Grundzüge der Theoretischen Informatik, WS 21/22: Musterlösung zum 10. Übungsblatt

Julian Dörfler

Aufgabe A10.1 (Unäre Konstruierbarkeit) *(4 Punkte)*
Beweisen Sie:

(a) Genau dann, wenn t zeitkonstruierbar ist, gibt es eine $O(t)$-zeitbeschränkte Turingmaschine M, die bei Eingabe 1^n die Ausgabe $1^{t(n)}$ produziert.

(b) Genau dann, wenn s platzkonstruierbar ist, gibt es eine $O(s)$-platzbeschränkte Turingmaschine M (mit zusätzlichem Eingabeband), die bei Eingabe 1^n die Ausgabe $1^{s(n)}$ produziert.

Lösung A10.1 (Unäre Konstruierbarkeit)

(a) Wenn t zeitkonstruierbar ist, gibt es eine Turingmaschine M, die auf Eingabe 1^n in Zeit $O(t(n))$ die Ausgabe $\mathsf{bin}(t(n))$ produziert. Wir ändern M so ab, dass danach $1^{t(n)}$ auf ein zusätzliches Band geschrieben wird. Dazu dekrementieren wir schrittweise die durch $\mathsf{bin}(t(n))$ dargestellte Zahl und fügen eine 1 auf dem neuen Band hinzu.

Für die Analyse der Laufzeit nutzen wir, dass das $t(n)$-fache (aufeinanderfolgende) dekrementieren eines 0,1-Bitstrings $O(t(n))$ Zeit benötigt (amortisierte Analyse!). Insgesamt ist die resultierende Turingmaschine also immer noch $O(t(n))$ zeitbeschränkt.

Umgekehrt, wenn es eine Turingmaschine M' gibt, die auf Eingabe 1^n in Zeit $O(t(n))$ die Ausgabe $1^{t(n)}$ produziert, können wir M' so abändern, dass danach für jede 1 des Ergebnisses auf einem zusätzlichen Band eine Binärzahl inkrementiert wird. Dies benötigt ebenfalls nur $O(t(n))$ Zeit, also ist t zeitkonstruierbar.

(b) Wenn s platzkonstruierbar ist, gibt es eine Turingmaschine M, die auf Eingabe 1^n mit $O(s(n))$ Platz die Ausgabe $\mathsf{bin}(s(n))$ produziert. Wir ändern M so ab, dass danach $1^{s(n)}$ auf ein zusätzliches Band geschrieben wird. Dazu dekrementieren wir schrittweise die durch $\mathsf{bin}(s(n))$ dargestellte Zahl und fügen eine 1 auf dem neuen Band hinzu.

Für die Analyse des Platzverbrauchs nutzen wir, dass die Zahl $\mathsf{bin}(s(n))$ $O(\log(s(n)))$ Zellen zur Speicherung benötigt und das Ergebnis $s(n)$ Zellen benötigt. Insgesamt ist die resultierende Turingmaschine also immer noch $O(s(n))$ platzbeschränkt.

Umgekehrt, wenn es eine Turingmaschine M' gibt, die auf Eingabe 1^n mit $O(t(n))$ Platz die Ausgabe $1^{s(n)}$ produziert, können wir M' so abändern, dass danach für jede 1 des Ergebnisses auf einem zusätzlichen Band eine Binärzahl inkrementiert wird. Dies benötigt ebenfalls nur $O(s(n)) + O(\log(s(n))) = O(s(n))$ Platz, also ist s platzkonstruierbar.

Aufgabe A10.2 (NTM vs DTM) *(4 Punkte)*
Zur Erinnerung: Eine DTM erkennt eine Sprache L, wenn sie auf allen $x \in L$ hält und akzeptiert und auf allen $x \notin L$ nicht hält oder hält und verwirft. Eine NTM erkennt eine Sprache L, wenn sie für alle $x \in L$ einen akzeptierenden Berechnungspfad hat und für alle $x \notin L$ keinen akzeptierenden Berechnungspfad hat.

Zeigen Sie, dass jede Sprache die von einer NTM erkannt wird, auch von einer DTM erkannt wird. (Man könnte dieses Resultat $\mathsf{RE} = \mathsf{NRE}$ nennen.)

Lösung A10.2 (NTM vs DTM) Sei N eine NTM, die L erkennt. Wir konstruieren nun eine DTM M die L rekursiv aufzählt wie folgt:

> Bei Eingabe x führe eine Breitensuche auf dem Konfigurationsgraph CG_N von N ausgehend von $\mathrm{SC}_N(x)$ aus. Sobald hierbei eine akzeptierende Konfiguration gefunden wird, akzeptiere. Falls die Breitensuche ohne Erfolg terminiert, verwirf.

Wenn nun $x \in L$ ist, dann gibt es eine akzeptierende Konfiguration C mit $\mathrm{SC}_N(x) \models_N C$. Da M eine Breitensuche[1] ausführt und jeder Knoten nur endlichen Grad hat, findet M in dieser Breitensuche also irgendwann C und akzeptiert somit x.

Wenn nun aber $x \notin L$ ist, dann gibt es keine akzeptierenden Konfigurationen C mit $\mathrm{SC}_N(x) \models_N C$. Somit wird eine Breitensuche von $\mathrm{SC}_N(x)$ aus entweder nicht terminieren oder erfolglos terminieren, M divergiert also oder verwirft.

Aufgabe A10.3 (Inklusion in NP) *(4 Punkte)*
Beweisen Sie, dass wenn ein Problem L einen Polynomialzeit-Verifizierer hat, dann gilt $L \in \mathsf{NP}$.

Lösung A10.3 (Inklusion in NP) Sei M ein Polynomialzeit-Verifizierer für L, und sei p ein Polynom, so dass für alle $x \in L$ gilt: $x \in L$ genau dann, wenn es ein $c \in \{0,1\}^{\leq p(|x|)}$ gibt mit $[x, c] \in L(M)$.

Wir bauen eine nichtdeterministische Polynomialzeit-Turing-Maschine N für die gilt $L(N) = L$. Auf Eingabe x berechnen wir zunächst $p(|x|)$ in $O(p(|x|))$ Schritten. Dann raten wir einen String c der Länge höchstens $p(|x|)$ in $O(p(|x|))$ Schritten. Schließlich simulieren wir M auf Eingabe $[x, c]$. Akzeptiert M, dann akzeptiert N.

Ist $x \in L$, dann rät N ein c mit $[x, c] \in L(M)$, also hat N einen akzeptierenden Berechnungspfad, also ist $x \in L(N)$. Sei nun $x \in L(N)$. N akzeptiert nur dann, wenn M ein Paar $[x, c]$ akzeptiert, wobei c auf dem entsprechenden Berechnungspfad von N geraten wurde. Nach Konstruktion ist $|c| \leq p(|x|)$. Aus $[x, c] \in L(M)$ folgt also $x \in L$.

Aufgabe A10.4 (UCONN) *(4 Punkte)*
Wir definieren

$$\mathsf{UCONN} = \{(G, s, t) \mid \text{Es gibt einen Pfad von } s \text{ nach } t \text{ in } G\},$$

[1] Machen Sie sich klar, wieso eine Tiefensuche hier nicht ausreichen würde

wobei wir hier ungerichtete Graphen verwenden.

Zeigen Sie, dass $\mathsf{UCONN} \in \mathsf{NL} := \mathsf{NSpace}(O(\log n))$.

Über eine geeignete Kodierung von G müssen Sie sich keine Gedanken machen. Sie dürfen annehmen, dass eine NTM in logarithmischem Platz für zwei Knoten u, v bestimmen kann, ob u und v mit einer Kante verbunden sind, sowie dass eine Kodierung eines einzelnen Knotens nur logarithmischen Platz benötigt. Ebenso kann die NTM Knoten nichdeterministisch "raten", das bedeutet, dass gestartet in einem speziellen Zustand, der (Teil-)Berechnungbaum n verschiedene Pfade hat und am Ende jeden Pfades ist jeweils die Kodierung eines der n Knoten von G auf ein Band geschrieben worden.

Zusatzaufgabe *(0 Bonuspunkte)*: Machen Sie sich klar, wie eine mögliche Kodierung aussieht und welche technischen Details bei der Konstruktion der obigen Subroutinen zu bewältigen sind.

Lösung A10.4 (UCONN) Wir konstruieren eine NTM M, die UCONN entscheidet, wie folgt:

M rät nacheinander einen Pfad $s = v_1, v_2, \ldots, v_r = t$ (dazu müssen jeweils nur zwei aufeinanderfolgende Knoten betrachtet werden) und prüft nun jeweils, ob v_i und v_{i+1} in G verbunden sind.

Dazu benötigt M jeweils nur logarithmischen Platz, es gilt also $\mathsf{UCONN} \in \mathsf{NL}$.