eine erfüllende Belegung.

(c)
- $v = 4$
- $p = 2$
- $S_i = \{s_i, s_i'\}, 1 \leq i \leq 4$
- $P_1 = \{s_1, s_3', s_4\}$
- $P_2 = \{s_1', s_2, s_3\}$

Aufgabe H12.5. (*München ist gerettet, Dietersheim ist in Not*) Nur Freundschaftspunkte

Diese Aufgabe ist optional und schließt das Kapitel Dr. Evilsparza ab.

Trotz all euren Bemühungen ist es Dr. Evilsparza gelungen, die heißumkämpfte Professur an der Freien Universität Dietersheim zu bekommen. Es ist nun wieder alles beim Alten: Dr. Evilsparza ist wieder Prof. Evilsparza und richtet großes Unheil an seiner Universität an. Der einzige Unterschied: das Unheil findet nun in Dietersheim und nicht mehr in München statt.

Eure Informatikerfreund:innen in Dietersheim sind davon ziemlich überfordert. Diese wilden Maschinen sind ihnen ganz unheimlich. Aufgebracht erzählen sie euch von ihrer neuesten Hausaufgabe, die natürlich niemand lösen kann. Doch nach all euren Abenteuern seid ihr nun Expertinnen, Maschinenkonstrukteure und -erzwingerinnen, echte Informatikerhandwerker:innen. Ein – nun aber wirklich – letztes Mal rettet ihr also die Welt und helft euren Dietersheimer Freund:innen bei ihren Hausaufgaben. Alle Augen sind auf euch. Die Angabe ist wie folgt:

"Sei $f : \mathbb{N} \to \mathbb{N}$ berechenbar und total. Zeigen Sie: Es existiert eine *entscheidbare* Sprache $L \notin \mathsf{TIME}(f(n))$. Verwenden Sie hierfür einen Diagonalisierungsansatz ähnlich zu Satz 5.33 aus der Vorlesung der Theoretischen Informatik der TU München."

Da kribbelt es euch sofort in den Fingern. Eure THEO-Instinke sagen euch, dass die Sprache

$$L \coloneqq \{w \in \{0,1\}^* \mid M_w[w] \text{ akzeptiert nicht in } \leq f(|w|) \text{ Schritten}\}.$$

zum Erfolg führen könnte...

Lösungsskizze. L ist entscheidbar: Wir simulieren bei Eingabe w die Maschine $M_w[w]$ für $f(|w|)$ Schritte (f ist total und berechenbar). Falls $M_w[w]$ in dieser Dauer akzeptiert, lehnen wir ab, sonst akzeptieren wir.

$L \notin \mathsf{TIME}(f(n))$: Angenommen $L \in \mathsf{TIME}(f(n))$. Dann gibt es eine Gödelnummer w für eine DTM M_w mit $L(M_w) = L$ und $\mathsf{time}_{M_w}(x) \leq f(|x|)$ für alle $x \in \Sigma^*$. Insbesondere gilt daher $\mathsf{time}_{M_w}(w) \leq f(|w|)$ (*). Nun folgt

$$\begin{aligned} w \in L(M_w) = L &\iff M_w \text{ akzeptiert } w \text{ und } \mathsf{time}_{M_w}(w) \leq f(|w|) && \text{(Def. } w \in L(M_w) \text{ und (*))} \\ &\iff M_w \text{ akzeptiert } w \text{ in maximal } f(|w|) \text{ Schritten} \\ &\iff w \notin L = L(M_w) && \text{(Def. } L\text{).} \end{aligned}$$

Widerspruch! Somit gilt $L \notin \mathsf{TIME}(f(n))$.

> The more I think about language, the more it amazes me that people ever understand each other at all.
>
> — Kurt Gödel

Bonus: Eine schöne Antwort zu "What is the enlightenment I'm supposed to attain after studying finite automata?"

Vielen Dank für eure Mitarbeit, vor allem bei den Abenteuern mit ~~Dr~~Prof. Evilsparza. Wir wünschen viel Erfolg bei der Klausur und Freude beim zukünftigen THEOretisieren.