

Zwei einfache Programmiertechniken

1. Merken im Zustand

- Nutzen Zustände als endlichen Speicher.

2. Markieren von Symbolen

- Nutzen Bandalphabet zur Markierung von Positionen des Bands.

DTM M_3 für Palindrome über $\{0,1\}$

δ	0	1	\sqcup	\triangleright
q_0	(q_1, \sqcup, R)	(q_2, \sqcup, R)	(q_6, \sqcup, R)	(q_0, \triangleright, R)
q_1	$(q_1, 0, R)$	$(q_1, 1, R)$	(q_3, \sqcup, L)	(q_7, \triangleright, R)
q_2	$(q_2, 0, R)$	$(q_2, 1, R)$	(q_4, \sqcup, L)	(q_7, \triangleright, R)
q_3	(q_5, \sqcup, L)	$(q_7, 1, L)$	(q_6, \sqcup, L)	(q_7, \triangleright, R)
q_4	$(q_7, 0, L)$	(q_5, \sqcup, L)	(q_6, \sqcup, L)	(q_7, \triangleright, R)
q_5	$(q_5, 0, L)$	$(q_5, 1, L)$	(q_0, \sqcup, R)	(q_7, \triangleright, R)

$q_6 = q_{\text{accept}}, q_7 = q_{\text{reject}}$

Im Zustand merken - Beispiel

$$L := \{w \in \Sigma^* \mid w = w_1 \dots w_n, \exists i, 2 \leq i \leq n: w_i = w_1\}$$

$$1) \quad \delta(q_0, \triangleright) = (q_0, \triangleright, R)$$

$$2) \quad \delta(q_0, \sqcup) = (q_2, \sqcup, L)$$

$$3) \quad \delta(q_0, a) = ([q_0, a], a, R) \quad \forall a \in \Sigma$$

$$4) \quad \delta([q_0, a], a) = (q_1, a, L) \quad \forall a \in \Sigma$$

$$5) \quad \delta([q_0, a], b) = ([q_0, a], b, R) \quad \forall a, b \in \Sigma, a \neq b$$

$$6) \quad \delta([q_0, a], \sqcup) = (q_2, \sqcup, L)$$

$$7) \quad \delta([q_0, a], \triangleright) = (q_2, \triangleright, R)$$

$$q_{\text{accept}} = q_1, \quad q_{\text{reject}} = q_2$$

DTM M'_3 für Palindrome über $\{0,1\}$

δ	0	1	\sqcup	\triangleright
q_0	$([q_0,0],\sqcup,R)$	$([q_0,1],\sqcup,R)$	(q_6,\sqcup,R)	(q_0,\triangleright,R)
$[q_0,0]$	$([q_0,0],0,R)$	$([q_0,0],1,R)$	$([q_1,0],\sqcup,L)$	(q_7,\triangleright,R)
$[q_0,1]$	$([q_0,1],0,R)$	$([q_0,1],1,R)$	$([q_1,1],\sqcup,L)$	(q_7,\triangleright,R)
$[q_1,0]$	(q_5,\sqcup,L)	$(q_7,1,L)$	(q_6,\sqcup,L)	(q_7,\triangleright,R)
$[q_1,1]$	$(q_7,0,L)$	(q_5,\sqcup,L)	(q_6,\sqcup,L)	(q_7,\triangleright,R)
q_5	$(q_5,0,L)$	$(q_5,1,L)$	(q_0,\sqcup,R)	(q_7,\triangleright,R)

$q_6 = q_{\text{accept}}, q_7 = q_{\text{reject}}$

Markieren von Symbolen

Element Distinctness:

$L := \{ \#w_1\# \dots \#w_n \mid w_i \in \{0,1\}^*, w_i \neq w_j \text{ für alle } i \neq j \}$

Beispiele

- $\#001\#111\#10\#11 \in L$
- $\#011\#11\#01\#10\#01 \notin L$

DTM für Element-Distinctness

1. Falls das erste Eingabesymbol nicht # ist, lehne ab, sonst ersetze # durch #' (markiere #).
2. Finde das nächste # rechts vom ersten # und ersetze es durch #'. Wird kein weiteres # gefunden, akzeptiere.
3. Teste, ob die beiden Folgen w_i , w_j rechts der Symbole #' gleich sind. Wenn ja, lehne ab.
4. Verschiebe Markierungen für den Vergleich des nächsten Paares von Folgen. Falls dieses Paar nicht mehr existiert, akzeptiere. Sonst gehe zu 3.

Element-Distinctness - Markierte Positionen

▷	#'	0	1	1	#'	1	1	#	0	1	#	1	0	#	0	1
---	----	---	---	---	----	---	---	---	---	---	---	---	---	---	---	---

Markierte # bei Vergleich w_1 und w_2 .

▷	#'	0	1	1	#	1	1	#'	0	1	#	1	0	#	0	1
---	----	---	---	---	---	---	---	----	---	---	---	---	---	---	---	---

Markierte # bei Vergleich w_1 und w_3 .

⋮

▷	#	0	1	1	#'	1	1	#	0	1	#	1	0	#'	0	1
---	---	---	---	---	----	---	---	---	---	---	---	---	---	----	---	---

Markierte # bei Vergleich w_2 und w_5 .

Element-Distinctness - Markierte Positionen

▷	#'	0	1	1	#'	1	1	#	0	1	#	1	0	#	0	1
---	----	---	---	---	----	---	---	---	---	---	---	---	---	---	---	---

Markierte # bei Vergleich w_1 und w_2 .

▷	#'	0	1	1	#	1	1	#'	0	1	#	1	0	#	0	1
---	----	---	---	---	---	---	---	----	---	---	---	---	---	---	---	---

Markierte # bei Vergleich w_1 und w_3 .

⋮

Es müssen jetzt noch die Paare verglichen werden.
Mit mehreren Bändern wäre das leichter!

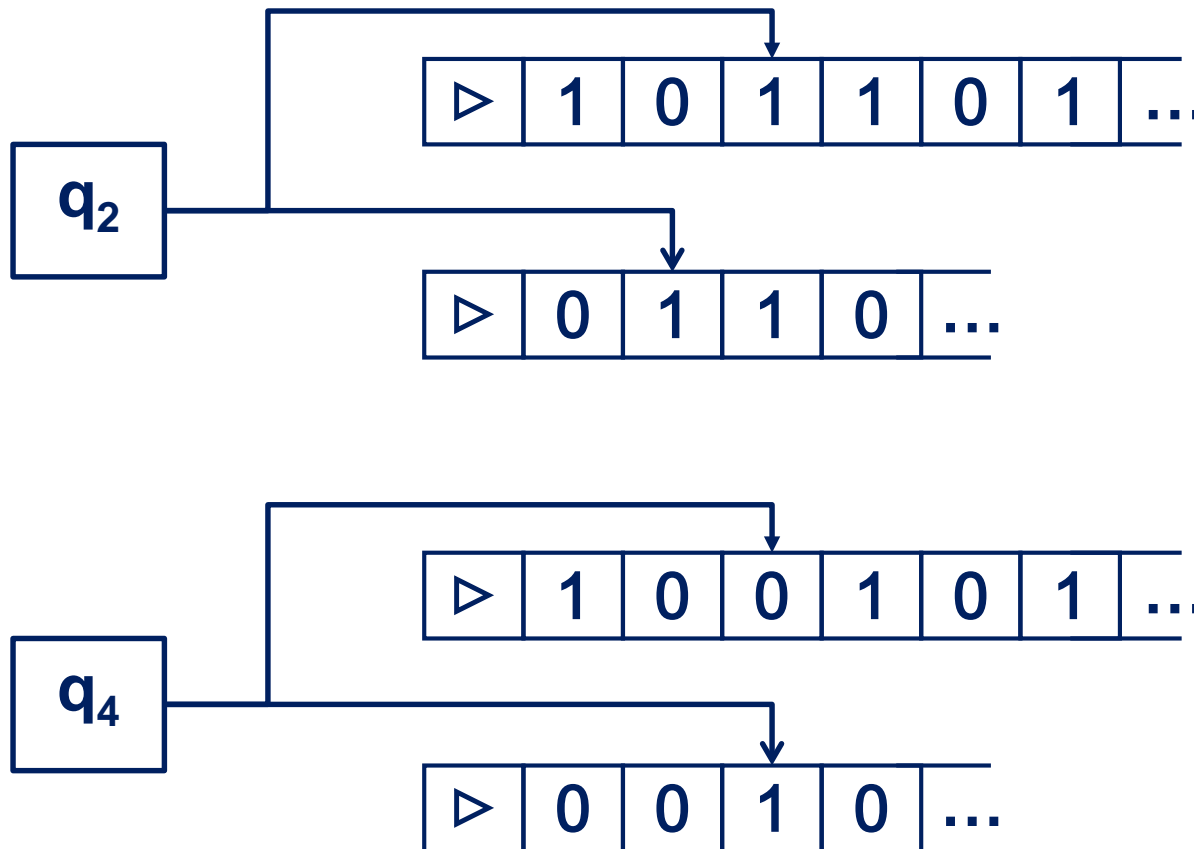
Mehrband-Turingmaschinen - Definition

Definition 2.7 Eine **deterministische k-Band Turingmaschine** (k-DTM) ist ein 4-Tupel $(Q, \Sigma, \Gamma, \delta)$. Für Q, Σ, Γ gelten die Anforderungen einer 1-Band DTM. Für die Übergangsfunktion δ gilt

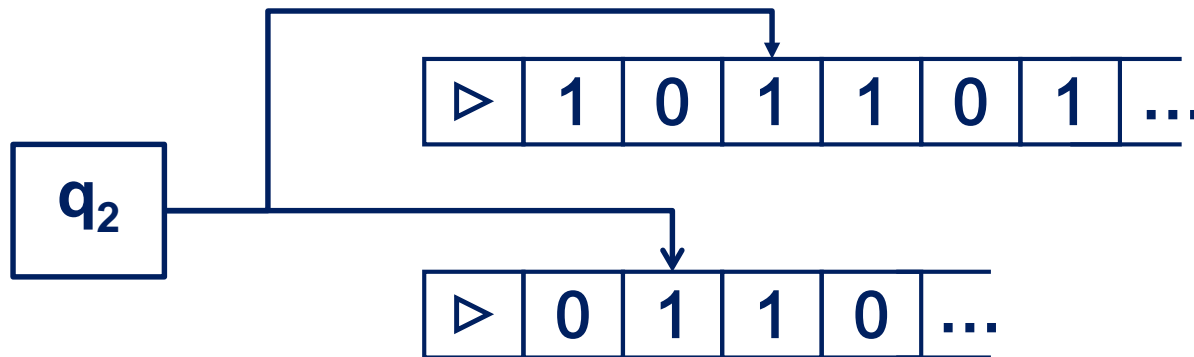
$$\delta: Q \setminus \{q_{\text{accept}}, q_{\text{reject}}\} \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R, S\}^k.$$

Dabei steht **S** für stationär, d.h., der Lesekopf wird nicht bewegt. Für das Startsymbol gelten dieselben Einschränkungen wie bei 1-Band DTMs.

Rechenschritt bei $\delta(q_2, 1, 1) = (q_4, 0, 0, S, R)$



Konfiguration einer 2-DTM



Konfiguration besteht aus

- Zustand,
- Inhalt der Bänder
- Position der Leseköpfe.

Konfiguration einer k-DTM

Übliche Kodierung einer Konfiguration:

$$(q, (v_1, w_1), \dots, (v_k, w_k)) \in Q \times ((\Gamma^*)^2)^k$$

wobei v_i, w_i die Zeichenketten vor und nach der Kopfposition auf Band i sind (der Kopf steht auf dem ersten Zeichen von w_i).

Startkonfiguration bei Eingabe $w \in \Sigma^*$ besteht aus

- Startzustand q_0 ,
- erstes Band enthält $\triangleright w$,
- alle anderen Bänder $\triangleright \sqcup \dots$ (d.h. $v_i = \varepsilon$ für alle i),
- alle Leseköpfe stehen auf \triangleright (d.h. $w_i = \triangleright$ für alle i).

Akzeptierende/ablehnende Konfiguration

- Zustand ist $q_{\text{accept}}, q_{\text{reject}}$.

Berechnung, Akzeptieren, Entscheiden, ...

wie bei 1-Band DTMs.

Beispiel: Addition zweier Binärzahlen

Eingabe: $w_1\#w_2$ für zwei Binärzahlen w_1 und w_2 , wobei die erste Binärzahl jeweils die niedrigste Binärstelle angibt

(**Beispiel:** 01#001 entspricht den Zahlen 2 und 4)

Ausgabe: das Ergebnis von w_1+w_2 auf irgendeinem Band

Strategie (Übung):

- Wir verwenden 3-Band Turingmaschine
- w_2 wird zunächst auf Band 2 verschoben
- Dann werden w_1 und w_2 auf Band 3 bitweise zu w_1+w_2 zusammenaddiert. Zum Schluss gehen wir in q_{accept} .

Mächtigkeit von k-DTMs

Satz 2.8 Wird die Sprache L von einer k -DTM entschieden (akzeptiert), so gibt es auch eine 1-Band Turingmaschine, die L entscheidet (akzeptiert).

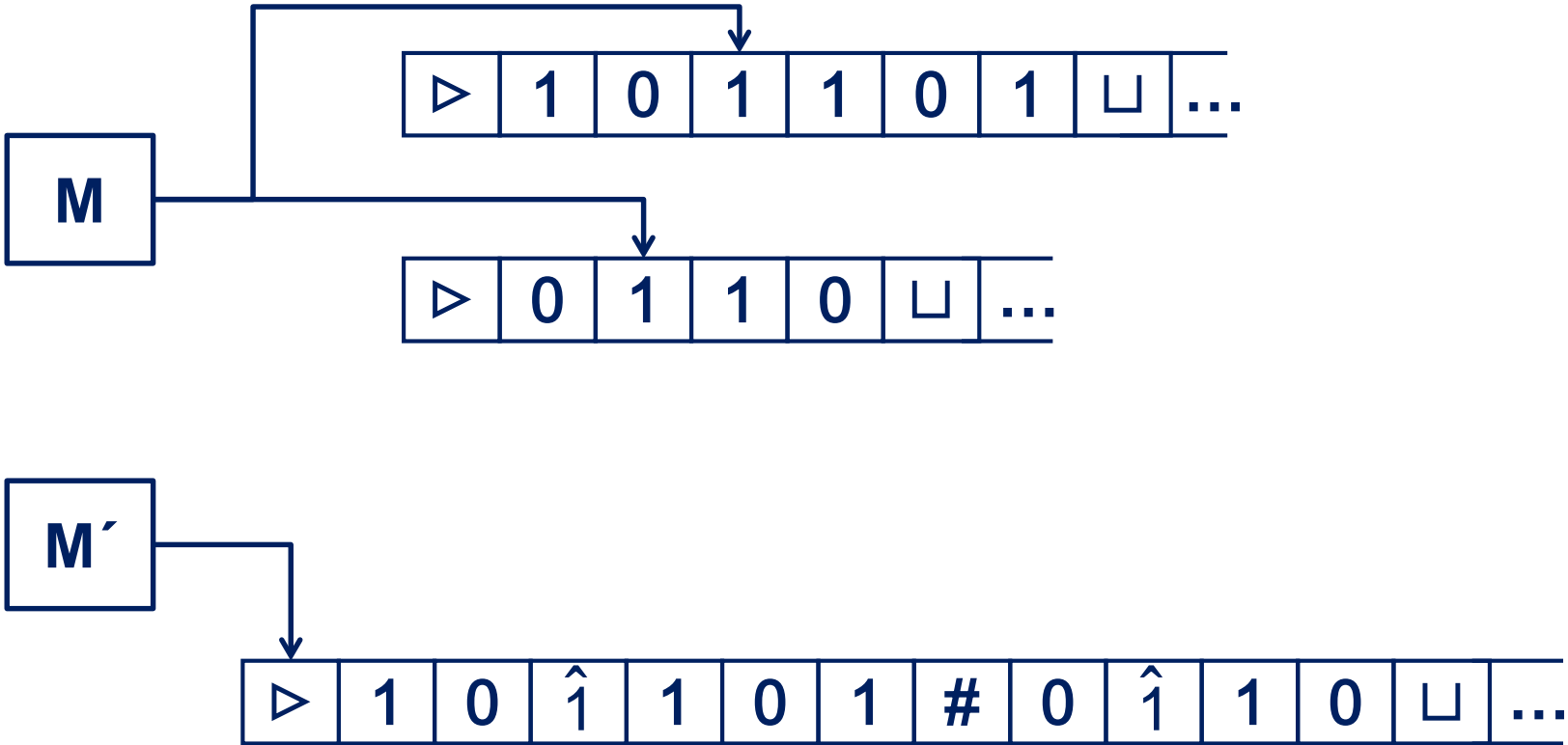
Beweis beruht auf Simulation einer k -DTM durch DTM.

Simulation von k-DTMs durch DTMs

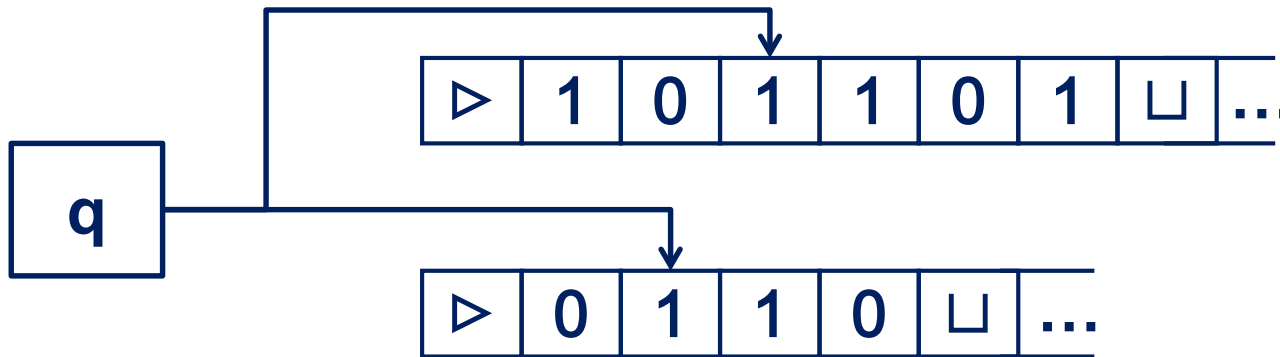
Simulation einer k-DTM durch DTM beruht auf zwei Techniken:

- Kodierung der Konfigurationen einer k-DTM M auf 1-Band DTM M' .
- Berechnung der Kodierung der Nachfolgekonfiguration aus Kodierung der Konfiguration.

Speicherung von 2 Bändern durch 1 Band



Speicherung der Bandinhalte im Kopf



$[p_1, (q, 1, 1)]$

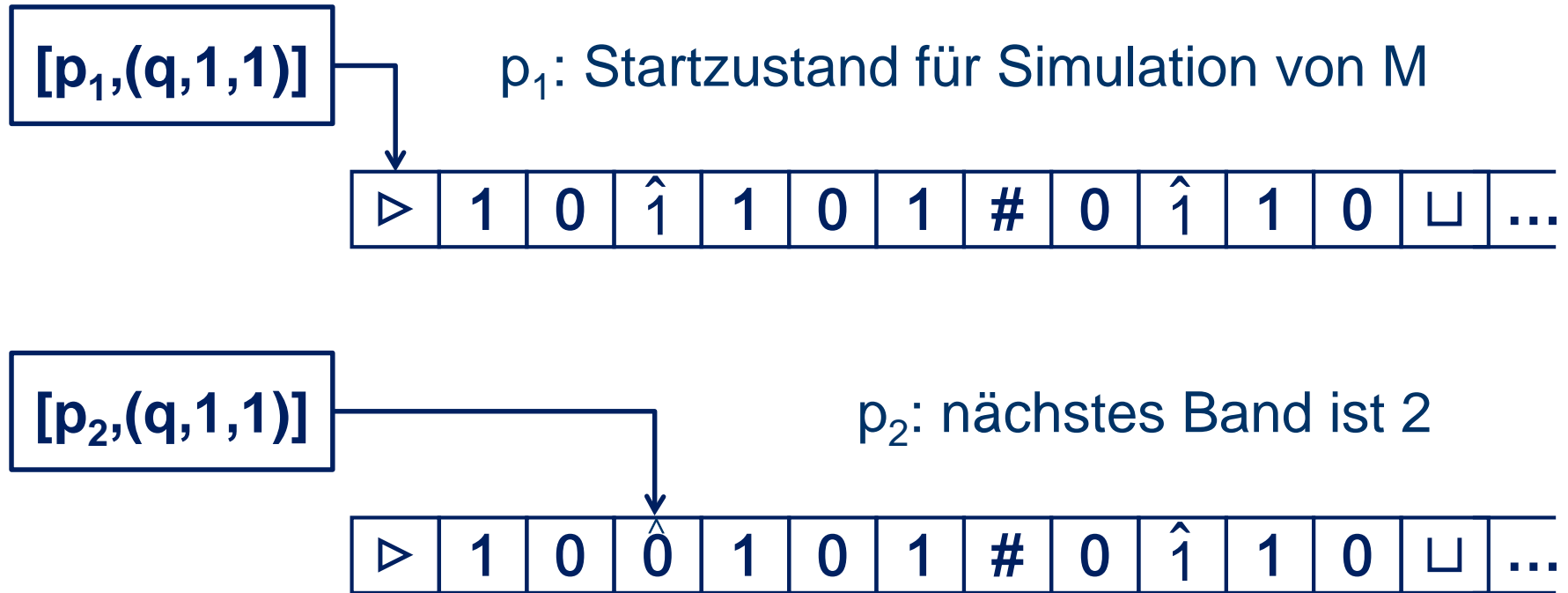
p_1 : Startzustand für Simulation von M



Simulation eines Rechenschritts

Beispiel: $\delta(q,1,1)=(q',0,0,S,R)$

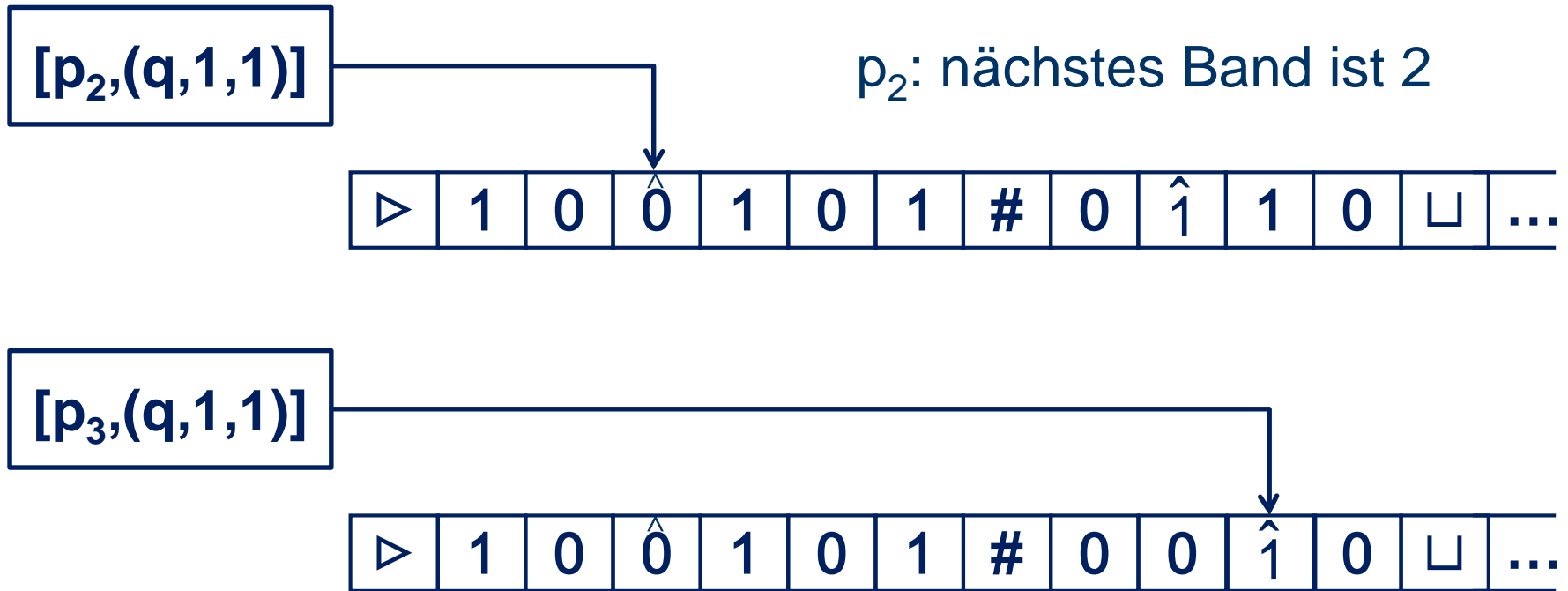
1.: Gehe zum markierten Zeichen in Band 1 und führe die notwendigen Änderungen durch.



Simulation eines Rechenschritts

Beispiel: $\delta(q,1,1)=(q',0,0,S,R)$

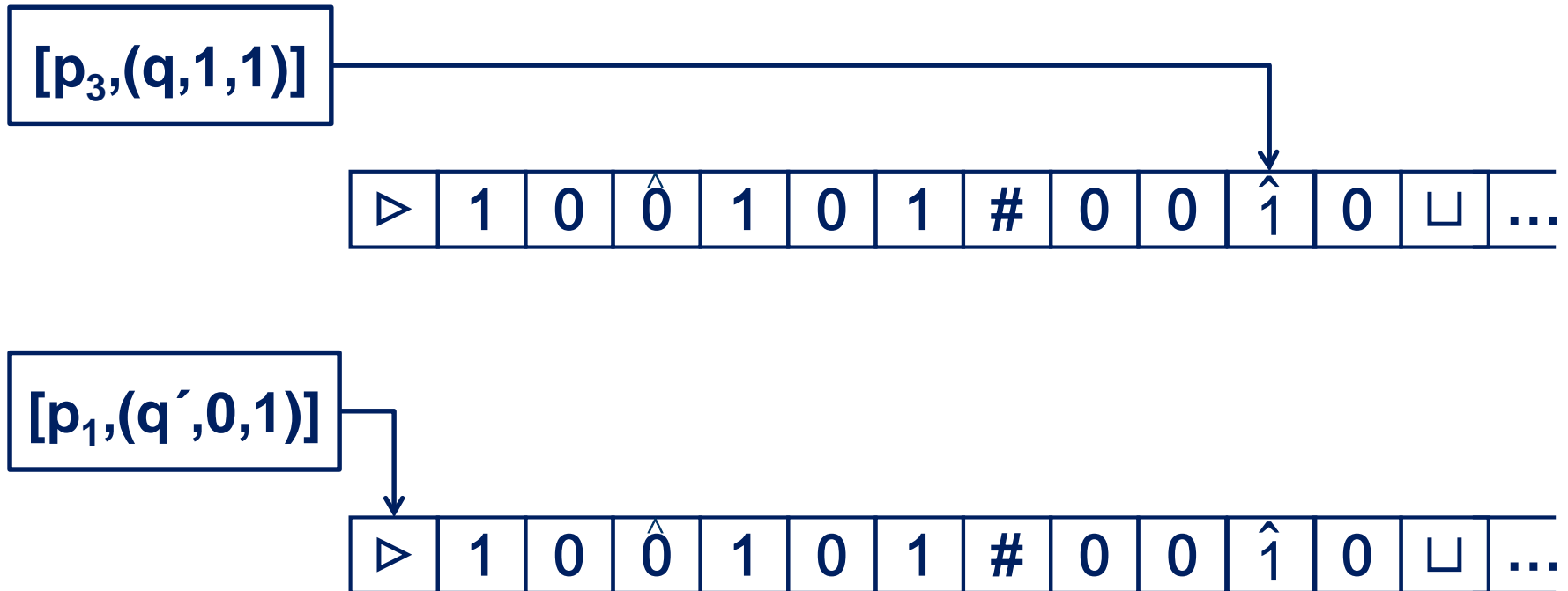
2.: Gehe zum markierten Zeichen in Band 2 und führe die notwendigen Änderungen durch.



Simulation eines Rechenschritts

Beispiel: $\delta(q,1,1)=(q',0,0,S,R)$

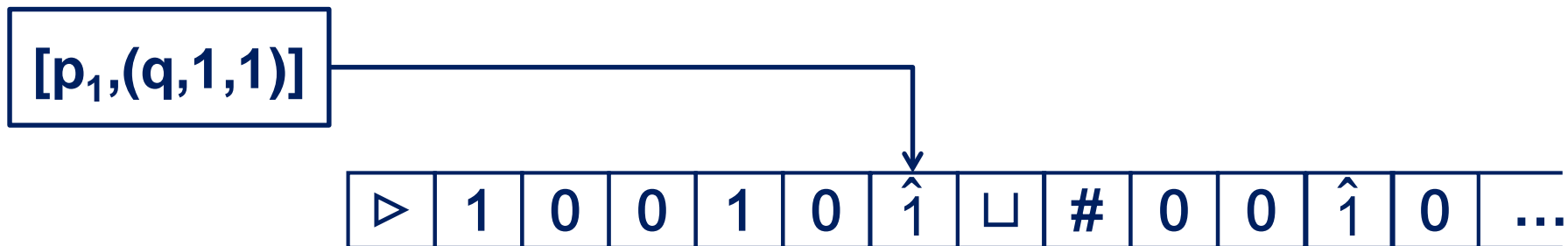
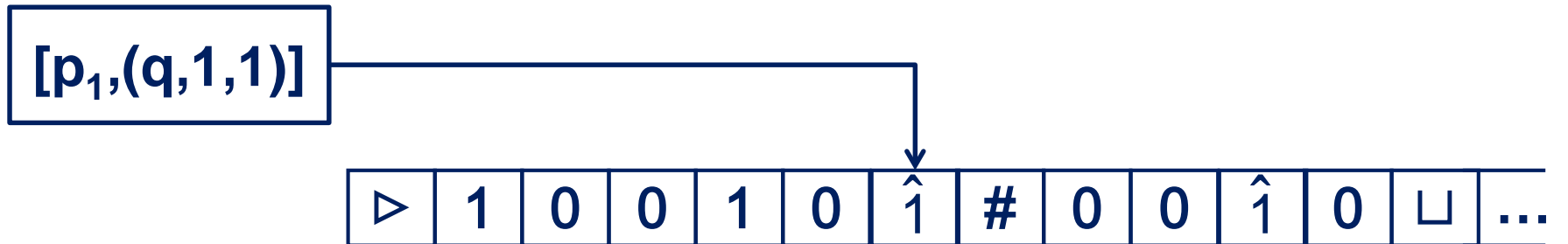
3.: Aktualisiere Zustand, kehre zum Anfang zurück und übernehme dabei markierte Zeichen in den Kopf.



Simulation eines Rechenschritts

Problem: Was machen wir, wenn der Platz auf einem Band i nicht ausreicht?

Lösung: schiebe Zeichen eine Position nach rechts.



k-Band Turingmaschinen für Sprachen

Element Distinctness:

$L := \{ \#w_1\# \dots \#w_n \mid w_i \in \{0,1\}^*, w_i \neq w_j \text{ für alle } i \neq j \}$

Lösung: 3-Band TM

- **Markiere wie vorher Wortpaare, die verglichen werden sollen, auf dem Eingabeband.**
- **Kopiere das erste markierte Wort auf Band 2 und das zweite auf Band 3.**
- **Vergleiche die Worte auf Band 2 und 3.**
- **Falls gleich, lehne die Eingabe ab, sonst markiere das nächste Wortpaar.**

Abstrahierung von TM zum Algorithmus

- Algorithmus verwendet v Variablen:
verwende dafür v zusätzliche Bänder in der TM
- Grundlegende Rechenoperationen:
einfach in TM umsetzbar
- If $\langle \text{Condition} \rangle$ then $\langle \text{Block1} \rangle$ else $\langle \text{Block2} \rangle$:

	(a_1, \dots, a_k)	(b_1, \dots, b_k)	...
q-Block für $\langle \text{Condition} \rangle$	Nach Test von $\langle \text{Condition} \rangle$ Sprung zu $\langle \text{Block1} \rangle$ oder $\langle \text{Block2} \rangle$		
q-Block für $\langle \text{Block1} \rangle$			
q-Block für $\langle \text{Block2} \rangle$			

Abstrahierung von TM zum Algorithmus

- for $i=1$ to n do **<Block>**:

	(a_1, \dots, a_k)	(b_1, \dots, b_k)	...
q-Block für $i=1$			
q-Block für <Block>			
q-Block für $i=i+1, i>n?$	$i \leq n$: springe zurück zum q-Block für <Block>		

- while **<Condition>** do **<Block>**:

	(a_1, \dots, a_k)	(b_1, \dots, b_k)	...
q-Block für <Condition>	<Condition> wahr: springe zu <Block>		
q-Block für <Block>	am Ende zurück zu <Condition>		

Churchsche These (1936)

Die im intuitiven Sinne berechenbaren Funktionen und Sprachen sind genau die, die durch Turingmaschinen berechenbar sind.

Warum sind Turingmaschinen tatsächlich ein geeignetes Modell?

- Menschliche Wahrnehmung ist endlich
(endliche Anzahl an Rezeptoren)
- Jeder realisierbare Rechner muss endlicher Natur sein
und den **physikalischen Gesetzen** folgen

Abschlusseigenschaften

\bar{L} : Komplementsprache zu L , d.h. $\bar{L} = \Sigma^* \setminus L$

Satz 2.10 Seien L_1, L_2 entscheidbare Sprachen, dann gilt:

1. \bar{L}_1 ist entscheidbar.
2. $L_1 \cap L_2$ ist entscheidbar.
3. $L_1 \cup L_2$ ist entscheidbar.

Satz 2.11 Seien L_1, L_2 rekursiv aufzählbare Sprachen, dann gilt:

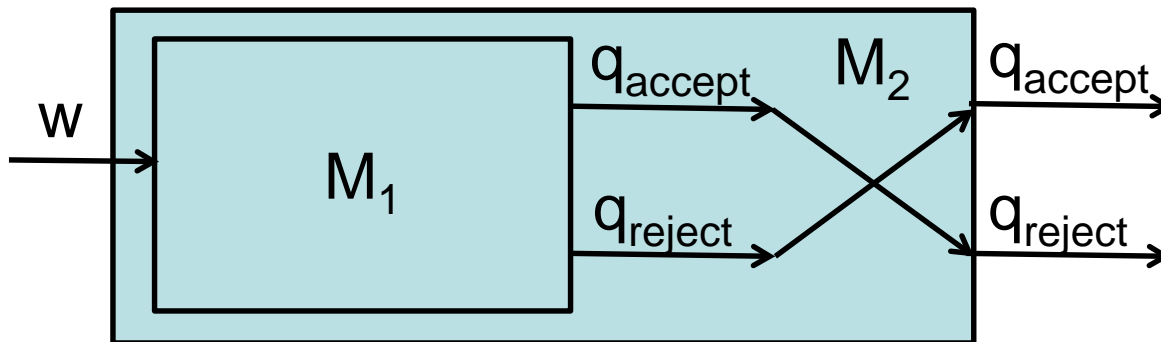
1. $L_1 \cap L_2$ ist rekursiv aufzählbar.
2. $L_1 \cup L_2$ ist rekursiv aufzählbar.

Abschlusseigenschaften

Satz 2.10 Seien L_1, L_2 entscheidbare Sprachen, dann gilt:

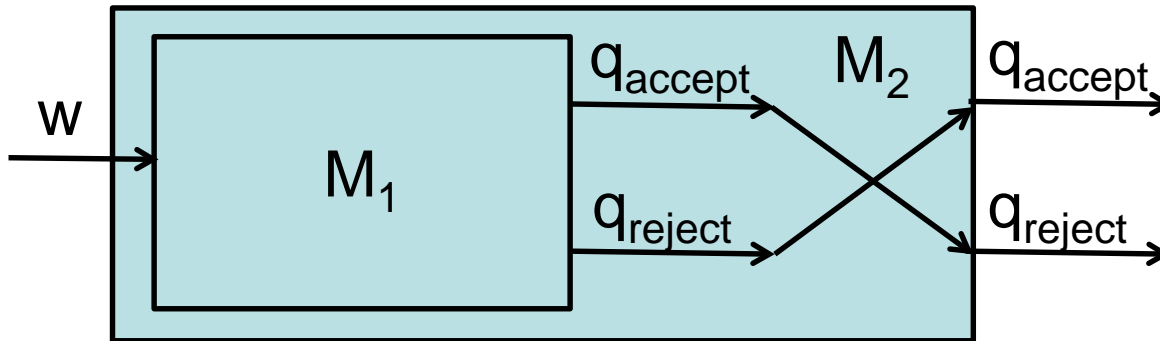
1. \bar{L}_1 ist entscheidbar.
2. $L_1 \cap L_2$ ist entscheidbar.
3. $L_1 \cup L_2$ ist entscheidbar.

Beweis von 1.: Sei M_1 die TM zu L_1 . TM M_2 zu \bar{L}_1 :



Abschlusseigenschaften

Beweis von 1.: Sei M_1 die TM zu L_1 . TM M_2 zu \bar{L}_1 :



M_2 : Q, Σ, Γ und δ genauso definiert wie in M_1 , nur dass q_{accept} und q_{reject} getauscht werden.

$x \in L_1$: M_1 akzeptiert $x \Rightarrow M_2$ lehnt x ab
 $x \notin L_1$: M_1 lehnt x ab $\Rightarrow M_2$ akzeptiert x } $L(M_2) = \bar{L}_1$

Schnitt/Vereinigung entscheidbarer Sprachen

Sei M_1 die DTM zu L_1 und M_2 die DTM zu L_2 .

M bei Eingabe w :

1. Kopiere die Eingabe auf Band 2.
2. Arbeite **parallel** wie M_1 auf Band 1 und wie M_2 auf Band 2.

Parallel arbeiten:

$\delta([q_1, q_2], a_1, a_2) = ([p_1, p_2], b_1, b_2, D_1, D_2)$, falls

$\delta(q_1, a_1) = (p_1, b_1, D_1)$ und $\delta(q_2, a_2) = (p_2, b_2, D_2)$

M für $L_1 \cap L_2$:

- Akzeptiere, sobald M_1 und M_2 akzeptieren
- Lehne Eingabe ab, wenn M_1 oder M_2 Eingabe ablehnen

Schnitt/Vereinigung entscheidbarer Sprachen

M für $L_1 \cap L_2$:

- Akzeptiere, sobald M_1 und M_2 akzeptieren
- Lehne Eingabe ab, wenn M_1 oder M_2 Eingabe ablehnen

$q_1 \notin \{q_{\text{accept}}^1, q_{\text{reject}}^1\}$ und $q_2 \notin \{q_{\text{accept}}^2, q_{\text{reject}}^2\}$:

$\delta([q_1, q_2], a_1, a_2) = ([p_1, p_2], b_1, b_2, D_1, D_2)$, falls
 $\delta(q_1, a_1) = (p_1, b_1, D_1)$ und $\delta(q_2, a_2) = (p_2, b_2, D_2)$

$q_2 \notin \{q_{\text{accept}}^2, q_{\text{reject}}^2\}$:

$\delta([q_{\text{accept}}^1, q_2], a_1, a_2) = ([q_{\text{accept}}^1, p_2], a_1, b_2, S, D_2)$, falls $\delta(q_2, a_2) = (p_2, b_2, D_2)$

Genauso für $[q_1, q_{\text{accept}}^2]$.

Wir setzen $[q_{\text{accept}}^1, q_{\text{accept}}^2]$ als q_{accept} und gehen von $[q_1, q_2]$ zu q_{reject} für alle $[q_1, q_2]$ mit $q_1 = q_{\text{reject}}^1$ oder $q_2 = q_{\text{reject}}^2$.

Durchschnitt entscheidbarer Sprachen

Sei M_1 die DTM zu L_1 und M_2 die DTM zu L_2 .

Alternativstrategie für M für $L_1 \cap L_2$ bei Eingabe w :

1. **Simuliere M_1 gestartet auf w . Falls M_1 w ablehnt, dann lehne w ab.**
2. **Falls M_1 w akzeptiert, dann simuliere M_2 gestartet mit w . Falls M_2 w akzeptiert, dann akzeptiere w , sonst lehne w ab.**

Zu zeigen: M entscheidet $L_1 \cap L_2$.

- **$w \in L_1 \cap L_2$: M_1 und M_2 akzeptieren $w \Rightarrow M$ akzeptiert w .**
- **$w \notin L_1 \cap L_2$: M_1 oder M_2 verwerfen $w \Rightarrow M$ verwirft w .**

Durchschnitt entscheidbarer Sprachen

Anschaulich:

M

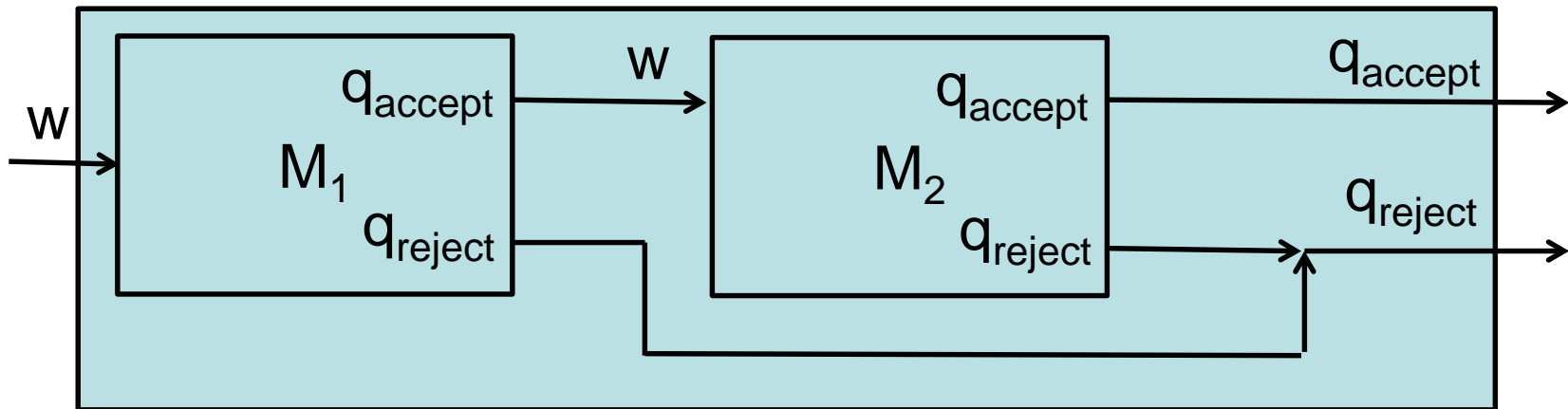


Schaubild in Beweisen zur Illustration hilfreich, aber trotzdem ist eine formale (algorithmische) Beschreibung zu leisten. Darüber hinaus ist formale Argumentation notwendig, dass DTM M tatsächlich die gewünschte Sprache entscheidet.

Schnitt/Vereinigung entscheidbarer Sprachen

Sei M_1 die DTM zu L_1 und M_2 die DTM zu L_2 .

M bei Eingabe w :

1. Kopiere die Eingabe auf Band 2.
2. Arbeite parallel wie M_1 auf Band 1 und wie M_2 auf Band 2.

Parallel arbeiten:

$\delta([q_1, q_2], a_1, a_2) = ([p_1, p_2], b_1, b_2, D_1, D_2)$, falls

$\delta(q_1, a_1) = (p_1, b_1, D_1)$ und $\delta(q_2, a_2) = (p_2, b_2, D_2)$

M für $L_1 \cup L_2$:

- Akzeptiere, sobald M_1 oder M_2 akzeptieren
- Lehne Eingabe ab, wenn M_1 und M_2 Eingabe ablehnen

Abschlusseigenschaften

Satz 2.11 Seien L_1, L_2 rekursiv aufzählbare Sprachen, dann gilt:

1. $L_1 \cap L_2$ ist rekursiv aufzählbar.
2. $L_1 \cup L_2$ ist rekursiv aufzählbar.

Beweis:

1. und 2. wie mit der Parallelmethode auf Folie 30

- Warum kann sie sequentielle Methode auf Folie 31 nicht angewandt werden?
- Warum ist \bar{L}_1 nicht notwendigerweise rekursiv aufzählbar?

Abschlusseigenschaften

Satz 2.12 L ist genau dann entscheidbar, wenn L und \bar{L} rekursiv aufzählbar sind.

Beweis:

- L und \bar{L} rekursiv aufzählbar: es gibt DTMs M und \bar{M} , die L und \bar{L} akzeptieren
- M' , die L entscheidet: lass M und \bar{M} **parallel** auf Eingabe w laufen
- akzeptiere, falls M w akzeptiert
- verwerfe, falls \bar{M} w akzeptiert
- $w \in L$: M akzeptiert $w \Rightarrow M'$ akzeptiert w
- $w \notin L$: \bar{M} akzeptiert $w \Rightarrow M'$ verwirft w