

15. Aufgabenblatt

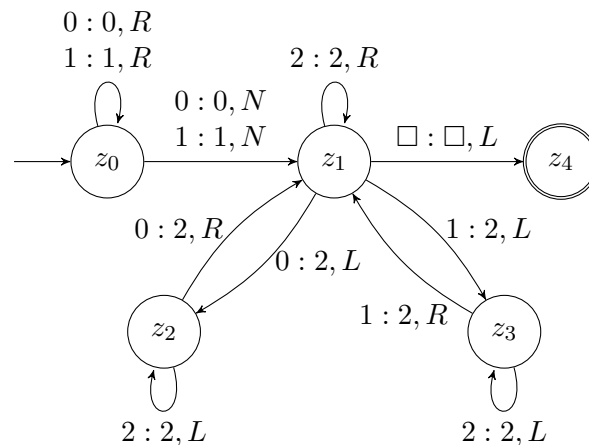
(Besprechung in den Tutorien 12.02.2024–18.02.2023)

Aufgabe 1. Analyse einer Turing-Maschine (schriftlicher Test WS20/21)

Betrachten Sie die folgende Turing-Maschine

$$M = (\{z_0, z_1, z_2, z_3, z_4\}, \{0, 1\}, \{0, 1, 2, \square\}, \delta, z_0, \square, \{z_4\}),$$

wobei δ die folgende graphische Darstellung hat:



- Wird das Wort 00110 von M akzeptiert?
- Zeigen Sie, dass es ein Eingabewort der Länge 3 gibt, das von M nicht akzeptiert wird.
- Geben Sie die von M akzeptierte Sprache $T(M)$ an (ohne Begründung).
- Ist die von M akzeptierte Sprache $T(M)$ in der Komplexitätsklasse P , das heißt, ist $T(M)$ von einer deterministischen Turing-Maschine in polynomieller Zeit entscheidbar?

Aufgabe 2. Polynomzeitreduktion (Schriftlicher Test WS 20/21)

Betrachten Sie die Probleme A und B, die wie folgt definiert sind:

PROBLEM A

Eingabe: Ein ungerichteter Graph $G = (V_G, E_G)$ und eine Zahl $k \in \mathbb{N}$.

Frage: Existiert eine Teilmenge $X \subseteq V_G$ mit $|X| = k$, sodass jede Kante in E_G einen Endpunkt in X hat?

PROBLEM B

Eingabe: Ein ungerichteter Graph $H = (V_H, E_H)$ und eine Zahl $\ell \in \mathbb{N}$.

Frage: Existiert eine Menge $Y \subseteq E_H$ mit $|Y| = \ell$, sodass es für jede Kante $e \in E_H \setminus Y$ mindestens eine Kante $f \in Y$ mit $f \cap e \neq \emptyset$ gibt?

Betrachten Sie folgende Reduktion von PROBLEM A auf PROBLEM B:

- Starte mit $V_H = \emptyset$ und $E_H = \emptyset$.
- Für jeden Knoten $v \in V_G$ füge einen Knoten $v' \in V_H$ hinzu.

3. Füge $2k$ Knoten v_1, \dots, v_k und v'_1, \dots, v'_k zu V_H hinzu.
 4. Für jedes $i \in \{1, 2, \dots, k\}$, füge die Kante $\{v_i, v'_i\}$ zu E_H hinzu.
 5. Für jede Kante $\{v, w\} \in E_G$ füge die Kante $\{v', w'\}$ zu E_H hinzu.
 6. Für jedes Paar von Kanten $\{v, w\} \in E_G$ und $\{x, y\} \in E_G$ mit $w = x$ und $v \neq y$ füge die Kante $\{v', y'\}$ zu E_H hinzu.
 7. Für jeden Knoten $v \in V_G$ und jedes $i \in \{1, 2, \dots, k\}$ füge die Kanten $\{v_i, v'\}$ zu E_H hinzu.
 8. Setze $\ell := k$.
- (a) Durch Weglassen eines der acht Schritte der Reduktion ergibt sich eine Polynomzeitreduktion von PROBLEM A auf PROBLEM B. Welcher der acht Schritte muss hierfür weggelassen werden? (Ohne Begründung)
- (b) Zeigen Sie, dass nach Weglassen des in (a) genannten Schrittes die Reduktion eine Polynomzeitreduktion von PROBLEM A auf PROBLEM B ist. Zeigen Sie dafür, dass die Reduktion in polynomieller Zeit berechnet werden kann, total ist und korrekt ist.

Aufgabe 3. „Lösungen“ einiger offener Probleme der Komplexitätstheorie

Wo liegen die Fehler in folgenden „Beweisen“?

- (a) Wir zeigen, dass $P \neq NP$. Angenommen, dass $P = NP$. Diese Aussage führen wir zu einem Widerspruch, indem wir zeigen, dass das spezielle Halteproblem K in NP liegt und damit entscheidbar ist.

Bekanntlich ist ein Problem in NP , falls es durch einen „Guess-and-Check“-Algorithmus gelöst werden kann. Wenn eine Turing-Maschine M auf dem leeren Band hält, dann existiert eine Zahl $k \in \mathbb{N}$, sodass M innerhalb von k Schritten hält. Wir können also eine Zahl k raten und dann M simulieren. Wenn M innerhalb von k Schritten hält, dann akzeptieren wir. Somit ist $K \in NP$.

- (b) Wir zeigen, dass $P = NP$. Bekanntlich ist das Problem CLIQUE NP -schwer. Wenn CLIQUE in Polynomzeit gelöst werden kann, ist also $P = NP$.

Sei $G = (V, E)$ ein Graph und $k \in \mathbb{N}$. Folgender Algorithmus bestimmt, ob G eine Clique der Größe k enthält.

Zunächst iteriert der Algorithmus über alle k -elementigen Teilmengen $S \subseteq V$. Für jedes S bestimmt der Algorithmus, ob S eine Clique ist. Dazu muss er bloß für jedes Paar $u, v \in S$ mit $u \neq v$ prüfen, ob $\{u, v\} \in E$. Wenn S eine Clique ist, dann akzeptiert der Algorithmus. Wenn keine der überprüften Mengen eine Clique ist, dann lehnt er ab.

Korrektheit: Angenommen, (G, k) ist eine Ja-Instanz für CLIQUE. Dann enthält G eine Clique S der Größe k . Da der Algorithmus über alle Teilmengen von V der Größe k iteriert, überprüft er auch S . Dann stellt er fest, dass S eine Clique ist und akzeptiert.

Angenommen, der Algorithmus akzeptiert (G, k) . Dann heißt das, dass der Algorithmus eine Menge S der Größe k gefunden hat, sodass S eine Clique ist. Folglich ist (G, k) eine Ja-Instanz für CLIQUE.

Laufzeit: Sei $n := |V|$. Es gibt $\binom{n}{k}$ Teilmengen von V der Größe k . Zudem gilt $\binom{n}{k} \in \mathcal{O}(n^k)$. Um zu überprüfen, ob eine Menge der Größe k eine Clique ist, genügen k^2 Schritte. Also ist die Laufzeit dieses Algorithmus $\mathcal{O}(n^k \cdot k^2) \subseteq \mathcal{O}(n^k \cdot n^2) = \mathcal{O}(n^{k+2})$. Dies ist ein Polynom $(k+2)$ -ten Grades.

- (c) Wir zeigen, dass $P \neq NP$, indem wir zeigen, dass 3-SAT nicht in Polynomzeit entschieden werden kann. Für eine Formel F mit n Variablen gibt es 2^n verschiedene Belegungen, also mehr als polynomiell. Zu überprüfen, ob eine erfüllende Belegung existiert, erfordert daher mehr als polynomiell viele Schritte, da mit weniger Schritten nicht alle Belegungen überprüft werden können.
- (d) Wir zeigen, dass $NP = PSPACE$, indem wir zeigen, dass 3-SAT in PSPACE liegt. Da 3-SAT NP-schwer ist und $PSPACE \subseteq NP$ bereits bekannt ist, folgt daraus die Aussage. Wir zeigen $3\text{-SAT} \in PSPACE$ durch Angabe eines Algorithmus, der nur polynomiell viel Speicher benutzt. Der Algorithmus iteriert über alle Belegungen einer Formel mit n Variablen und überprüft, ob sie die Formel erfüllen. Um eine solche Belegung zu kodieren, genügen n Bits. Zu jedem Zeitpunkt stehen auf dem Band der Turing-Maschine nur die Formel F , sowie die aktuelle Belegung. Dann überprüft der Algorithmus, ob diese Belegung die Formel erfüllt. Um von einer Belegung zur nächsten zu iterieren, muss kein zusätzlicher Speicher verwendet werden.