Musterlösung Teilklausur

A1 a) • Eine Turingmaschine ist ein 5-Tupel $M = (Q, \Sigma, \delta, q_0, q_F)$

wobei: Q – endliche Zustandsmenge
Σ – Eingabealphabet
δ – endliche Liste von Zustandsübergängen

$(q,x) \to (q',x')$ mit $q, q' \in Q$

$x \in \overline{\Sigma}$
$x' \in \overline{\Sigma}$ oder $x' = R, L$

mit $\overline{\Sigma} = \Sigma \cup \{\#\}$
für jedes Paar (q,x) gibt es höchstens eine solche Regel, d.h.

δ ist partielle Funktion $Q \times \overline{\Sigma} \to Q \times (\overline{\Sigma} \cup \{R, L\})$

$q_0 \in Q$ – Startzustand; $q_F \in Q$ – Endzustand

b) • Konfigurationen sind Paare (q, w), $q \in Q$,
1 $w \in \overline{\Sigma}^* \times \overline{\Sigma} \times \overline{\Sigma}^*$ (Bandinhalt mit Kopfposition)

• Folgekonfigurationsrelation $\vdash$:

$(q, s_0 \dots \underline{s_i}\, s_{i+1} \dots s_n) \vdash (q, s_0 \dots s_i \underline{s_{i+1}} \dots s_n)$
falls $(q, s_i) \to (q', R)$ in δ

2 —"— —"— $\vdash (q', s_0 \dots s_{i-1}\, s_i \dots s_n)$
falls $(q, s_i) \to (q', L)$ in δ

— " — — " — $\vdash (q', s_0 \dots s_{i-1}\, y\, s_{i+1} \dots s_n)$
falls $(q, s_i) \to (q', y)$ in δ
mit $y \in \overline{\Sigma}$

• — " — Haltekonfiguration falls $\delta(q, s_i)$ nicht definiert

• Berechnung: Endliche oder unendliche Folge
2 von $s_0 \dots s_n$
$K_0 \vdash K_1 \vdash K_2 \vdash \dots$ von Konfigurationen mit

– $K_0 = (q_0, \underline{s_0} \dots s_n)$ Initialkonfiguration
– falls die Folge endlich ist, ist die letzte Konfiguration eine Haltekonfiguration

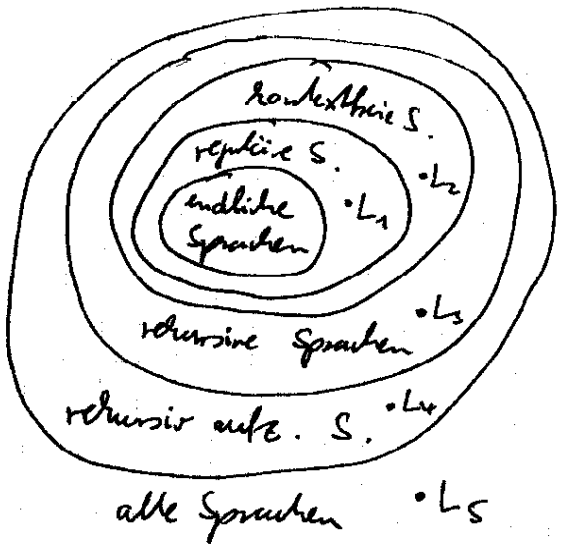
• akzeptierende Berechnung: obige Folge die endlich ist und die
2 Haltekonfiguration am Ende enthält q_F
• nicht akz. Berechnung: sonst

A'1

b) ~~Eine formale Sprache L heißt rekursiv aufzählbar, falls es eine TM M gibt, die L akzeptiert: $L = L(M)$.~~

~~Die Sprache L heißt rekursiv falls es eine TM M gibt die L akzeptiert und auf jede Eingabe hält.~~

c)



$L_1 = \{a^n \mid n \in \mathbb{N}\}$

$L_2 = \{a^n b^n \mid n \in \mathbb{N}\}$

$L_3 = \{a^n b^n c^n \mid n \in \mathbb{N}\}$

$L_4 = \{c(M)w \mid M \text{ TM, die auf Eingabe } w \text{ hält}\} = L_{\text{null}}$

$L_5 = \{c(M) \mid c(M) \notin L(M)\} = L_{\text{code}}$

Idee der Arbeitsweise:

- von links nach rechts laufen und für jedes gefundene „a" ein „c" suchen und mit Spezialsymbol $ überschreiben
- gleichzeitig analog nach „b" suchen und für jedes gefundene „b" dieses und ein zugehöriges „d" löschen
- etc.

Bemerke Modifikation von TM mit zusätzlichen Bandsymbolen:

$M = (Q, \Sigma, \Sigma \cup \{\$\}, \delta, q_0, q_F)$ mit

$Q = \{q_0, q_F, q_1, \ldots, q_8\}$

δ:

$(q_0, \#) \to (q_F, \#)$

$(q_0, a) \to (q_1, \$)$ „a" gefunden … „c" suchen (a löschen und)

$(q_0, b) \to (q_3, \#)$

$(q_0, c) \to (q_5, \#)$

$(q_0, d) \to (q_7, \#)$

c-Suchmodul:

$(q_1, x) \to (q_1, R)$ $x = \$, b, a, d$ nach rechts suchen

$(q_1, c) \to (q_2, \$)$ c gefunden; wieder zurück laufen

$(q_2, x) \to (q_2, L)$ $x = a, b, d, \$$

$(q_2, \#) \to (q_0, R)$

a, b, d-Suchmodul sind analog zu obigem!

$(q_0, \$) \to (q_0, R)$ schon überschriebene Zeichen überlesen

A3)

(4)

a) ☹: $M = \{ f: \Sigma^* \to \Sigma^* \}$ mit $\Sigma = \{1\}$ ist überabzählbar.

Ann.: M ist abzählbar, d.h. $M = \{f_0, f_1, f_2, \dots\}$

Definiere

$$g: \Sigma^* \to \Sigma^* \text{ durch}$$

$$g(1^n) = f_n(1^n)1 \quad .$$

Dann gilt für alle $n \in \mathbb{N}$: $g \neq f_n$.

Also ist g nicht in der Auflistung von M enthalten ↯

b) Jede TM, die eine Funktion $f: \Sigma^* \to \Sigma^*$ berechnet, kann durch ein Codewort über $\{0,1\}$ kodiert werden.

Da $\{0,1\}^*$ abzählbar ist, gibt es daher nur abzählbar viele solche TMen und daher höchstens abzählbar viele berechenbare Funktionen $f: \Sigma^* \to \Sigma^*$.

Da aber insgesamt überabzählbar viele Funktionen dieses Typs existieren können nicht alle davon auch berechenbar sein.

a) **Satz:** Jede nicht-triviale Eigenschaft von formalen Sprachen ist unentscheidbar. Genauer: für jede nicht-triviale Eigenschaft P von formalen Sprachen ist die Sprache

$$L_P = \{c(M) \mid L(M) \text{ hat Eigenschaft } P\}$$

nicht entscheidbar.

Eine Eigenschaft heißt trivial falls alle Sprachen oder keine Sprache die Eigenschaft hat.

b) Es gilt:

$$L = \{c(M) \mid L(M) = \Sigma^*\}$$

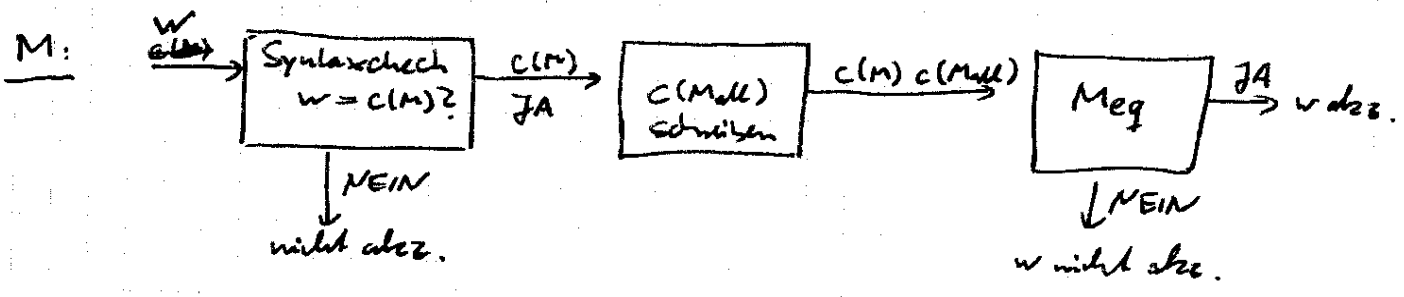
Die Eigenschaft gleich Σ^* zu sein ist nicht-trivial:
Σ^* hat diese Eigenschaft; $\varnothing$ hat sie nicht.

Nach dem Satz von Rice ist L daher nicht entscheidbar.

c) $\not\Leftarrow$: L_{eq} ist nicht entscheidbar

Ann.: L_{eq} ist entscheidbar.

Dann ex. die TM M_{eq} mit $L_{eq} = L(M_{eq})$, die auf jede Eingabe hält.
Wir benutzen M_{eq}, um eine TM M für die Sprache L aus b) zu erhalten:



Dabei ist M_{all} die folgende feste TM, die jedes Wort akzeptiert

$$M_{all} = (\{q_0\}, \Sigma, \varnothing, q_0, q_0);$$

es gilt $L(M_{all}) = \Sigma^*$.

Die obige Maschine M hält auf jede Eingabe, da alle ihre Bestandteile dies tun.

Im letzten Schritt wird getestet, ob $L(M) = L(M_{all}) = \Sigma^*$ gilt, also ob das eingegebene Codewort $c(M)$ zu L gehört. Die Maschine M entscheidet also L aus b). $\lightning$

Eingabe: zwei Matrizen $A=(a_{ij})$ und $(b_{ij})=B$

Ausgabe: Matrixprodukt $A \cdot B = C$

```
for i = 1,...,n do
    for j = 1,...,n do                | Einträge von A·B schrittweise berechnen
        c_ij := 0;
        for k = 1,...,n do            | Skalarprodukt berechnen
            c_ij := c_ij + a_ik · b_kj
        od
    od
od
```

Laufzeit: Zuweisungen und arithmetische Operationen dauern einen Zeitschritt. Dann gilt

$$T(n) = n \cdot n \cdot (1 + 2 \cdot n) = n^2 + 2n^3 \in O(n^3).$$