Lösungsblatt 11

Aufgabe 1

Sei $G = (V, E)$ ein Graph und $S \subseteq V$ eine beliebige Menge von Knoten. S heißt *stabil*, falls keine zwei Knoten aus S durch eine Kante verbunden sind, d. h. :

$$S \text{ stabil} \iff \forall u, v \in S \colon \{u, v\} \notin E.$$

Zeigen Sie, dass das folgende Entscheidungsproblem NP-vollständig ist:

<u>Stabilität</u>
Eingabe: Ein Graph G und eine Zahl $k \in \mathbb{N}$.
Frage: Gibt es eine stabile Menge S der Größe $|S| = k$?

Lösung

<u>Stabilität $\in \mathsf{NP}$</u>

Mit *guess and check*: Die nichtdeterministische Turingmaschine rät für jeden Knoten, ob dieser in S enthalten ist und überprüft, ob es genau k Knoten in S gibt und ob irgendeine Kante zwei Knoten aus S miteinander verbindet.

<u>Stabilität ist NP-hart</u>

Wir reduzieren Clique auf Stabilität. Sei x die Kodierung einer Clique-Instanz (G, k) mit $G = (V, E)$. Wähle $f(x)$ als Kodierung einer Stabilität-Intanz $(\overline{G}, k)$ mit dem *Komplementgraphen* $\overline{G} = (V, \binom{V}{2} \setminus E)$ und derselben Zahl k. f ist total und in Polynomialzeit berechenbar und für alle Kodierungen x von Clique-Instanzen gilt:

$$\begin{aligned} x \in \textsc{Clique} &\iff \exists S \subseteq V, |S| = k \colon \forall u, v \in S, u \neq v \colon \{u, v\} \in E \\ &\iff \exists S \subseteq V, |S| = k \colon \forall u, v \in S, u \neq v \colon \{u, v\} \notin \overline{E} \\ &\iff f(x) \in \textsc{Stabilität}. \end{aligned}$$

Da Clique NP-hart ist, ist es Stabilität auch.

Hinweis zur Notation: Für eine Menge A und eine natürliche Zahl k ist $\binom{A}{k}$ die Menge aller k-elementigen Teilmengen von A, z. B.:

$$\binom{\{a, b, c\}}{2} = \{\{a, b\}, \{a, c\}, \{b, c\}\}.$$

Auf ähnliche Weise schreibt man oft 2^A für die Potenzmenge $\mathcal{P}(A)$ und B^A für die Menge $A \to B$ aller Funktionen von A nach B. Ein Vorteil dieser Notationen ist, dass dann für endliche Mengen die Gleichungen $|\binom{A}{k}| = \binom{|A|}{k}$, $|2^A| = 2^{|A|}$ und $|B^A| = |B|^{|A|}$ gelten.

Aufgabe 2

Sei $f\colon \mathbb{N} \to \mathbb{N}$ eine beliebige Funktion. Welche der folgenden Aussagen implizieren welche? Beweisen Sie Ihre Antworten.

(a) f ist berechenbar

(b) f ist zeitkonstruierbar

(c) f ist platzkonstruierbar

Lösung

Es gilt folgende (strikte) Hierarchie:

$$f \text{ ist zeitkonstruierbar} \overset{(1)}{\Longrightarrow} f \text{ ist platzkonstruierbar} \overset{(2)}{\Longrightarrow} f \text{ ist berechenbar.}$$

Beweis von (1)

Sei f zeitkonstruierbar. Dann gibt es eine deterministische k-Band-Turingmaschine M, die bei Eingabe a^n nach genau $f(n)$ Schritten hält. Diese verwendet natürlich höchstens $f(n)$ Felder auf jedem Arbeitsband. Sei dann M' eine deterministische $(k+1)$-Band-Turingmaschine, die sich auf den ersten k Bändern wie M verhält und auf Band $k+1$ immer das aktuelle Feld markiert und danach den Leseschreibkopf nach rechts bewegt. Verändert man M' nun so, dass diese den Bandinhalt von Band $k+1$ auf alle anderen Bänder kopiert, so erhält man eine deterministische Turingmaschine gemäß der Definition der Platzkonstruierbarkeit. Somit ist f platzkonstruierbar.

Bemerkung: Die Umkehrung dieser Implikation gilt nicht, da die Funktion $f\colon \mathbb{N} \to \mathbb{N}$, $n \mapsto \log n$ platz-, aber nicht zeitkonstruierbar ist.

Beweis von (2)

Sei f platzkonstruierbar. Dann gibt es eine deterministische Turingmaschine M, die bei Eingabe a^n genau $f(n)$ Felder auf den Arbeitsbändern markiert (z. B. mit 1) und dann hält. Verändert man M nun so, dass sie vor Beginn der Berechnung das Eingabewort durch seine Unärdarstellung ersetzt und nach dem halten die Binärdarstellung von $1^{f(n)}$ berechnet und diese ausgibt, so erhält man eine deterministische Turingmaschine, die f berechnet.

Bemerkung: Die Umkehrung dieser Implikation gilt nicht, da die Funktion $f\colon \mathbb{N} \to \mathbb{N}$, $n \mapsto \log\log n$ berechenbar, aber nicht platzkonstruierbar ist.

Aufgabe 3

Sei $G = (V, E)$ ein gerichteter Graph. Ein *Pfad der Länge* ℓ ist ein Tupel $(v_0, \ldots, v_\ell) \in V^{\ell+1}$ mit $(v_{i-1}, v_i) \in E$ für alle $i \in [\ell]$. Ein Pfad, in dem die erste und letzte Komponente gleich sind, heißt *Kreis*. Kreise der Länge 0 nennt man *trivial*. Ein *gerichteter kreisfreier Graph* (engl. *directed acyclic graph*, kurz: *DAG*) ist ein gerichteter Graph, der keine nichttrivialen Kreise enthält.

Zeigen Sie, dass das folgende Entscheidungsproblem NL-vollständig bezüglich $\leq_{\log}$ ist:

DAGAP

Eingabe:	Ein gerichteter Graph $G = (V, E)$ und zwei Knoten $s, t \in V$.
Frage:	Ist G ein gerichteter kreisfreier Graph mit einem Pfad von s nach t?

Hinweis: Sie dürfen wieder die NL-Vollständigkeit des folgenden Problems annehmen:

GAP

Eingabe:	Ein gerichteter Graph $G = (V, E)$ und zwei Knoten $s, t \in V$.
Frage:	Gibt es in G einen Pfad von s nach t?

Diese wird in Vorlesungseinheit 39 bewiesen.

Lösung

Wir zeigen, dass DAGAP in NL liegt und NL-hart ist.

DAGAP $\in$ NL

Der NL-Algorithmus aus Vorlesungsfolie 31.1 löst das Problem für beliebige Graphen, also insbesondere auch für azyklische Graphen.

DAGAP ist NL-hart

Wir zeigen GAP $\leq_{\log}$ DAGAP. Sei x eine Kodierung einer GAP-Instanz bestehend aus einem Graph $G = (V, E)$ und zwei Knoten $s, t \in V$. Wähle $f(x)$ als Kodierung einer DAGAP-Instanz bestehend aus einem DAG $G' = (V', E')$ und zwei Knoten $s', t' \in V'$ mit $V' = V \times \{1, \dots, |V|\}$,

$$E' = \Big\{ \big((u, i), (v, i + 1)\big) \,\Big|\, (u, v) \in E \cup \{(t, t)\} \wedge 1 \leq i < |V| \Big\},$$

$s' = (s, 1)$ und $t' = (t, |V|)$. Von G werden also $|V|$ Kopien erstellt, durchnummeriert und die Kanten der i-ten Kopie in die $(i + 1)$-te gelenkt. f ist total und in logarithmischem Platz berechenbar.

Dass G' für jedes G ein DAG ist, folgt daraus, dass die zweite Komponente eines Knotens in G' entlang einer Kante immer steigt. Enthielte G' einen Kreis, so gäbe es eine Kante $((u, i), (v, j))$ mit $i > j$, was unmöglich ist. Außerdem gilt

$$\begin{aligned}
x \in \text{GAP} &\iff \text{es gibt in } G \text{ einen Pfad } (u_1, \dots, u_\ell) \text{ mit } u_1 = s,\ u_\ell = t,\ \ell \leq |V| \\
&\qquad \text{und } u_i \neq u_j \text{ für alle } 1 \leq i < j \leq \ell \\
&\iff \text{es gibt in } G' \text{ einen Pfad } ((u_1, 1), \dots, (u_\ell, \ell), (u_\ell, \ell + 1), \dots, (u_\ell, |V|)) \\
&\qquad \text{mit } u_1 = s,\ u_\ell = t,\ \ell \leq |V| \text{ und } u_i \neq u_j \text{ für alle } 1 \leq i < j \leq \ell \\
&\iff f(x) \in \text{DAGAP}
\end{aligned}$$

für alle Kodierungen x von GAP-Instanzen.

Aufgabe 4

Entscheiden Sie für jedes der gegebenen Klassenpaare, welche Klasse in der jeweils anderen als Teilmenge enthalten ist und welche nicht. Beweisen Sie Ihre Antworten.

1. $\mathsf{NTIME}(n^2)$ und $\mathsf{DSPACE}(n^3)$
2. $\mathsf{DTIME}(3n^2 + (\log n)^4)$ und $\mathsf{DTIME}(n^2 + 1)$

Lösung

Für die Beweise wurden folgende Aussagen aus der Vorlesung verwendet:

(a) Beziehungen zwischen:

- $\mathsf{DTIME}(f)$ und $\mathsf{DTIME}(\mathcal{O}(f))$
- $\mathsf{NTIME}(f)$ und $\mathsf{NTIME}(\mathcal{O}(f))$
- $\mathsf{DSPACE}(f)$ und $\mathsf{DSPACE}(\mathcal{O}(f))$
- $\mathsf{NSPACE}(f)$ und $\mathsf{NSPACE}(\mathcal{O}(f))$

(b) Beziehungen zwischen $\mathsf{DTIME}(f)$, $\mathsf{NTIME}(f)$, $\mathsf{DSPACE}(f)$ und $\mathsf{NSPACE}(f)$.

(c) Der Satz von Savitch

(d) Die Hierarchiesätze

Machen Sie sich bitte klar, an welchen Stellen welche Aussagen verwendet wurden und überprüfen Sie, dass alle notwendigen Bedingungen der jeweiligen Aussagen erfüllt sind.

1. Wegen

$$\mathsf{NTIME}(n^2) \subseteq \mathsf{DSPACE}(n^2) \subsetneq \mathsf{DSPACE}(n^3)$$

ist $\mathsf{NTIME}(n^2)$ eine echte Teilmenge von $\mathsf{DSPACE}(n^3)$.

2. Wegen

$$\begin{aligned}\mathsf{DTIME}(3n^2 + (\log n)^4) &= \mathsf{DTIME}(\mathcal{O}(3n^2 + (\log n)^4)) = \mathsf{DTIME}(\mathcal{O}(n^2)) \\ &= \mathsf{DTIME}(\mathcal{O}(n^2 + 1)) = \mathsf{DTIME}(n^2 + 1)\end{aligned}$$

sind beide Klassen gleich.