Formale Grundlagen der Informatik

7

Turing-Maschinen

Motivation

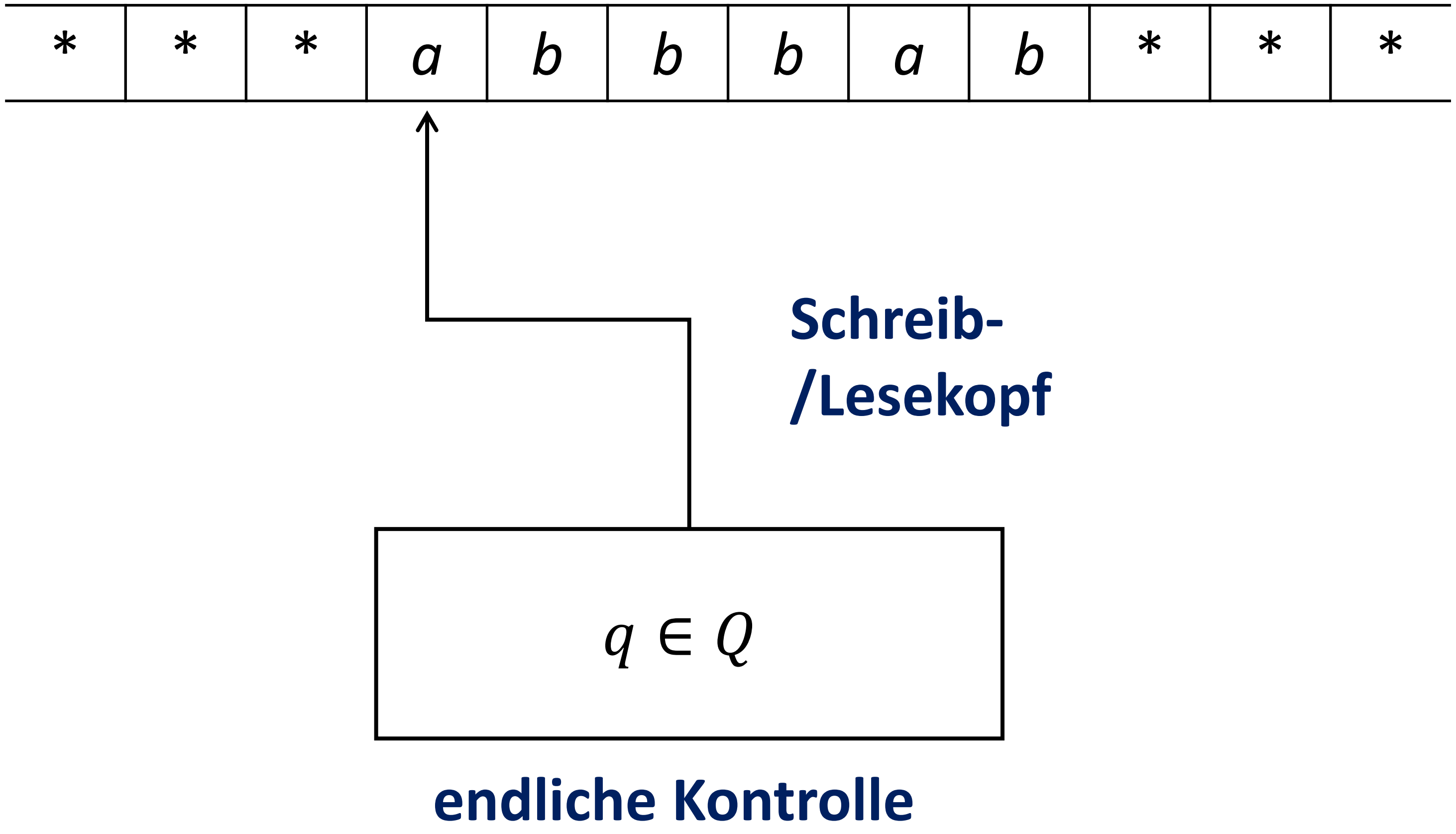
- Alle Varianten endlicher Automaten charakterisieren die regulären Sprachen.
  - DEAs, NEAs, ε-NEAs
  - zweiseitige DEAs
- *Idee:* Zusätzliche Erweiterung von endlichen Automaten:

  Schreib-/Lesekopf (statt nur Lesekopf)

➢ Automaten können „Notizen“ machen
(und diese wieder lesen, da zweiseitig)

➢ Erweiterung des Zustandsraums durch unbegrenztes Schreib-/Leseband
(unbegrenzt viel Information merken)

Turing-Maschine – Idee



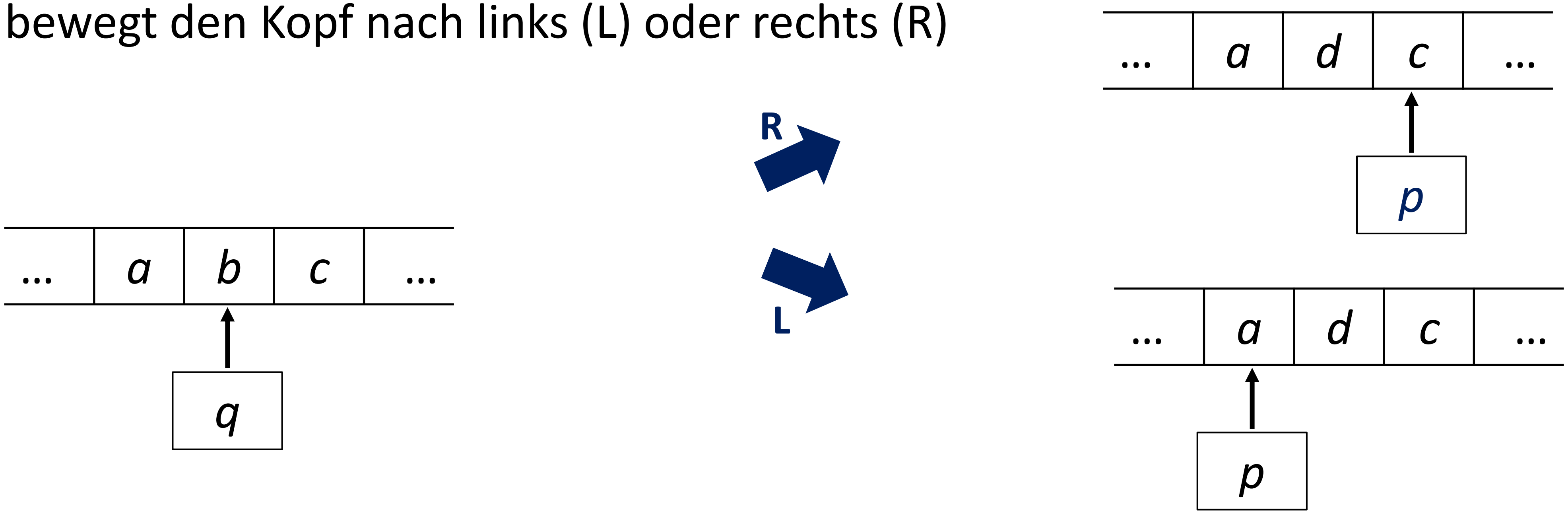
* für leere Zelle

Band beidseitig unbegrenzt
→ unbegrenzt viele Notizen

am Anfang: Kopf liest erstes Eingabesymbol im Startzustand

Turing-Maschine – Arbeitsweise

- liest in einem ihrer (endlich vielen) Zustände ein Bandsymbol
- wechselt den Zustand
- ersetzt das gelesene Bandsymbol
- bewegt den Kopf nach links (L) oder rechts (R)



Turing-Maschine – Komponenten

- endliche Menge von Zuständen
- Eingabealphabet
- Bandalphabet (*muss mindestens alle Eingabebuchstaben enthalten*)
- Startzustand
- Symbol für die leere Zelle
- Menge von Endzuständen (*in denen die TM ihre Arbeit beendet*)

Turing-Maschine – Definition

Eine **(deterministische) Turing-Maschine (D)TM** ist ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, *, F)$, wobei folgendes gilt:

- Q ist eine endliche Menge von Zuständen;
- Σ ist ein Alphabet der Eingabesymbole;
- Γ ist ein Alphabet der Bandsymbole mit $\Sigma \subseteq \Gamma$;
- $\delta: (Q \setminus F) \times \Gamma \to Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}$;
- $* \in \Gamma \setminus \Sigma$ ist das Symbol für die leere Zelle;
- $q_0 \in Q$ ist der Startzustand;
- $F \subseteq Q$ ist die Menge der Endzustände.

Konfigurationen einer TM

- **Konfiguration:** Wort aus $\Gamma^* Q \Gamma^*$ (o.B.d.A.: $Q \cap \Sigma = \emptyset$),
- *Interpretation* von Konfiguration uqv:
  - q ist der aktuelle Zustand
  - uv ist der aktuelle Bandinhalt;
    alle Symbole vor und hinter uv sind das Blank-Symbol *
  - Schreib-/Lesekopf befindet sich über dem ersten Buchstaben von v.

➢ Es gibt mehrere **äquivalente** Konfigurationen:

- Wenn uqv eine Konfiguration ist, dann beschreibt jede Konfiguration $*^i\, uqv\, *^j$ $(i \geq 0, j \geq 0)$ dieselbe Situation der TM.
- Wählen immer die Konfiguration, die im jeweiligen Kontext am besten geeignet ist.

Berechnungen einer TM

- **Konfigurationsübergang** von M:

1. $a_1 \ldots a_{i-1} q a_i a_{i+1} \ldots a_n \vdash a_1 \ldots a_{i-1} b p a_{i+1} \ldots a_n$ gdw. $\delta(q, a_i) = (p, b, \mathrm{R})$
2. $a_1 \ldots a_{i-1} q a_i a_{i+1} \ldots a_n \vdash a_1 \ldots a_{i-2} p a_{i-1} b a_{i+1} \ldots a_n$ gdw. $\delta(q, a_i) = (p, b, \mathrm{L})$

- Sei $\overset{*}{\vdash}$ die reflexive und transitive Hülle von $\vdash$, *d.h.*
  $K \overset{*}{\vdash} K$ und $K_1 \overset{*}{\vdash} K_k$ falls $K_1 \vdash K_2 \vdash \cdots \vdash K_k$
  für Konfigurationen $K, K_1, K_2, \ldots, K_k$.

- $L(M) = \{\, w \in \Sigma^* \mid q_0 w \overset{*}{\vdash} upv$ für gewisse $u, v \in \Gamma^*$ und ein $p \in F \,\}$

Anfangskonfiguration

Endkonfiguration

TM – Beispiel

$$M = (\{q_0, q_1, q_2, q_3, q_4, f\}, \{a, b\}, \{a, b, *\}, \delta, q_0, *, \{f\})$$

	a	b	$*$
q_0	$(q_1, *, \mathrm{R})$	(q_4, b, R)	$(f, *, \mathrm{R})$
q_1	(q_1, a, R)	(q_1, b, R)	$(q_2, *, \mathrm{L})$
q_2	(q_4, a, R)	$(q_3, *, \mathrm{L})$	$(q_4, *, \mathrm{R})$
q_3	(q_3, a, L)	(q_3, b, L)	$(q_0, *, \mathrm{R})$
q_4	(q_4, a, R)	(q_4, b, R)	$(q_4, *, \mathrm{R})$

$$\begin{aligned} q_0 ab &\vdash\ * q_1 b \\ &\vdash\ * b q_1 * \\ &\vdash\ * q_2 b * \\ &\vdash\ q_3 *** \\ &\vdash\ * q_0 ** \\ &\vdash\ ** f * \end{aligned}$$

M akzeptiert ab

$$\begin{aligned} q_0 abb &\vdash\ * q_1 bb \\ &\vdash\ * b q_1 b \\ &\vdash\ * bb q_1 * \\ &\vdash\ * b q_2 b * \\ &\vdash\ * q_3 b ** \\ &\vdash\ q_3 * b ** \\ &\vdash\ * q_0 b * \\ &\vdash\ * b q_4 ** \\ &\vdash\ * b * q_4 * \end{aligned}$$

M wird nie halten

TM – Beispiel

$$M = (\{q_0, q_1, q_2, q_3, q_4, f\}, \{a, b\}, \{a, b, *\}, \delta, q_0, *, \{f\})$$

	a	b	$*$
q_0	$(q_1, *, \mathrm{R})$	(q_4, b, R)	$(f, *, \mathrm{R})$
q_1	(q_1, a, R)	(q_1, b, R)	$(q_2, *, \mathrm{L})$
q_2	(q_4, a, R)	$(q_3, *, \mathrm{L})$	$(q_4, *, \mathrm{R})$
q_3	(q_3, a, L)	(q_3, b, L)	$(q_0, *, \mathrm{R})$
q_4	(q_4, a, R)	(q_4, b, R)	$(q_4, *, \mathrm{R})$

⟵ *ist offenbar Fehlerzustand …*

*… **Konvention:** Wir dürfen derartige Fehlerzustände weglassen und alle Tabelleneinträge, die zu ihnen führen, leer lassen …*

TM – Beispiel

$$M = (\{q_0, q_1, q_2, q_3, q_4, f\}, \{a, b\}, \{a, b, *\}, \delta, q_0, *, \{f\})$$

	a	b	$*$
q_0	$(q_1, *, \mathrm{R})$		$(f, *, \mathrm{R})$
q_1	(q_1, a, R)	(q_1, b, R)	$(q_2, *, \mathrm{L})$
q_2		$(q_3, *, \mathrm{L})$	
q_3	(q_3, a, L)	(q_3, b, L)	$(q_0, *, \mathrm{R})$

$\longrightarrow L(M) = \{\, a^n b^n \mid n \geq 0 \,\}$

- $\varepsilon \in L(M)$ *Sonst*:
- Jedes akzeptierte Wort muss mit a beginnen. Es wird gelöscht.
- Dann bewegt sich der Kopf über den Rest des Eingabewortes nach rechts hinweg.
- Der Kopf bewegt sich dann zum letzten Buchstaben nach links zurück.
- Der letzte Buchstabe muss ein b sein, das dann gelöscht wird.
- Dann bewegt sich der Kopf zum (nun) ersten Buchstaben und beginnt dort von vorn.

Definition von Turing-Maschinen

Zwei Aufgabentypen:

1. **Die formale Definition einer TM ist anzugeben**
   → Tupel und Überführungstabelle angeben

2. **Zeigen, dass (für ein Beispiel/eine Aufgabe) eine TM existiert**
   → Anweisungen ähnlich zu Pseudocode, die die Arbeitsschritte der TM beschreiben, dürfen verwendet werden…

   … *solange sicher ist, dass diese Schritte „ohne weitere Kreativität" in eine formale TM-Definition umgesetzt werden können!!!*

Informale Beschreibung von TM

... *solange sicher ist, dass diese Schritte „ohne weitere Kreativität" in eine formale TM-Definition umgesetzt werden können!!!*

- *Beispiele:*
  - Bewege den Kopf zur ersten leeren Zelle hinter dem Bandinhalt.
  - Bewege den Kopf zurück zum ersten Buchstaben des Bandinhalts.
  - Lösche jeden zweiten Buchstaben des Eingabewortes (durch * ersetzen).
  - Jede Aktion, die mithilfe eines DEAs durchgeführt werden kann:
    - *z.B.* Prüfe, ob sich 0 und 1 im Eingabewort immer abwechseln.
    - *z.B.* Prüfe, ob in der Eingabe mindestens zwei Nullen aufeinanderfolgen.
    - *z.B.* Prüfe, ob der vorletzte Buchstabe im Eingabewort eine 1 ist.
    - *z.B.* Prüfe, ob sich eine gerade Anzahl von Nullen im Eingabewort befinden.

Informale Beschreibung von TM

… *solange sicher ist, dass diese Schritte „ohne weitere Kreativität“ in eine formale TM-Definition umgesetzt werden können!!!*

- *Gegenbeispiele:*
  - Sortiere die Buchstaben des Eingabewortes (alphabetisch) aufsteigend.
  - Gehe zur Mitte des Eingabewortes.

➢ **Dann verfeinern!!!**

Z.B. Mitte finden: Solange Buchstaben noch unmarkiert sind, wiederhole:

1. Markiere den ersten unmarkierten Buchstaben.
2. Bewege den Kopf zum letzten noch unmarkierten Buchstaben und markiere diesen.
3. Bewege den Kopf zum ersten noch unmarkierten Buchstaben.

Beispiel Mitte finden – Überführungstabelle

z.B. für $\Sigma = \{a, b\}$ verwenden wir $\Gamma \subseteq \{a, b, A, B, *\}$

	a	b	A	B	$*$
q_0	(q_1, A, R)	(q_1, B, R)	„gefunden“	„gefunden“	„leer“
q_1	(q_1, a, R)	(q_1, b, R)	(q_2, A, L)	(q_2, B, L)	$(q_2, *, \mathrm{L})$
q_2	(q_3, A, L)	(q_3, B, L)	„gefunden“	„gefunden“	
q_3	(q_3, a, L)	(q_3, b, L)	(q_0, A, R)	(q_0, B, R)	

Nachdem die Mitte gefunden wurde, können nun weitere Aktionen folgen …

Nichtdeterministische TM – Definition

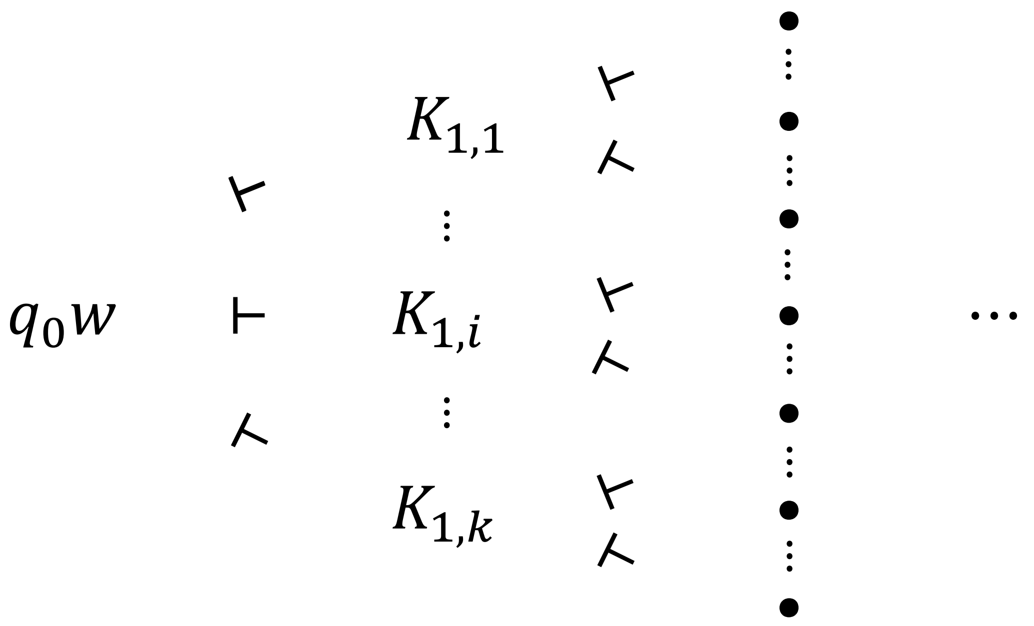
- Eine **nichtdeterministische Turing-Maschine NTM** ist ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, *, F)$, wobei die Überführungsfunktion als Funktion
$$\delta: (Q \setminus F) \times \Gamma \to 2^{Q \times \Gamma \times \{\mathrm{L},\mathrm{R}\}}$$
und die anderen Komponenten wie bei einer DTM definiert sind.
- Der Begriff **Konfiguration** ist wie bei DTM definiert.
- **Konfigurationsübergang** von M:

1. $a_1 \dots a_{i-1} q a_i \dots a_n \vdash a_1 \dots b p a_{i+1} \dots a_n$ gdw. $(p, b, \mathrm{R}) \in \delta(q, a_i)$
2. $a_1 \dots a_{i-1} q a_i a_{i+1} \dots a_n \vdash a_1 \dots a_{i-2} p a_{i-1} b a_{i+1} \dots a_n$ gdw. $(p, b, \mathrm{L}) \in \delta(q, a_i)$

- Die Hülle $\overset{*}{\vdash}$ und die Sprache $L(M)$ sind wie bei einer DTM definiert.

Berechnungen einer NTM

Seien q_0 der Startzustand und w das Eingabewort.



Es entsteht ein **Berechnungsbaum**, dessen Knoten Konfigurationen sind und die Anfangskonfiguration $q_0 w$ die Wurzel ist.

Das Eingabewort w wird genau dann **akzeptiert**, wenn <u>es</u> einen Pfad von $q_0 w$ zu einer Endkonfiguration <u>gibt</u>.

$$L(M) = \{\, w \in \Sigma^* \mid q_0 w \vdash^* upv \ \text{ für gewisse } u, v \in \Gamma^* \text{ und ein } p \in F \,\}$$

Äquivalenz von NTM und DTM

Beobachtung 7.1: Jede DTM kann als eine spezielle NTM angesehen werden, für die $|\delta(q,a)| = 1$ für alle $q \in Q \setminus F$ und alle $a \in \Gamma$ gilt.

Satz 7.2: Zu jeder NTM M kann eine äquivalente DTM M' konstruiert werden.

Idee: Brute Force,
also systematisches Durchprobieren aller nichtdeterministischen Berechnungen

Konstruktion einer DTM M' aus einer NTM M

- Sei r die kleinste Zahl, so dass $|\delta(q,a)| \leq r$ für alle $(q,a) \in (Q \setminus F) \times \Gamma$.
- Nummerieren für jedes Paar (q,a) die möglichen Berechnungsschritte mit $1, 2, \ldots$ (bis maximal r).
- Jede Berechnung auf einem Eingabewort ist eine Folge von Zahlen aus 1 bis r. Die Berechnungen sollen in aufsteigender Sortierung dieser Folgen geschehen.
- *Bsp.:*

	a	b
q_0	1: (q_0, a, R) 2: (q_1, a, R)	1: (q_0, b, R)
q_1	1: (q_2, a, R)	1: (q_0, b, R)
q_2	1: (f, a, L)	1: (f, a, L)

mögliche Berechnungen auf $bbaaa$:

1,
11
111, 112
1111, 1112,
1121
11211 (*haltend*)

Konstruktion einer DTM M' aus einer NTM M

- Jede Berechnung auf einem Eingabewort ist eine Folge von Zahlen aus 1 bis r.
  Die Berechnungen sollen in aufsteigender Sortierung dieser Folgen geschehen.
- Konstruieren die DTM M' so, dass sie folgende Arbeitsschritte realisiert:

  Falls das Eingabewort w ist, wiederhole für alle endlichen Folgen s aus Zahlen von 1 bis r (nach ihren Zahlenwerten in aufsteigender Anordnung):

  1. Erzeuge hinter dem Eingabewort ein Trennsymbol (sagen wir #).
  2. Kopiere w direkt hinter dieses Symbol (#), gefolgt von einem Trennsymbol (sagen wir $).
  3. Erzeuge die nächste Folge s hinter dem zweiten Trennsymbol ($), anfangs die Folge 1.
  4. Simuliere Berechnung s von M auf der Kopie der Eingabe.
     Stoppe, falls die Simulation stoppt.
     (*Falls nötig Platz schaffen durch Umkopieren in weitere leere Zellen*)

Konstruktion einer DTM M' aus einer NTM M

Falls das Eingabewort w die Länge n hat, wiederhole für alle endlichen Folgen s (in aufsteigender Anordnung):

1. Erzeuge hinter dem Eingabewort ein Trennsymbol (sagen wir #).
2. Kopiere w direkt hinter dieses Symbol (#), gefolgt von einem Trennsymbol (sagen wir $).
3. Erzeuge die nächste Folge s hinter dem zweiten Trennsymbol ($), anfangs die Folge 1.
4. Simuliere Berechnung s von M auf der Kopie der Eingabe.
   Stoppe, falls die Simulation stoppt.

- Falls $w \in L(M)$, dann stoppt M auf der Eingabe w in mindestens einer ihrer möglichen Berechnungen.
  Dann wird M' diese Berechnung irgendwann simulieren und ebenfalls stoppen.
  Dann gilt $w \in L(M')$.
- Falls $w \notin L(M)$, dann hält M in keiner ihrer Berechnungen auf der Eingabe w.
  Dann wird auch M' nie stoppen und es gilt $w \notin L(M')$. $\blacksquare$