Formale Sprachen und Komplexitätstheorie

WS 2018/19

Robert Elsässer

Organisatorisches

Vorlesung:

- Di 11:15 – 12:45 T.01

Proseminar:

- Di 13:00 – 13:45 T.01
- Mi 14:15 – 15:00 T.03
- Beginn: nächste Woche

Organisatorisches

Heimübungen:

- Jede Woche ein Übungsblatt (dienstags)
- 1. Blatt diese Woche
- Abgabe: Dienstag bis 11:00 Uhr
- Erfolgreicher Abschluss (Gruppen zu 2-4 Personen)
  - mind. 50% der erreichbaren Punkte
  - mind. einmal korrekt Vorrechnen
  - zwei Tests während des Semesters (50% der Gesamtbewertung)
- Erste bewertete Aufgabe: übernächste Woche
- Vorstellung *einer* Musterlösung im Proseminar

Organisatorisches

Proseminargruppen:

- PLUSonline
  - erreichbar über https://online.uni-salzburg.at/plus_online/webnav.ini
  - ITS-Login und Passwort
  - Bis zu 25 Teilnehmer
  - eine Gruppe
  - Übungsaufgaben und Vorlesungsfolien über die Webseite der Vorlesung: http://fl.cosy.sbg.ac.at
- Abmeldung bis zum 25.10.2017, 23:55 Uhr
  - Nach diesem Zeitpunkt ist keine Abmeldung mehr möglich und es folgt eine Bewertung der Leistungen am Ende des Semesters

Organisatorisches

Klausur:

- Eine Korrelation mit den PS-Aufgaben ist zu erwarten
- Es gab in der Vergangenheit einen direkten Zusammenhang zwischen PS-Teilnahme bzw. -abgabe und gutem Abschneiden bei Klausuren

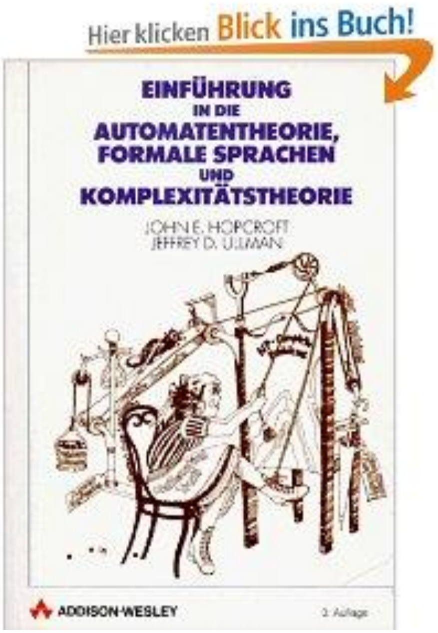
Sprechzeiten:

- Di 10:00 – 11:00
- (Raum 2.23, Jakob-Haringer-Straße 2)

Organisatorisches

Literatur:

- John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman: *Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie*, 3. Auflage, Pearson Studium, 2011
- Die Vorlesungsfolien wurden unter Verwendung der Folien und des Skriptes der Vorlesung *„Einführung in die Berechenbarkeit, Komplexität und Formale Sprachen"*
  von Prof. Dr. Johannes Blömer,
  Prof. Dr. Friedhelm Meyer auf der Heide
  bzw. Prof. Dr. Christian Scheideler erstellt. Die meisten Bilder, Definitionen und Beschreibungen wurden aus den oben genannten Unterlagen übernommen.
- Die Folien wurden von Eva Lugstein (ehem. Studienassistentin am FB Cowi) überarbeitet.



Quelle: amazon.de

Motivation

- Was lässt sich mit dem Computer lösen?
  Wie effizient lassen sich einzelne Probleme lösen?
- Grenzen und Möglichkeiten eines Rechners.
- Eine geeignete Formalisierung ist Voraussetzung
  für eine systematische Lösung.
- Als Ausdrucksmittel muss man passende Kalküle
  und Notationen anwenden können.

Ziele

- Einen Überblick über grundlegende Formalisierungsmethoden zu bekommen
- Die für die Methoden typische Techniken zu erlernen
- Techniken an typischen Beispielen anzuwenden

Insgesamt soll erlernt werden:

- Aufgaben präzise zu formalisieren und zu analysieren
- berechenbare von unberechenbaren Problemen zu unterscheiden
- festzustellen, ob ein Problem effiziente Lösungen haben kann

Durchführung

Zu jedem Bereich soll(en):

- mit einigen typischen Beispielen motivierend hineingeführt werden
- der konzeptionelle Kern der Methode vorgestellt werden
- Anwendungstechniken an Beispielen gezeigt und in den Proseminaren erfahren werden
- an einem durchgehenden Beispiel größere Zusammenhänge gelernt werden

Inhaltsangabe

- Einleitung, Motivation
- Turingmaschinen
- Arbeitstechniken

} Einführung

- Unentscheidbare Probleme
- Das Halteproblem
- Reduktionen

} Berechenbarkeit

- Zeitkomplexität
- Die Klassen P und NP
- NP-Vollständigkeit
- NP-vollständige Probleme

} Komplexität

- Formale Sprachen und Automaten
- Kellerautomaten und kontextfreie Sprachen
- Kontextsensitive Sprachen

} Formale Sprachen

1. Einführung

$\mathrm{ALG}(n)$

```
for t = 1 to ∞
    for x = 1 to t
        for y = 1 to t
            for z = 1 to t
                if x^n + y^n = z^n
                    return (x, y, z)
```

$\mathrm{ALG}(n)$ hält bei Eingabe n genau dann, wenn $x^n + y^n = z^n$ für irgendwelche natürlichen Zahlen x, y, z.

(vgl. großen Satz von Fermat)

1. Einführung

Gibt es einen Algorithmus HALTE, der

- als Eingabe einen beliebigen Algorithmus ALG und eine Eingabe w für ALG erhält und
- entscheidet, ob ALG bei Eingabe w hält?

Satz von Turing:

Einen solchen Algorithmus kann es nicht geben.

1. Einführung

- Was sind die wesentlichen Möglichkeiten und Grenzen von Computern?

- **Berechenbarkeit und Komplexität:**
  - Was ist ein Problem?
    - Berechnung einer Funktion, Optimierungsproblem, Entscheiden einer Sprache
    - Multiplikation zweier Matrizen, Kürzeste Wege finden, usw…
  - Wie modelliert man einen Computer?
    - Registermaschinen (RAM), λ-Kalkulus, μ-Rekursion, Turingmaschinen, usw…

1. Einführung

- **Problem des Handlungsreisenden:**
  – Wien, Krems, St. Pölten, Wiener Neustadt, Mürzzuschlag, Graz, Wolfsberg, Klagenfurt, Villach, Spital, Lienz, Brenner, Innsbruck, Kitzbühel, Bischofshofen, Hallein, Salzburg, Braunau, Ried, Wels, Linz, Enns, Amstetten, Zwettl, Stockerau

1. Einführung

- Was sind die wesentlichen Möglichkeiten und Grenzen von Computern?

- **Berechenbarkeit und Komplexität:**
  - Was ist ein Problem?
    - Berechnung einer Funktion, Optimierungsproblem, Entscheiden einer Sprache
    - Multiplikation zweier Matrizen, Kürzeste Wege finden, usw…
  - Wie modelliert man einen Computer?
    - Registermaschinen (RAM), λ-Kalkulus, μ-Rekursion, Turingmaschinen, usw…

1. Einführung

- Alle sinnvollen Rechenmodelle liefern aus unserer Sicht die gleichen Ergebnisse
  - **Churchsche These**

- Es gibt Probleme, die algorithmisch nicht gelöst werden können (z.B. das Halteproblem)
  - **Berechenbarkeit**

- Manche Problem können zwar algorithmisch, aber nicht effizient gelöst werden (z.B. das Problem des Handlungsreisenden)
  - **Komplexität**

1. Einführung

- Wie werden Sprachen beschrieben?
  - Mit Hilfe von Grammatiken – syntaktisch korrekte Java-Programme

- Wie analysiert man Worte?
  - Mit Hilfe der Syntaxanalyse

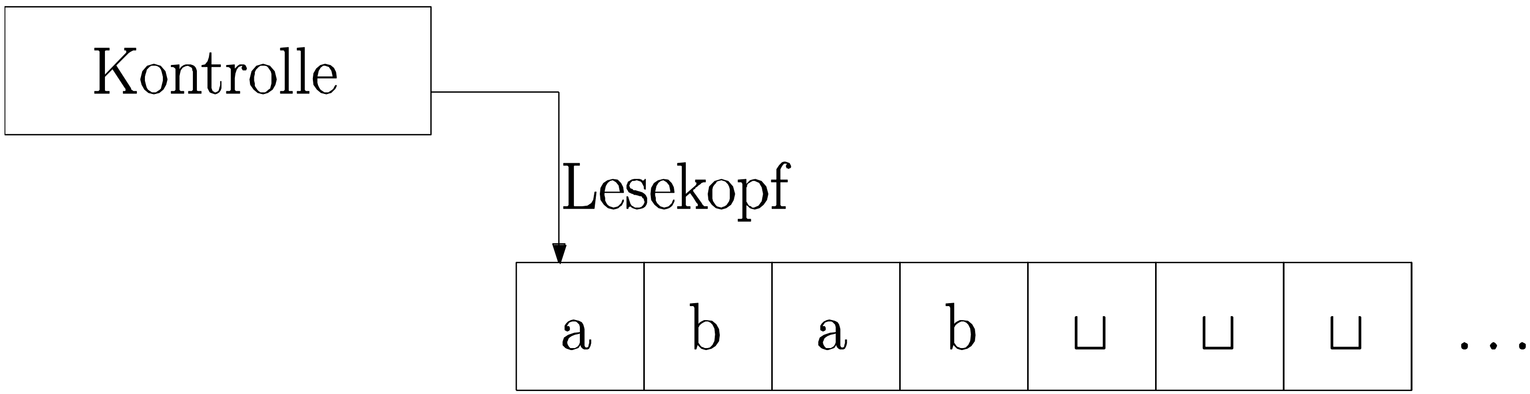
- Einfachster Fall: Ist x in L? – Automaten
  - Ist ein Programm syntaktisch korrekt?

- Verschiedene mächtige Grammatiken und Automatentypen:
  - Reguläre Ausdrücke, kontextfreie Grammatiken, kontextsensitive Grammatiken, …

1. Einführung

- **Turingmaschine**
  - Arbeitet auf unbeschränktem Band
  - Eingabe steht zu Beginn am Anfang des Bands
  - Auf dem Rest des Bandes steht t (Blank)
  - Position auf dem Band wird durch den sog. *Lesekopf* beschrieben

Turingmaschine



- Der jeweils nächste Rechenschritt ist eindeutig festgelegt durch den aktuellen Zustand und das aktuell gelesene Zeichen.
- Der Rechenschritt überschreibt das aktuelle Zeichen, bewegt den Kopf nach rechts oder nach links und verändert den Zustand.

Alan Turing

Quelle: rutherfordjournal.org

- Studium in Cambridge
- Entschlüsselung von Enigma-Verschlüsselungen
- Professor in Manchester
- Nach Alan Turing wird der renommierteste Preis in der Informatik benannt
- „The Imitation Game“

1. Einführung

- Teste, ob ein Wort in der folgenden Sprache liegt:
  - $L = \{w\#w \mid w \text{ in } \{0,1\}^*\}$

- **Vorgehensweise:**
  - Von links nach rechts über das Wort laufen
  - Erstes Zeichen links merken und markieren
  - Erstes Zeichen rechts von # vergleichen und markieren
  - Für alle Zeichen wiederholen bis Blank erreicht
  - Rechts dürfen dann nur noch Blanks folgen
  - Falls Zeichen an einer Stelle nicht übereinstimmen ablehnen, sonst am Ende akzeptieren

1. Einführung

$0 \ 1 \ 1 \ 0 \ 0 \ 0 \ \# \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \ t$ ...

$x \ 1 \ 1 \ 0 \ 0 \ 0 \ \# \ 0 \ 1 \ 1 \ 0 \ 0 \ 0 \ t$ ...

$x \ 1 \ 1 \ 0 \ 0 \ 0 \ \# \ x \ 1 \ 1 \ 0 \ 0 \ 0 \ t$ ...

$x \ 1 \ 1 \ 0 \ 0 \ 0 \ \# \ x \ 1 \ 1 \ 0 \ 0 \ 0 \ t$ ...

$x \ x \ 1 \ 0 \ 0 \ 0 \ \# \ x \ 1 \ 1 \ 0 \ 0 \ 0 \ t$ ...

$x \ x \ x \ x \ x \ x \ \# \ x \ x \ x \ x \ x \ x \ t$ ... accept

1. Einführung

Definition

Eine *(deterministische 1-Band) Turingmaschine* (DTM) wird beschrieben durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$.

Dabei sind Q, Σ, Γ endliche, nichtleere Mengen und es gilt:

- Σ ist Teilmenge von Γ
- t in $\Gamma \setminus \Sigma$ ist das *Blanksymbol* (auch ⊔)
- Q ist die *Zustandsmenge*
- Σ ist das *Eingabealphabet*
- Γ ist das *Bandalphabet*
- q_0 in Q ist der *Startzustand*
- q_{accept} in Q ist der akzeptierende Endzustand
- q_{reject} in Q ist der ablehnende Endzustand
- $\delta: Q \setminus \{q_{accept}, q_{reject}\} \times \Gamma \to Q \times \Gamma \times \{L, R\}$ ist die (partielle) *Übergangsfunktion*. Sie ist für kein Argument aus $\{q_{accept}, q_{reject}\} \times \Gamma$ definiert.

1. Einführung

- Initial:
  - Eingabe steht links auf dem Band
  - Der Rest des Bands ist leer
  - Kopf befindet sich ganz links
- Berechnungen finden entsprechend der Übergangsfunktion statt
- Wenn der Kopf sich am linken Ende befindet und nach links bewegen soll, bleibt er an seiner Position
- Wenn q_{accept} oder q_{reject} erreicht wird, ist die Bearbeitung beendet

1. Einführung

Momentaufnahme einer Turingmaschine:

- Bei Bandinschrift uv (dabei beginnt u am linken Ende des Bandes und hinter v stehen nur Blanks)
- Zustand q
- Kopf auf erstem Zeichen von v

Konfiguration $C = uqv$

1. Einführung

- Gegeben: Konfigurationen C_1, C_2
- Wir sagen: **Konfiguration $\boldsymbol{C_1}$ führt zu $\boldsymbol{C_2}$,** falls die TM von C_1 in einem Schritt zu C_2 übergehen kann

Formal:

- Seien a, b, c in Γ, u, v in Γ^* und Zustände q_i, q_j gegeben
- Wir sagen:
  - uaq_ibv führt zu uq_jacv, falls $\delta(q_i, b) = (q_j, c, L)$ und
  - uaq_ibv führt zu $uacq_jv$, falls $\delta(q_i, b) = (q_j, c, R)$

1. Einführung

- Startkonfiguration:
  - $q_0 w$, wobei w die Eingabe ist
- Akzeptierende Konfiguration:
  - Konfigurationen mit Zustand q_{accept}
- Ablehnende Konfiguration:
  - Konfigurationen mit Zustand q_{reject}
- Haltende Konfiguration:
  - akzeptierende oder ablehnende Konfigurationen

1. Einführung

Definition

Eine Turingmaschine M akzeptiert eine Eingabe w, falls es eine Folge von Konfigurationen $C_1, C_2, \dots, C_k$ gibt, sodass

1. C_1 ist die Startkonfiguration von M bei Eingabe w
2. C_i führt zu C_{i+1}
3. C_k ist eine akzeptierende Konfiguration

- Die von M akzeptierten Worte bilden die von M akzeptierte Sprache $L(M)$.
- Eine Turingmaschine M entscheidet die Sprache $L(M)$, wenn jede Eingabe in einer haltenden Konfiguration C_k resultiert.

1. Einführung

Definition

- Eine Sprache L heißt **rekursiv aufzählbar,** falls es eine Turingmaschine M gibt, die L *akzeptiert.*
- Eine Sprache L heißt **rekursiv** oder **entscheidbar,** falls es eine Turingmaschine M gibt, die L *entscheidet.*

1. Einführung

Gesucht: Turingmaschine, die $L = \{0^{2^n} \mid n \geq 0\}$ entscheidet

Arbeitsweise:

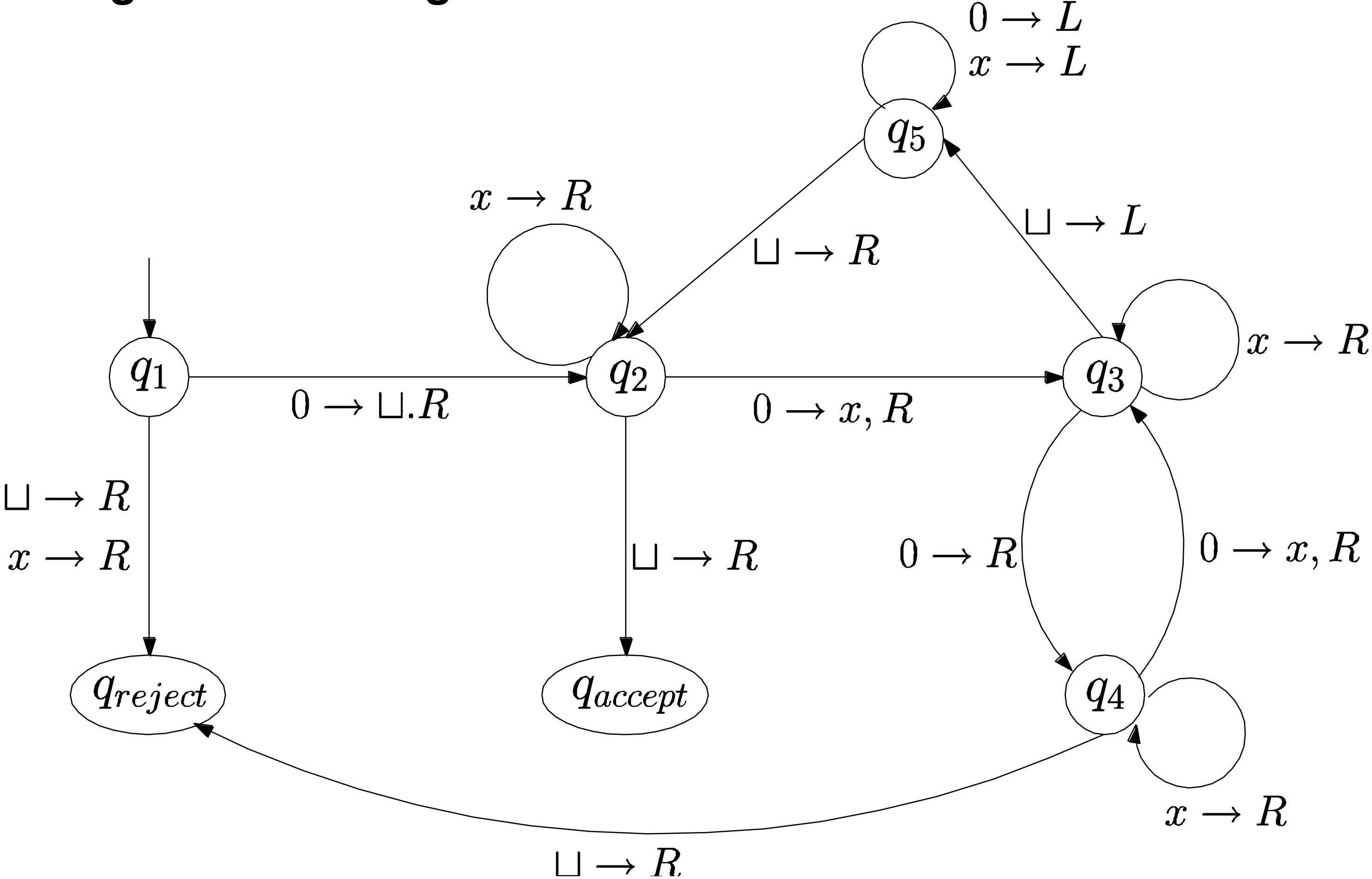
1. Gehe von links nach rechts über die Eingabe und ersetze jede zweite 0 durch x
2. Wenn nur eine 0 auf dem Band ist, akzeptiere
3. Falls die Anzahl der 0en ungerade ist, lehne ab
4. Bewege den Kopf zurück an das linke Ende

1. Einführung

Definition der Turingmaschine

- $Q = \{q_1, q_2, q_3, q_4, q_5, q_{accept}, q_{reject}\}$
- $\Sigma = \{0\}$
- $\Gamma = \{0, x, t\}$
- q_1: Startzustand
- q_{accept}: Akzeptierender Endzustand
- q_{reject}: Ablehnender Endzustand
- δ: Übergangsfunktion

Darstellung von δ als Diagramm

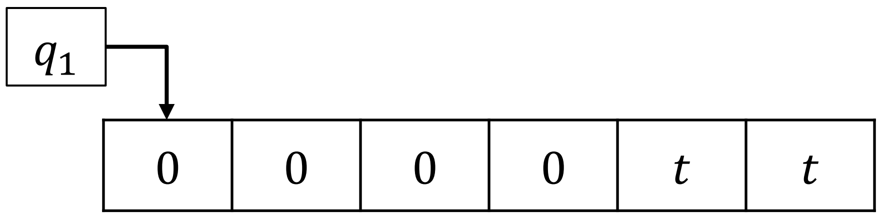


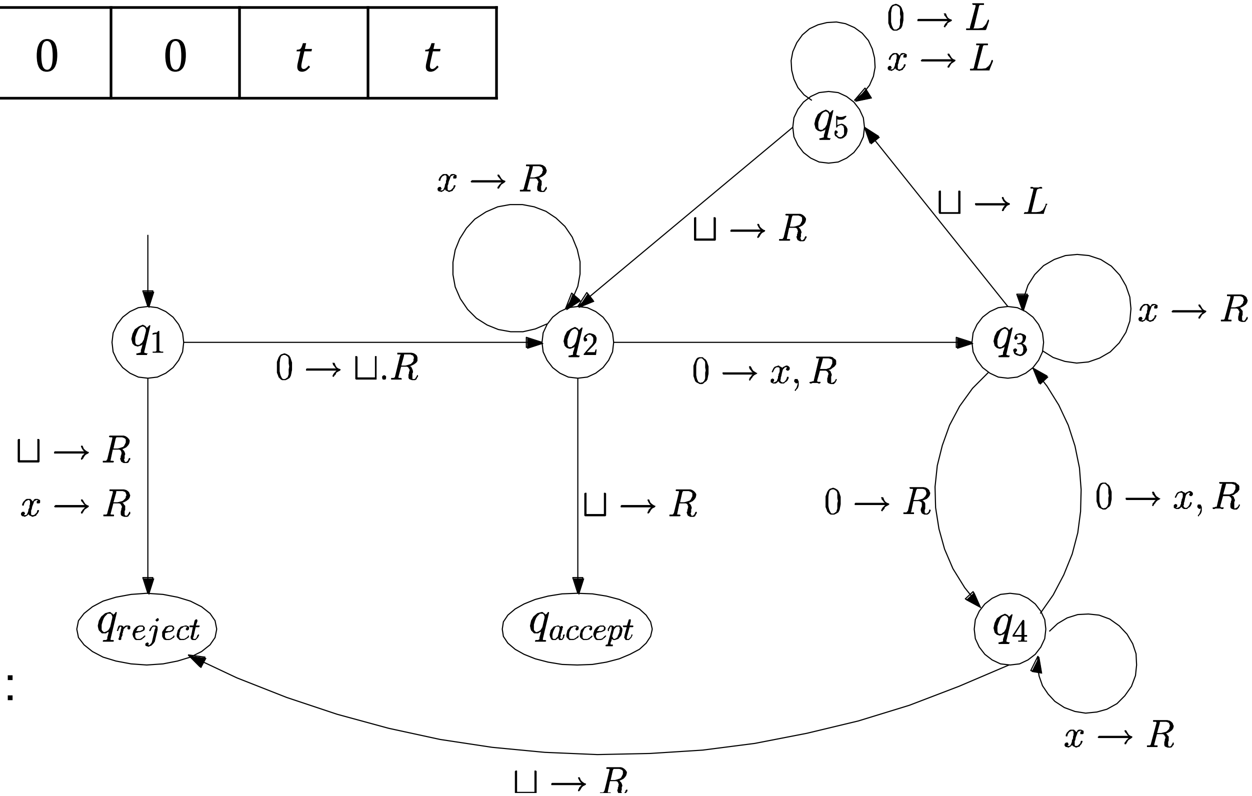
1. Einführung

Darstellung von δ als Tabelle

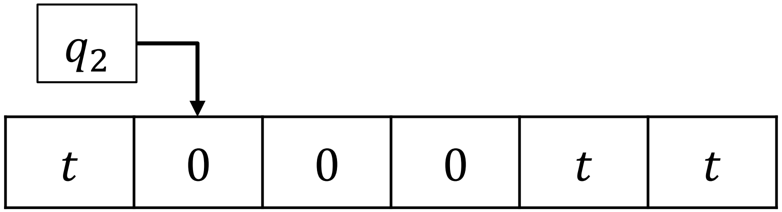
$\boldsymbol{\delta}$	$\mathbf{0}$	$\boldsymbol{x}$	$\boldsymbol{t}$
q_1	(q_2, t, R)	(q_{reject}, x, R)	(q_{reject}, t, R)
q_2	(q_3, x, R)	(q_2, x, R)	(q_{accept}, t, R)
q_3	$(q_4, 0, R)$	(q_3, x, R)	(q_5, t, L)
q_4	(q_3, x, R)	(q_4, x, R)	(q_{reject}, t, R)
q_5	$(q_5, 0, L)$	(q_5, x, L)	(q_2, t, R)

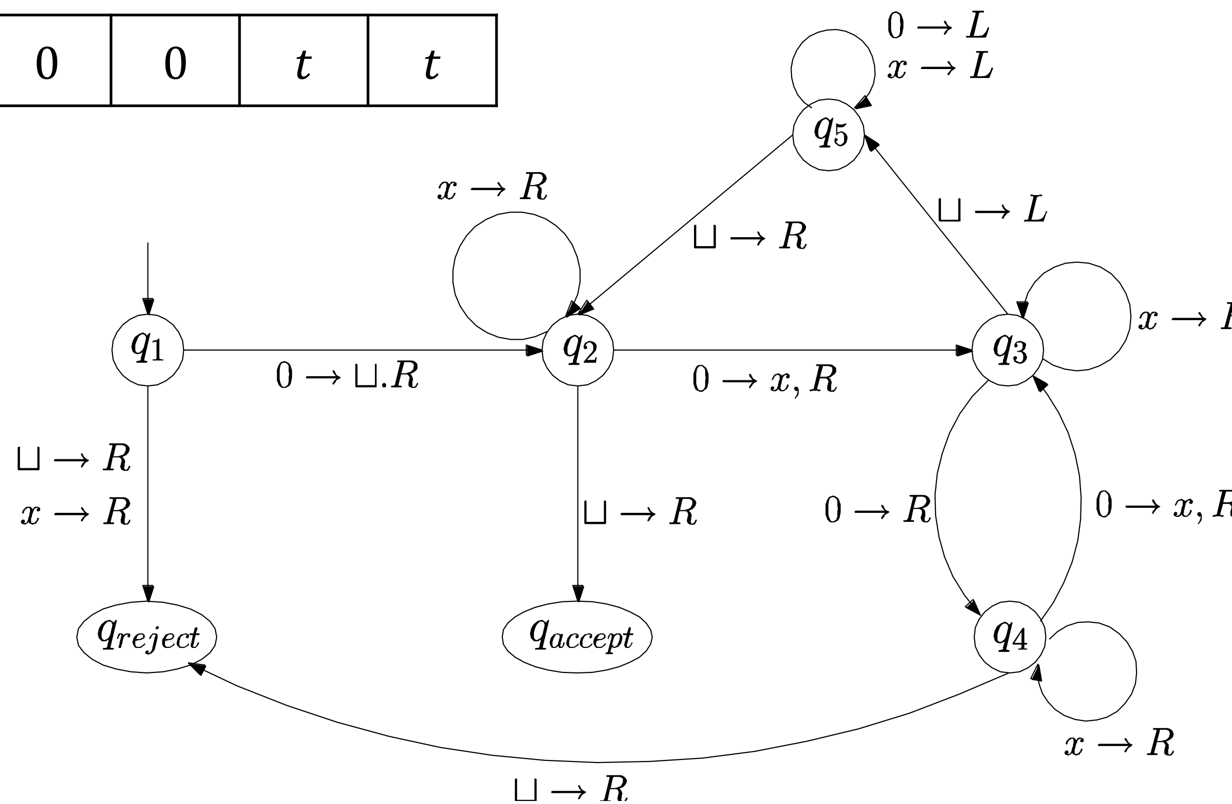
1. Einführung





- Beispiel für das Wort: 0000
- Startkonfiguration: $q_1 0000$

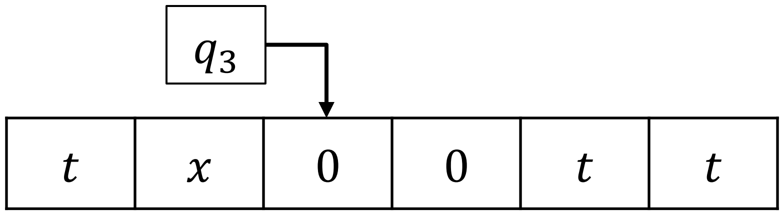


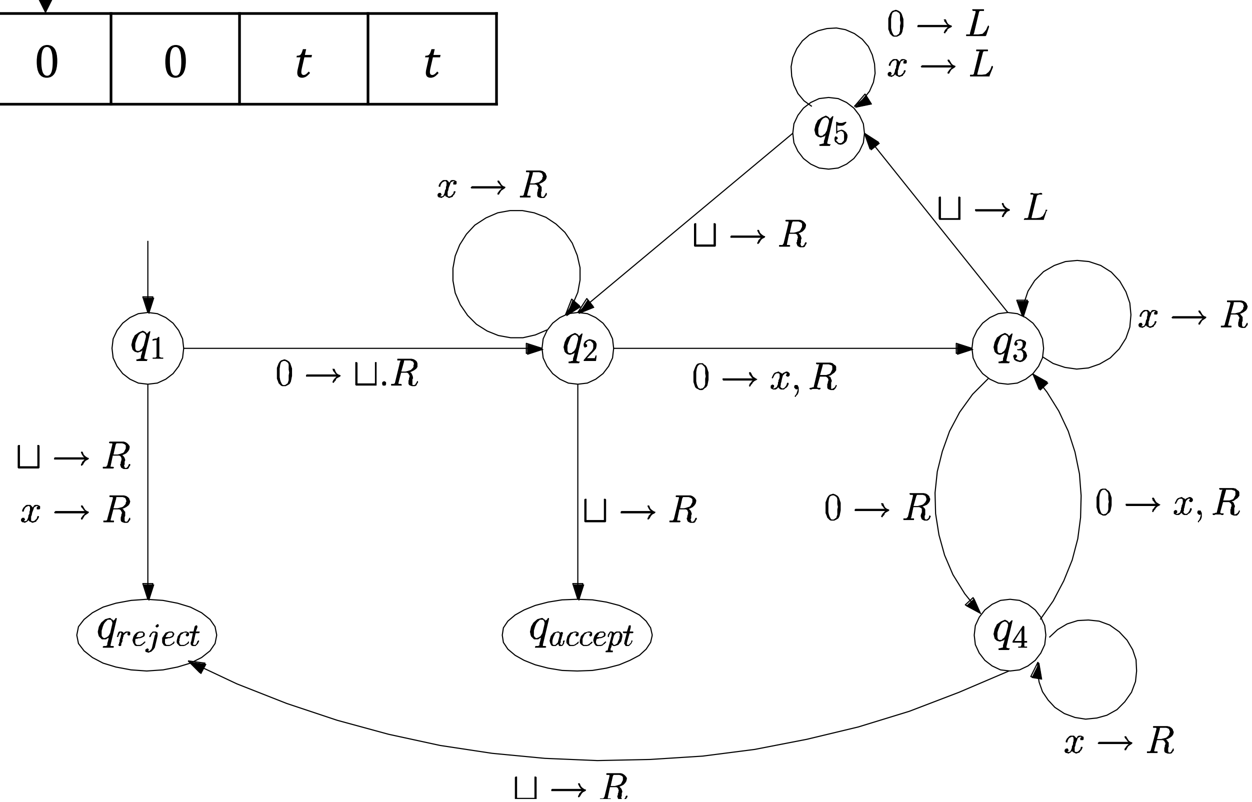


- Beispiel für das Wort: 0000

- Konfiguration: tq_2000

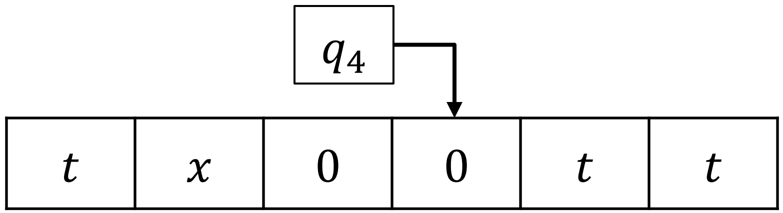
1. Einführung

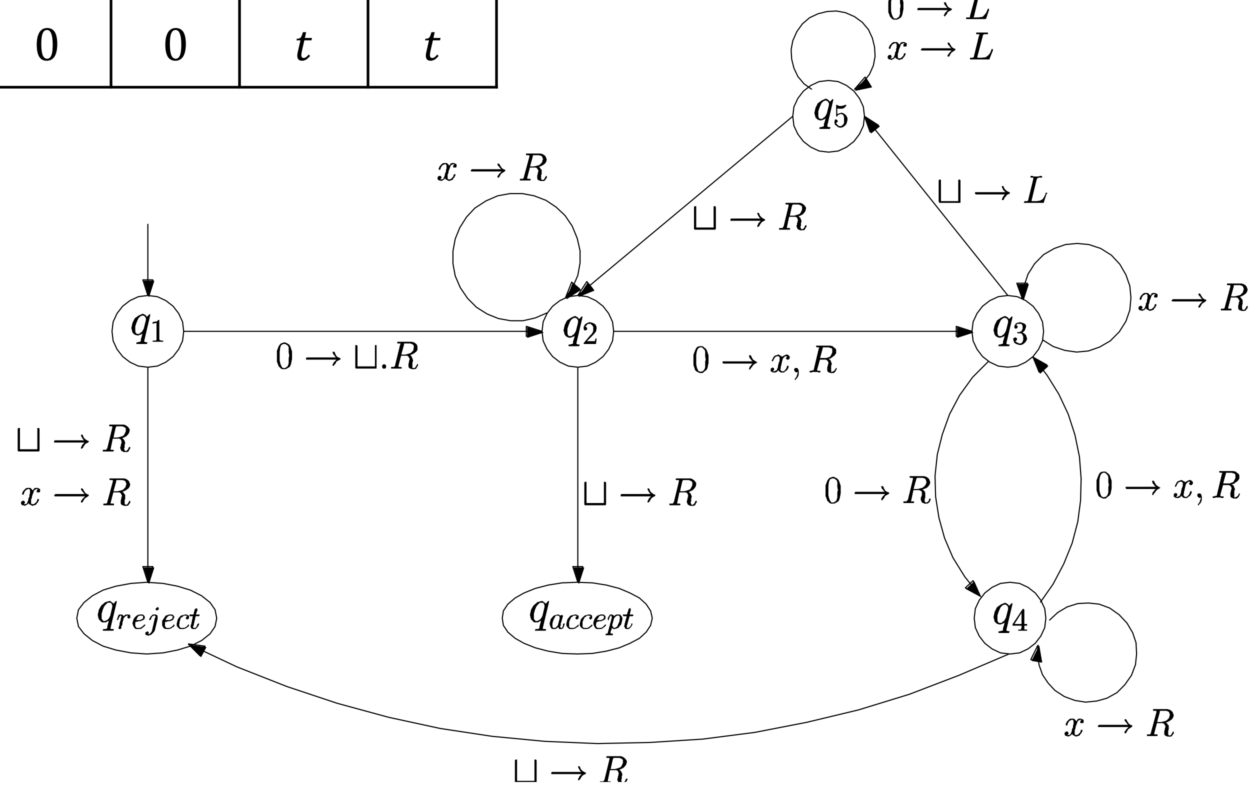




- Beispiel für das Wort: 0000

- Konfiguration: txq_300

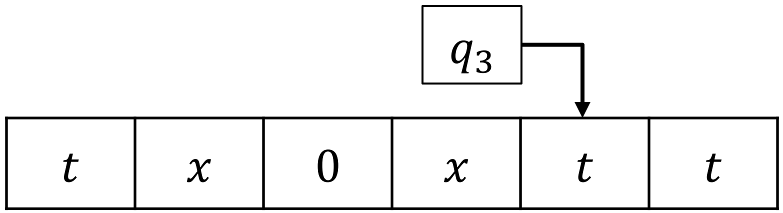


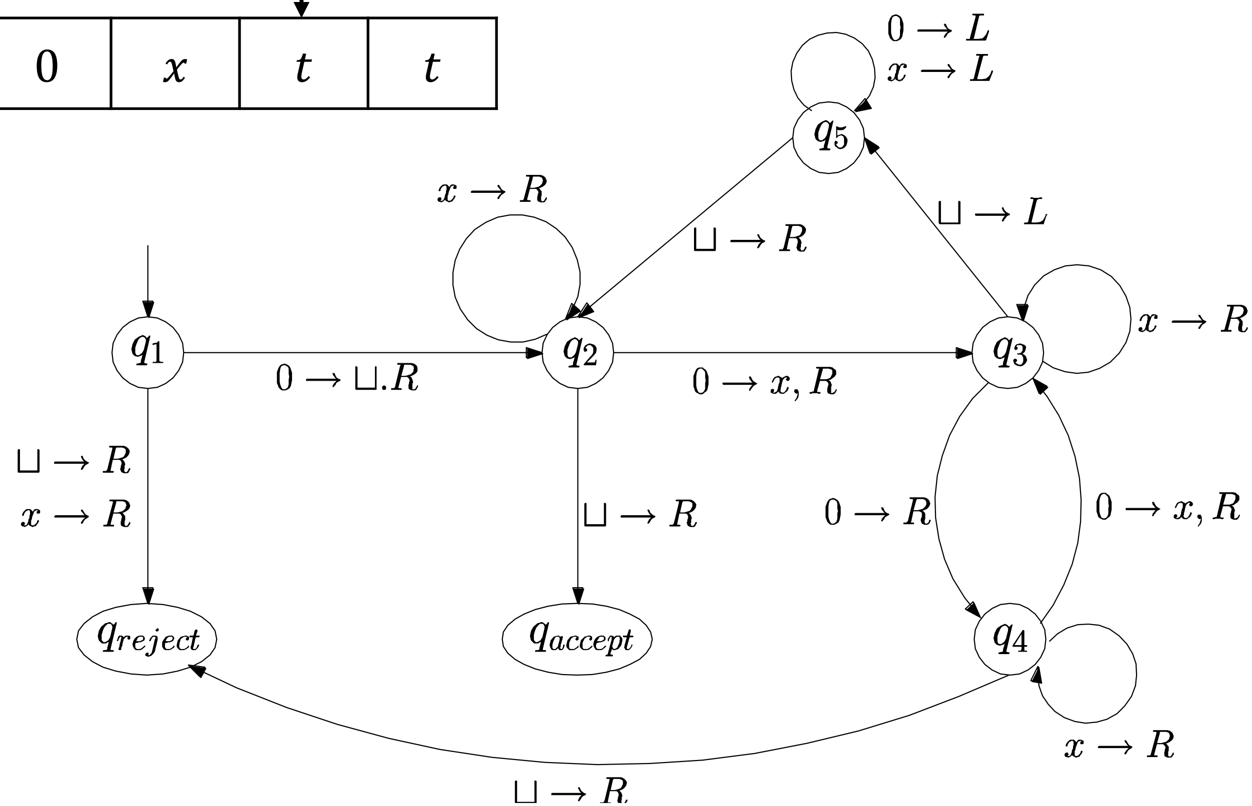


- Beispiel für das Wort: 0000

- Konfiguration: $tx0q_40$

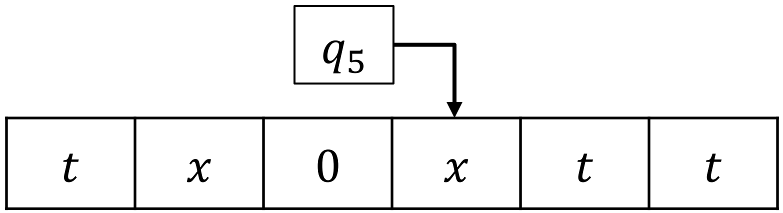
1. Einführung

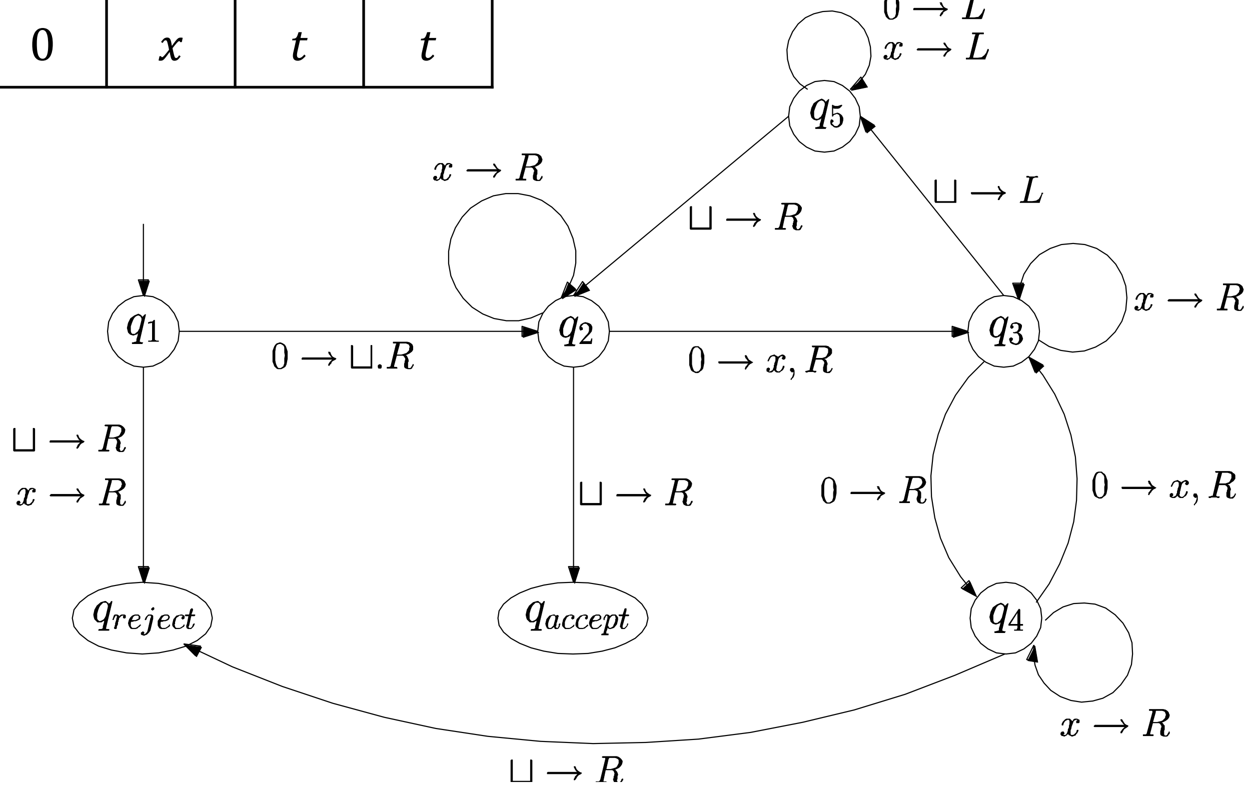




- Beispiel für das Wort: 0000

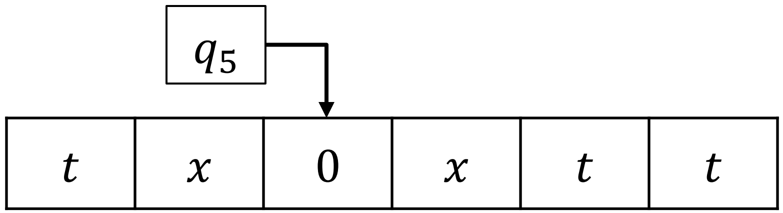
- Konfiguration: $tx0xq_3t$



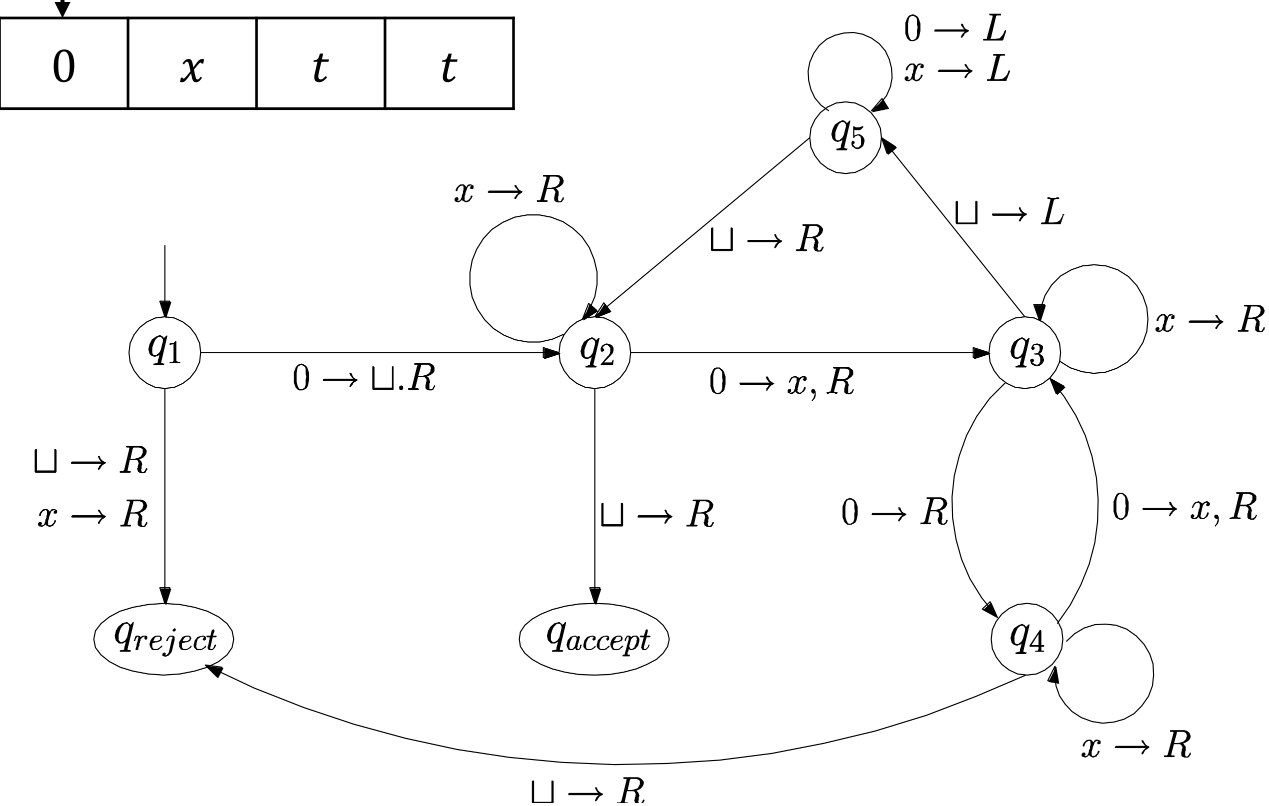


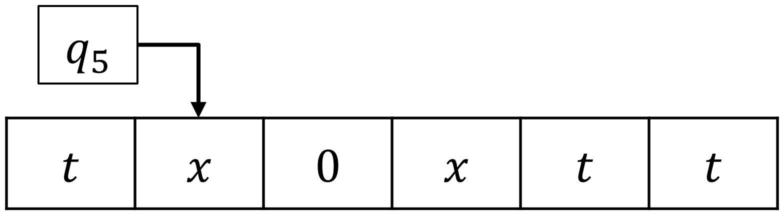
- Beispiel für das Wort: 0000

1. Einführung

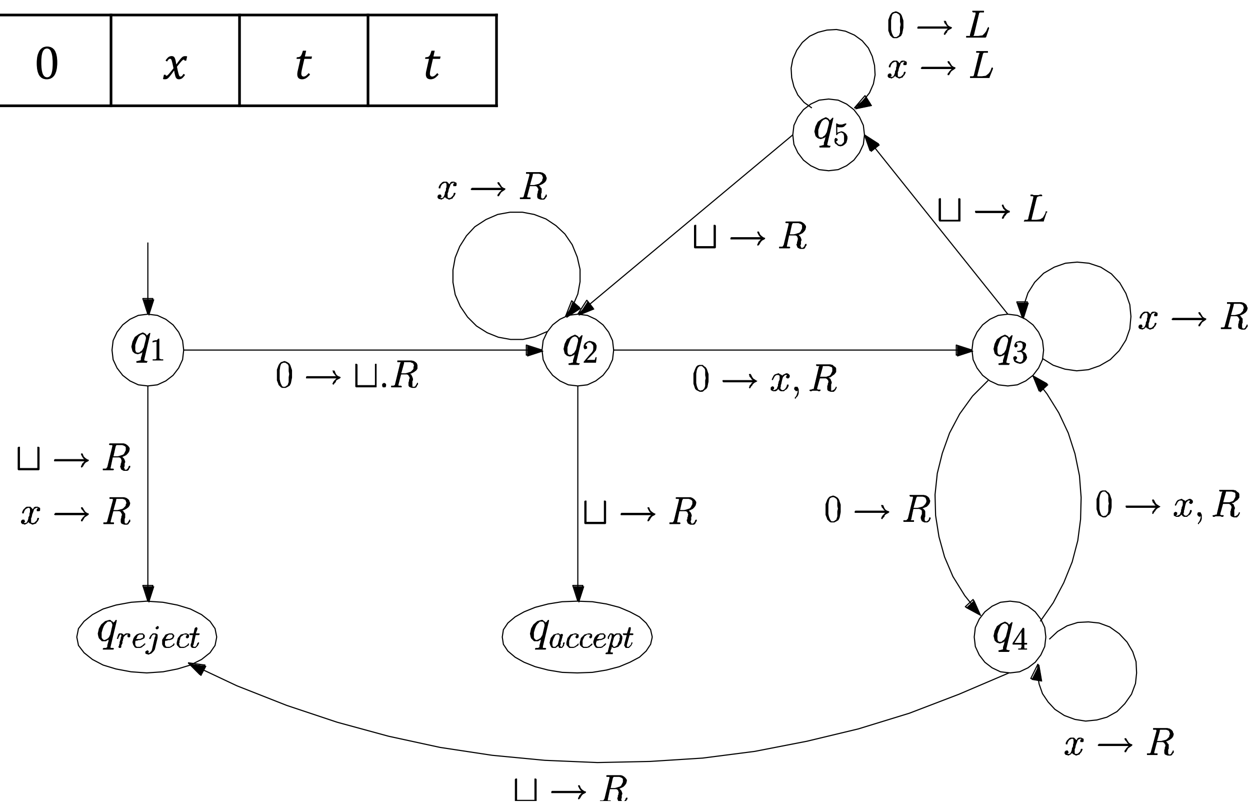


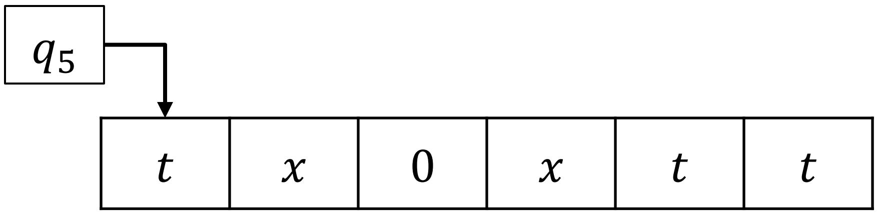
- Beispiel für das Wort: 0000

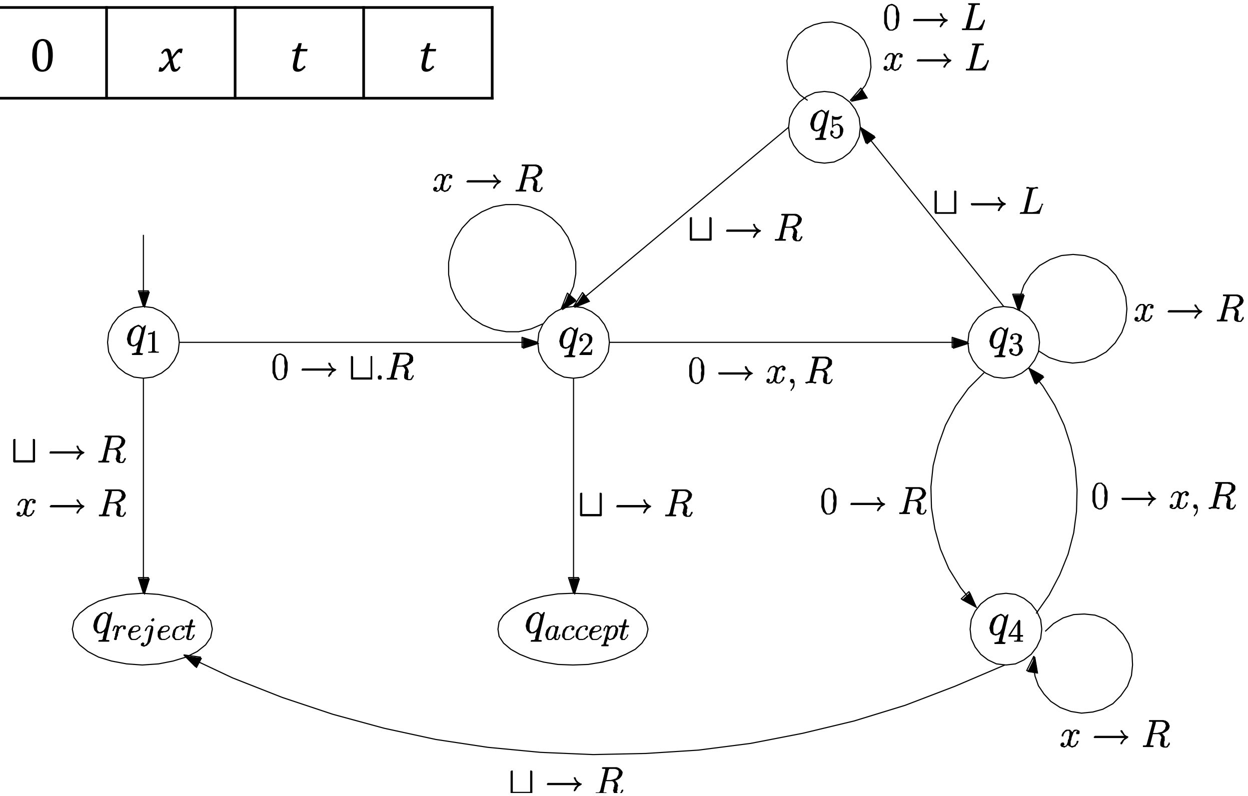




- Beispiel für das Wort: 0000

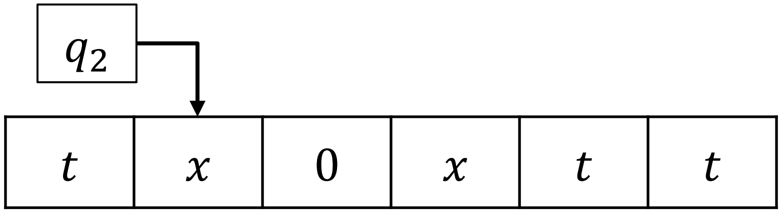




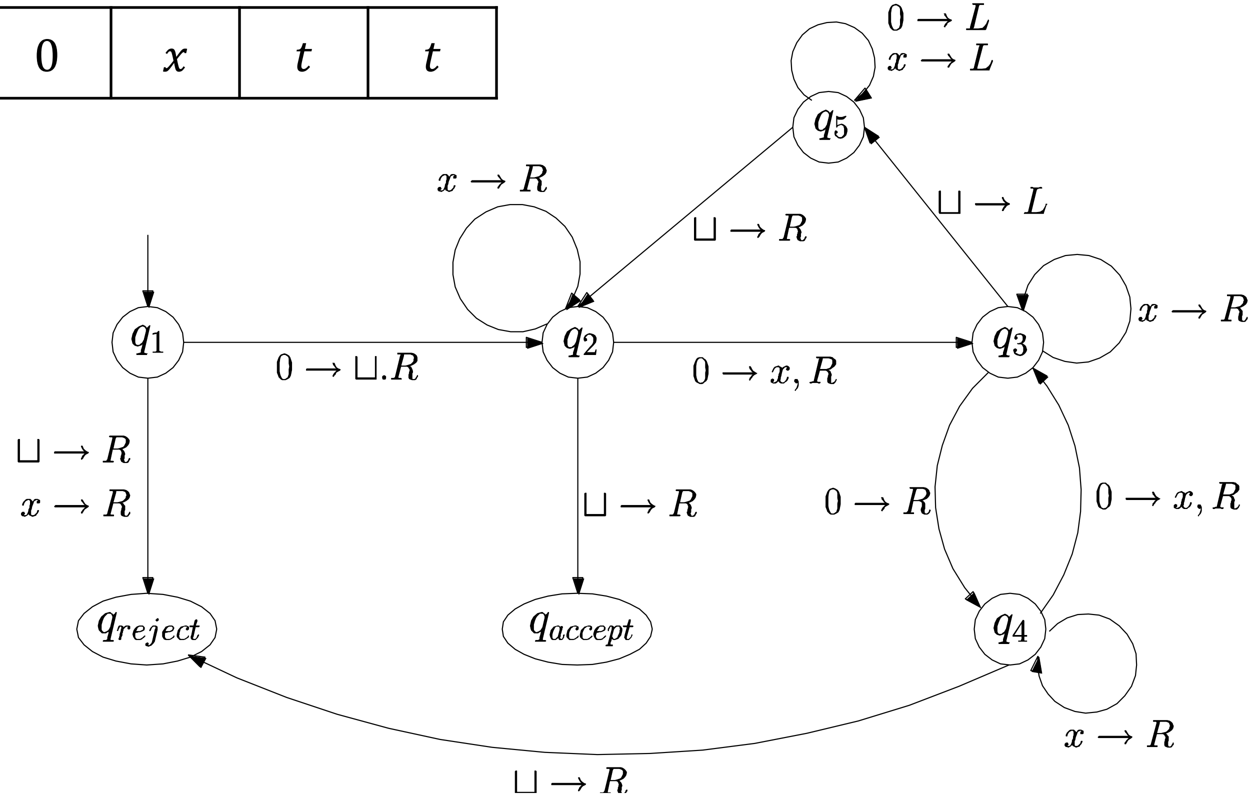


- Beispiel für das Wort: 0000

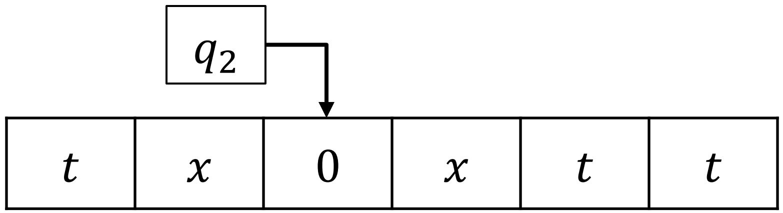
1. Einführung



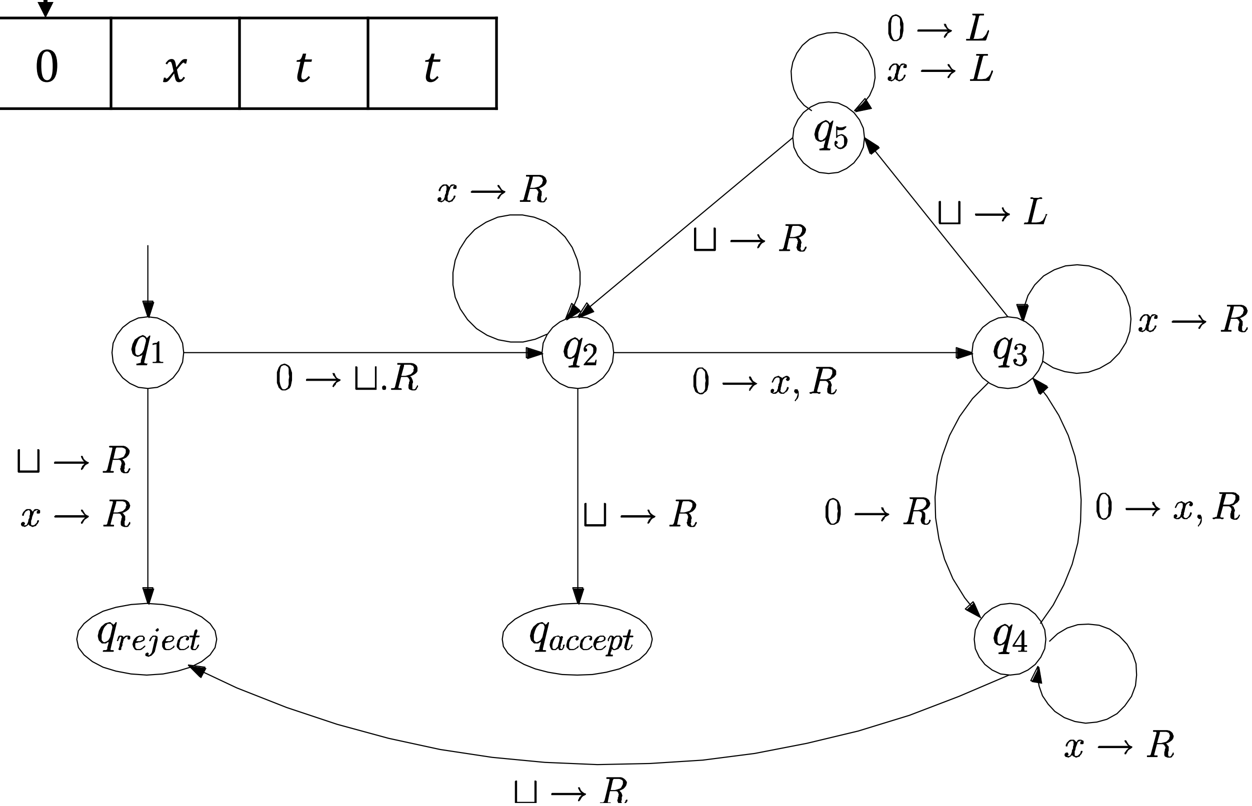
- Beispiel für das Wort: 0000

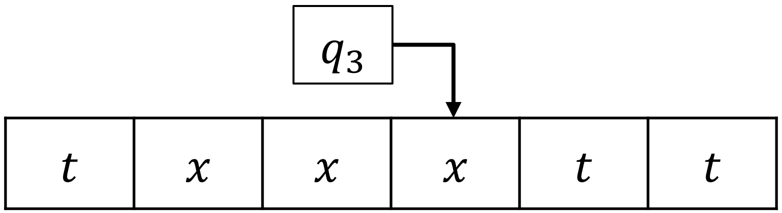


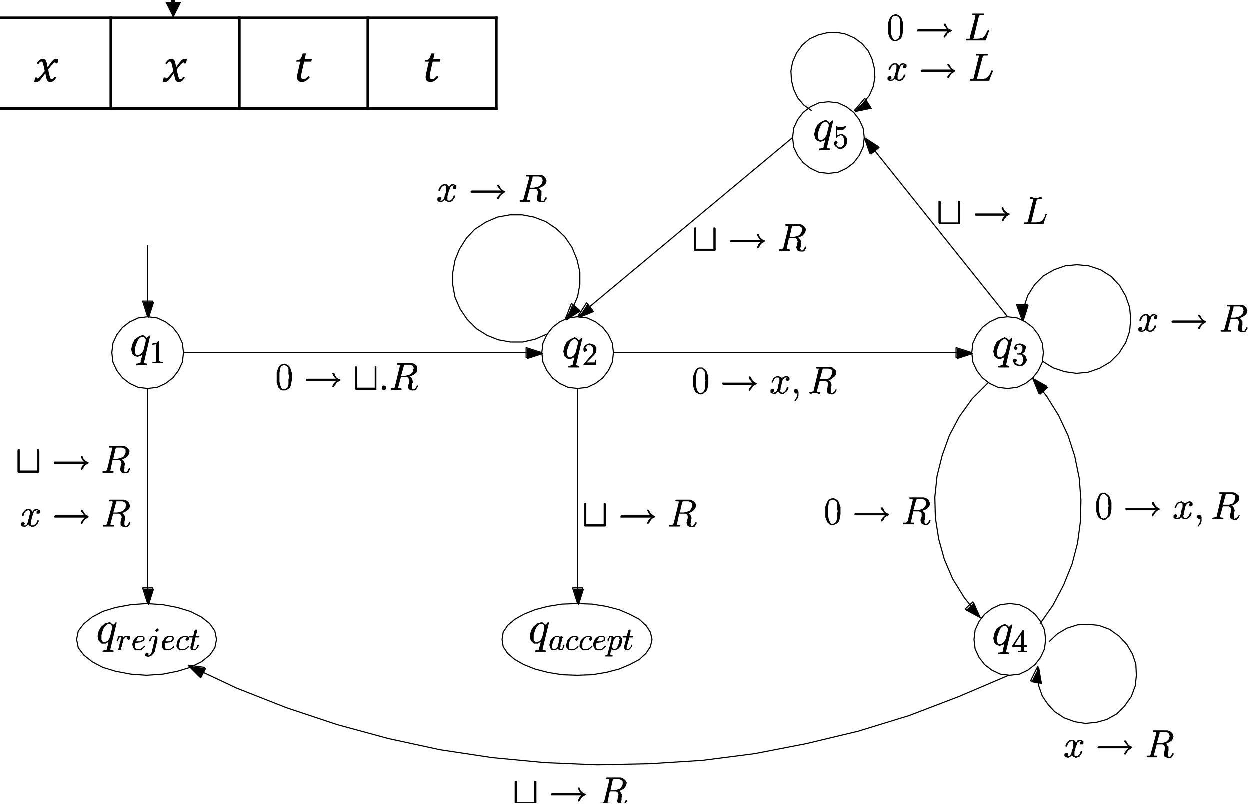
1. Einführung



- Beispiel für das Wort: 0000

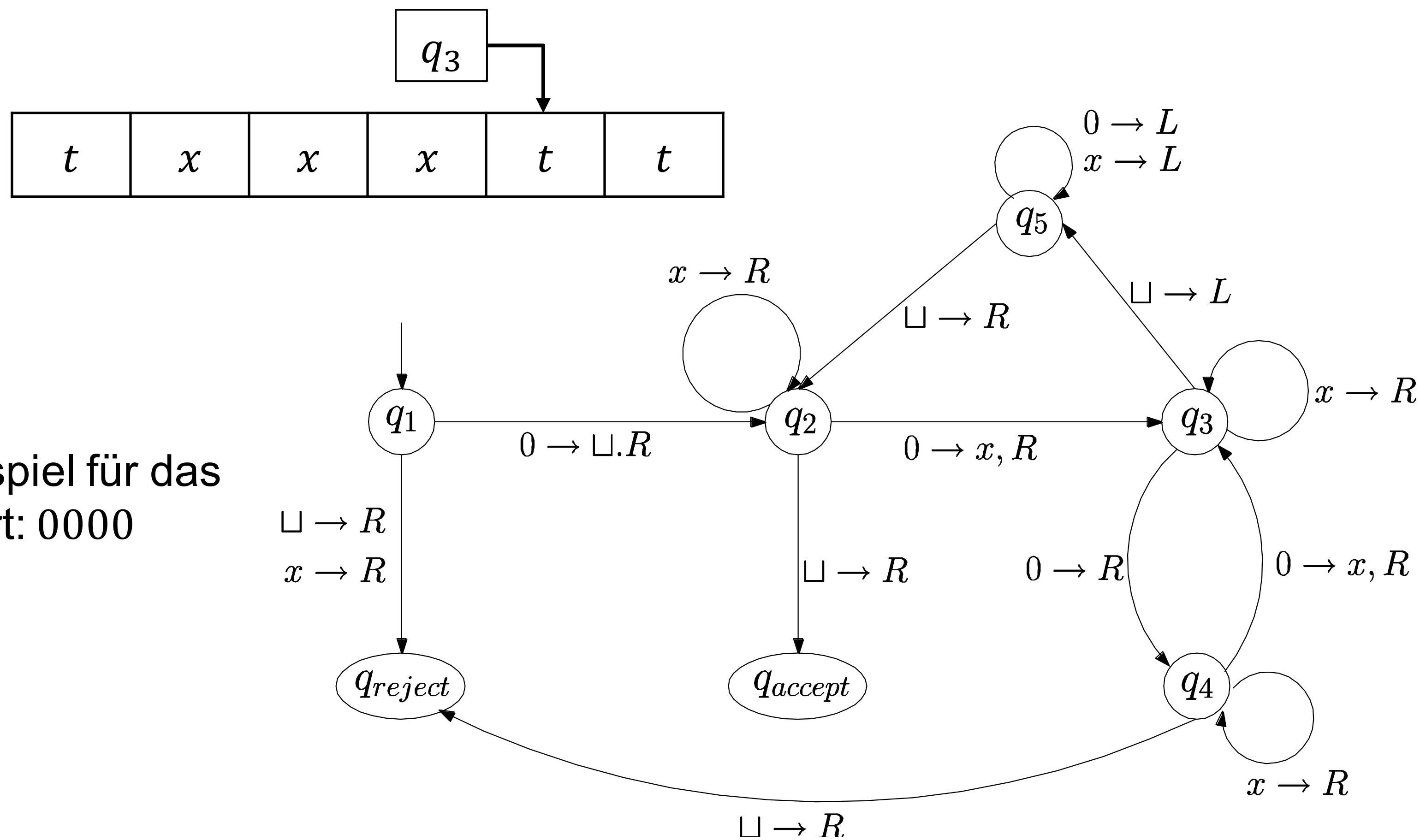






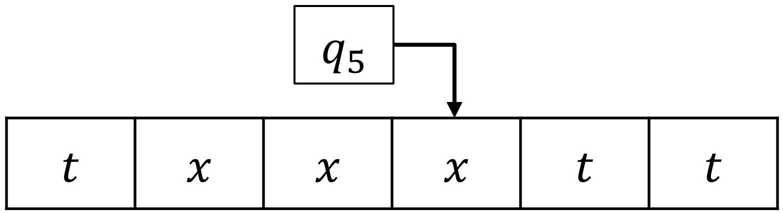
- Beispiel für das Wort: 0000

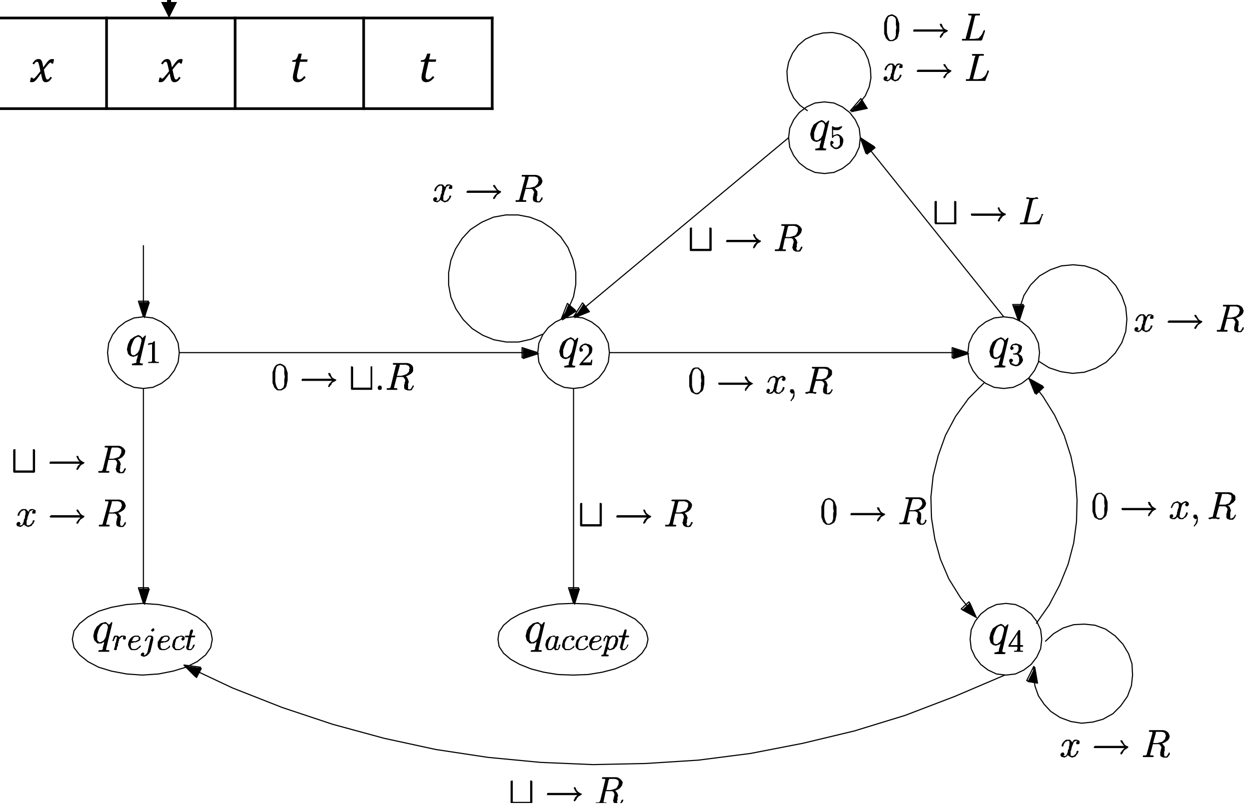
1. Einführung



- Beispiel für das Wort: 0000

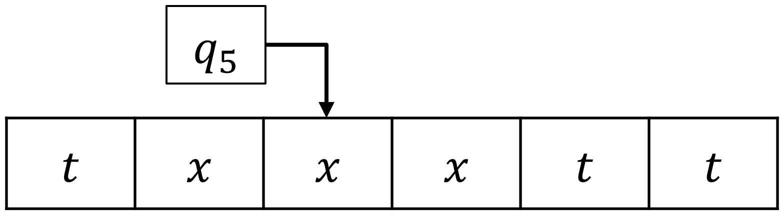
1. Einführung



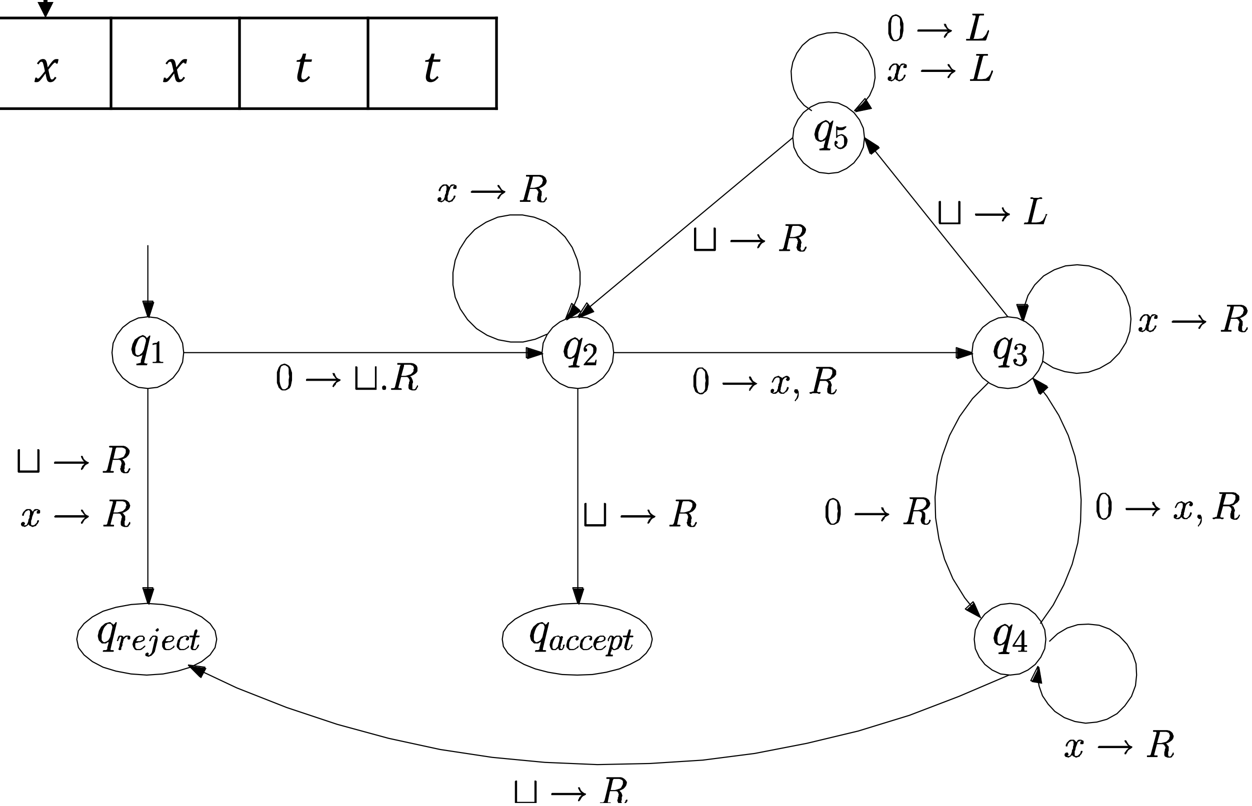


- Beispiel für das Wort: 0000

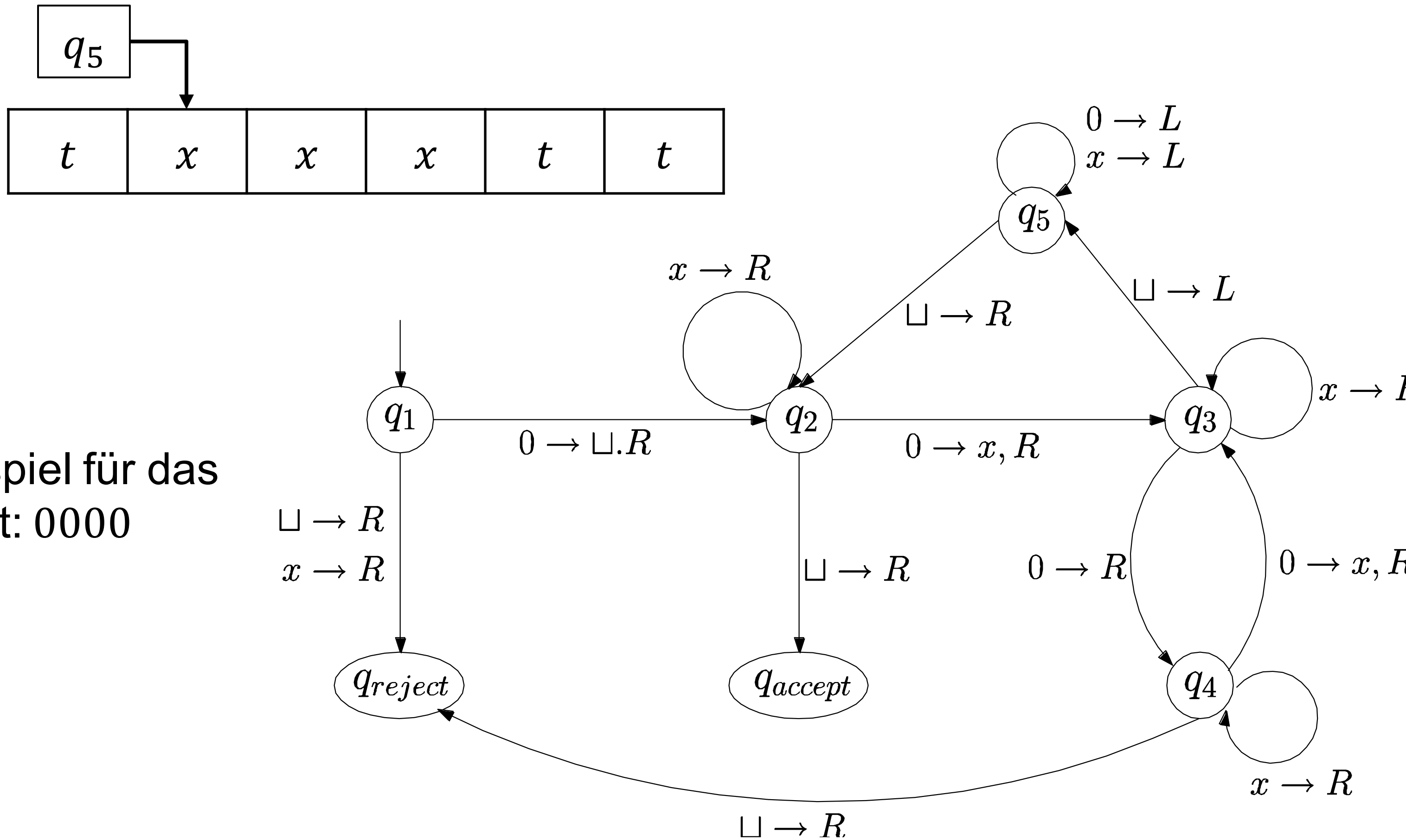
1. Einführung



- Beispiel für das Wort: 0000

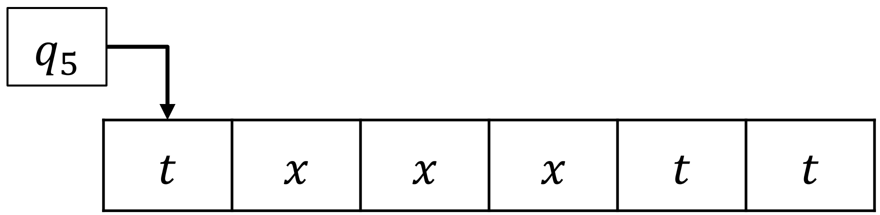


1. Einführung

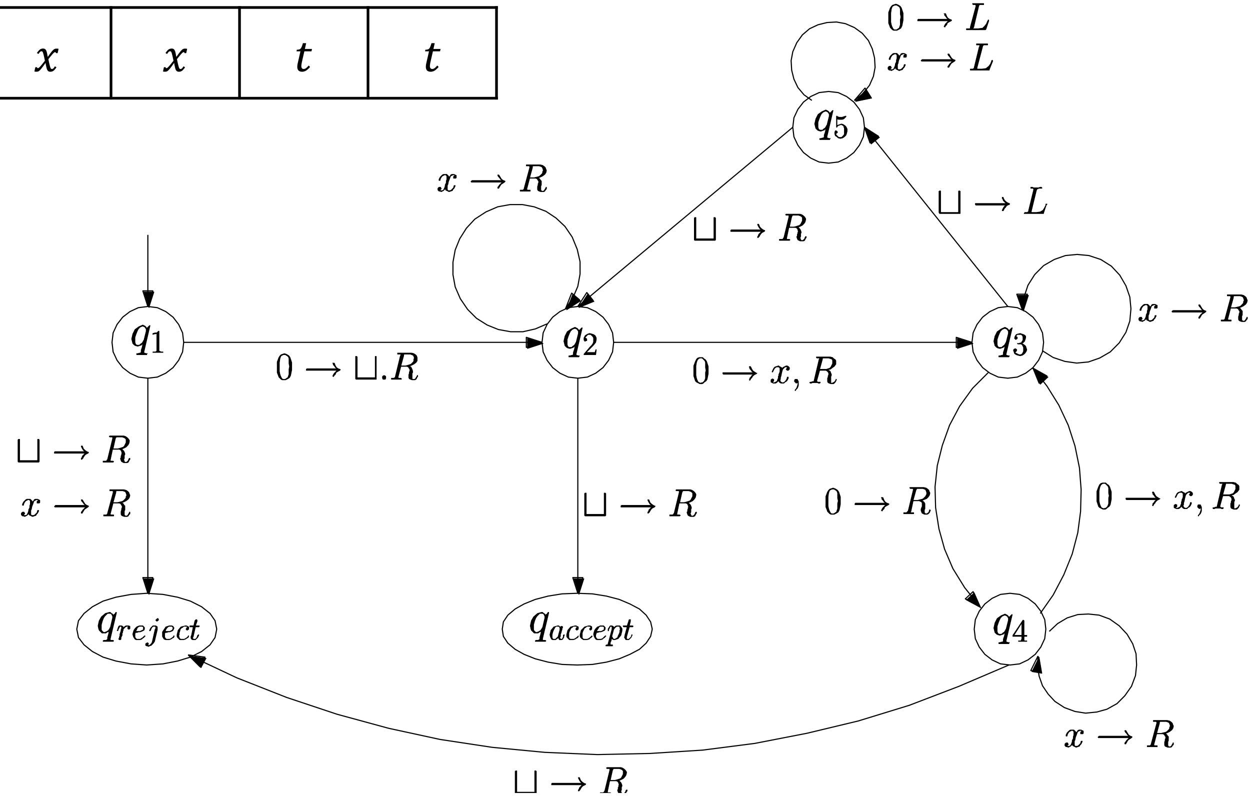


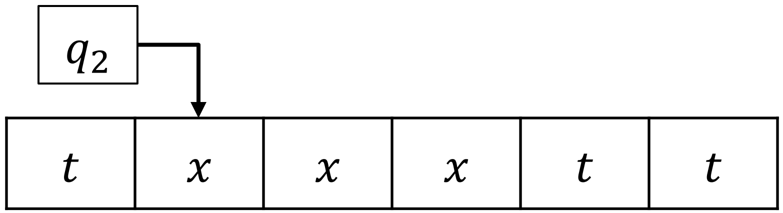
- Beispiel für das Wort: 0000

1. Einführung

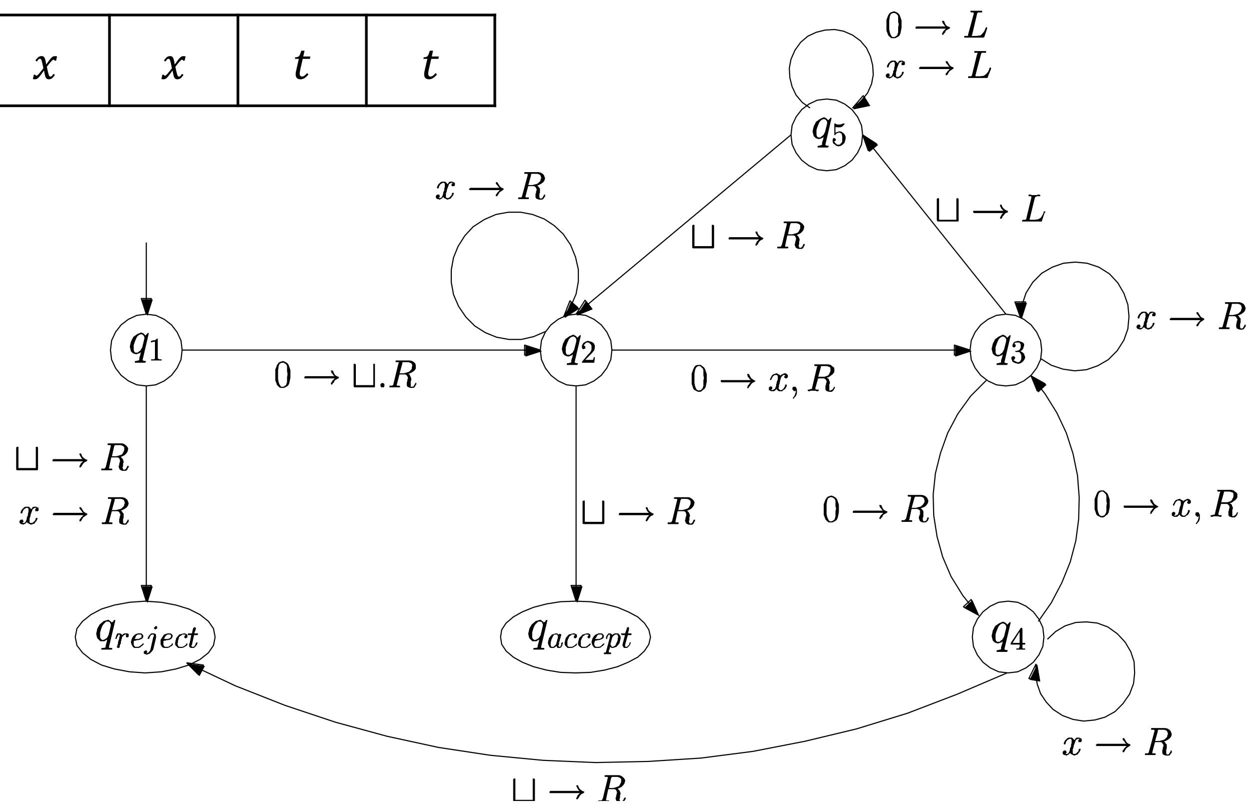


- Beispiel für das Wort: 0000

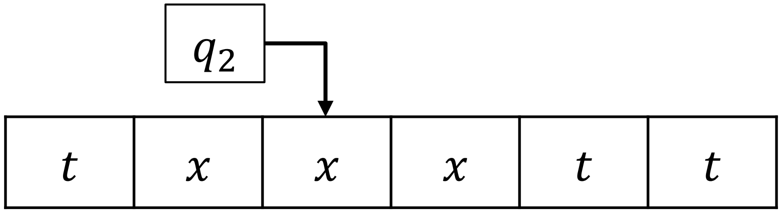


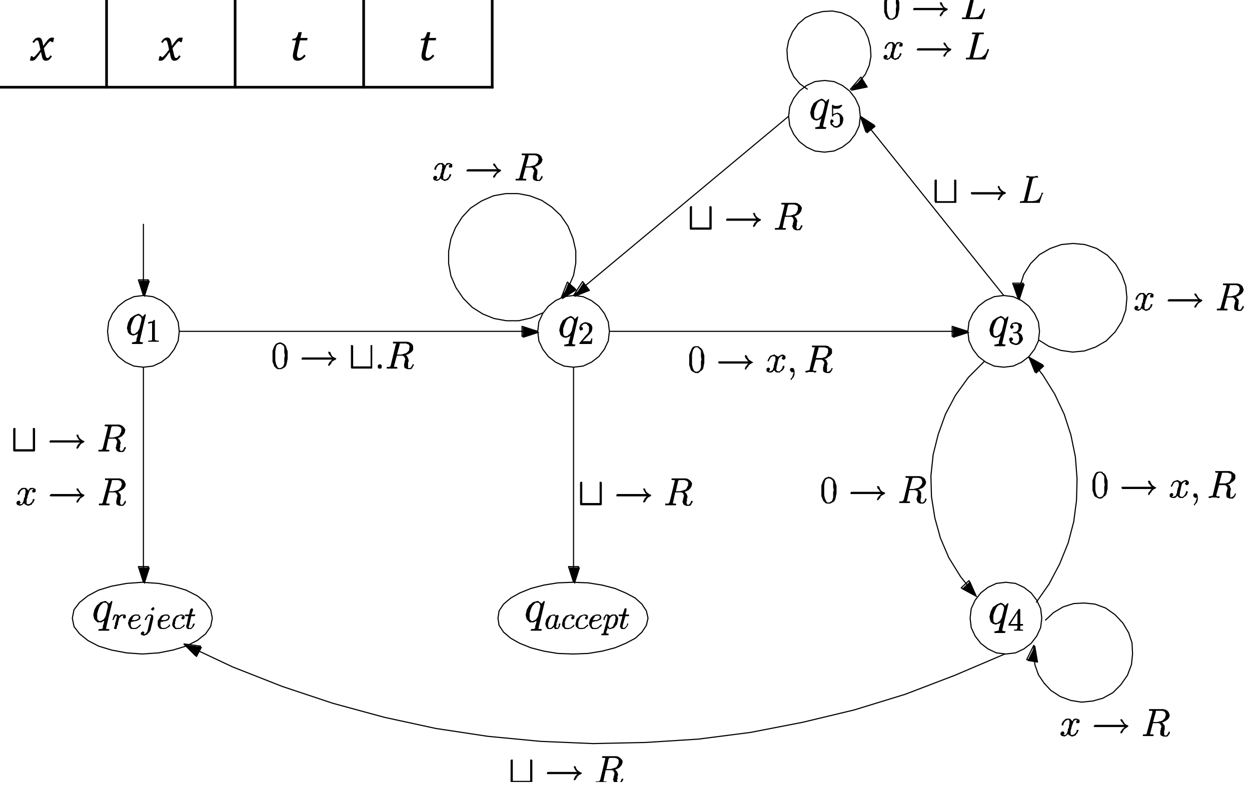


- Beispiel für das Wort: 0000



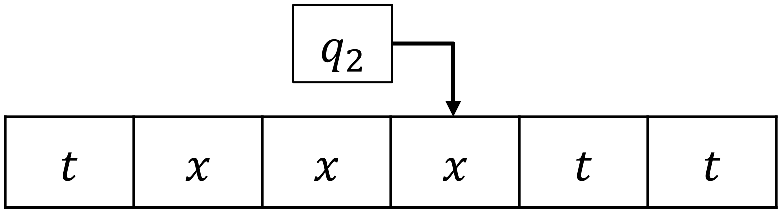
1. Einführung



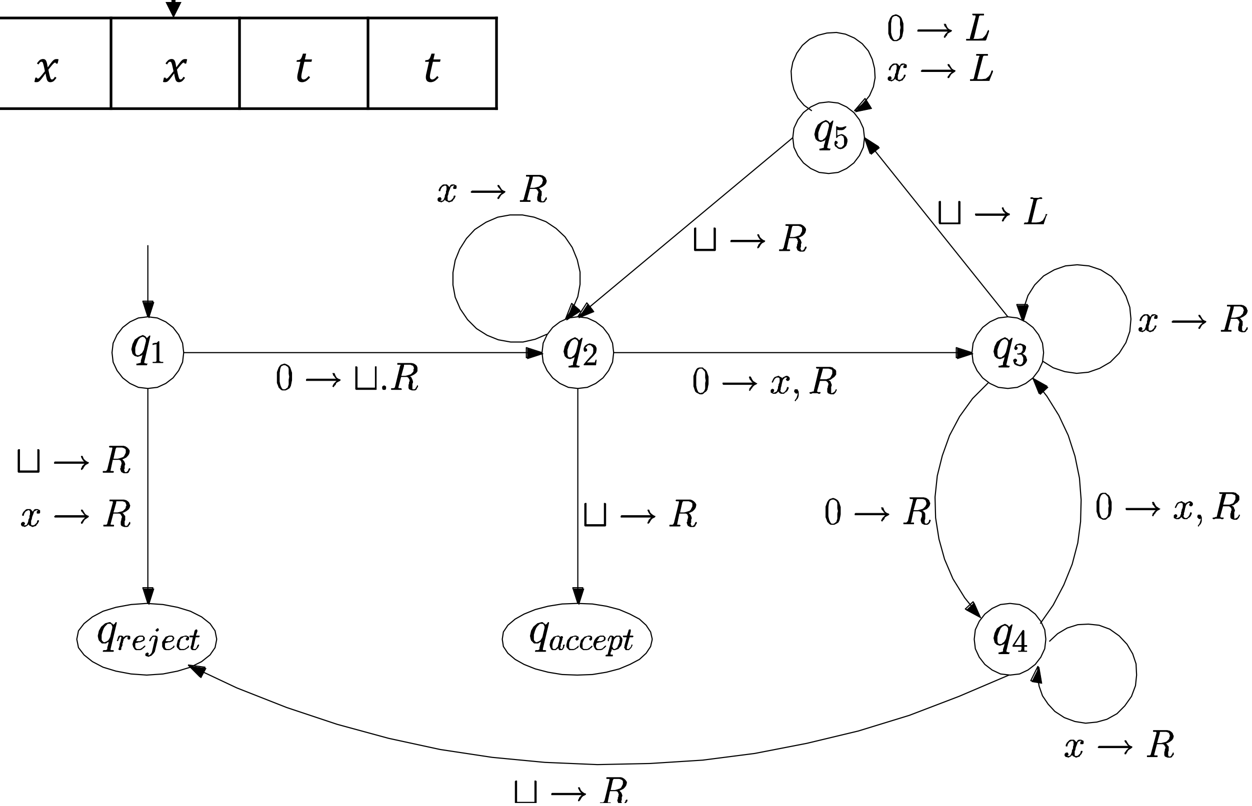


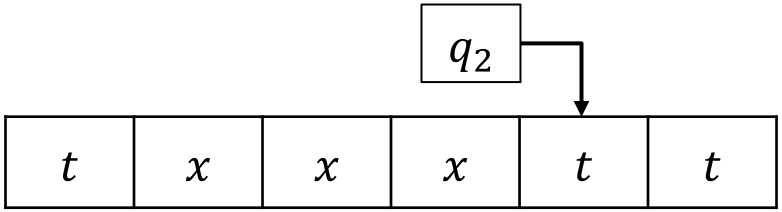
- Beispiel für das Wort: 0000

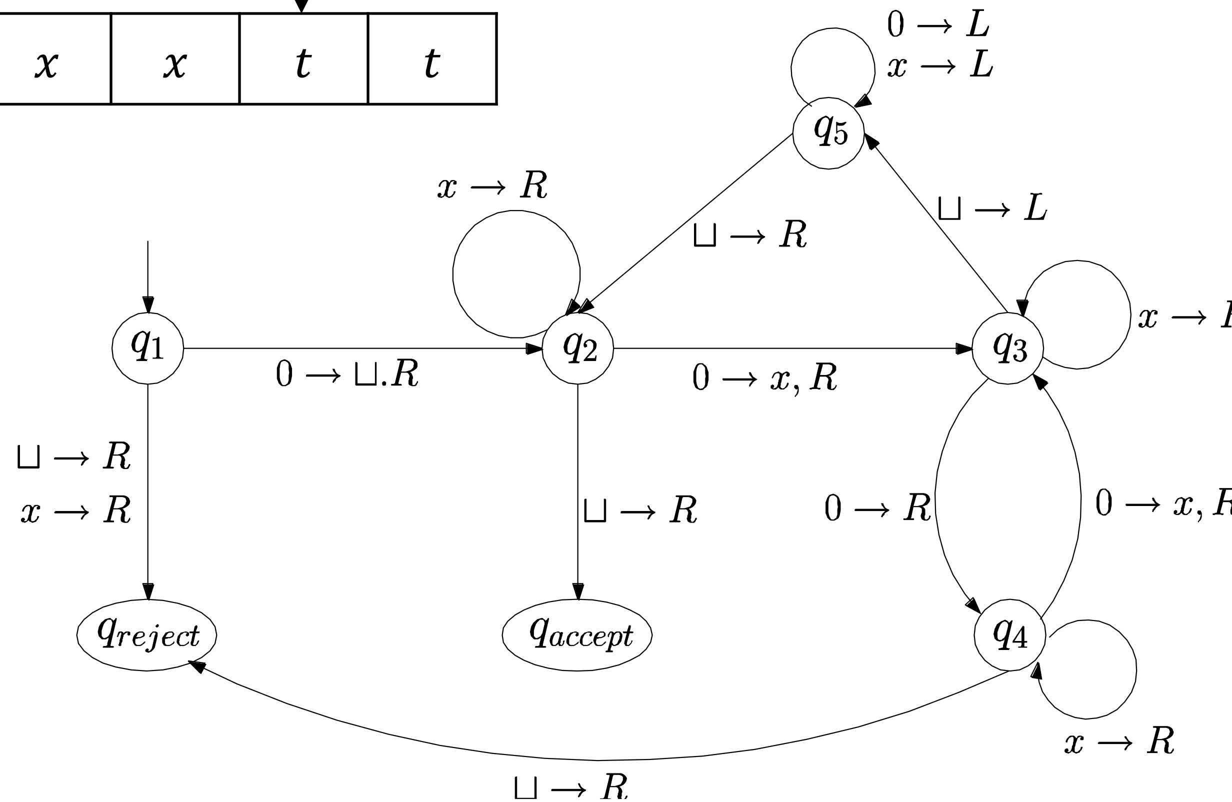
1. Einführung



- Beispiel für das Wort: 0000

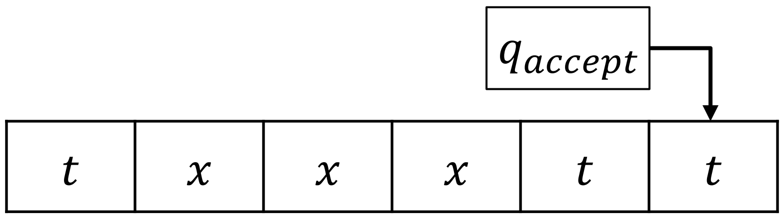


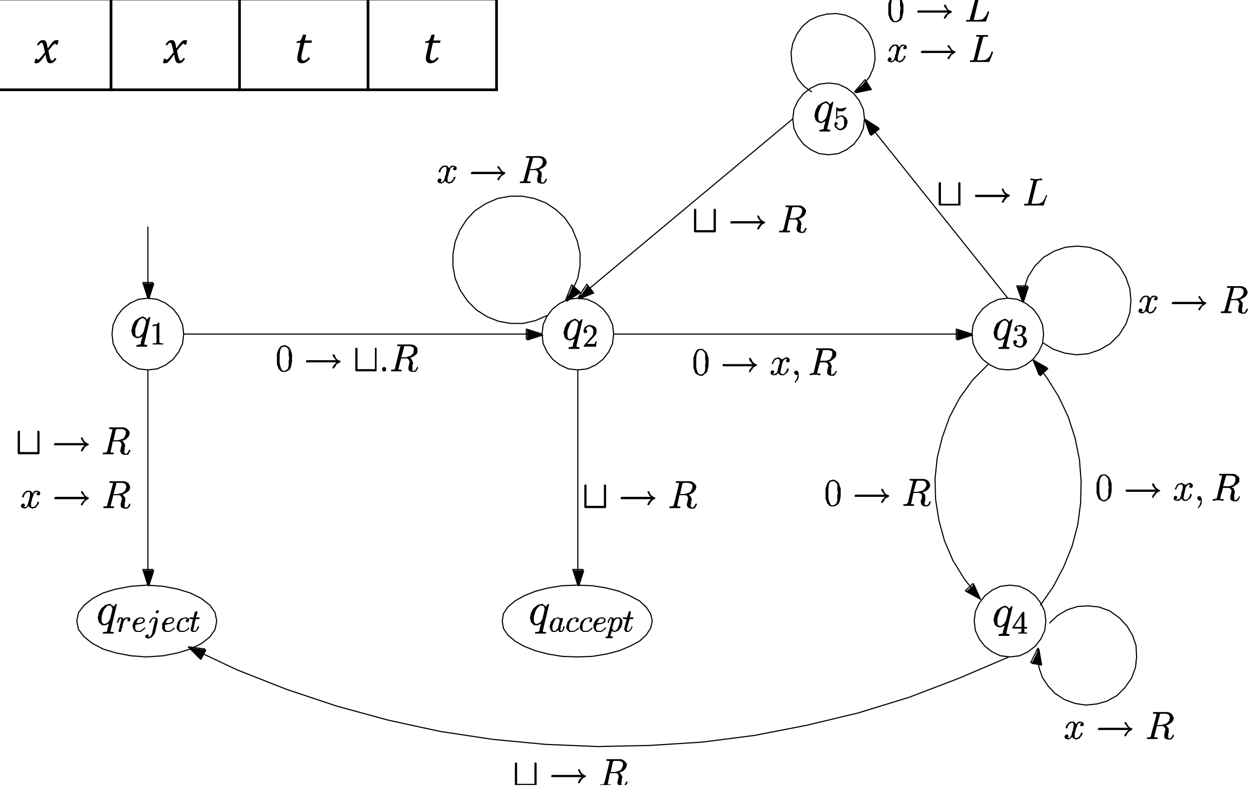




- Beispiel für das Wort: 0000

1. Einführung





- Beispiel für das Wort: 0000
- Endkonfiguration: $txxxtq_{accept}$

1. Einführung

Definition

- Eine Turingmaschine M berechnet die Funktion $f : \Sigma^* \to \Gamma^* \setminus \{t\}$ falls für alle w aus Σ^* die Berechnung von M mit der Eingabe w in einer akzeptierenden Konfiguration hält und dabei der Bandinhalt $f(w)$ ist.

1. Einführung

- Eine Mehrband- oder k-Band Turingmaschine (k-Band DTM) hat k Bänder mit je einem Kopf.
- Die Übergangsfunktion ist dann von der Form $\delta: Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R, S\}^k$
- Zu Beginn steht die Eingabe auf Band 1, sonst stehen überall Blanks. Die Arbeitsweise ist analog zu 1-Band-DTMs definiert.

1. Einführung

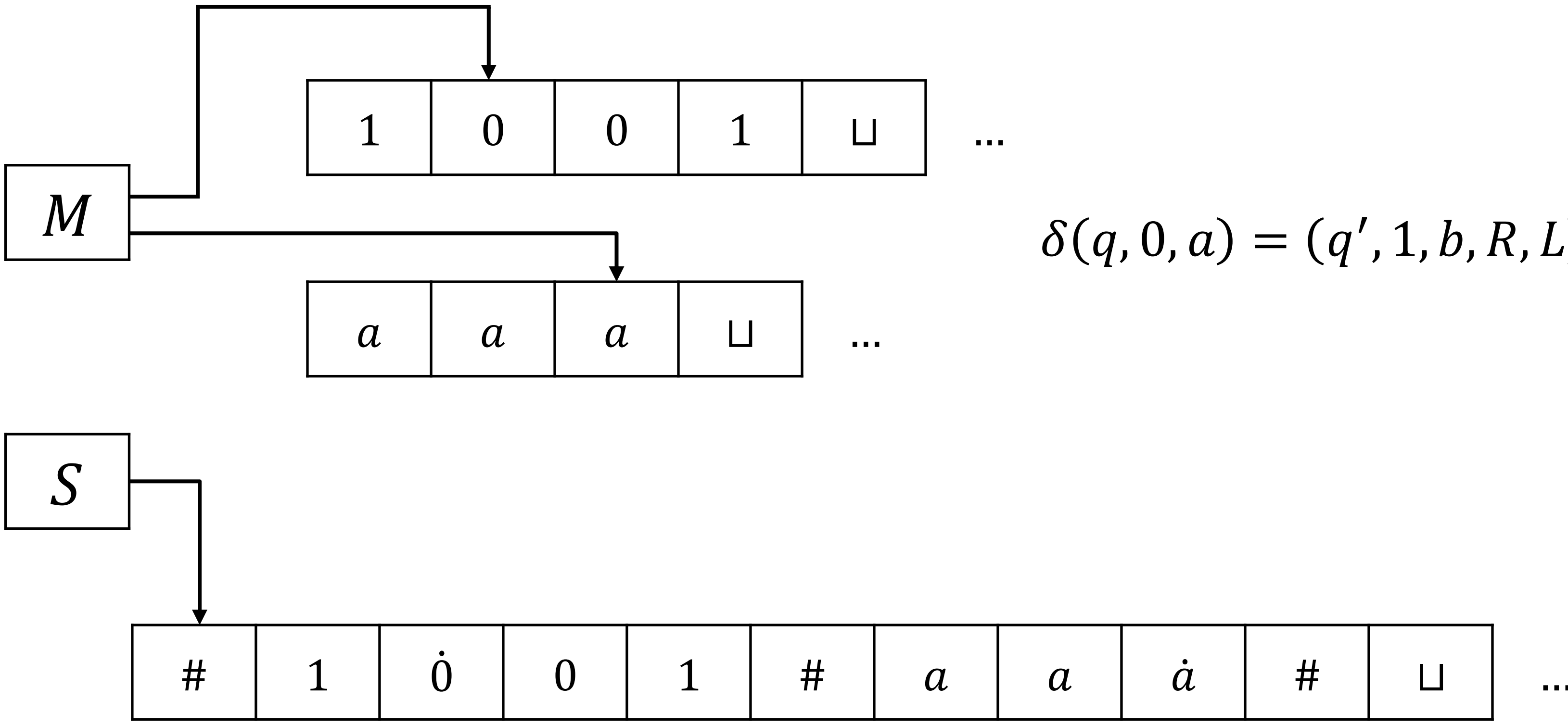
Satz

Zu jeder Mehrband-Turingmaschine gibt es eine äquivalente 1-Band-Turingmaschine.

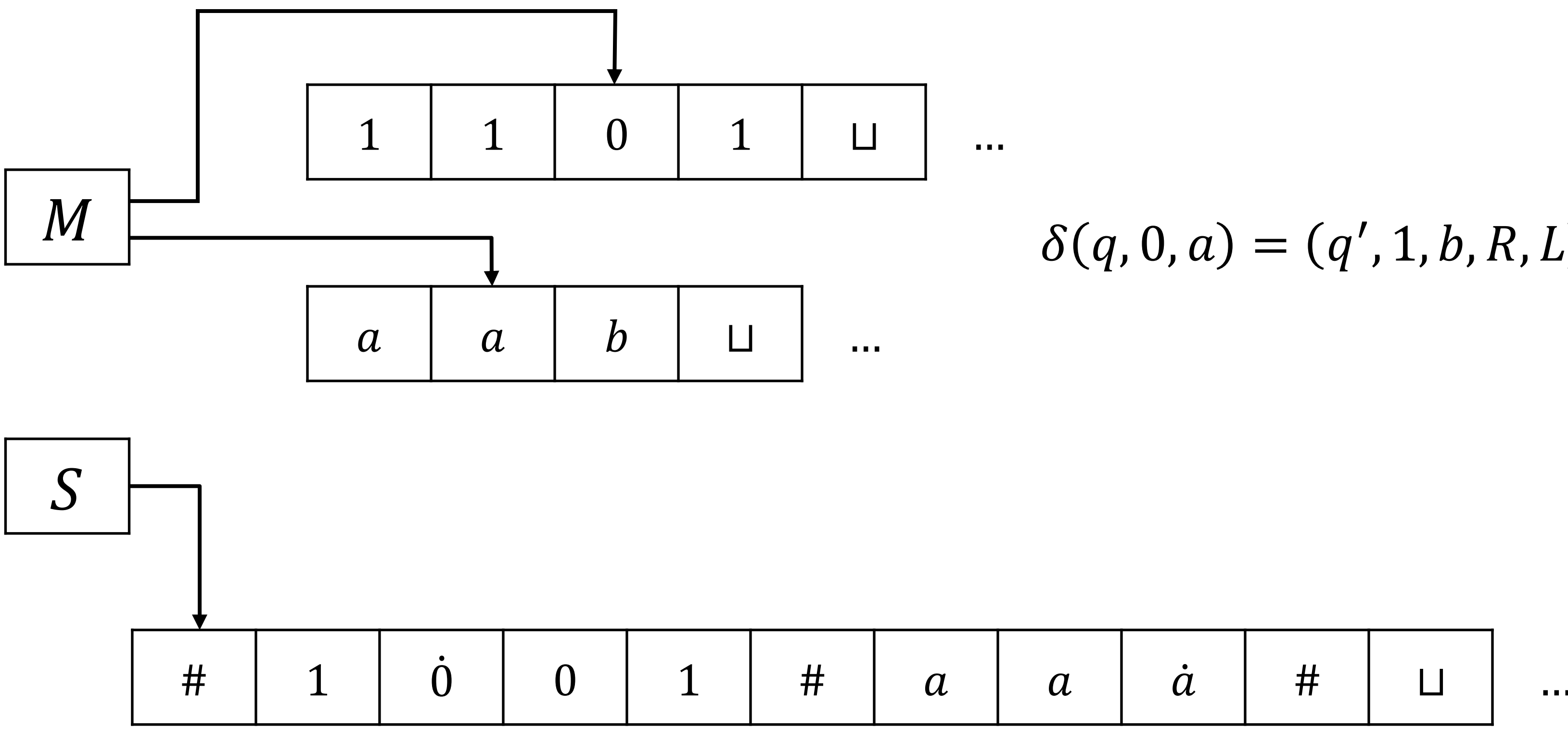
Beweis

Idee:
Simuliere Mehrband-DTM M auf 1-Band-DTM S.

1. Einführung

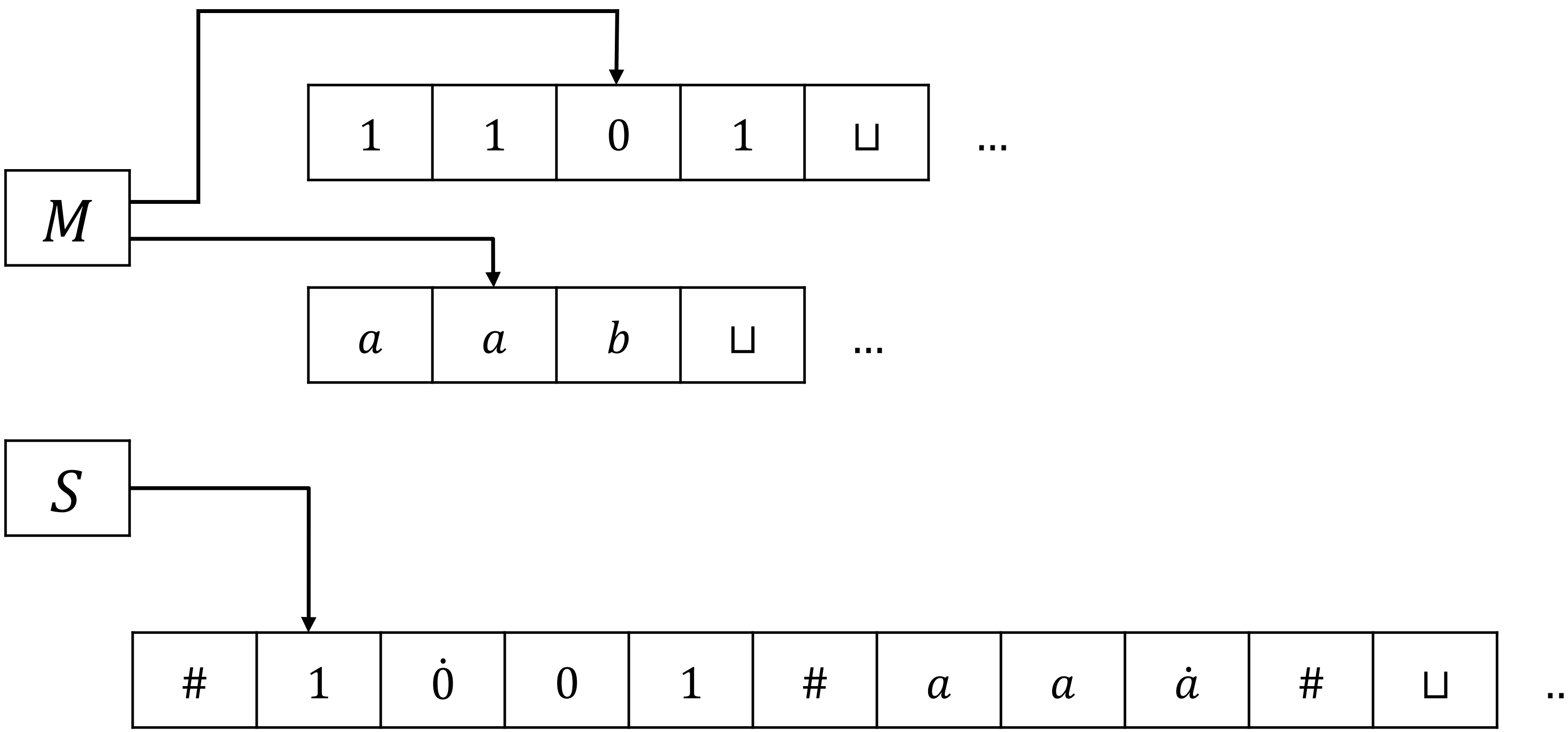


$$\delta(q, 0, a) = (q', 1, b, R, L)$$

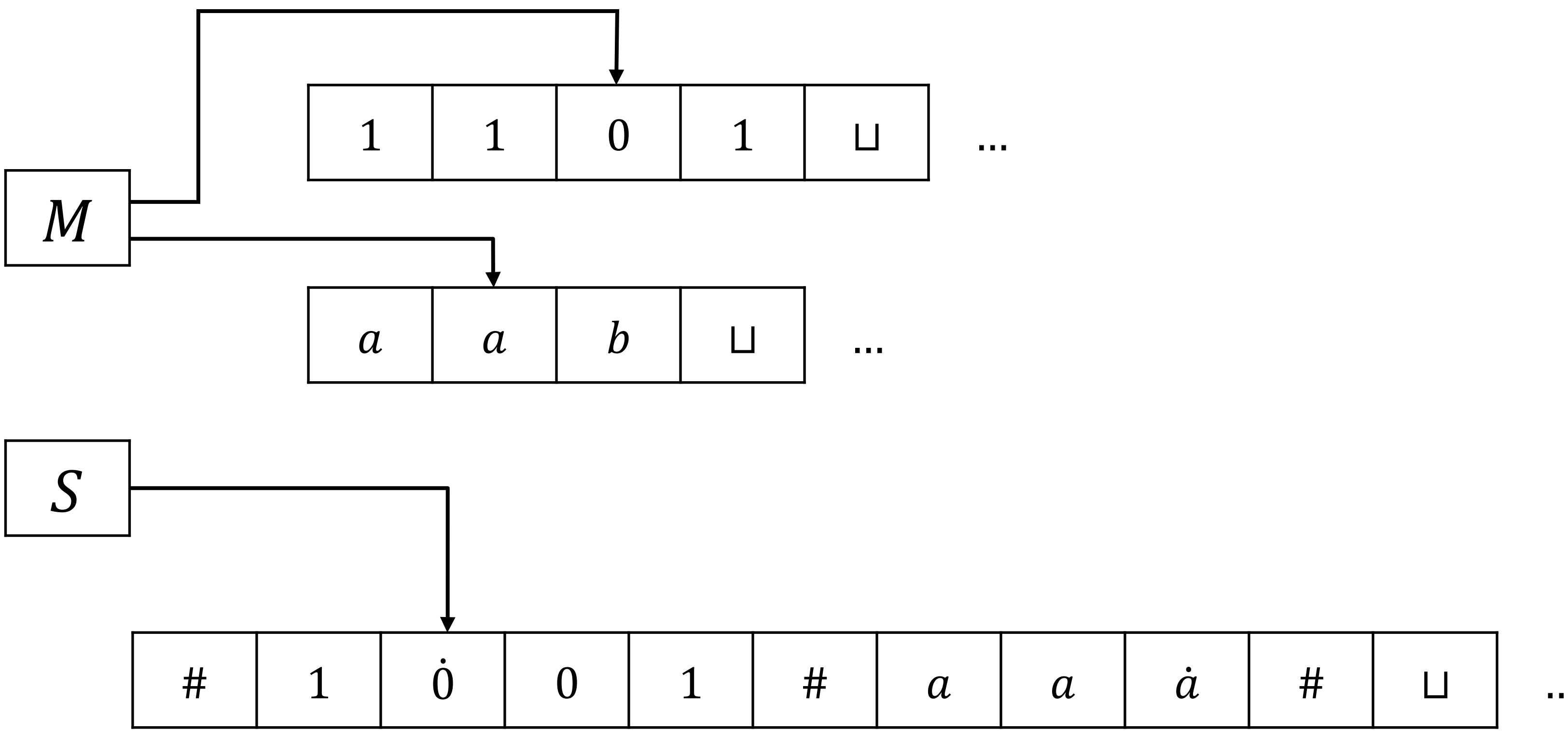


$$\delta(q, 0, a) = (q', 1, b, R, L)$$

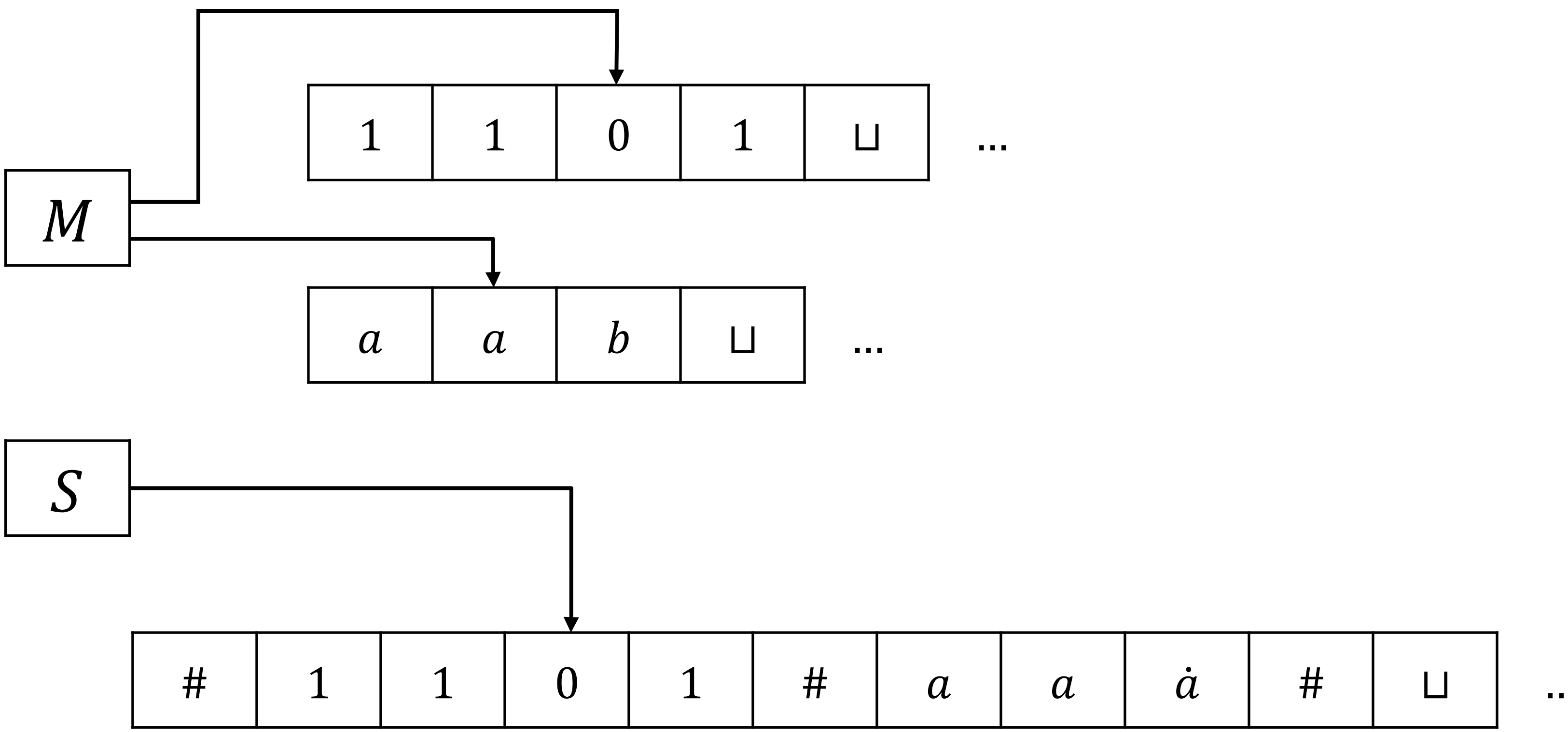
1. Einführung



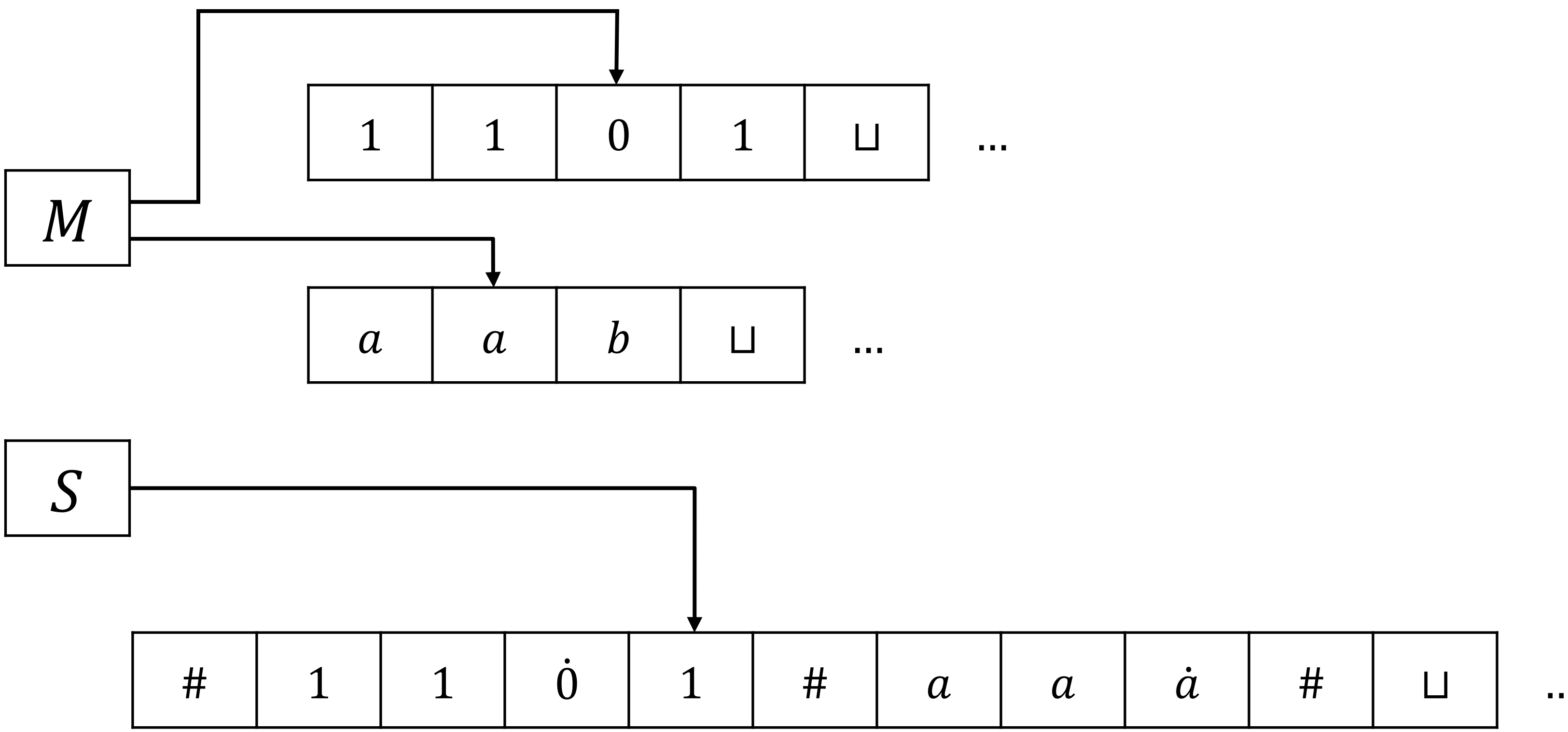
1. Einführung

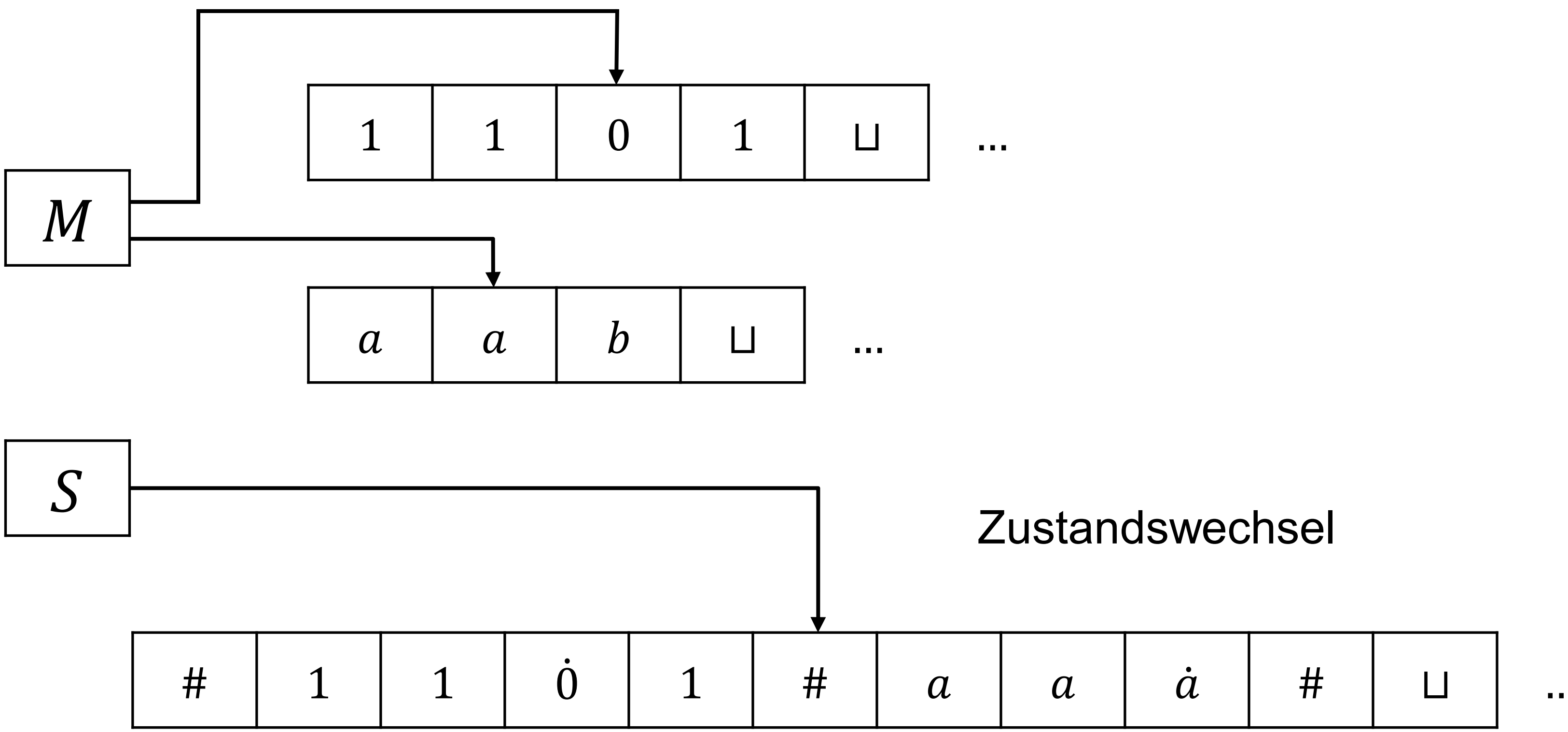


1. Einführung

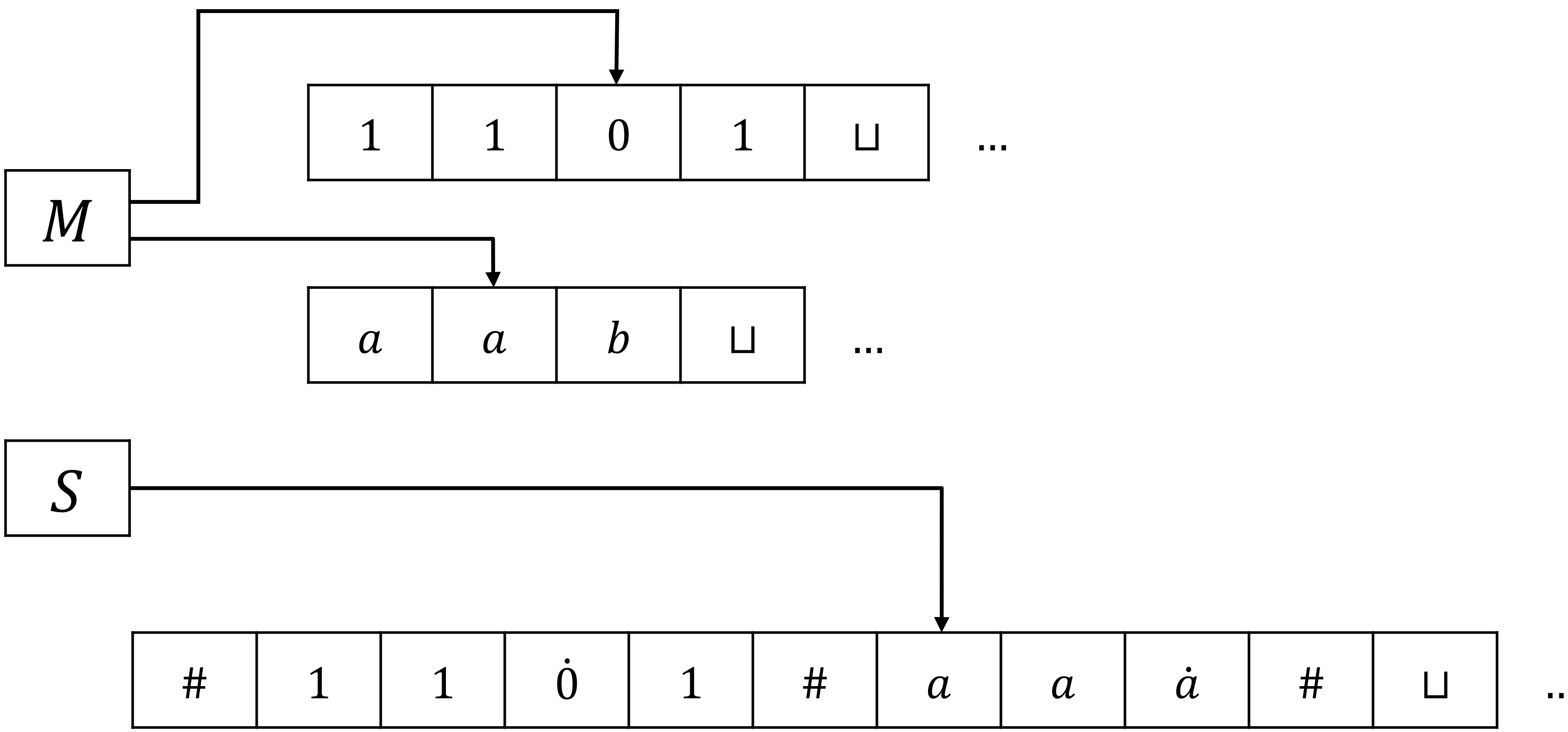


1. Einführung

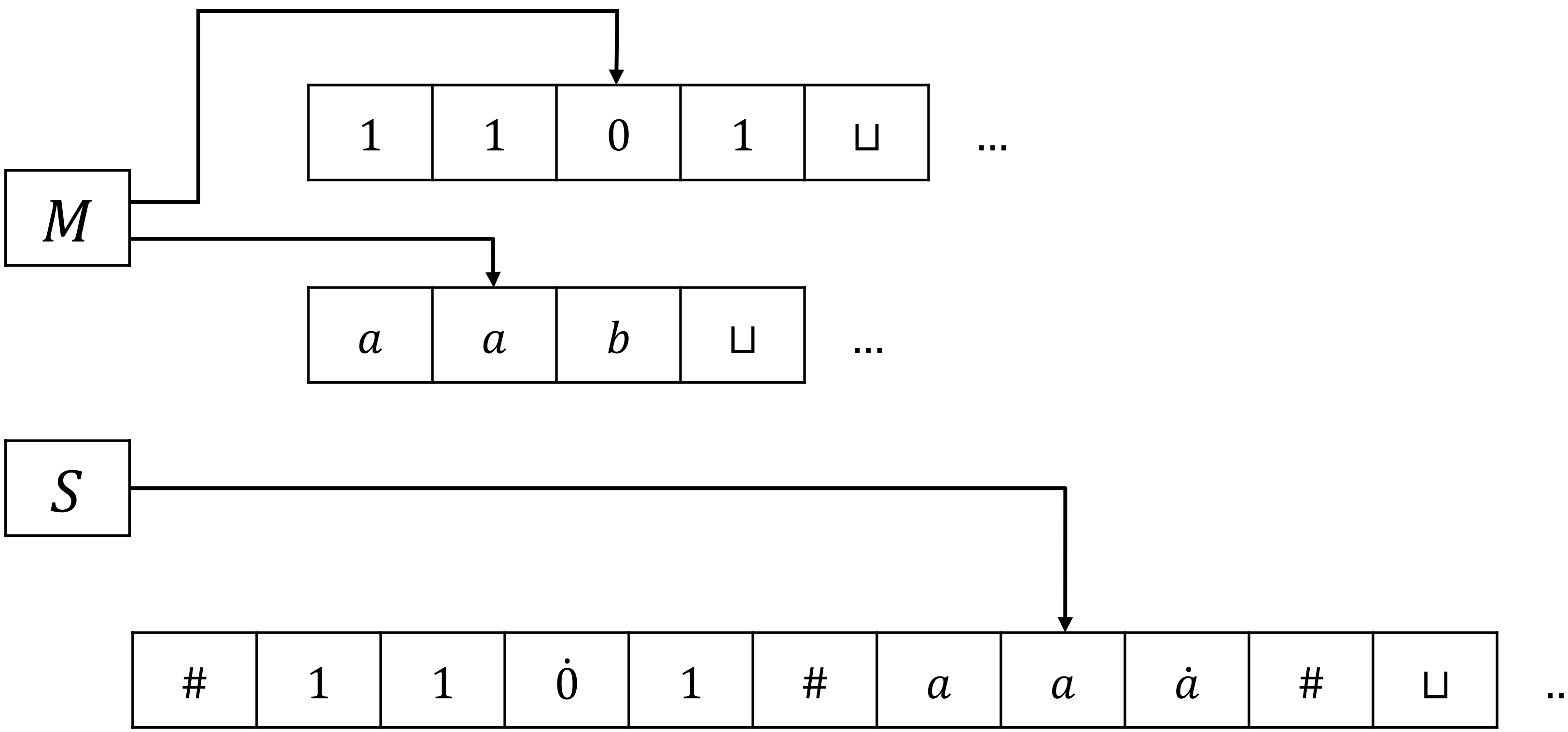



1 1 0 1 ⊔ ...
M
a a b ⊔ ...
S
Zustandswechsel
1 1 0̇ 1 # a a ȧ # ⊔ ...

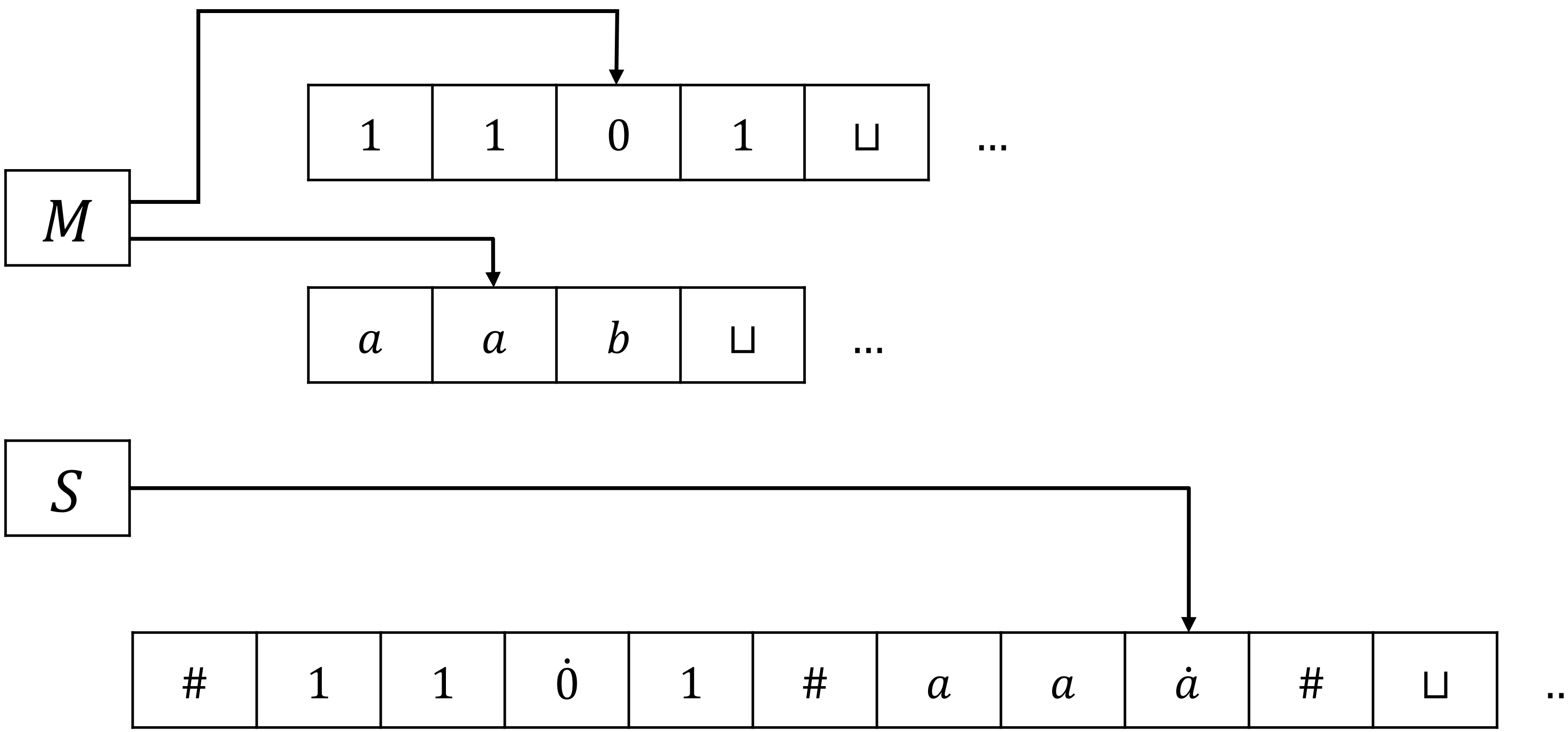
1. Einführung



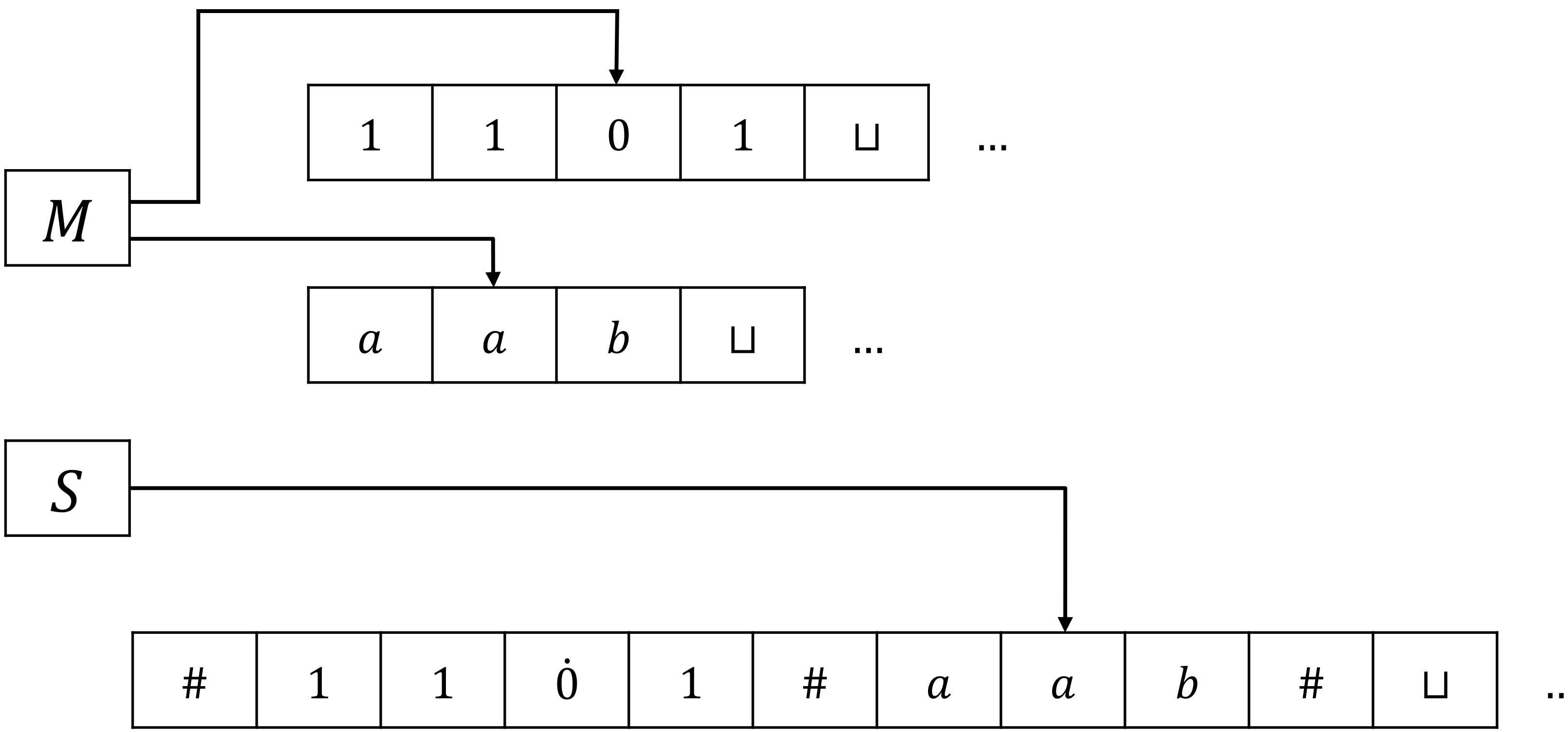
1. Einführung



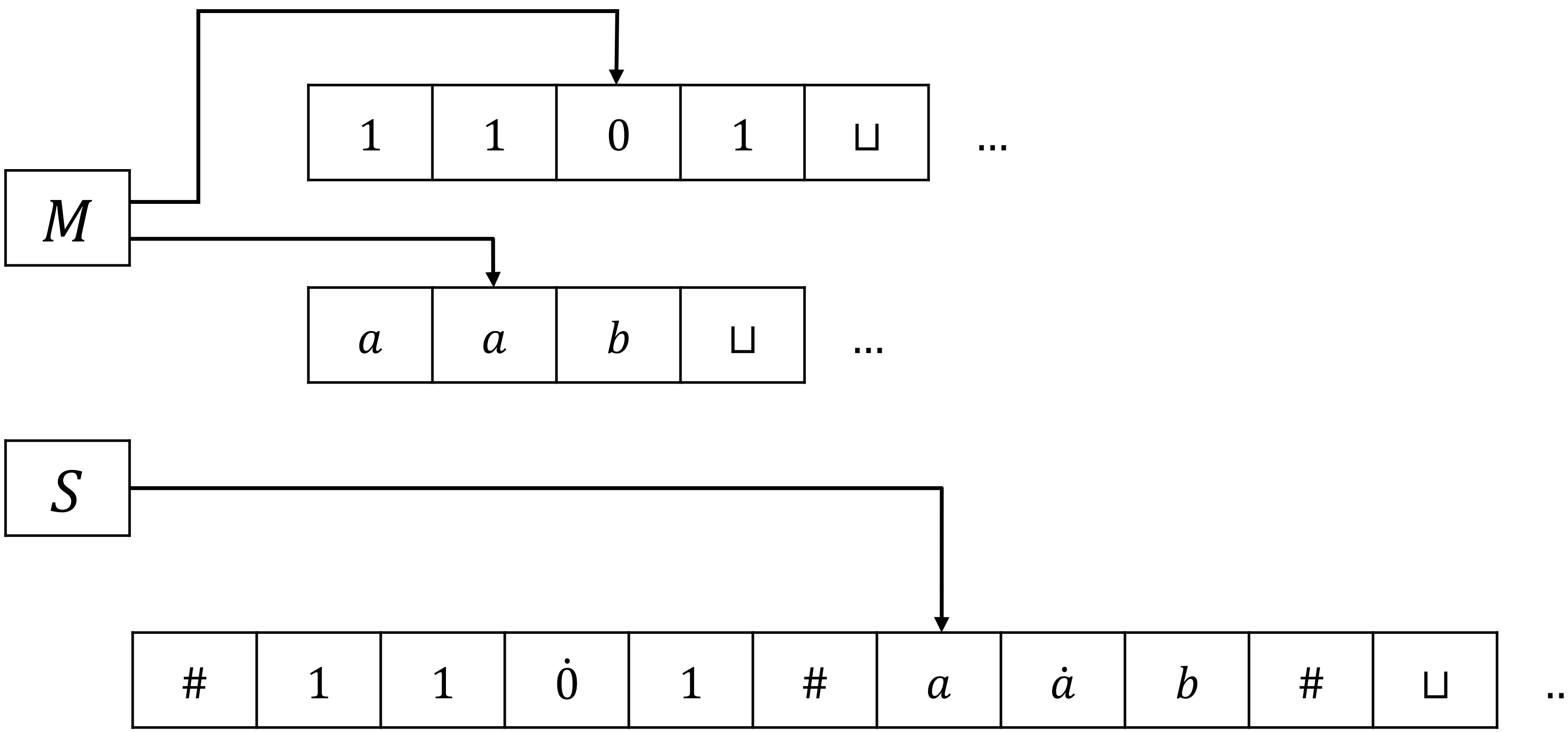
1. Einführung



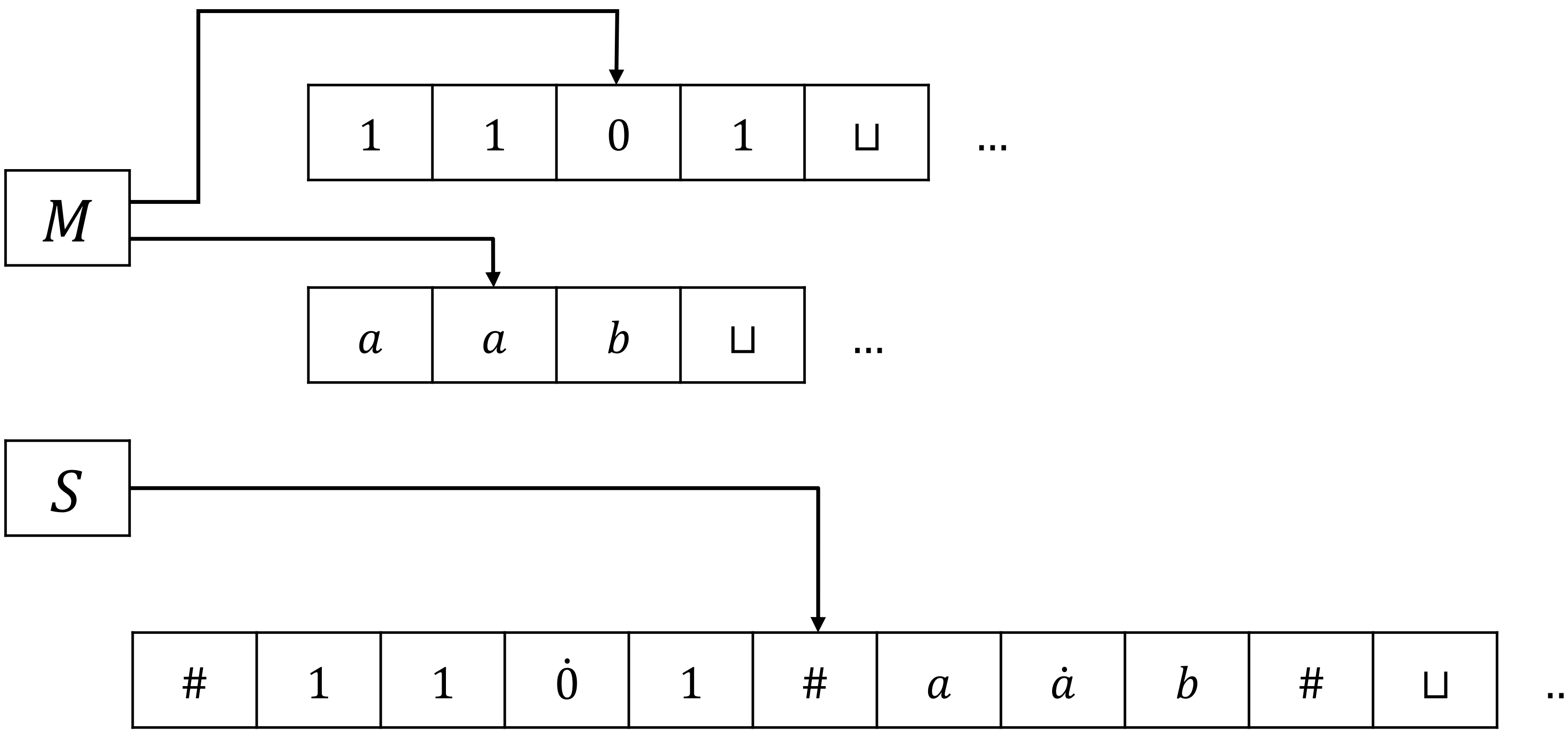
1. Einführung



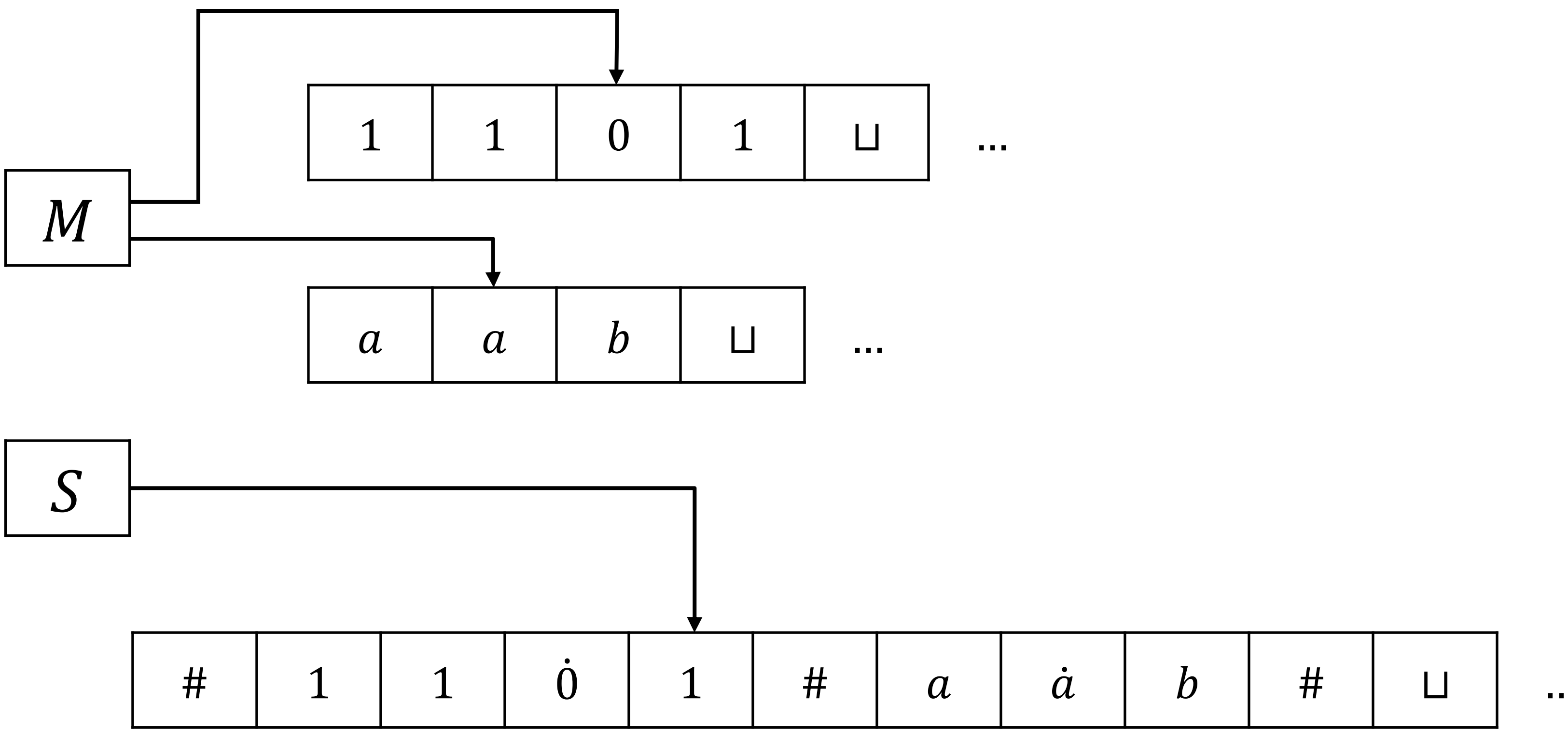
1. Einführung



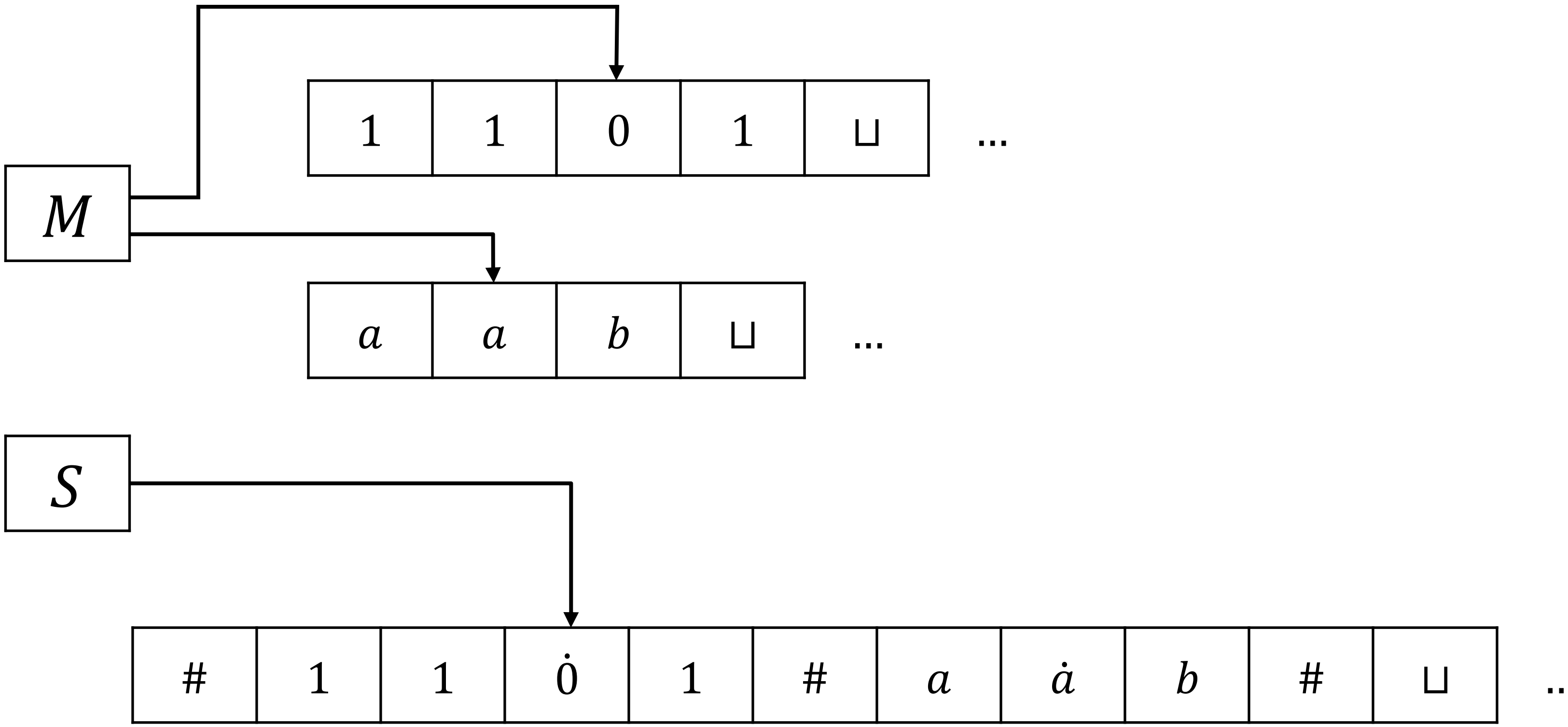
1. Einführung



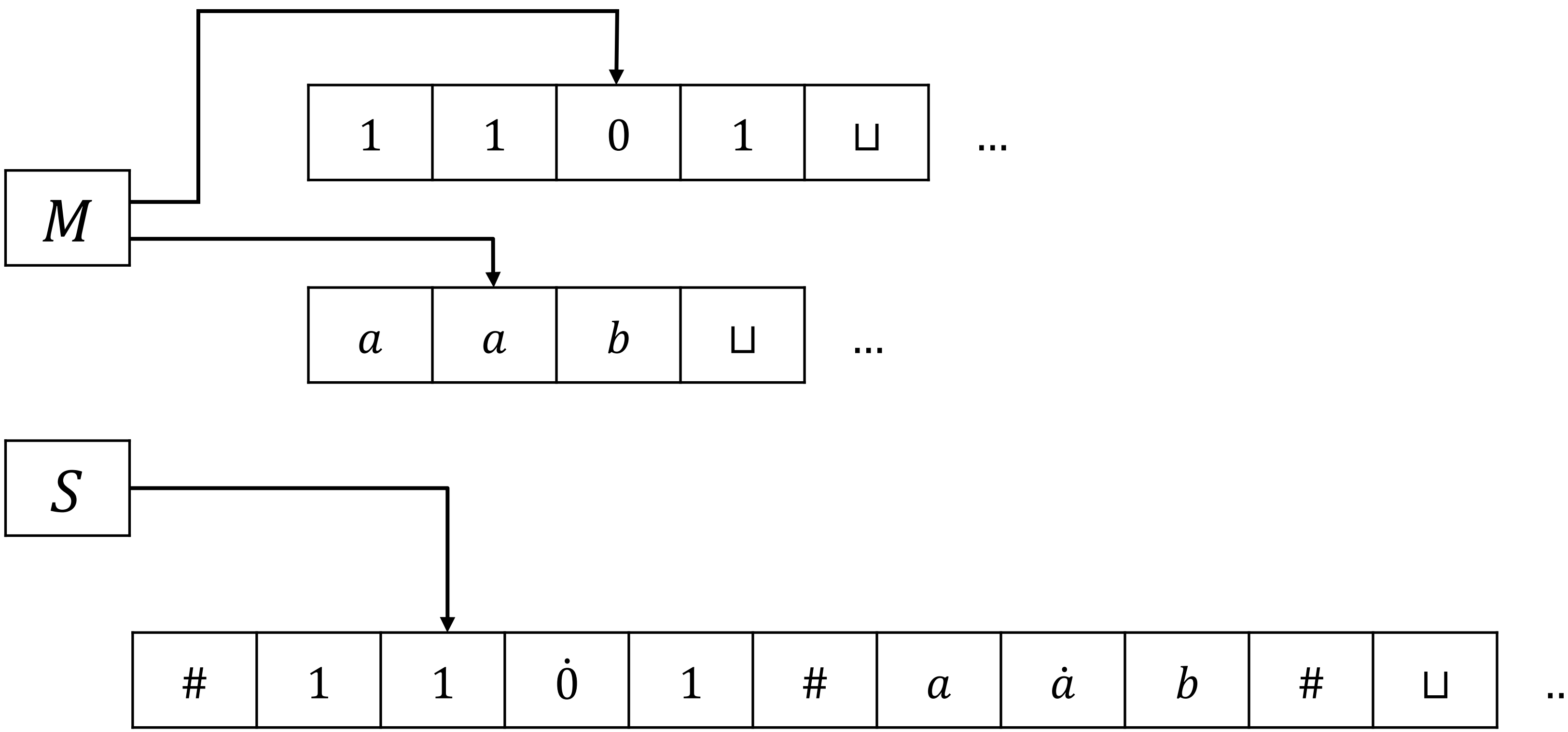
1. Einführung



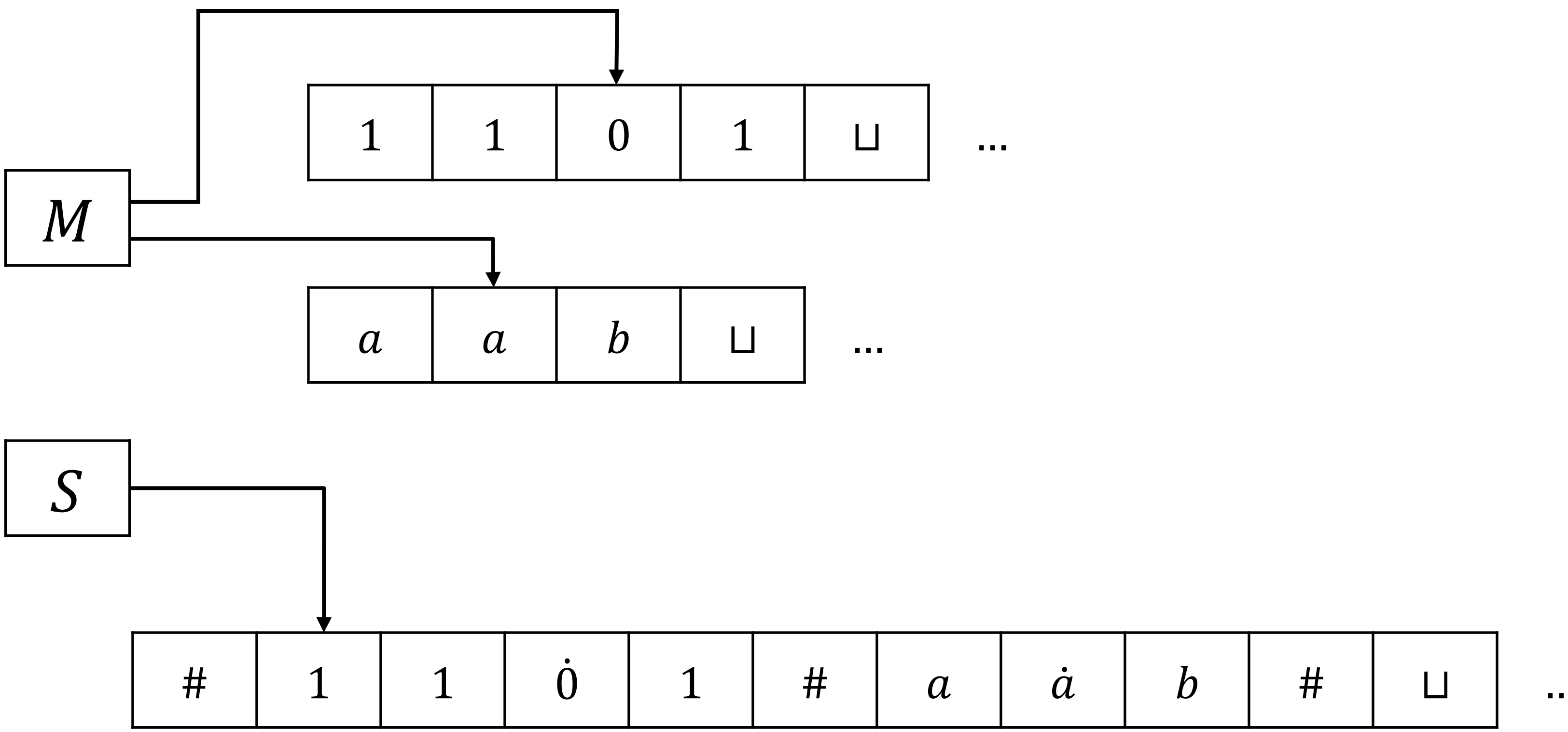
1. Einführung



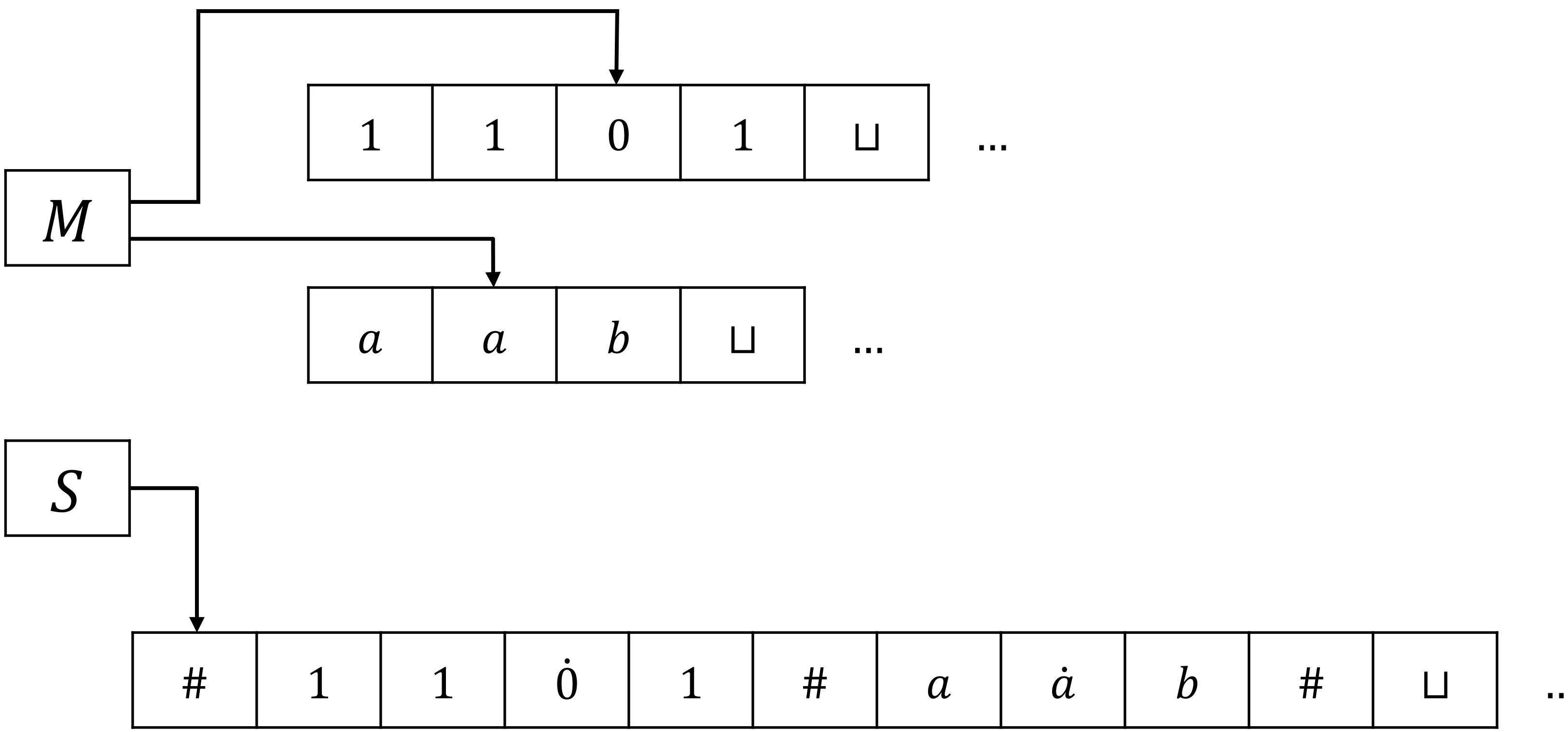
1. Einführung



1. Einführung



1. Einführung



1. Einführung

Korollar

Eine Sprache L ist genau dann rekursiv aufzählbar, wenn es eine Mehrband-Turingmaschine gibt, die L akzeptiert.

Formale Sprachen und Komplexitätstheorie

WS 2018/19

Robert Elsässer

Inhaltsangabe

- Einleitung, Motivation
- Turingmaschinen
- Arbeitstechniken

} Einführung

- Unentscheidbare Probleme
- Das Halteproblem
- Reduktionen

} Berechenbarkeit

- Zeitkomplexität
- Die Klassen P und NP
- NP-Vollständigkeit
- NP-vollständige Probleme

} Komplexität

- Formale Sprachen und Automaten
- Kellerautomaten und kontextfreie Sprachen
- Kontextsensitive Sprachen

} Formale Sprachen

1. Einführung

Gibt es einen Algorithmus HALTE, der

- als Eingabe einen beliebigen Algorithmus ALG und eine Eingabe w für ALG erhält und
- entscheidet, ob ALG bei Eingabe w hält?

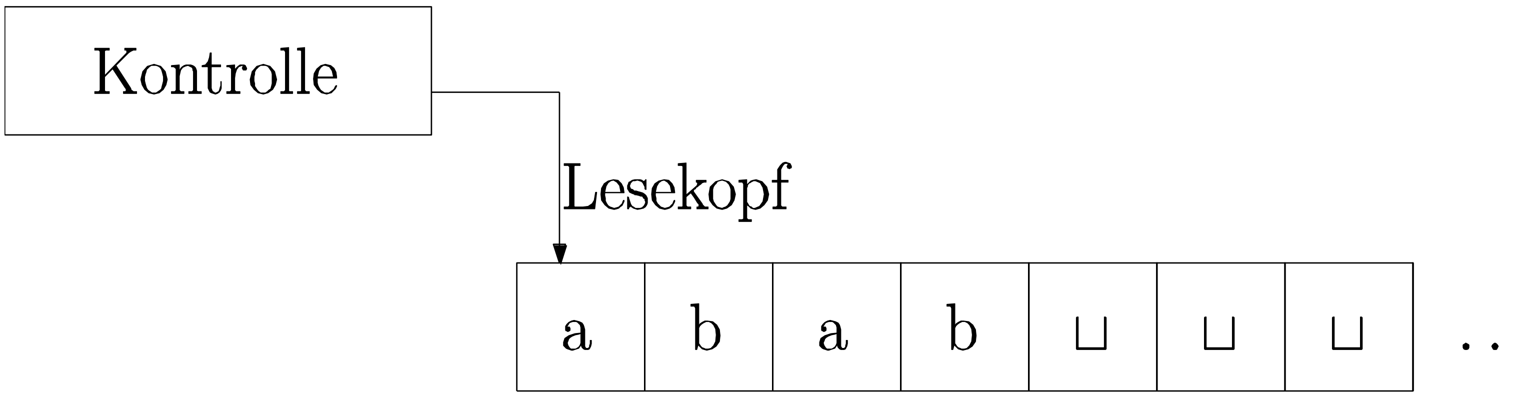
Satz von Turing:

Einen solchen Algorithmus kann es nicht geben.

1. Einführung

- **Turingmaschine**
  - Arbeitet auf unbeschränktem Band
  - Eingabe steht zu Beginn am Anfang des Bands
  - Auf dem Rest des Bandes steht t (Blank)
  - Position auf dem Band wird durch den sog. *Lesekopf* beschrieben

Turingmaschine



- Der jeweils nächste Rechenschritt ist eindeutig festgelegt durch den aktuellen Zustand und das aktuell gelesene Zeichen.
- Der Rechenschritt überschreibt das aktuelle Zeichen, bewegt den Kopf nach rechts oder nach links und verändert den Zustand.

1. Einführung

Definition

Eine *(deterministische 1-Band) Turingmaschine* (DTM) wird beschrieben durch ein 7-Tupel $M = \left(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject}\right)$.

Dabei sind Q, Σ, Γ endliche, nichtleere Mengen und es gilt:

- Σ ist Teilmenge von Γ
- t in $\Gamma \setminus \Sigma$ ist das *Blanksymbol* (auch ⊔)
- Q ist die *Zustandsmenge*
- Σ ist das *Eingabealphabet*
- Γ ist das *Bandalphabet*
- q_0 in Q ist der *Startzustand*
- q_{accept} in Q ist der akzeptierende Endzustand
- q_{reject} in Q ist der ablehnende Endzustand
- $\delta: Q \setminus \left\{q_{accept}, q_{reject}\right\} \times \Gamma \to Q \times \Gamma \times \{L, R\}$ ist die (partielle) *Übergangsfunktion*.
  Sie ist für kein Argument aus $\left\{q_{accept}, q_{reject}\right\} \times \Gamma$ definiert.

1. Einführung

- Initial:
  - Eingabe steht links auf dem Band
  - Der Rest des Bands ist leer
  - Kopf befindet sich ganz links
- Berechnungen finden entsprechend der Übergangsfunktion statt
- Wenn der Kopf sich am linken Ende befindet und nach links bewegen soll, bleibt er an seiner Position
- Wenn q_{accept} oder q_{reject} erreicht wird, ist die Bearbeitung beendet

Momentaufnahme einer Turingmaschine:

- Bei Bandinschrift uv (dabei beginnt u am linken Ende des Bandes und hinter v stehen nur Blanks)
- Zustand q
- Kopf auf erstem Zeichen von v

Konfiguration $C = uqv$

1. Einführung

- Gegeben: Konfigurationen C_1, C_2
- Wir sagen: **Konfiguration $\boldsymbol{C_1}$ führt zu $\boldsymbol{C_2}$,** falls die TM von C_1 in einem Schritt zu C_2 übergehen kann

Formal:

- Seien a, b, c in Γ, u, v in Γ^* und Zustände q_i, q_j gegeben
- Wir sagen:
  - uaq_ibv führt zu uq_jacv, falls $\delta(q_i, b) = (q_j, c, L)$ und
  - uaq_ibv führt zu $uacq_jv$, falls $\delta(q_i, b) = (q_j, c, R)$

1. Einführung

- Startkonfiguration:
  - $q_0 w$, wobei w die Eingabe ist

- Akzeptierende Konfiguration:
  - Konfigurationen mit Zustand q_{accept}

- Ablehnende Konfiguration:
  - Konfigurationen mit Zustand q_{reject}

- Haltende Konfiguration:
  - akzeptierende oder ablehnende Konfigurationen

1. Einführung

Definition

Eine Turingmaschine M akzeptiert eine Eingabe w, falls es eine Folge von Konfigurationen $C_1, C_2, \ldots, C_k$ gibt, sodass

1. C_1 ist die Startkonfiguration von M bei Eingabe w
2. C_i führt zu C_{i+1}
3. C_k ist eine akzeptierende Konfiguration

- Die von M akzeptierten Worte bilden die von M akzeptierte Sprache $L(M)$.
- Eine Turingmaschine entscheidet eine Sprache, wenn jede Eingabe in einer haltenden Konfiguration C_k resultiert.

1. Einführung

Definition

- Eine Sprache L heißt **rekursiv aufzählbar,** falls es eine Turingmaschine M gibt, die L *akzeptiert.*
- Eine Sprache L heißt **rekursiv** oder **entscheidbar,** falls es eine Turingmaschine M gibt, die L *entscheidet.*

1. Einführung

- Eine Mehrband- oder k-Band Turingmaschine (k-Band DTM) hat k Bänder mit je einem Kopf.
- Die Übergangsfunktion ist dann von der Form $\delta: Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R, S\}^k$
- Zu Beginn steht die Eingabe auf Band 1, sonst stehen überall Blanks. Die Arbeitsweise ist analog zu 1-Band-DTMs definiert.

1. Einführung

Satz

Zu jeder Mehrband-Turingmaschine gibt es eine äquivalente 1-Band-Turingmaschine.

Beweis

Idee:
Simuliere Mehrband-DTM M auf 1-Band-DTM S.

2. Berechenbarkeit

Simulationstechniken:

- Merken im Zustand
  - Nutzen Zustände als endlichen Speicher
- Markieren von Symbolen
  - Nutzen Bandalphabet zur Markierung von Positionen des Bandes

2. Berechenbarkeit

Im Zustand merken:

$L \coloneqq \{w \mid w = w_1 \dots w_n, \exists i, 2 \leq i \leq n : w_i = w_1\}$

1. $\delta(q_0, t) = (q_2, t, R)$
2. $\delta(q_0, a) = ([q_0, a], a, R)$ für alle a aus Σ
3. $\delta([q_0, a], a) = (q_1, a, R)$
4. $\delta([q_0, a], b) = ([q_0, a], b, R)$
5. $\delta([q_0, a], t) = (q_2, t, R)$

$q_{accept} = q_1, q_{reject} = q_2$

2. Berechenbarkeit

Element-Distinctness:

$L \coloneqq \left\{\#w_1\#w_2 \ldots \#w_n \mid w_i \text{ aus } \{0,1\}^*, w_i \neq w_j \text{ für alle } i \neq j\right\}$

Beispiele:

- $\#011\#001\#01\#00$ ist in L
- $\#011\#001\#01\#00\#001$ ist nicht in L

2. Berechenbarkeit

Element-Distinctness:

$L := \{\#w_1\#w_2 \ldots \#w_n \mid w_i \text{ aus } \{0,1\}^*, w_i \neq w_j \text{ für alle } i \neq j\}$

Beispiele:

- $\#011\#001\#01\#00$ ist in L
- $\#011\#001\#01\#00\#001$ ist nicht in L

Turingmaschine für Element-Distinctness:

1. Falls das erste Eingabesymbol nicht # ist, lehne ab – sonst ersetze # durch #′.
   Wenn kein weiteres # gefunden, akzeptiere.

2. Finde das nächste # 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.

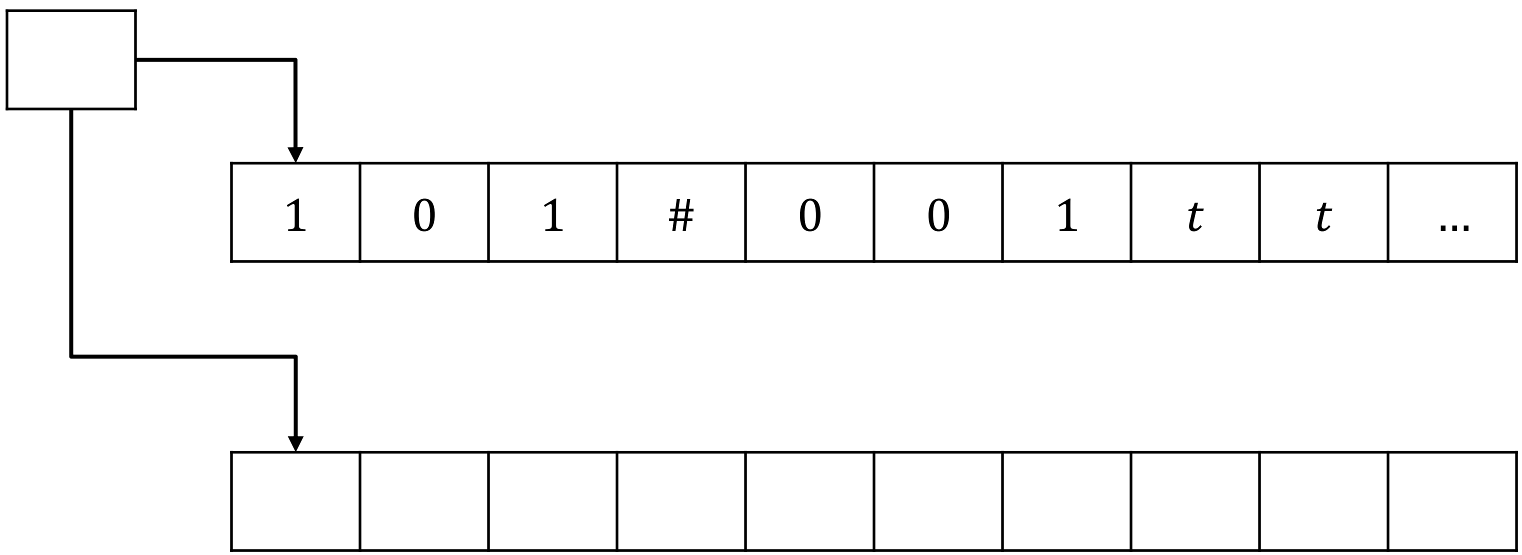
4. Verschiebe Markierungen für den Vergleich des nächsten Paares von Folgen. Falls dieses Paar nicht mehr existiert, akzeptiere. Sonst gehe zu Schritt 3.

2. Berechenbarkeit

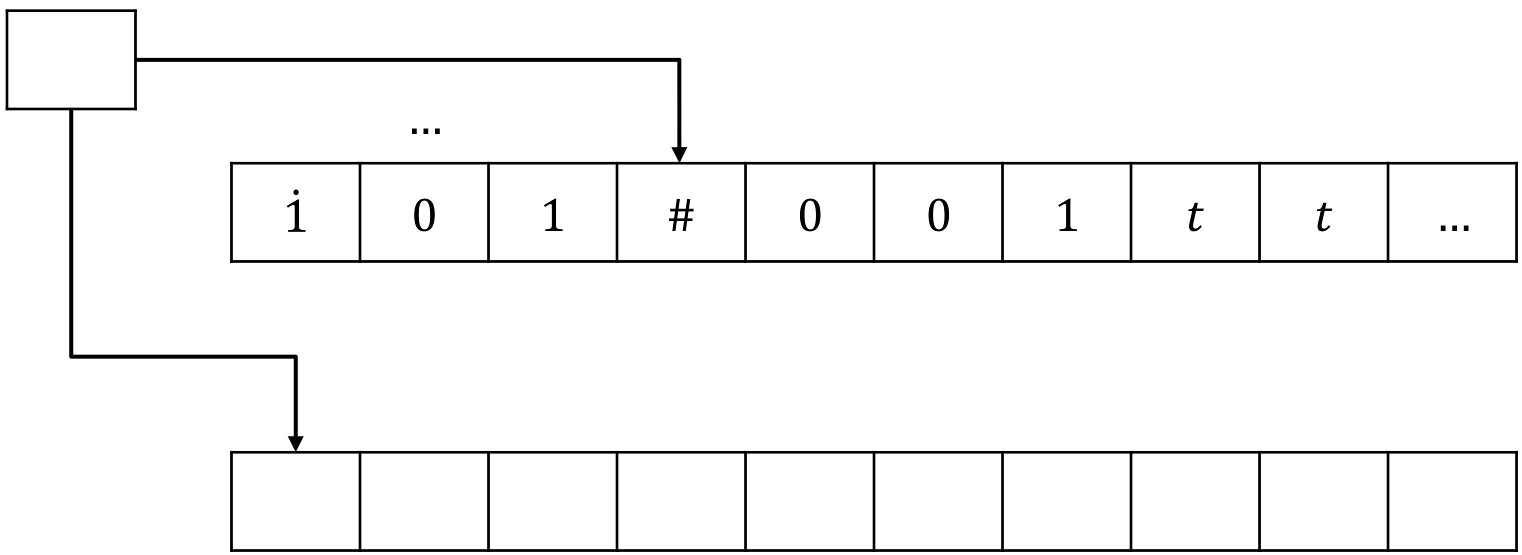
Berechnung, Akzeptieren, Entscheiden, … k-Band Turingmaschinen

- **Beispiel:** Addition von zwei Binärzahlen
- **Eingabe:** $w_1\#w_2$ für zwei Binärzahlen w_1 und w_2 (z.B. $100\#1000$ entspricht den Zahlen 4 und 8).
- **Ausgabe:** das Ergebnis $w_1 + w_2$ auf irgendeinem Band
- **Strategie:**
  - Verwende eine 3-Band Turing-Maschine
  - w_2 wird zunächst auf Band 2 geschrieben
  - w_1 und w_2 werden bitweise auf Band 3 zusammenaddiert. Zum Schluss geht man in q_{accept}.

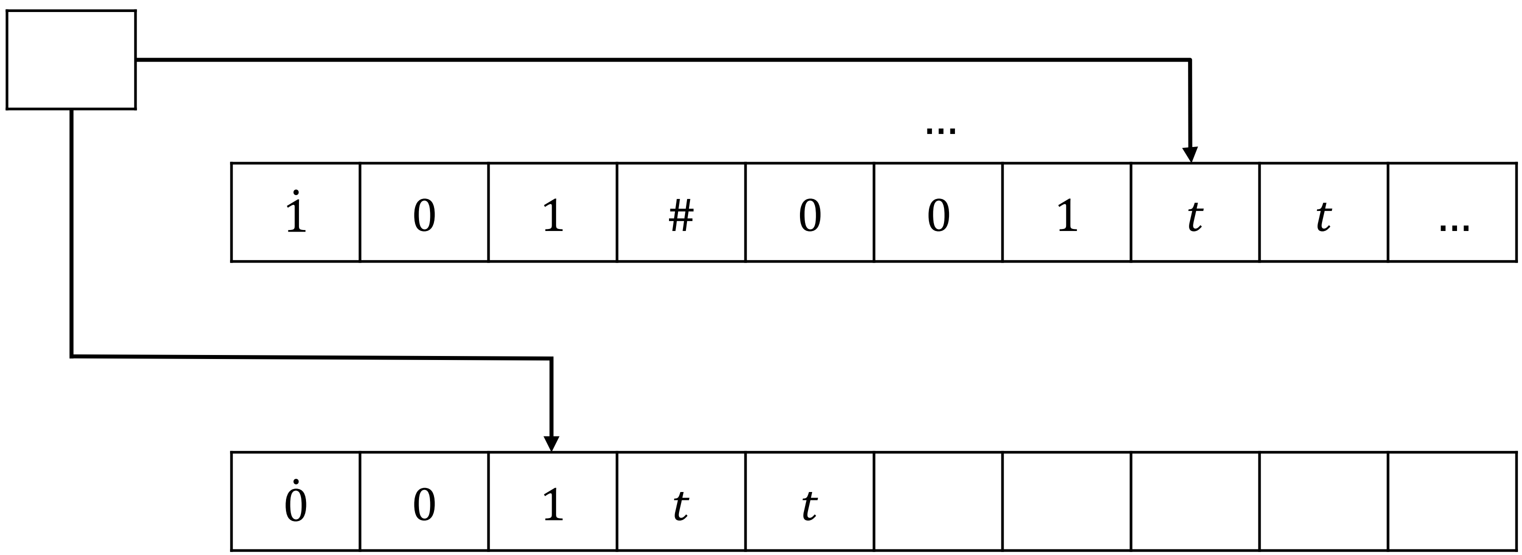
2. Berechenbarkeit



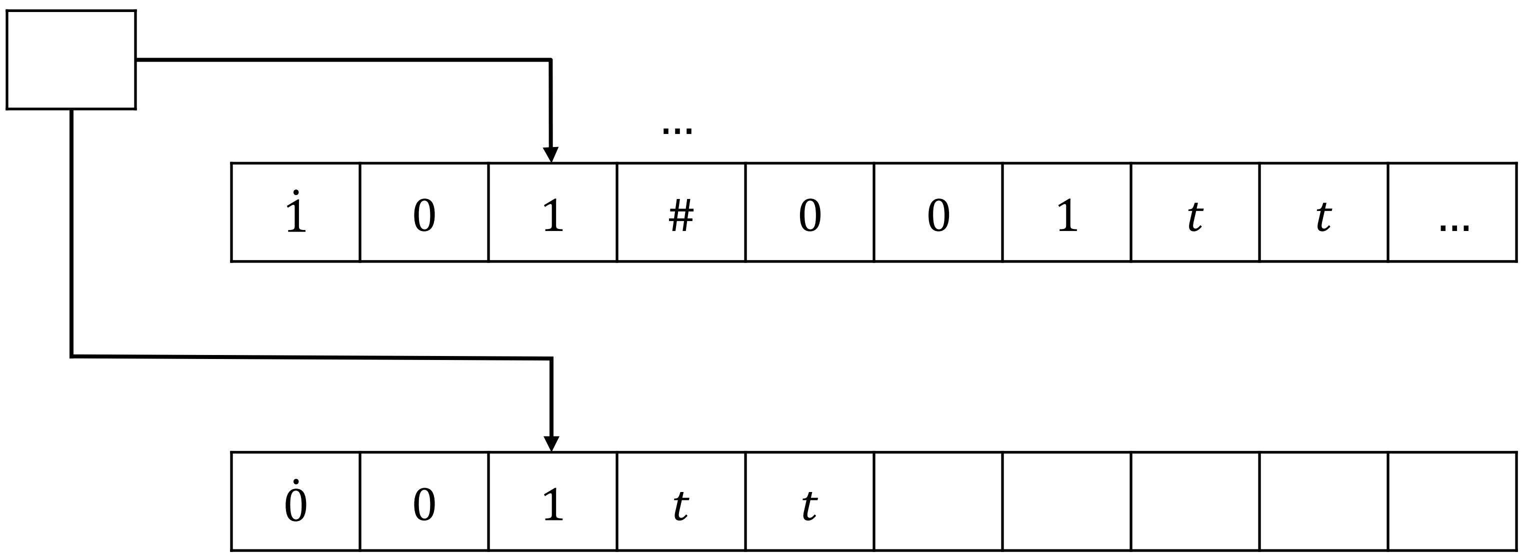
2. Berechenbarkeit



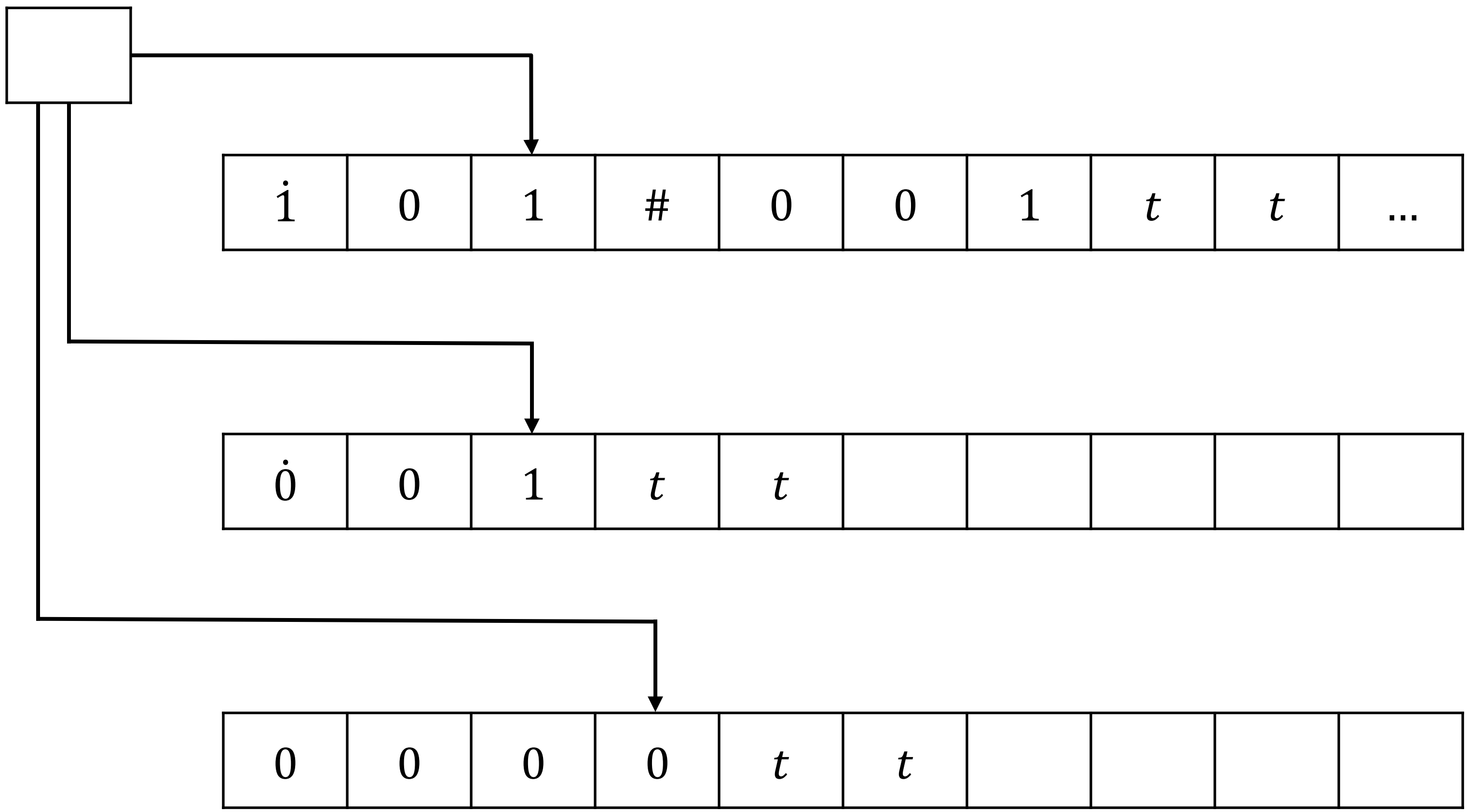
2. Berechenbarkeit

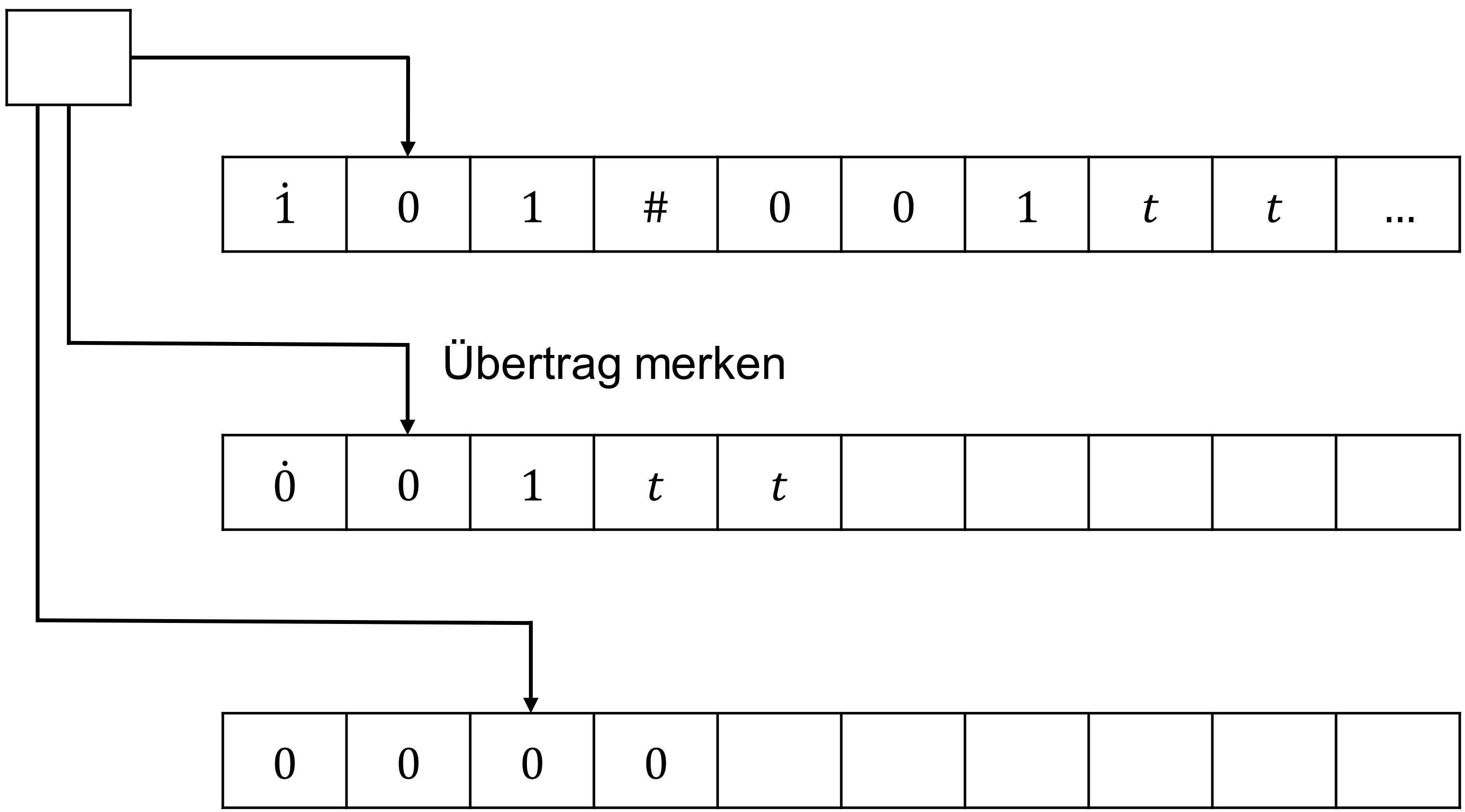


2. Berechenbarkeit



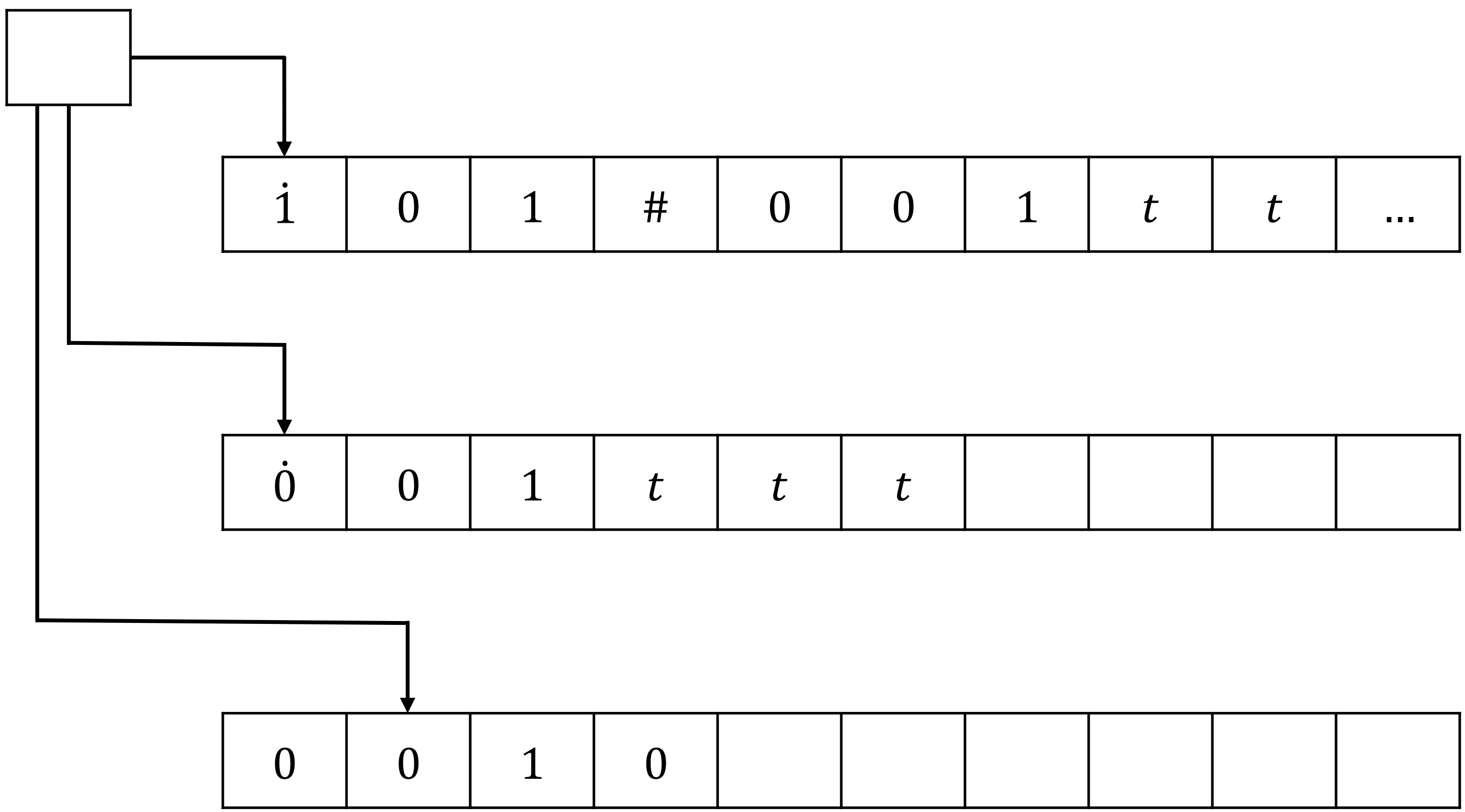
2. Berechenbarkeit



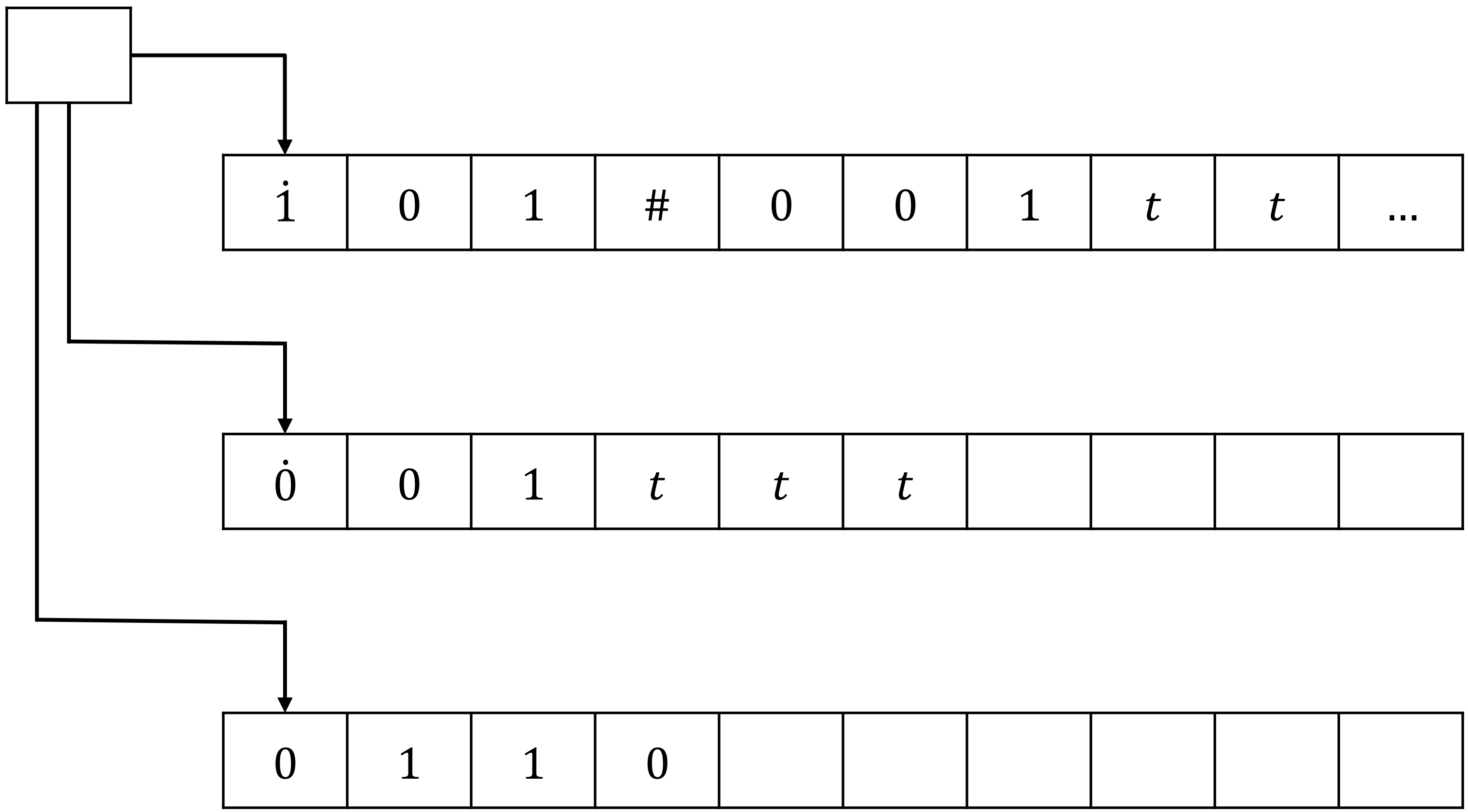


$\dot{1}$ 0 1 # 0 0 1 t t ...
Übertrag merken
$\dot{0}$ 0 1 t t
0 0 0 0

2. Berechenbarkeit



2. Berechenbarkeit



2. Berechenbarkeit

Church‘sche These (1936)

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

Warum sind Turingmaschinen ein geeignetes Modell?

- Menschliche Wahrnehmung ist endlich.
- Jeder realisierbare Rechner muss endlicher Natur sein und den physikalischen Gesetzen folgen.

Abschlusseigenschaften

$\overline{L}$: Komplementsprache zu $L - \overline{L} = \Sigma^* \setminus L$

Satz

Seien L_1 und L_2 entscheidbare Sprachen. Dann gilt:

1. $\overline{L_1}$ ist entscheidbar
2. $L_1 \cap L_2$ ist entscheidbar
3. $L_1 \cup L_2$ ist entscheidbar

Satz

Seien L_1 und 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

2. Berechenbarkeit

Abschlusseigenschaften

Satz

Eine Sprache L ist genau dann entscheidbar, wenn L und $\overline{L}$ rekursiv aufzählbar sind.

2. Berechenbarkeit

- **Universelle Turingmaschinen**
  - Bislang *special purpose Computer*:
    eine Sprache – eine Turing-Maschine
  - Allgemein programmierbare Turing-Maschinen:
    universelle Turing-Maschinen
  - Erhalten als Eingabe die Beschreibung einer Turingmaschine und simulieren diese Maschine
  - Benötigen dafür eine einheitliche Beschreibung von Turingmaschinen durch sog. *Gödel-Nummern*

2. Berechenbarkeit

Standardisierungen

- Betrachten nur 1-Band Turing-Maschinen
- Standardalphabet $\Sigma = \{0,1\}, \Gamma = \{0,1,t\}$
- andere Alphabete können durch Standardalphabete kodiert werden
- Turingmaschinen mit anderen Alphabeten können durch Turingmaschinen mit Standardalphabeten simuliert werden.

Definition Gödelnummern

Sei M eine 1-Band-Turingmaschine mit

$$Q = \{q_0, \dots, q_n\},$$
$$q_{accept} = q_{n-1},$$
$$q_{reject} = q_n.$$

Sei $X_1 = 0, X_2 = 1, X_3 = t, D_1 = L, D_2 = R$.

Wir kodieren $\delta(q_i, X_j) = (q_k, X_l, D_m)$ durch $0^{i+1}10^j10^{k+1}10^l10^m$.

$Code_r$: Kodierung des r-ten Eintrags für $\delta, 1 \leq r \leq 4(n-1)$

Gödelnummer $\langle M \rangle = 111Code_1 11Code_2 11 \dots 11Code_g 111$

2. Berechenbarkeit

Definition Universelle Turingmaschine

Eine Turingmaschine M_0 heißt universell, falls für jede 1-Band-Turingmaschine M und jedes x aus $\{0,1\}^*$ gilt:

- M_0 gestartet mit $\langle M \rangle x$ hält genau dann, wenn M gestartet mit x hält.
- M_0 akzeptiert $\langle M \rangle x$ genau dann, wenn M das Wort x akzeptiert.

Satz

Es gibt eine universelle 2-Band Turingmaschine.

Formale Sprachen und Komplexitätstheorie

WS 2018/19

Robert Elsässer

1. Einführung

Definition

Eine *(deterministische 1-Band) Turingmaschine* (DTM) wird beschrieben durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$.

Dabei sind Q, Σ, Γ endliche, nichtleere Mengen und es gilt:

- Σ ist Teilmenge von Γ
- t in $\Sigma \cap \Gamma$ ist das *Blanksymbol* (auch ⊔)
- Q ist die *Zustandsmenge*
- Σ ist das *Eingabealphabet*
- Γ ist das *Bandalphabet*
- q_0 in Q ist der *Startzustand*
- q_{accept} in Q ist der akzeptierende Endzustand
- q_{reject} in Q ist der ablehnende Endzustand
- $\delta: Q \setminus \{q_{accept}, q_{reject}\} \times \Gamma \to Q \times \Gamma \times \{L, R\}$ ist die (partielle) *Übergangsfunktion*. Sie ist für kein Argument aus $\{q_{accept}, q_{reject}\} \times \Gamma$ definiert.

1. Einführung

Momentaufnahme einer Turingmaschine:

- Bei Bandinschrift uv (dabei beginnt u am linken Ende des Bandes und hinter v stehen nur Blanks)
- Zustand q
- Kopf auf erstem Zeichen von v

Konfiguration $C = uqv$

1. Einführung

Definition

- Eine Sprache L heißt **rekursiv aufzählbar,**
  falls es eine Turingmaschine M gibt, die L *akzeptiert.*

- Eine Sprache L heißt **rekursiv** oder **entscheidbar,**
  falls es eine Turingmaschine M gibt, die L *entscheidet.*

1. Einführung

- Eine Mehrband- oder k-Band Turingmaschine (k-Band DTM) hat k Bänder mit je einem Kopf.
- Die Übergangsfunktion ist dann von der Form $\delta: Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R, S\}^k$
- Zu Beginn steht die Eingabe auf Band 1, sonst stehen überall Blanks. Die Arbeitsweise ist analog zu 1-Band-DTMs definiert.

2. Berechenbarkeit

- **Universelle Turingmaschinen**
  - Bislang *special purpose Computer*:
    eine Sprache – eine Turing-Maschine
  - Allgemein programmierbare Turing-Maschinen:
    universelle Turing-Maschinen
  - Erhalten als Eingabe die Beschreibung einer Turingmaschine und simulieren diese Maschine
  - Benötigen dafür eine einheitliche Beschreibung von Turingmaschinen durch sog. *Gödel-Nummern*

2. Berechenbarkeit

Definition Gödelnummern

Sei M eine 1-Band-Turingmaschine mit

$$Q = \{q_0, \dots, q_n\},$$
$$q_{accept} = q_{n-1},$$
$$q_{reject} = q_n.$$

Sei $X_1 = 0, X_2 = 1, X_3 = t, D_1 = L, D_2 = R$.

Wir kodieren $\delta(q_i, X_j) = (q_k, X_l, D_m)$ durch $0^{i+1}10^j10^{k+1}10^l10^m$.

$Code_r$: Kodierung des r-ten Eintrags für $\delta, 1 \leq r \leq 4(n-1)$

Gödelnummer $\langle M \rangle = 111Code_1 11Code_2 11 \dots 11Code_g 111$

Kurt Gödel

Quelle: www.numbersleuth.org

- Studium an der Universität Wien
- Dozenturen in Wien und Princeton
- Rennomiertester Preis in der theoretischen Informatik wird nach Gödel benannt

2. Berechenbarkeit

$M = \left(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject}\right)$

$\Sigma = \{0,1\}$

$\Gamma = \Sigma \cup \{t\}$

$\delta(q_0, 0) = \left(q_{reject}, 0, R\right)$

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

$\delta(q_0, t) = \left(q_{accept}, t, R\right)$

$L = \{1^n \mid n \geq 0\}$

Gödel-Nummer:

1110101000101001101001010010011010001001000100111

2. Berechenbarkeit

Definition Universelle Turingmaschine

Eine Turingmaschine M_0 heißt **universell**, falls für jede 1-Band-Turingmaschine M und jedes x aus $\{0,1\}^*$ gilt:

- M_0 gestartet mit $\langle M \rangle x$ hält genau dann, wenn M gestartet mit x hält.
- M_0 akzeptiert $\langle M \rangle x$ genau dann, wenn M das Wort x akzeptiert.

Satz

Es gibt eine universelle 2-Band Turingmaschine.

2. Berechenbarkeit

Die Sprache $\mathrm{Gödel}$:

Sprache $\mathrm{Gödel} \coloneqq \{w \text{ aus } \{0,1\}^* \mid w \text{ ist die Gödel-Nummer einer DTM}\}$

Lemma

Die Sprache $\mathrm{Gödel}$ ist entscheidbar.

Die Sprache States:

Sprache $\mathrm{States} \coloneqq \{(\langle M \rangle, d) \mid M \text{ besitzt mindestens } d \text{ Zustände}\}$

Lemma

Die Sprache States ist entscheidbar.

Das Halteproblem

$H := \{(\langle M\rangle, x)\ M \text{ ist DTM, die gestartet mit Eingabe } x \text{ hält}\}$

Satz

Das Halteproblem ist rekursiv aufzählbar.

Die Sprache Useful

$$\text{Useful} := \left\{ \begin{matrix} (\langle M \rangle, q) \mid M \text{ ist DTM mit Zustand } q, \text{ und es gibt eine Eingabe } w, \text{ so} \\ \text{dass } M \text{ gestartet mit } w \text{ in den Zustand } q \text{ gerät} \end{matrix} \right\}$$

Satz

Useful ist rekursiv aufzählbar.

2. Berechenbarkeit

Aufzählung von binären Eingabefolgen:

- für alle natürlichen Zahlen i sei $w_i = w$, falls $bin(i) = 1w$
- damit werden alle möglichen w aus $\{0,1\}^*$ aufgezählt

Aufzählung von Turingmaschinen:

M_i ist:

- M_{reject}, falls i keine Gödelnummer ist
- M, falls $bin(i)$ die Gödelnummer der DTM M ist, d.h. $\langle M \rangle = bin(i)$

Die Sprache Diag

$\mathrm{Diag} \coloneqq \{w \text{ in } \{0,1\}^* \mid w = w_i \text{ und die DTM } M_i \text{ akzeptiert } w \text{ nicht}\}$

Satz

Die Sprache Diag ist nicht rekursiv aufzählbar.

2. Berechenbarkeit

	M_1	M_2	M_3	$\dots$ $\dots$	M_7	$\dots$ $\dots$	M_i	$\dots$ $\dots$
w_1	na	na	na		na		na	
w_2	na	na	na		na		na	
w_3	na	na	na		na		na	
$\vdots$								
w_7	na	na	na		na		a	
$\vdots$								
w_i	na	na	na		na		a	

Diagonale

Tabelle für Akzeptanz/Nichtakzeptant von DTMs

Quelle: Skript Blömer

Formale Sprachen und Komplexitätstheorie

WS 2018/19

Robert Elsässer

1. Einführung

Definition

Eine *(deterministische 1-Band) Turingmaschine* (DTM) wird beschrieben durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$.

Dabei sind Q, Σ, Γ endliche, nichtleere Mengen und es gilt:

- Σ ist Teilmenge von Γ
- t in Γ ist das *Blanksymbol* (auch ⊔)
- Q ist die *Zustandsmenge*
- Σ ist das *Eingabealphabet*
- Γ ist das *Bandalphabet*
- q_0 in Q ist der *Startzustand*
- q_{accept} in Q ist der akzeptierende Endzustand
- q_{reject} in Q ist der ablehnende Endzustand
- $\delta: Q \setminus \{q_{accept}, q_{reject}\} \times \Gamma \to Q \times \Gamma \times \{L, R\}$ ist die (partielle) *Übergangsfunktion*. Sie ist für kein Argument aus $\{q_{accept}, q_{reject}\} \times \Gamma$ definiert.

1. Einführung

Momentaufnahme einer Turingmaschine:

- Bei Bandinschrift uv (dabei beginnt u am linken Ende des Bandes und hinter v stehen nur Blanks)
- Zustand q
- Kopf auf erstem Zeichen von v

Konfiguration $C = uqv$

1. Einführung

Definition

- Eine Sprache L heißt **rekursiv aufzählbar,**
  falls es eine Turingmaschine M gibt, die L *akzeptiert.*
- Eine Sprache L heißt **rekursiv** oder **entscheidbar,**
  falls es eine Turingmaschine M gibt, die L *entscheidet.*

1. Einführung

- Eine Mehrband- oder k-Band Turingmaschine (k-Band DTM) hat k Bänder mit je einem Kopf.
- Die Übergangsfunktion ist dann von der Form $\delta\colon Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R, S\}^k$
- Zu Beginn steht die Eingabe auf Band 1, sonst stehen überall Blanks. Die Arbeitsweise ist analog zu 1-Band-DTMs definiert.

2. Berechenbarkeit

- **Universelle Turingmaschinen**
  - Bislang *special purpose Computer*:
    eine Sprache – eine Turing-Maschine
  - Allgemein programmierbare Turing-Maschinen:
    universelle Turing-Maschinen
  - Erhalten als Eingabe die Beschreibung einer Turingmaschine und simulieren diese Maschine
  - Benötigen dafür eine einheitliche Beschreibung von Turingmaschinen durch sog. *Gödel-Nummern*

2. Berechenbarkeit

Definition Gödelnummern

Sei M eine 1-Band-Turingmaschine mit

$$Q = \{q_0, \dots, q_n\},$$
$$q_{accept} = q_{n-1},$$
$$q_{reject} = q_n.$$

Sei $X_1 = 0, X_2 = 1, X_3 = t, D_1 = L, D_2 = R$.

Wir kodieren $\delta(q_i, X_j) = (q_k, X_l, D_m)$ durch $0^{i+1}10^j10^{k+1}10^l10^m$.

$Code_r$: Kodierung des r-ten Eintrags für $\delta, 1 \leq r \leq 4(n-1)$

Gödelnummer $\langle M \rangle = 111Code_1 11Code_2 11 \dots 11Code_g 111$

2. Berechenbarkeit

$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$

$\Sigma = \{0,1\}$

$\Gamma = \Sigma \cup \{t\}$

$\delta(q_0, 0) = (q_{reject}, 0, R)$

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

$\delta(q_0, t) = (q_{accept}, t, R)$

$L = \{1^n \mid n \geq 0\}$

Gödel-Nummer:

$1110101000101001101001010010011010001001000100111$

2. Berechenbarkeit

Definition Universelle Turingmaschine

Eine Turingmaschine M_0 heißt **universell**, falls für jede 1-Band-Turingmaschine M und jedes x aus $\{0,1\}^*$ gilt:

- M_0 gestartet mit $\langle M\rangle x$ hält genau dann, wenn M gestartet mit x hält.
- M_0 akzeptiert $\langle M\rangle x$ genau dann, wenn M das Wort x akzeptiert.

Satz

Es gibt eine universelle 2-Band Turingmaschine.

2. Berechenbarkeit

Die Sprache $\mathrm{Gödel}$:

Sprache $\mathrm{Gödel} \coloneqq \{w \text{ aus } \{0,1\}^* \mid w \text{ ist die Gödel−Nummer einer DTM}\}$

Lemma

Die Sprache $\mathrm{Gödel}$ ist entscheidbar.

Die Sprache States:

Sprache $\mathrm{States} \coloneqq \{(\langle M\rangle, d) \mid M \text{ besitzt mindestens } d \text{ Zustände}\}$

Lemma

Die Sprache States ist entscheidbar.

Das Halteproblem

$H := \{(\langle M \rangle, x) \mid M \text{ ist DTM, die gestartet mit Eingabe } x \text{ hält}\}$

Satz

Das Halteproblem ist rekursiv aufzählbar.

Die Sprache Useful

$$\mathrm{Useful} \coloneqq \left\{ (\langle M \rangle, q) \mid \begin{array}{l} M \text{ ist DTM mit Zustand } q \text{, und es gibt eine Eingabe } w, \\ \text{so dass } M \text{ gestartet mit } w \text{ in den Zustand } q \text{ gerät} \end{array} \right\}$$

Satz

Die Sprache Useful ist rekursiv aufzählbar.

2. Berechenbarkeit

Aufzählung von binären Eingabefolgen:

- für alle natürlichen Zahlen i sei $w_i = w$, falls $bin(i) = 1w$
- damit werden alle möglichen w aus $\{0,1\}^*$ aufgezählt

Aufzählung von Turingmaschinen:

M_i ist:

- M_{reject}, falls i keine Gödelnummer ist
- M, falls $bin(i)$ die Gödelnummer der DTM M ist, d.h. $\langle M\rangle = bin(i)$

2. Berechenbarkeit

Die Sprache Diag

$\mathrm{Diag} \coloneqq \{w \text{ in } \{0,1\}^* \mid w = w_i \text{ und die DTM } M_i \text{ akzeptiert } w \text{ nicht}\}$

Satz

Die Sprache Diag ist **nicht** rekursiv aufzählbar.

2. Berechenbarkeit

	M_1	M_2	M_3	… …	M_7	… …	M_i	… …
w_1	na	na	na		na		na	
w_2	na	na	na		na		na	
w_3	na	na	na		na		na	
$\vdots$								
w_7	na	na	na		na		a	
$\vdots$								
w_i	na	na	na		na		a	

Diagonale

Tabelle für Akzeptanz/Nichtakzeptanz von DTMs

Quelle: Skript Johannes Blömer, Universität Paderborn

Reduktionen

Formalisierung von

- Sprache A ist nicht schwerer als Sprache B

Idee

- Algorithmus/DTM für B kann genutzt werden,
  um A zu akzeptieren/entscheiden.

Zwei einfache Sprachen

$P \coloneqq \{w \text{ in } \{0,1\}^* \mid w \text{ ist ein Palindrom}\}$

$$XOR \coloneqq \left\{ \begin{matrix} (a,b,c) \text{ in } \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \mid a,b,c \text{ haben die gleiche} \\ \text{Länge und } a \oplus b = c \end{matrix} \right\}$$

$f\colon \{0,1\}^* \to \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$

$w \to \left(w, w^R, 0^{|w|}\right)$

2. Berechenbarkeit

Von XOR und f zu P

M_P bei Eingabe w

1. Berechne mit M_f das Tripel $f(w) = (w, w^R, 0^{|w|})$.
2. Simuliere M_{XOR} mit Eingabe $f(w)$.
3. Falls M_{XOR} die Eingabe $f(w)$ akzeptiert, akzeptiere w.
4. Falls M_{XOR} die Eingabe $f(w)$ ablehnt, lehne w ab.

M_{XOR} entscheidet XOR, M_f berechnet f.

2. Berechenbarkeit

Definition Reduktionen

L' heißt reduzierbar auf L, falls es eine Funktion $f: \{0,1\}^* \rightarrow \{0,1\}^*$ gibt mit

1. Für alle w aus $\{0,1\}^*$ gilt:
   w ist in L' genau dann, wenn $f(w)$ in L
2. Funktion f ist berechenbar, d.h., es gibt eine DTM M_f, die die Funktion f berechnet.

f heißt Reduktion von L' auf L, geschrieben $L' \leq L$.

2. Berechenbarkeit

Definition

Eine DTM M berechnet die Funktion $f\colon \Sigma^* \rightarrow \Gamma$, falls für alle w aus Σ^* die Berechnung von M mit Eingabe w in einer akzeptierenden Konfiguration hält und dabei der Bandinhalt $f(w)$ ist.

Hierbei werden ▶ und alle t ignoriert.

Lemma

Seien L' und L Sprachen mit $L' \leq L$. Dann gilt:

1. Ist L entscheidbar, so ist auch L' entscheidbar.
2. Ist L rekursiv aufzählbar, so ist auch L' rekursiv aufzählbar.

2. Berechenbarkeit

Von L und f zu L'

M' bei Eingabe w

1. Berechne mit M_f die Folge $f(w)$.
2. Simuliere M mit Eingabe $f(w)$.
3. Falls M die Eingabe $f(w)$ akzeptiert, akzeptiere w.
4. Falls M die Eingabe $f(w)$ ablehnt, lehne w ab.

Akzeptanz- und Halteproblem

$H \coloneqq \{\langle M \rangle x \mid M \text{ ist DTM, die gestartet mit Eingabe } x \text{ hält}\}$

$A \coloneqq \{\langle M \rangle x \mid M \text{ ist DTM, die die Eingabe } x \text{ akzeptiert}\}$

Lemma

Das Halteproblem kann auf das Akzeptanzproblem reduziert werden.

$$H \leq A$$

Formale Sprachen und Komplexitätstheorie

WS 2018/19

Robert Elsässer

1. Einführung

Definition

Eine *(deterministische 1-Band) Turingmaschine* (DTM) wird beschrieben durch ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$.

Dabei sind Q, Σ, Γ endliche, nichtleere Mengen und es gilt:

- Σ ist Teilmenge von Γ
- t in $\Sigma \cap \Gamma$ ist das *Blanksymbol* (auch ⊔)
- Q ist die *Zustandsmenge*
- Σ ist das *Eingabealphabet*
- Γ ist das *Bandalphabet*
- q_0 in Q ist der *Startzustand*
- q_{accept} in Q ist der akzeptierende Endzustand
- q_{reject} in Q ist der ablehnende Endzustand
- $\delta: Q \setminus \{q_{accept}, q_{reject}\} \times \Gamma \to Q \times \Gamma \times \{L, R\}$ ist die (partielle) *Übergangsfunktion*.
  Sie ist für kein Argument aus $\{q_{accept}, q_{reject}\} \times \Gamma$ definiert.

1. Einführung

Definition

- Eine Sprache L heißt **rekursiv aufzählbar,** falls es eine Turingmaschine M gibt, die L *akzeptiert.*
- Eine Sprache L heißt **rekursiv** oder **entscheidbar,** falls es eine Turingmaschine M gibt, die L *entscheidet.*

1. Einführung

- Eine Mehrband- oder k-Band Turingmaschine (k-Band DTM) hat k Bänder mit je einem Kopf.
- Die Übergangsfunktion ist dann von der Form $\delta\colon Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R, S\}^k$
- Zu Beginn steht die Eingabe auf Band 1, sonst stehen überall Blanks. Die Arbeitsweise ist analog zu 1-Band-DTMs definiert.

2. Berechenbarkeit

- **Universelle Turingmaschinen**
  - Bislang *special purpose Computer*:
    eine Sprache – eine Turing-Maschine
  - Allgemein programmierbare Turing-Maschinen:
    universelle Turing-Maschinen
  - Erhalten als Eingabe die Beschreibung einer Turingmaschine und simulieren diese Maschine
  - Benötigen dafür eine einheitliche Beschreibung von Turingmaschinen durch sog. *Gödel-Nummern*

2. Berechenbarkeit

Definition Gödelnummern

Sei M eine 1-Band-Turingmaschine mit

$$Q = \{q_0, \dots, q_n\},$$
$$q_{accept} = q_{n-1},$$
$$q_{reject} = q_n.$$

Sei $X_1 = 0, X_2 = 1, X_3 = t, D_1 = L, D_2 = R$.

Wir kodieren $\delta\big(q_i, X_j\big) = (q_k, X_l, D_m)$ durch $0^{i+1}10^j10^{k+1}10^l10^m$.

$Code_r$: Kodierung des r-ten Eintrags für $\delta, 1 \le r \le 4(n-1)$

Gödelnummer $\langle M \rangle = 111Code_1 11Code_2 11 \dots 11Code_g 111$

2. Berechenbarkeit

Definition Universelle Turingmaschine

Eine Turingmaschine M_0 heißt **universell**, falls für jede 1-Band-Turingmaschine M und jedes x aus $\{0,1\}^*$ gilt:

- M_0 gestartet mit $\langle M \rangle x$ hält genau dann, wenn M gestartet mit x hält.
- M_0 akzeptiert $\langle M \rangle x$ genau dann, wenn M das Wort x akzeptiert.

Satz

Es gibt eine universelle 2-Band Turingmaschine.

2. Berechenbarkeit

Die Sprache $\mathrm{Gödel}$:

Sprache $\mathrm{Gödel} \coloneqq \{w \text{ aus } \{0,1\}^* \mid w \text{ ist die Gödel}-\text{Nummer einer DTM}\}$

Lemma

Die Sprache $\mathrm{Gödel}$ ist entscheidbar.

Die Sprache States:

Sprache $\mathrm{States} \coloneqq \{(\langle M \rangle, d) \mid M \text{ besitzt mindestens } d \text{ Zustände}\}$

Lemma

Die Sprache States ist entscheidbar.

Das Halteproblem

$H \coloneqq \{(\langle M \rangle, x) \mid M \text{ ist DTM, die gestartet mit Eingabe } x \text{ hält}\}$

Satz

Das Halteproblem ist rekursiv aufzählbar.

Die Sprache Useful

$$\mathrm{Useful} \coloneqq \left\{ \begin{array}{l} (\langle M\rangle, q) \mid M \text{ ist DTM mit Zustand } q, \text{ und es gibt eine Eingabe } w, \\ \qquad\qquad\quad \text{so dass } M \text{ gestartet mit } w \text{ in den Zustand } q \text{ gerät} \end{array} \right\}$$

Satz

Die Sprache Useful ist rekursiv aufzählbar.

Aufzählung von binären Eingabefolgen:

- für alle natürlichen Zahlen i sei $w_i = w$, falls $bin(i) = 1w$
- damit werden alle möglichen w aus $\{0,1\}^*$ aufgezählt

Aufzählung von Turingmaschinen:

M_i ist:

- M_{reject}, falls i keine Gödelnummer ist
- M, falls $bin(i)$ die Gödelnummer der DTM M ist, d.h. $\langle M \rangle = bin(i)$

Die Sprache Diag

$\mathrm{Diag} \coloneqq \{w \text{ in } \{0,1\}^* \mid w = w_i \text{ und die DTM } M_i \text{ akzeptiert } w \text{ nicht}\}$

Satz

Die Sprache Diag ist **nicht** rekursiv aufzählbar.

2. Berechenbarkeit

Reduktionen

Formalisierung von

- Sprache A ist nicht schwerer als Sprache B

Idee

- Algorithmus/DTM für B kann genutzt werden, um A zu akzeptieren/entscheiden.

2. Berechenbarkeit

Definition Reduktionen

L' heißt reduzierbar auf L, falls es eine Funktion $f: \{0,1\}^* \rightarrow \{0,1\}^*$ gibt mit

1. Für alle w aus $\{0,1\}^*$ gilt:
   w ist in L' genau dann, wenn $f(w)$ in L
2. Funktion f ist berechenbar, d.h., es gibt eine DTM M_f, die die Funktion f berechnet.

f heißt Reduktion von L' auf L, geschrieben $L' \leq L$.

2. Berechenbarkeit

Definition

Eine DTM M berechnet die Funktion $f: \Sigma^* \to \Gamma$, falls für alle w aus Σ^* die Berechnung von M mit Eingabe w in einer akzeptierenden Konfiguration hält und dabei der Bandinhalt $f(w)$ ist.

Hierbei werden ▶ und alle t ignoriert.

Lemma

Seien L' und L Sprachen mit $L' \leq L$. Dann gilt:

1. Ist L entscheidbar, so ist auch L' entscheidbar.
2. Ist L rekursiv aufzählbar, so ist auch L' rekursiv aufzählbar.

2. Berechenbarkeit

Lemma

Seien L' und L Sprachen mit $L' \leq L$. Dann gilt:

1. Ist L entscheidbar, so ist auch L' entscheidbar.
2. Ist L rekursiv aufzählbar, so ist auch L' rekursiv aufzählbar.

Korollar

Seien L' und L Sprachen mit $L' \leq L$. Dann gilt:

1. Ist L' nicht entscheidbar, so ist auch L nicht entscheidbar.
2. Ist L' nicht rekursiv aufzählbar, so ist auch L nicht rekursiv aufzählbar.

Von L und f zu L'

M' bei Eingabe w

1. Berechne mit M_f die Folge $f(w)$.
2. Simuliere M mit Eingabe $f(w)$.
3. Falls M die Eingabe $f(w)$ akzeptiert, akzeptiere w.
4. Falls M die Eingabe $f(w)$ ablehnt, lehne w ab.

Akzeptanz- und Halteproblem

$H \coloneqq \{\langle M \rangle x \mid M \text{ ist DTM, die gestartet mit Eingabe } x \text{ hält}\}$

$A \coloneqq \{\langle M \rangle x \mid M \text{ ist DTM, die die Eingabe } x \text{ akzeptiert}\}$

Lemma

Das Halteproblem kann auf das Akzeptanzproblem reduziert werden.

$$H \leq A$$

Akzeptanzproblem und die Sprache Useful

$A \coloneqq \{\langle M \rangle x \mid M \text{ ist DTM, die die Eingabe } x \text{ akzeptiert}\}$

$$\text{Useful} \coloneqq \left\{ \begin{array}{l} (\langle M \rangle, q) \mid M \text{ ist DTM mit Zustand } q, \text{ und es gibt eine Eingabe } w, \\ \qquad\qquad\qquad \text{so dass } M \text{ gestartet mit } w \text{ in den Zustand } q \text{ gerät} \end{array} \right\}$$

Lemma

Das Akzeptanzproblem kann auf die Sprache Useful reduziert werden.

$$A \leq \text{Useful}$$

Halteproblem

$H \coloneqq \{\langle M \rangle x \mid M \text{ ist DTM, die gestartet mit Eingabe } x \text{ hält}\}$

Satz

Das Halteproblem ist nicht entscheidbar.

Das Komplement des Halteproblems

$$\overline{H} := \left\{ \begin{array}{r} w \text{ aus } \{0,1\}^* \mid w \text{ ist nicht von der Form} \langle M \rangle x \text{ für eine DTM } M, \text{ oder} \\ w = \langle M \rangle x, \text{ wobei } M \text{ gestartet mit Eingabe } x \text{ nicht hält} \end{array} \right\}$$

Korollar

Das Komplement des Halteproblems ist nicht rekursiv aufzählbar.

Korollar

Die Klasse der rekursiv aufzählbaren Sprachen ist von der Klasse der entscheidbaren Sprachen verschieden und nicht gegen Komplementbildung abgeschlossen.

Akzeptanzproblem und die Sprache Useful

$A := \{\langle M \rangle x \mid M \text{ ist DTM, die die Eingabe } x \text{ akzeptiert}\}$

$$\text{Useful} := \left\{ \begin{array}{l} (\langle M \rangle, q) \mid M \text{ ist DTM mit Zustand } q, \text{ und es gibt eine Eingabe } w, \\ \qquad\qquad\quad \text{so dass } M \text{ gestartet mit } w \text{ in den Zustand } q \text{ gerät} \end{array} \right\}$$

Satz

Das Akzeptanzproblem *A* und die Sprache Useful sind nicht entscheidbar.

Halteproblem mit leerem Band

$H_0 \coloneqq \{\langle M \rangle \mid M \text{ ist DTM, die gestartet mit Eingabe } \varepsilon \text{ hält}\}$

Satz

Das Halteproblem mit leerem Band H_0 ist nicht entscheidbar.

Totalitätsproblem

$T_o \coloneqq \{\langle M \rangle \mid M \text{ hält bei jeder Eingabe}\}$

Endlichkeitsproblem

$E_o \coloneqq \{\langle M \rangle \mid M \text{ hält für endlich viele Eingaben}\}$

Äquivalenzproblem

$Q_o \coloneqq \{\langle M \rangle, \langle M' \rangle \mid M \text{ und } M' \text{ akzeptieren die gleiche Sprache}\}$

Satz

Das Äquivalenzproblem und das Totalitätsproblem sind nicht rekursiv aufzählbar.

2. Berechenbarkeit

Der Satz von Rice

Satz

Sei $\mathcal{R}$ die Menge aller berechenbaren Funktionen und sei S eine nicht-triviale Teilmenge von $\mathcal{R}$. Dann ist die Sprache

$$L(S) := \{\langle M \rangle \mid M \text{ berechnet eine Funktion aus } S\}$$

nicht entscheidbar.