

Präsenzblatt 0

Aufgabe 0.1

Sei $G = (V, E)$ ein ungerichteter Graph mit $V = \{0, \dots, n-1\}$. Beschreiben Sie eine Kodierung von G

- in Adjazenzmatrix-Darstellung über dem Alphabet $\Sigma = \{0, 1\}$.
- in Adjazenzlisten-Darstellung über dem Alphabet $\Sigma = \{0, 1, \#\}$.
- in Adjazenzlisten-Darstellung über dem Alphabet $\Sigma = \{0, 1\}$.

Aufgabe 0.2

Sei \mathcal{M} die Menge aller DFAs über dem Alphabet $\Sigma = \{0, 1\}$. Analog zur Gödelnummerierung von Turingmaschinen betrachten wir eine präfixfreie injektive Abbildung $c: \mathcal{M} \rightarrow \Sigma^+$. Gibt es für eine geeignete Nummerierung c einen universellen DFA, also einen DFA, der eine Eingabe $c(M)w$ für $M \in \mathcal{M}$ und $w \in \Sigma^*$ genau dann akzeptiert, wenn der DFA M die Eingabe w akzeptiert?

Aufgabe 0.3

Wir betrachten eine Grammatik $G = (\Sigma, V, S, P)$.

- Zeigen Sie, dass es eine Turingmaschine gibt, die die Sprache $L(G)$ entscheidet, wenn P nur Regeln der Form $\alpha \rightarrow \beta$ mit $\alpha, \beta \in (V \cup \Sigma)^*$ und $|\beta| > |\alpha|$ und Regeln der Form $A \rightarrow a$ für $A \in V$ und $a \in \Sigma$ enthält.
- Gibt es eine Turingmaschine, die die Sprache $L(G)$ entscheidet, wenn keine Einschränkungen an die Regeln gestellt werden?

Aufgabe 0.4

Wir betrachten zwei Probleme A und B . Problem A lässt sich durch folgenden Algorithmus lösen:

- Berechne aus einer Eingabe x der Länge n für Problem A in Zeit $O(n^3)$ eine Eingabe y der Länge $N = O(n^3)$ für Problem B .
- Rufe einen Algorithmus für Problem B mit Eingabe y auf.
- Gib die Ausgabe dieses Algorithmus als Ausgabe für Problem A aus.

Die Komplexität eines Problems ist $O(f(n))$, wenn es einen Algorithmus für das Problem mit Laufzeit $O(f(n))$ gibt. Ansonsten sagen wir, dass das Problem Komplexität $\Omega(f(n))$ hat.

Beantworten Sie die folgenden Fragen und begründen Sie Ihre Antworten:

- Was kann man über die Komplexität von Problem A aussagen, wenn Problem B Komplexität $O(N^2)$ hat, wobei N die Länge einer Eingabe für Problem B ist?
- Was lässt sich über die Komplexität von Problem A schlussfolgern, wenn wir wissen, dass es keinen Algorithmus für Problem B mit Laufzeit $O(e^N)$ gibt?
- Was lässt sich über die Komplexität von Problem B aussagen, wenn wir einen Algorithmus für Problem A mit Laufzeit $O(n)$ kennen?
- Was kann man über die Komplexität von Problem B schlussfolgern, wenn die Komplexität von Problem A $\Omega(e^n)$ ist?
- Was kann man über die Komplexität von Problem B schlussfolgern, wenn die Komplexität von Problem A $\Omega(n^2)$ ist?