Formale Sprachen und Komplexitätstheorie

WS 2019/20

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?

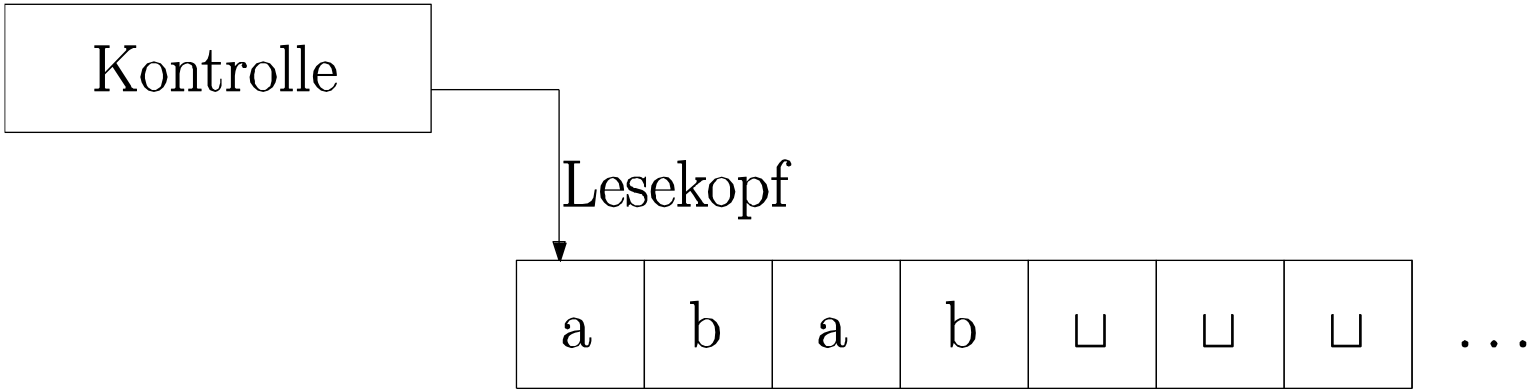
Satz von Turing:

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

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 $\sqcup$)
- 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
- $\delta: Q \setminus \{q_{accept}, q_{reject}\} \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ 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

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 $a, b, c \text{ in } \Gamma, u, v \text{ in } \Gamma^*$ 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:

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

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:

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

Beispiele:

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

2. Berechenbarkeit

Element-Distinctness:

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

Beispiele:

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

2. Berechenbarkeit

Turingmaschine für Element-Distinctness:

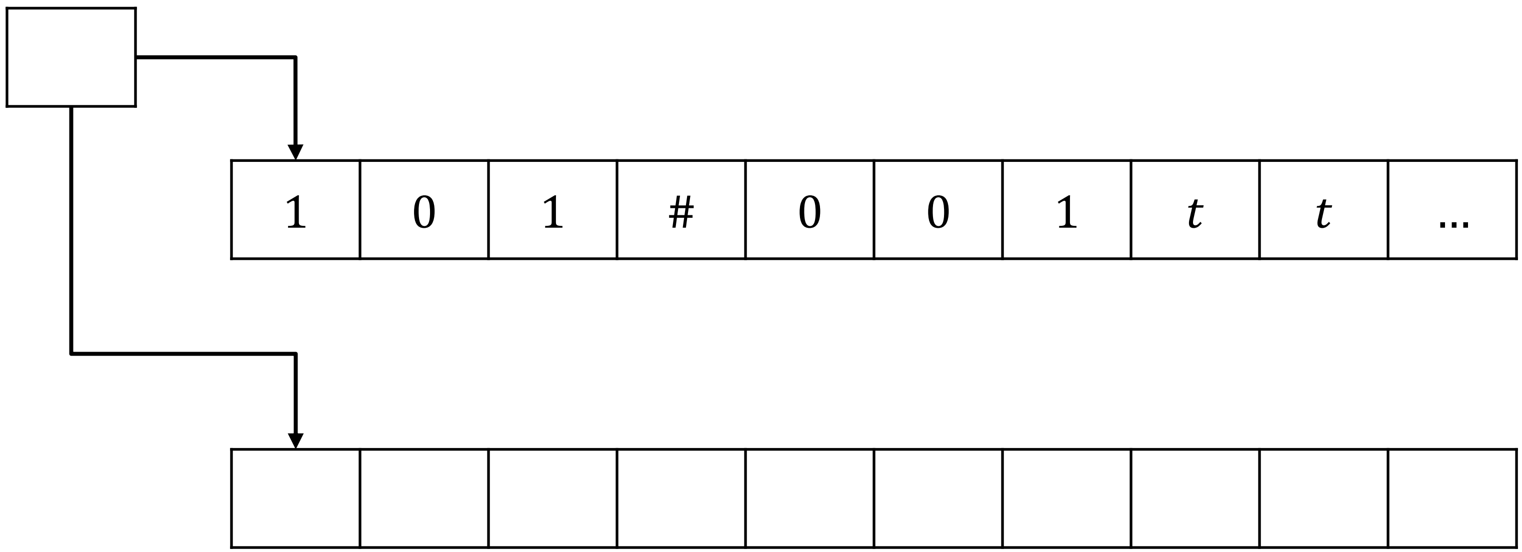
1. Falls das erste Eingabesymbol nicht # ist, lehne ab – sonst ersetze # durch #′.
   Wenn kein weiteres # gefunden, akzeptiere.
2. Finde das nächste # und ersetze es durch #′.
   Wird kein weiteres # gefunden, akzeptiere.
3. Teste, ob die beiden Folgen w_i, w_j rechts der Symbole #′ gleich sind. Wenn 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

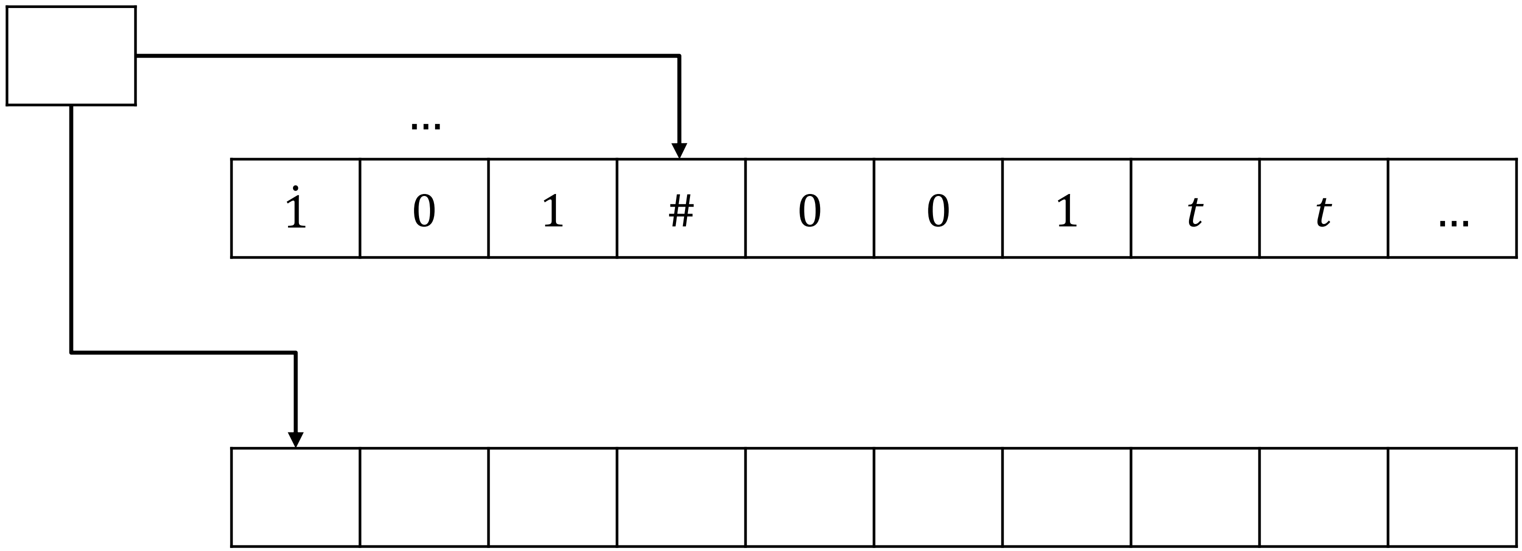
Berechnung, Akzeptieren, Entscheiden, … k-Band Turingmaschinen

- **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}.

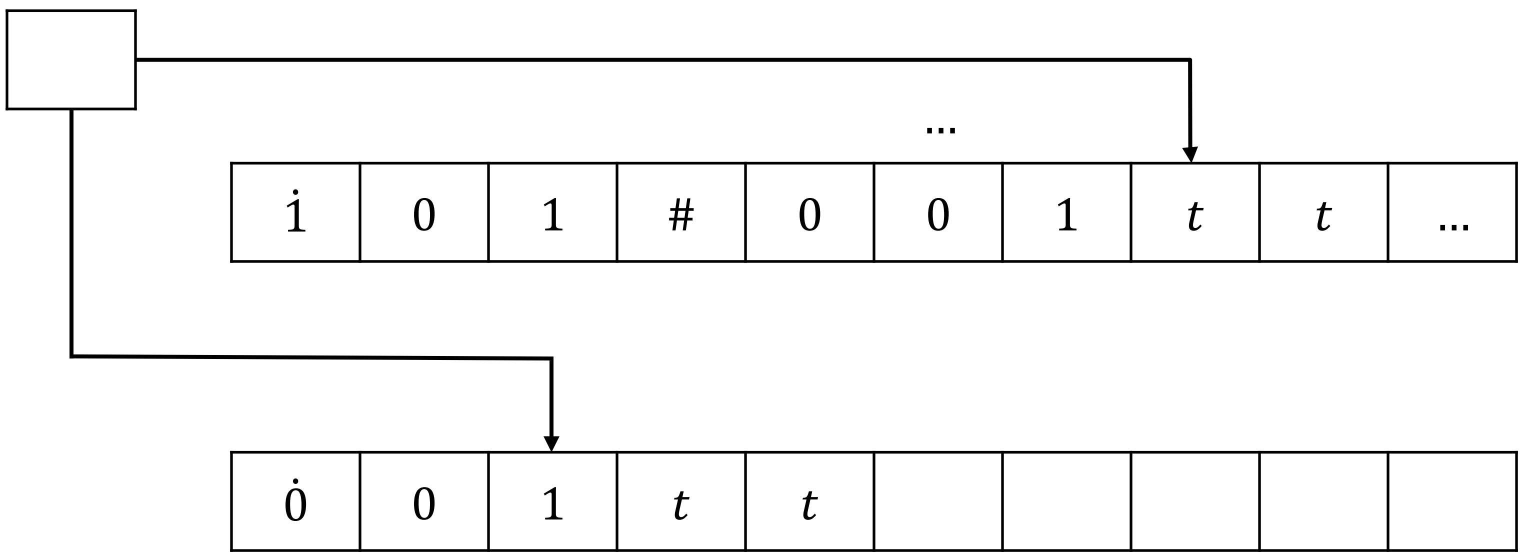
2. Berechenbarkeit



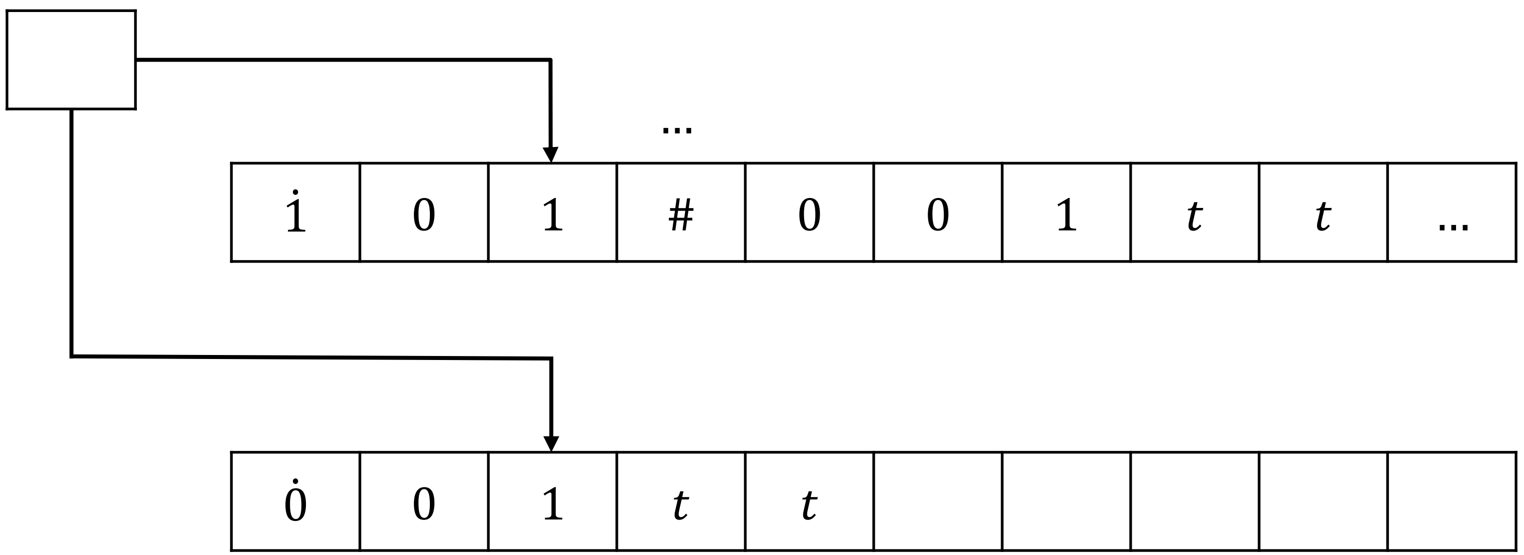
2. Berechenbarkeit



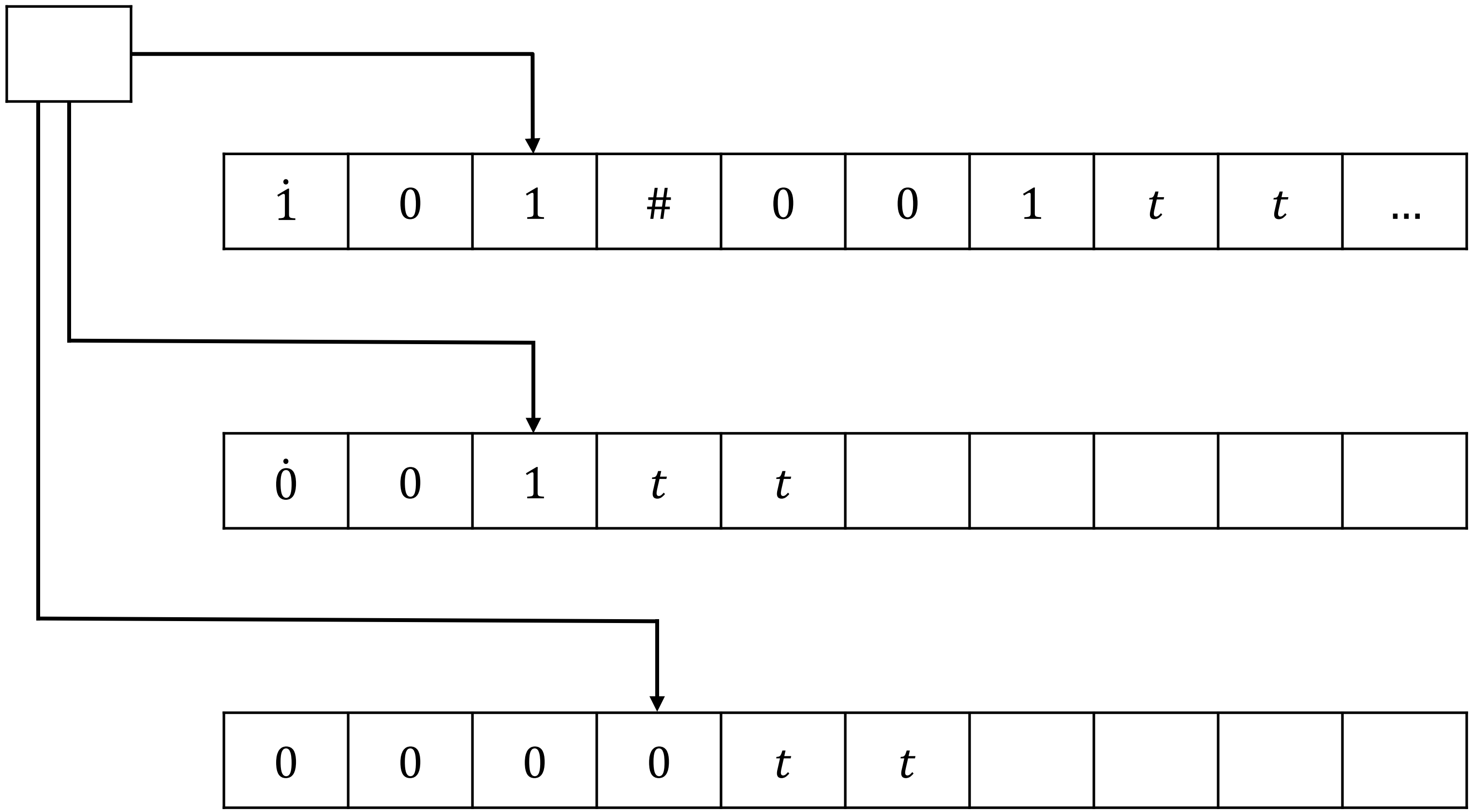
2. Berechenbarkeit

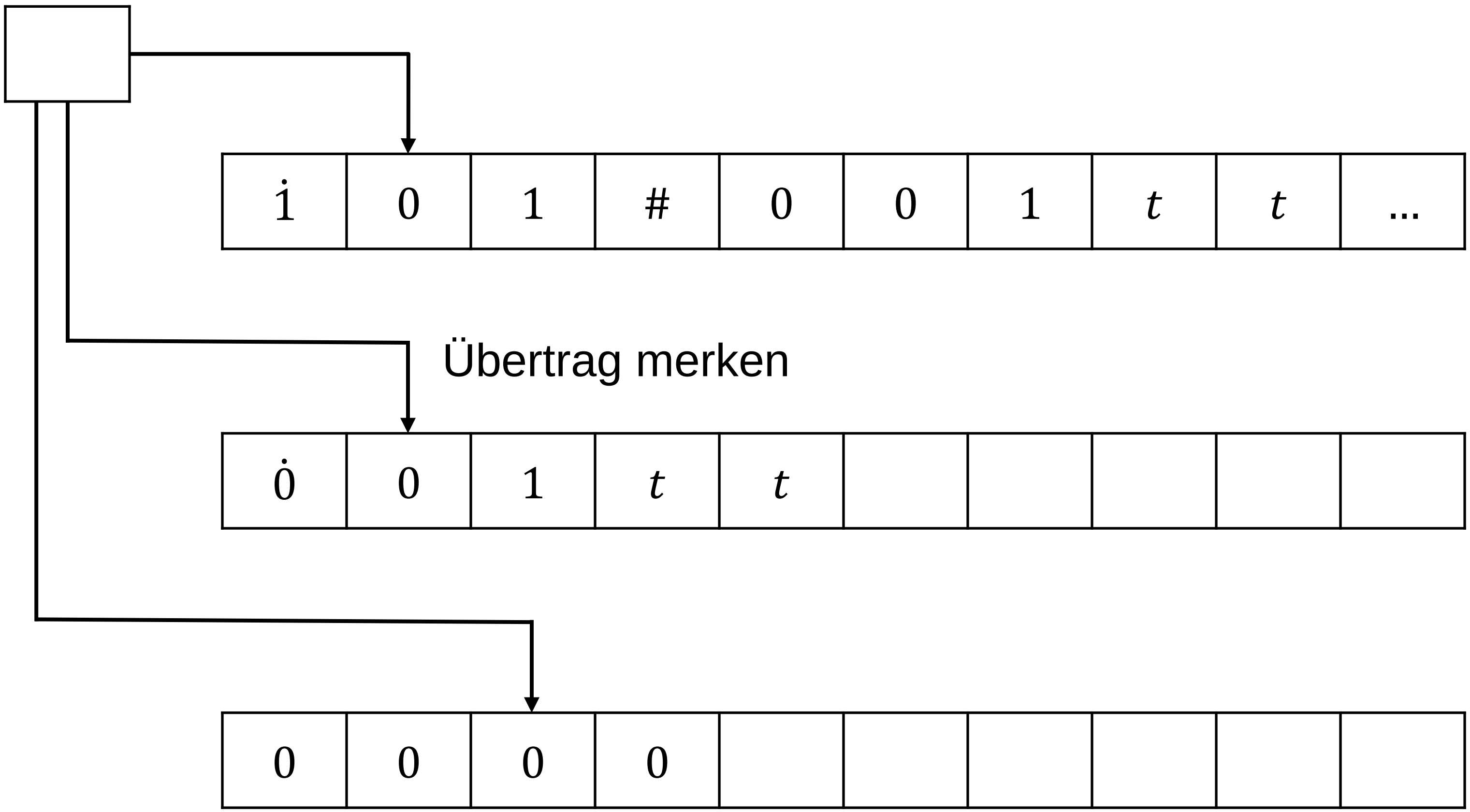


2. Berechenbarkeit



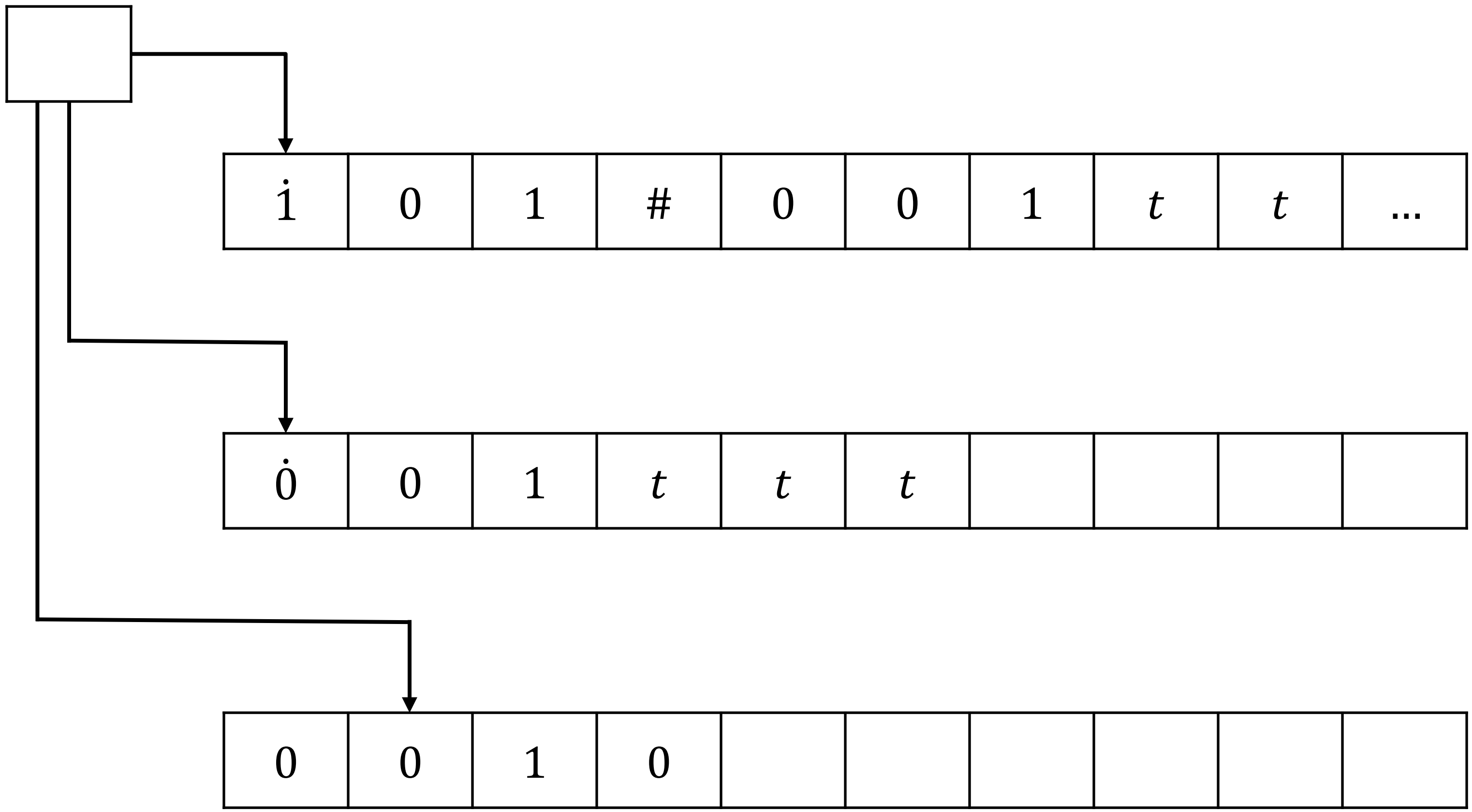
2. Berechenbarkeit



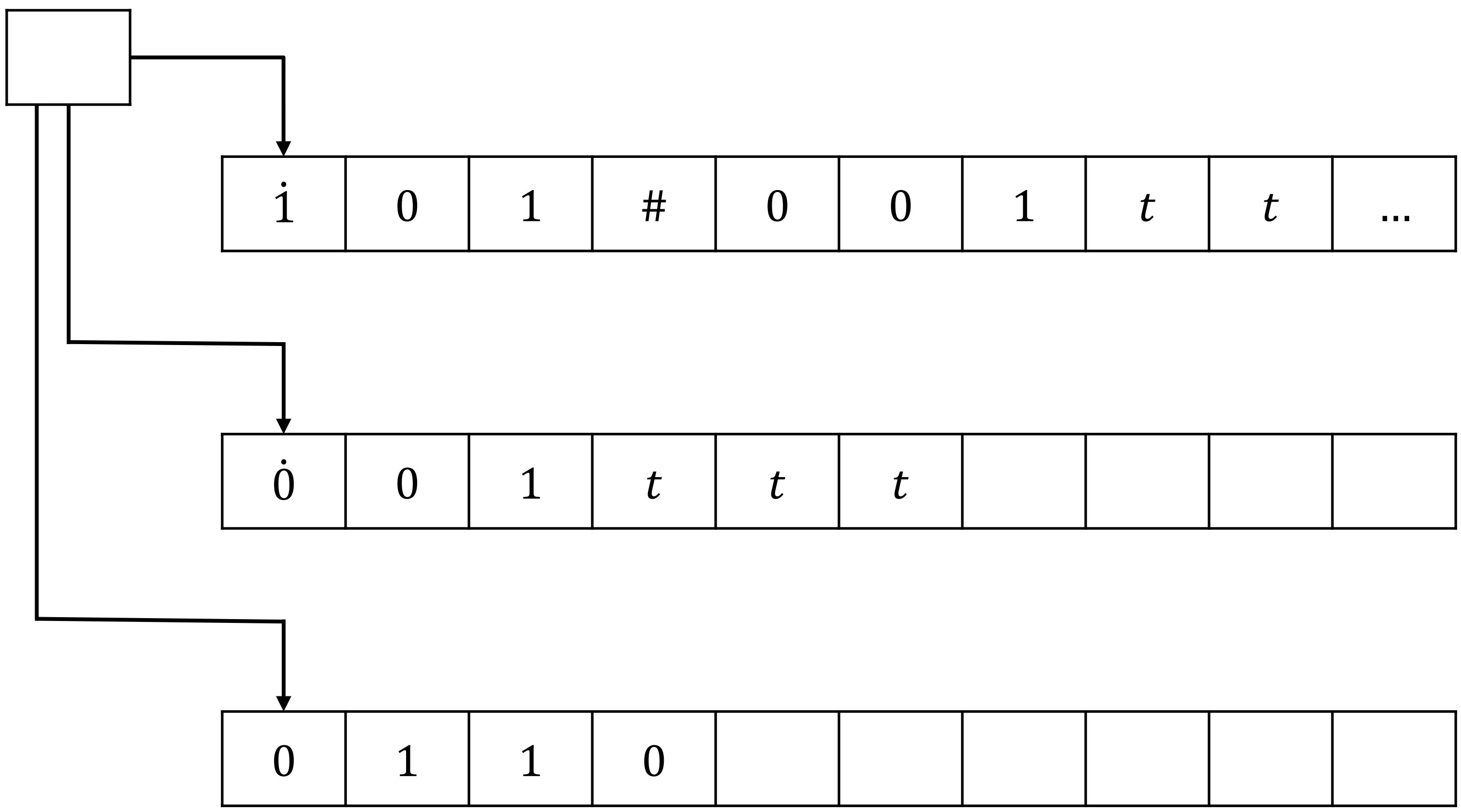


$\dot{1}$ 0 1 # 0 0 1 t t ...
Übertrag merken
$\dot{0}$ 0 1 t t
0 0 0 0

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.

2. Berechenbarkeit

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

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.

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

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.