Prof. J. Esparza
S. Sickert, J. Krämer

Technische Universität München
KEINE ABGABE

Einführung in die theoretische Informatik

Sommersemester 2017 – Übungsblatt Lösungsskizze 10

Übungsblatt

Wir unterscheiden zwischen Übungs- und Abgabeblättern. Auf diesem *Übungsblatt* finden Sie eine Übersicht über die Kernaspekte, die Sie in Kalenderwoche 27 in den Tutorien diskutieren, üben und vertiefen. Die Aufgaben auf diesem Blatt dienen dem Üben und Verstehen des Vorlesungsstoffes, sowie dem *eigenständigen Erarbeiten* der Kernaspekte. Außerdem sollen Ihnen diese Aufgaben auch helfen, ein Gefühl dafür zu bekommen, was Sie inhaltlich in der Klausur erwartet. Klausuraufgaben können jedoch deutlich von den hier gestellten Aufgaben abweichen. Abschreiben und Auswendiglernen von Lösungen wird Ihnen daher keinen dauerhaften Erfolg in der Vorlesung bringen. Fragen zu den Übungsblättern können Sie montags bis donnerstags von 12 Uhr bis 14 Uhr in der *THEO-Sprechstunde* in Raum 03.11.034 stellen.

Kernaspekte

K10.1 korrektes Wiedergeben der folgenden Definitionen und Algorithmen

- intuitiv berechenbar
- totale/partielle/echt partielle Funktion
- nicht-deterministische/deterministische Turing-Maschine
- abzählbar
- überabzählbar
- Konfiguration einer Turing-Maschine
- akzeptierte Sprache einer Turing-Maschine
- Turing-berechenbar

K10.2 für zwei gegebene Turing-Maschinen M_1 und M_2 die sequentielle Komposition angeben
K10.3 für drei gegebene Turing-Maschinen M, M_1 und M_2 eine Fallunterscheidung nach den Endzuständen von M angeben
K10.4 für eine gegebene Turing-Maschine M eine Schleife angeben, die solange M ausführt, solange nicht 0 auf dem Band steht
K10.5 zu einer Sprache L eine Turing-Maschine M angeben, so dass $\mathsf{L}(\mathsf{M}) = \mathsf{L}$
K10.6 begründet entscheiden, ob gegebene Beispiele neu eingeführte Definitionen erfüllen
K10.7 Aussagen mit neu eingeführten Definitionen beweisen oder widerlegen

Definition (Alternative Akzeptanzbedingungen für Turing-Maschinen)

In der Vorlesung wurde die Annahme gemacht, dass die Übergangsfunktion δ einer Turingmaschine folgende Eigenschaft erfüllt:

$$\delta(q, \mathsf{a}) \textit{ ist nicht definiert für alle } q \in \mathsf{F},\ \mathsf{a} \in \Gamma.$$

Sei $\mathcal{M}_A$ die Menge der Turingmaschinen, die diese Annahme erfüllen, und sei $\mathcal{M}$ die Menge aller Turingmaschinen. Es gilt somit $\mathcal{M}_A \subsetneq \mathcal{M}$.
Für $\mathsf{M} \in \mathcal{M}$ mit $\mathsf{M} = (\mathsf{Q}, \Sigma, \Gamma, \delta, \mathsf{q}_0, \square, \mathsf{F})$ definiere:

- $\mathsf{L}_F(\mathsf{M}) = \{w \in \Sigma^* \mid \exists \alpha, \beta \in \Gamma^*, f \in \mathsf{F}.\ (\varepsilon, \mathsf{q}_0, w) \rightarrow^*_{\mathsf{M}} (\alpha, f, \beta)\}$.
  (Menge der Wörter, für die die Maschine einen Endzustand irgendwann besucht.)
- $\mathsf{L}_H(\mathsf{M}) = \{w \in \Sigma^* \mid \exists \alpha, \beta \in \Gamma^*, q \in \mathsf{Q}.\ (\varepsilon, \mathsf{q}_0, w) \rightarrow^*_M (\alpha, q, \beta)$ und $\delta(q, \textit{first}(\beta))$ ist nicht definiert$\}$.
  (Menge der Wörter, für die die Maschine hält.)
- $\mathsf{L}_{HF}(\mathsf{M}) = \{w \in \Sigma^* \mid \exists \alpha, \beta \in \Gamma^*, f \in \mathsf{F}.\ (\varepsilon, \mathsf{q}_0, w) \rightarrow^*_M (\alpha, f, \beta)$ und $\delta(f, \textit{first}(\beta))$ ist nicht definiert$\}$.
  (Menge der Wörter, für die die Maschine in einem Endzustand hält.)

AUFGABE 10.1.

Stufe B

Begründen Sie folgende Aussagen, indem Sie eine passende Konstruktion angeben.

(a) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}_A$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}$ mit $\mathsf{L}_{\mathsf{F}}(\mathsf{M}) = \mathsf{L}_H(\mathsf{M}')$.
(b) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}_A$ mit $\mathsf{L}_H(\mathsf{M}) = \mathsf{L}_F(\mathsf{M}')$.
(c) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}_A$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}$ mit $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_F(\mathsf{M}')$.
(d) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}_A$ mit $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_F(\mathsf{M}')$.
(e) Für jede Turing-Maschine $\mathsf{M} \in \mathcal{M}$ gibt es eine Turing-Maschine $\mathsf{M}' \in \mathcal{M}$ mit $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_H(\mathsf{M}')$.

Lösungsskizze

(a) Füge einen neuen Fangzustand q_t ein und mache die Transitionsfunktion total für Nichtendzustände. Formal:

$$\mathsf{Q}' = \mathsf{Q} \cup \{q_t\} \qquad \delta' = \delta \cup \{(q_t, a, q_t, a, N) \mid a \in \Gamma\} \cup \{(q, a, q_t, a, N) \mid a \in \Gamma \wedge q \in \mathsf{Q} \setminus \mathsf{F} \wedge \delta(q, a) \text{ ist undefiniert}\}$$

(b) Füge einen neuen Endzustand q_f ein und mache die Transitionsfunktion total (außer für q_f). Formal:

$$\mathsf{Q}' = \mathsf{Q} \cup \{q_f\} \qquad \mathsf{F}' = \{q_f\} \qquad \delta' = \delta \cup \{(q, a, q_f, a, N) \mid a \in \Gamma \wedge q \in \mathsf{Q} \wedge \delta(q, a) \text{ ist undefiniert}\}$$

(c) Trivial, da $\mathcal{M}_A \subsetneq \mathcal{M}$.

(d) Lösche alle ausgehenden Kanten von Endzuständen. Formal:

$$\delta' = \delta \setminus \mathsf{F} \times \Gamma \times \mathsf{Q} \times \Gamma \times \{L, R, N\}$$

(e) Lösche alle ausgehenden Kanten von Endzuständen und mache die Transitionsfunktion total für Nichtendzustände. Formal:

$$\mathsf{Q}' = \mathsf{Q} \cup \{q_t\}$$
$$\delta' = (\delta \cup \{(q_t, a, q_t, a, N) \mid a \in \Gamma\} \cup \{(q, a, q_t, a, N) \mid a \in \Gamma \wedge q \in \mathsf{Q} \setminus \mathsf{F} \wedge \delta(q, a) \text{ ist undefiniert}\}) \setminus \mathsf{F} \times \Gamma \times \mathsf{Q} \times \Gamma \times \{L, R, N\}$$

AUFGABE 10.2.

Stufe B

Entscheiden Sie, ob die folgenden Aussagen korrekt sind und begründen Sie Ihre Antwort kurz.

(a) Es gibt eine Turingmaschine, die den Kopf nie weiter als vier Schritte von der Startposition weg bewegt und eine unendliche Sprache akzeptiert.

(b) Sei $\mathsf{M} \in \mathcal{M}_A$ eine Turingmaschine. Dann existiert eine TM $\mathsf{M}' \in \mathcal{M}$, so dass $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_F(\mathsf{M}')$ und M' hat nur einen Zustand.

(c) Sei $\mathsf{M} \in \mathcal{M}_A$ eine Turingmaschine, die ihren Kopf immer nur nach links bewegt. Dann gilt:

$$\mathsf{L}_F(\mathsf{M}) \in \{A\Sigma^* \mid A \subseteq \Sigma\}$$

Lösungsskizze

(a) Ja. Sei $\mathsf{q}_0 \in \mathsf{F}$, dann erkennt die TM Σ^*.

(b) Nein. Entweder $\mathsf{q}_0 \in \mathsf{F}$, dann erkennt die TM Σ^*, oder $\mathsf{q}_0 \notin \mathsf{F}$, dann erkennt die TM $\emptyset$. Somit können keine weiteren Sprachen erkannt werden.

(c) Nein. Es kann zwar nur das erste Zeichen gelesen werden, aber auch dessen Abwesenheit kann erkannt werden. Somit haben wir:

$$\mathsf{L}_F(\mathsf{M}) \in \{A\Sigma^* \cup B \mid A \subseteq \Sigma, B \subseteq \{\varepsilon\}\}$$

AUFGABE 10.3.

Stufe C

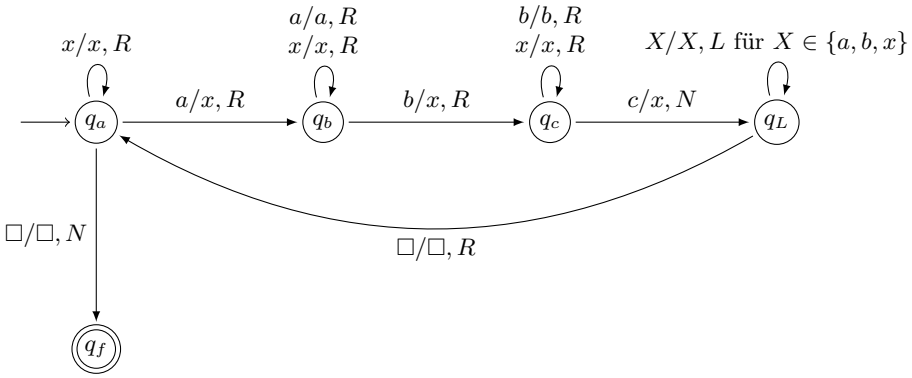
Geben Sie für die beiden angegebenen Sprachen die jeweils passende TM M_i an.

$$\text{(a) } \mathsf{L}_F(\mathsf{M}_1) = \{\mathsf{a}^n\mathsf{b}^n\mathsf{c}^n \mid n \in \mathbb{N}_0\} \qquad \text{(b) } \mathsf{L}_F(\mathsf{M}_2) = \{\mathsf{a}^n\mathsf{b}^{n^2} \mid n \in \mathbb{N}_0\}$$

Lösungsskizze

Wir schreiben $\square$ für eine leere Bandzelle.

(a) Sei TM $M_1 = (\{q_a, q_b, q_c, q_L, q_F\}, \{a, b, c\}, \Sigma \cup \{x, \square\}, \delta, q_a, \square, \{q_F\})$.



(b) Idee: Benutze den a-Bereich auch als Zähler.
(Alternativ: Schreibe b^{n^2} zu $b^{(n-1)^2}$ um, in dem man ein a und $2n-1$ b löscht.)

(i) Terminiere sofort bei leerer Eingabe, sonst gehe zu (c):

$$\mathsf{q}_0 \xrightarrow{\square/\square,N} f \qquad \mathsf{q}_0 \xrightarrow{a/a,N} q$$

(ii) Laufe einmal bis ganz nach links und lösche alle Markierungen auf dem Weg dorthin

$$p \xrightarrow{\substack{a'/a,L \\ d'/d,L}} p \qquad p \xrightarrow{\square/\square,R} q$$

(iii) Laufe zum ersten a von links und lösche es:

$$q \xrightarrow{d/d,R} q \qquad q \xrightarrow{a/d,L} r$$

(iv) Laufe wieder zum linkesten Symbol:

$$r \xrightarrow{d/d,L} r \qquad r \xrightarrow{\square/\square,R} s$$

(v) Suche erstes a bzw. d von links und markiere es:

$$s \xrightarrow{\substack{d'/d',R \\ a'/a',R}} s \qquad s \xrightarrow{\substack{d/d',R \\ a/a',L}} t$$

(vi) Suche rechtestes b:

$$t \xrightarrow{\substack{a/a,R \\ b/b,R}} t \qquad t \xrightarrow{\square/\square,L} u$$

(vii) Lösche es:

$$u \xrightarrow{b/\square,R} v$$

(viii) Suche rechtestes a' bzw. d':

$$v \xrightarrow{\substack{b/b,L \\ a/a,L \\ d/d,L}} v \qquad v \xrightarrow{\substack{a'/a',R \\ d'/d',R}} w$$

Falls noch unmarkietes a oder d übrig, markiere wechsle in Zustand t (goto (f)), falls noch ein b übrig ist, wechsle in Zustand p (goto (b))

$$w \xrightarrow{\substack{a/a',L \\ d/d',L}} t \qquad w \xrightarrow{b/b,R} p \qquad w \xrightarrow{\square/\square} x$$

(ix) ansonsten akzeptiere, falls Eingabe nur noch aus markierten d' besteht:

$$x \xrightarrow{d'/d',N} f$$

AUFGABE 10.4. Stufe C

Wir konstruieren eine TM, die La-Ola-Wellen simuliert. Geben Sie hierzu eine deterministische TM an, welche als Eingabe ein Wort $w \in \{u, m, o\}^*$ mit $|w| \geq 3$ erwartet, wobei $w = a_0 \dots a_{l-1}$ den aktuellen Zustand einer La-Ola-Welle beschreibt, wobei sich die Welle bei a_{l-1} wieder bei a_0 fortsetzen sollen ($\mathrm{mod} l$). Die Buchstaben von w beschreiben die aktuelle Armhaltung (**u**nten, **m**ittig, **o**ben) des Zuschauers auf Platz i.
Die DTM soll zuerst prüfen, dass w eine zulässige La-Ola-Welle ist, d.h.

$$\forall i \in \mathbb{Z}_l. \quad \begin{array}{ll} & (a_i = o \to a_{i-1 \bmod l} a_i a_{i+1 \bmod l} \in \mathsf{L}((o|m)o(o|m))) \\ \wedge & (a_i = u \to a_{i-1 \bmod l} a_i a_{i+1 \bmod l} \in \mathsf{L}((u|m)u(u|m))) \\ \wedge & (a_i = m \to a_{i-1 \bmod l} a_i a_{i+1 \bmod l} \in \mathsf{L}(umo|omu)) \end{array}$$

und anschließend die Welle um eine Position nach links verschieben, d.h. falls w zulässig ist, soll die DTM mit der Ausgabe $\dots \square a_1 \dots a_{l-1} a_0 \square \dots$ terminieren. Falls die Eingabe jedoch nicht zulässig ist, soll die DTM mit leerem Band terminieren.

Lösungsskizze
Zur Vereinfachung prüft man zunächst nur, dass die Eingabe die korrekte Struktur hat, hierzu speichert man in der Kontrolle z.B. a_0, a_{i-2}, a_{i-1}, um entscheiden zu können, ob a_i korrekt ist und ob a_{i-1} korrekt ist. Sei $L_A = \mathsf{L}((o|m)o(o|m)) \cup \mathsf{L}((u|m)u(u|m)) \cup \mathsf{L}(umo|omu)$.
Speicher initialisieren, wobei (x, x', y, z) in x das Zeichen a_0, in x' das Zeichen a_1, in y das Zeichen a_{i-2} und in z das Zeichen a_{i-1} speichert:

$$(\varepsilon,\varepsilon,\varepsilon,\varepsilon) \xrightarrow[x\in\{u,o,m\}]{x/x,R} (x,\varepsilon,x,\varepsilon) \qquad (x,\varepsilon,x,\varepsilon) \xrightarrow[x'\in\{u,o,m\}]{x'/x',R} (x,x',x,x') \qquad (x,\varepsilon,x,\varepsilon) \xrightarrow{\square/\square,L} \mathsf{del}$$

Auf Zulässigkeit testen an Positionen $i = 1, \ldots, l-2$:

$$(x,x',y,z) \xrightarrow[(y,z,a)\in L_A]{a/a,R} (x,x',z,a) \qquad (x,x',y,z) \xrightarrow[(y,z,a)\notin L_A]{a/a,R} \mathsf{mvr}$$

Auf Zulässigkeit testen an Positionen $i = 0$ und $i = l-1$:

$$(x,x',y,z) \xrightarrow[(y,z,x)\in L_A \wedge (z,x,x')\in L_A]{\square/\square,L} (x) \qquad (x,x',y,z) \xrightarrow[(z,a,x)\notin L_A \vee (z,x,z')\notin L_A]{\square/\square,L} \mathsf{del}$$

Eingabe um eine Positionen nach Links shiften:

$$(x) \xrightarrow[x,y\in\{u,m,o\}]{y/x,L} (y) \qquad (x) \xrightarrow[x,y\in\{u,m,o\}]{\square/\square,R} \mathsf{fin}$$

Rechtes Ende suchen, um Eingabe zu löschen:

$$\mathsf{mvr} \xrightarrow[x\in\{u,o,m\}]{x/x,R} \mathsf{mvr} \qquad \mathsf{mvr} \xrightarrow{\square/\square,L} \mathsf{del}$$

Eingabe löschen:

$$\mathsf{del} \xrightarrow[x\in\{u,m,o\}]{x/\square,L} \mathsf{del} \qquad \mathsf{del} \xrightarrow{\square/\square,N} \mathsf{del}$$

AUFGABE 10.5. (*k-PDA*) Stufe D

Im Folgenden sagen wir, dass ein k-PDA ein PDA ist, der k Stacks zur Verfügung hat. In jedem Schritt kann der PDA in Abhängigkeit vom aktuellen Zustand, dem gelesenen Eingabezeichen und den Symbolen, die zuoberst auf jedem der k Stacks liegen, in einen neuen Zustand wechseln und jeden der k Stacks wie im Fall eines gewöhnlichen PDAs modifizieren.

(a) Geben Sie eine formale Definition für k-PDAs an.
(b) Geben Sie eine Sprache an, die von einem 2-PDA, aber von keinem 1-PDA akzeptiert wird.
(c) Geben Sie eine allgemeine Übersetzung von einem PDA A in eine Turingmaschine M an, so dass $\mathsf{L}_\varepsilon(A) = \mathsf{L}_F(\mathsf{M})$. Verwenden Sie bei Bedarf eine TM mit mehreren Bändern.
(d) Beschreiben Sie wie Sie das Verfahren aus (c) erweitern können, so dass ein k-PDA A in eine Turingmaschine M übersetzt werden kann.
(e) Geben Sie eine allgemeine Übersetzung von einer Turingmaschine M in einen 2-PDA A an, so dass $\mathsf{L}_F(\mathsf{M}) = \mathsf{L}_F(A)$.
(f) Zeigen Sie unter Verwendung der vorherigen Ergebnisse, dass jeder k-PDA A ($k \geq 3$) von einem 2-PDA A' simuliert werden kann, d.h. $\mathsf{L}_\varepsilon(A) = \mathsf{L}_\varepsilon(A')$. Somit können beliebig viele Stacks immer durch genau zwei Stacks simuliert werden.
(g) Skizzieren Sie eine direkte Übersetzung von einem k-PDA zu einem 2-PDA an.

Lösungsskizze

(a) $M = (\mathsf{Q}, \Sigma, \Gamma, \delta, \mathsf{q}_0, Z_0^k, \mathsf{F})$ mit $\delta : \mathsf{Q} \times (\Sigma \cup \{\varepsilon\}) \times \Gamma^k \to \mathcal{P}_e(\mathsf{Q} \times (\Gamma^*)^k)$.

(b) Ein 2-PDA kann die Sprache $\mathsf{L} = \{\mathsf{a}^n\mathsf{b}^n\mathsf{c}^n \mid n \in \mathbb{N}_0\}$ akzeptieren, indem er zuerst alle as auf den ersten, dann alle bs auf den zweiten Stack schiebt, dann für jedes c jeweils ein Zeichen von beiden Stacks popt und schließlich nur dann akzeptiert, falls nach Lesen der Eingabe beide Stacks leer sind. Wie aber in der Vorlesung gezeigt worden ist, ist L nicht kontextfrei und somit existiert kein PDA, der L akzeptiert.

(c) Sei $P = (\mathsf{Q}_P, \Sigma, \Gamma, \delta_P, \mathsf{q}_0, \bot)$ ein PDA. Wir konstruieren nun ein TM $\mathsf{M} = (\mathsf{Q}_\mathsf{M}, \Sigma, \Sigma \cup \Gamma \cup \{\square\}, \delta_\mathsf{M}, \mathsf{q}_0, \square, \{f\})$ mit $\mathsf{Q}_\mathsf{M} = \mathsf{Q}_P \cup \mathsf{Q}_P \times \Gamma_P \cup \{f\}$, so dass M gerade $L_\varepsilon(P)$ akzeptiert. Wir nehmen an, dass der PDA mit leerem Keller akzeptiert und ein explizites Bottom-Symbol $\bot$ verwendet. Die TM simuliert den Stack des PDA auf einem zweiten Band, wobei das zweite Band oBdA. zu Beginn mit $\bot$ initialisiert ist. Das rechteste Symbol des zweiten Bandes entspricht dann dem obersten Symbol auf dem Stack des simulierten PDA. Damit:

$$\begin{aligned}\delta_{\mathsf{M}}(p,a,X) &= \{(q,a,Y,R,N) \mid pX \xrightarrow{a} qY \in \delta_P\}\\ &\cup \{(q,a,\square,R,L) \mid pX \xrightarrow{a} q\varepsilon \in \delta_P\}\\ &\cup \{((q,Y),a,Z,R,R) \mid pX \xrightarrow{a} qYZ \in \delta_P\}\\ \delta_{\mathsf{M}}(p,a,X) &= \{(q,a,Y,N,N) \mid pX \xrightarrow{\varepsilon} qY \in \delta_P\}\\ &\cup \{(q,a,\square,N,L) \mid pX \xrightarrow{\varepsilon} q \in \delta_P\}\\ &\cup \{((q,Y),a,Z,N,R) \mid pX \xrightarrow{\varepsilon} qYZ \in \delta_P\}\end{aligned}$$

zzgl.

$$\begin{aligned}\delta_{\mathsf{M}}((q,Y),a,\square) &= \{(q,a,Y,N,N)\} && \text{für alle } a \in \Sigma \cup \{\square\}, q \in \mathsf{Q}_P, Y \in \Gamma_P\\ \delta_{\mathsf{M}}(q,\square,\square) &= \{(f,\square,\square,N,N)\} && \text{für alle } q \in \mathsf{Q}_P\end{aligned}$$

(d) Wir verwenden anstatt einer 2-Band eine $(k+1)$-Band TM für einen k-PDA.

(e) Idee: Jeder 2-PDA kann eine TM mit einem Band simulieren, indem der erste Stack alle Symbole links vom Lesekopf, der zweite Stack alle Symbole rechts vom Lesekopf speichert. Sei $\mathsf{M} = (\mathsf{Q}, \Sigma, \Gamma, \delta, \mathsf{q}_0, \square, \mathsf{F})$ die zu simulierende TM. Dann konstruieren wir A folgend:

$$A = (\mathsf{Q} \cup \{q_{\mathsf{read}}, q_{\mathsf{rev}}\}, \Sigma, \Gamma \cup \{\bot\}, \delta, q_{\mathsf{read}}, (\bot,\bot), \mathsf{F})$$

Zuerst kopieren wir die Eingabe auf das Band:

$$\begin{aligned}q_{\mathsf{read}}(X,\bot) &\xrightarrow{\varepsilon} q_{\mathsf{rev}}(X,\bot) & q_{\mathsf{read}}(X,\bot) &\xrightarrow{a} q_{\mathsf{read}}(aX,\bot) && \text{für alle } a \in \Sigma, X \in \Sigma \cup \{\bot\}\\ q_{\mathsf{rev}}(\bot,Y) &\xrightarrow{\varepsilon} \mathsf{q}_0(\bot,Y) & q_{\mathsf{rev}}(X,Y) &\xrightarrow{\varepsilon} q_{\mathsf{rev}}(\varepsilon,XY) && \text{für alle } X \in \Sigma\end{aligned}$$

Wir simulieren wie folgt die TM:

$$\begin{aligned}q(X,Y) &\xrightarrow{\varepsilon} p(\varepsilon,XZ) && \text{für alle } X \in \Gamma, (q,Y,p,Z,L) \in \delta\\ q(X,Y) &\xrightarrow{\varepsilon} p(X,Z) && \text{für alle } X \in \Gamma, (q,Y,p,Z,N) \in \delta\\ q(X,Y) &\xrightarrow{\varepsilon} p(ZX,\varepsilon) && \text{für alle } X \in \Gamma, (q,Y,p,Z,R) \in \delta\end{aligned}$$

Weiterhin kann bei Bedarf auf $\bot$ immer ein $\square$ erzeugt werden.

$$q(\bot,X) \xrightarrow{\varepsilon} q(\square\bot,X) \qquad q(X,\bot) \xrightarrow{\varepsilon} q(X,\square\bot) \qquad \text{für alle } q \in \mathsf{Q}, X \in \Gamma \cup \{\bot\}$$

(f) Eine $(k+1)$-Band-TM kann jeden k-PDA simulieren. Jede $k+1$-Band-TM kann von einer 1-Band-TM simuliert werden. Damit kann jeder 2-PDA jeden k-PDA simulieren und ist so mächtig wie jede TM.

(g) Wir verwenden k-Trennzeichen um die k-Stacks zu unterscheiden. Alle Stacks werden sequentiell mit Trennzeichen auf einem Stack gespeichert. In dem Kontrollzustand speichern wir dann die obersten Elemente von jedem Stack. Das erreichen wir in, dem wir den gesamten Stack durchsuchen und auf dem zweiten Stack speichern. Wir führen dann die Transition aus und speichern den Effekt. Wir kopieren dann nochmal alle Stackinhalte und machen das Update.