Satz: Sei s raumkonstruierbar, dann gibt es eine Sprache $L \in DSPACE(s)$, die nicht mit Platzbedarf $o(s)$ entschieden werden kann.

Beweis (Diagonalisierung):

Für eine TM M sei $\langle M \rangle$ eine Kodierung von M in einem geeigneten Alphabet (z.B. als Binärzahl). $\langle M \rangle$ heißt Gödelisierung von M.

Sei nun $\hat{M}$ eine TM mit Eingabe w, $n = |w|$, die wie folgt arbeitet:

i) markiere $s(n)$ Bandzellen (s raumkonstruierbar!)

1 Die Gödelisierung von Turingmaschinen

Umwandlung muß effizient berechenbar sein

Aus technischen Gründen wollen wir Turingmaschinen in natürliche Zahlen umwandeln. Sei $M = (\Sigma, Z, \delta, z_0, z_1)$ eine 1-Band TM mit $\Sigma = \{a_0, \ldots, a_k\}$ und $Z = \{z_0, \ldots, z_l\}$, dann kann man die Übergangsfunktion δ wie folgt kodieren:

$$\delta(z_i, a_j) \ni (z_{i'}, a_{j'}, \sigma)$$

wird zu

$$\#\#\mathrm{bin}(i)\#\mathrm{bin}(j)\#\mathrm{bin}(i')\#\mathrm{bin}(j')\#\mathrm{bin}(m),$$

wobei

$$m = \begin{cases} 0, & \text{falls } \sigma = L \\ 1, & \text{falls } \sigma = R \\ 2, & \text{falls } \sigma = N. \end{cases}$$

Dabei ist $\mathrm{bin}(x)$ wieder die Binärkodierung von x. Die Befehle unserer Turingmaschine schreiben wir in einer festgelegten Reihenfolge auf und bekommen so ein Wort über $\{\#, 0, 1\}$. Nun kodieren wir

$$\begin{aligned} 0 &\mapsto 00, \\ 1 &\mapsto 01 \text{ und} \\ \# &\mapsto 11. \end{aligned}$$

Die so berechnete Zahl nennen wir *Gödelnummer*[1] der Maschine M. Die Konstruktion verdeutlicht, dass für jede beliebige Turingmaschine leicht ihre Gödelnummer berechenbar ist. Ebenfalls einsichtig ist, dass man auch aus einer Gödelnummer leicht wieder die ursprüngliche Turingmaschine gewinnen kann. Weiterhin kann man sich leicht überlegen, dass nicht jede natürliche Zahl in binärer Darstellung eine Turingmaschine beschreibt. Dieses Problem können wir aber relativ leicht reparieren. Sei $\hat{M}$ eine beliebige aber festgelegte Turingmaschine, dann sei

$$M_w =_{\mathrm{def}} \begin{cases} M, & \text{falls } w \text{ die Gödelnummer der Turingmaschine } M \text{ ist} \\ \hat{M}, & \text{sonst.} \end{cases}$$

Mit der *Gödelisierung* ($\triangleq$ Kodierung einer Turingmaschine durch eine Gödelnummer) haben wir erreicht, dass eine Turingmaschine die Eingabe einer anderen Turingmaschine M sein kann, d.h. die Turingmaschine M kann Berechnungen durchführen, die bestimmte Eigenschaften dieser Turingmaschine testet. Mit Hilfe dieser Idee definieren wir

Definition 1.1 (Spezielles Halteproblem)

(2)

spätere Simulation verläßt den markierten Bereich $\Rightarrow$ ablehnen

ii) falls w nicht die Form $\langle M \rangle 1 0^*$ für eine TM M hat, dann ablehnen

iii) simuliere M auf w. Benötigt M mehr als $2^{s(n)}$ Schritte, dann lehne ab

iv) falls M akz, dann ablehnen, sonst akz

Sei A die Sprache, die $\hat{M}$ entscheidet. Offensichtlich gilt $A \in DSPACE(s)$

Annahme: $A \in DSPACE(s')$, dann ex. eine TM M', die A entscheidet und Platz s' benötigt.

3

Wähle w von der Form $\langle M' \rangle 1 0^*$ lang genug, sodass M' auf w Platzbedarf $\leq s(|w|)$ und Zeitbedarf $\leq 2^{s(|w|)}$ hat. („Padding")

Aufgrund dieser Wahl erreicht $\hat{M}$ auf w den Schritt iv, ihrer Berechnung.

Also gilt:

$w \in A$ gdw M' akz w (Annahme)
gdw $\hat{M}$ akz w nicht (Konstr. von $\hat{M}$, Schritt iv)
gdw $w \notin A$ Widerspruch!

$\Rightarrow$ Annahme falsch! Also $A \notin DSPACE(s')$! #

<u>Folgerung:</u> Sei $s_1 = o(s_2)$, s_2 raumkonstruierbar, dann gilt

$$\mathrm{DSPACE}(s_1) \subsetneq \mathrm{DSPACE}(s_2)$$

<u>Satz:</u> Sei $t_1 = o(t_2 / \log t_2)$, t_2 „zeitkonstruierbar", dann gilt

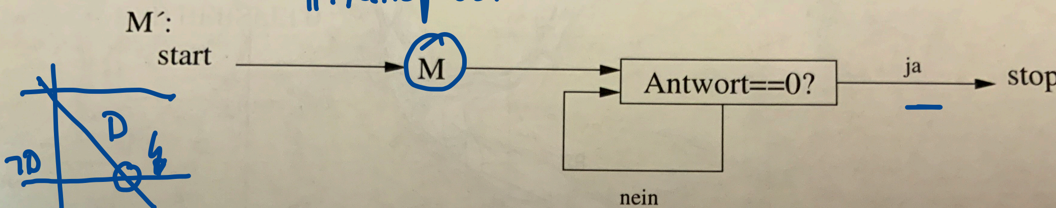
$$\mathrm{DTIME}(t_1) \subsetneq \mathrm{DTIME}(t_2)$$

<u>Beweis:</u> ähnliche Diagonalisierungsmethode

<u>Folgerung:</u> Es gilt $\mathrm{NL} \subsetneq \mathrm{PSPACE}$ und $\mathrm{P} \subsetneq \mathrm{EXP}$

Beweis: Angenommen H wäre entscheidbar, dann wäre die charakteristische Funktion c_H mit Hilfe der TM M berechenbar. Wir bauen nun die Turingmaschine M in die Maschine M' wie folgt um:

„Halteproblem"



Damit ergibt sich die folgende Situation: Die Maschine M' stoppt gdw. M die Antwort 0 ausgibt. Berechnet M die Antwort 1, dann gelangt M' in eine Endlosschleife und hält nicht an. Sei w' die Gödelnummer von M', dann gilt:

$$
\begin{array}{lll}
M' \text{ mit Eingabe } w' \text{ hält} & \text{gdw.} & M \text{ mit Eingabe } w' \text{ gibt } 0 \text{ aus} \\
& \text{gdw.} & c_H(w') = 0 \\
& \text{gdw.} & w' \notin H \\
& \text{gdw.} & M_{w'} \text{ hält bei Eingabe } w' \text{ nicht an} \\
& \text{gdw.} & M' \text{ mit Eingabe } w' \text{ hält nicht.}
\end{array}
$$

Dies ist ein Widerspruch zu der ursprünglichen Annahme, d.h. die Annahme war falsch und es kann keine Turingmaschine geben, die die charakteristische Funktion c_H berechnet. Damit ist das spezielle Halteproblem H nicht entscheidbar. #

4. Übungsblatt

Aufgabe 1

Beo: $\underbrace{n+n+n+\ldots+n}_{n\text{-mal}} = n \cdot n = n^2$

Idee: $2n$ kann man leicht durch Anhängen der Eingabe erreichen. Dies kann man leicht erreichen, indem man schon kopierte Zeichen d. Eingabe markiert

$$\square A N N A \square \vdash^{*} \square A N N A A \square$$

Arbeitsweise d. TM

i) markiere das erste Zeichen der Eingabe mit
einem c (≙ copied) und einem r (≙ repeated)
Also aus 1 wird 1^{c}_{r}

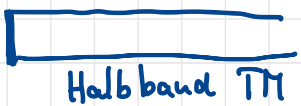
ii) kopiere alle Zeichen d. Eingabe und hänge diese
an. Noch zu kopierende Zeichen sind nicht mit
c markiert.

iii) Wandere von rechts nach links und markiere
das erste Zeichen der ursprünglichen Eingabe das
nicht mit r markiert ist.

iv) wenn noch nicht alle Zeichen der ursprünglichen Eingabe markiert sind wiederhole ab Schritt ii,

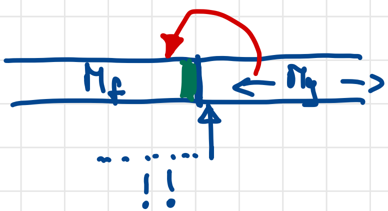
⇒ Platzbedarf: $n \cdot n = n^2$

Aufgabe 2:



i) $f+g$: Konstruiere M_{f+g} wie folgt

a) simuliere M_f und markiere die letzte (rechts!) benutzte Bandzelle gesondert



b) laufe zum ersten Blank rechts und simuliere M_g dahinter. Immer wenn M_g in eine von M_f beschriebene Zelle schreiben will, dann verschiebe alles von M_g erzeugte um eine Stelle nach rechts

ii) $f \cdot g$: Konstruiere $M_{f \cdot g}$ wie folgt

a) simuliere M_f und markiere die erste der $f(|x|)$ Zellen gesondert

b) schreibe beginnend beim ersten Zeichen „in" jede markierte Zelle das durch M_g erzeugte Wort / Ausgabe. Die i-te Version / Simulation von $g(|x|)$ überschreibt die i-te Zelle. Der benötigte Platz wird durch Verschieben nach links und rechts geschaffen.

c) stoppe, wenn das letzte Zeichen von M_f überschrieben wurde

iii) $f^g = \overbrace{f \cdot f \cdot \ldots \cdot f}^{g\text{-mal}}$ $|x| = n$ 5

Behandle den Sonderfall $f=1$ gesondert

a) markiere $g(|x|)$ Zellen blau

b) schreibe „in" die linkeste blaue Zelle $f(|x|)$ rote Zellen (evtl. Platz schaffen wie ii))

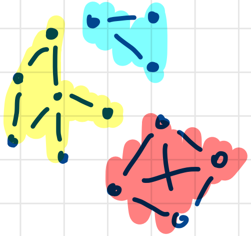
c) Solange noch blaue Zellen vorhanden sind

- verschiebe alle roten Zellen um 1 nach rechts und lösche so die linkeste blaue Zelle
- schreibe „in" jede rote Zelle $f(|x|)$ neue rote Zellen.

(v) n^k, analog zu (c) oder via Aufgabe 1 und $n^k = n \cdot n^{k-1}$ und der Raumkonstruierbarkeit von n.

Aufgabe 4: Folgende det. TM entscheidet CONNECTED

G



Eingabe: (G, k), $G = (V, E)$, $k \in \mathbb{N}$

i, prüfe ob G ein ungerichteter Graph mit Knoten V und Kante E

ii, $i = 0$; // Zähler für die Zusammenhangskomp.

iii, solange es unmarkierte Knoten gibt

iv, $i = i + 1$;

7

v, wähle einen unmarkierten Knoten und färbe ihn grün

vi, solange noch grüne Knoten v, unmarkierte Knoten u und $(v, u) \in \bar{E}$ markiere u grün

vii, falls $i \leq k$, dann akzeptieren, sonst ablehnen