Kodierungen

§2.1 Definition (partielle Funktion; *partial function*)

Seien A, B Mengen. Relation $\rho \subseteq A \times B$ ist **partielle Funktion**, geschrieben $\rho\colon A \dashrightarrow B$, falls für jedes $a \in A$ höchstens ein $b \in B$ mit $(a, b) \in \rho$ existiert.

Notizen

- Übliche Funktionsschreibweisen auch für partielle Funktionen
- Jede Funktion ist partielle Funktion
- **Definitionsbereich** partieller Funktion $f\colon A \dashrightarrow B$ ist $f^{-1}(B)$
  (Elemente des Vorbereiches A, für die f definiert ist)

$$f^{-1}(B) = \{a \in A \mid \exists b \in B\colon f(a) = b\}$$

- $f^{-1}(B) = A$ für jede Funktion $f\colon A \to B$

Kodierungen

Vereinbarungen

- Beschränkung auf partielle Funktionen

  $f\colon \mathbb{N}^k \dashrightarrow \mathbb{N}$ und $g\colon \Sigma^* \dashrightarrow \Delta^*$ (für Alphabete Σ, Δ)

- 2 Kodierungen für natürliche Zahlen
  - **Unäre** Kodierung: $n \in \mathbb{N}$ repräsentiert durch $a^n = \underbrace{a \cdots a}_{n \text{ mal}}$

    Aus $f\colon \mathbb{N}^k \dashrightarrow \mathbb{N}$ wird $g\colon \{a, \#\}^* \dashrightarrow \{a\}^*$ mit
    $g(a^{n_1} \# a^{n_2} \# \cdots \# a^{n_k}) = a^{f(n_1, \ldots, n_k)}$
  - **Binäre** Kodierung: $n \in \mathbb{N}$ repräsentiert durch $\mathrm{bin}(n) \in \{0,1\}^*$

    Aus $f\colon \mathbb{N}^k \dashrightarrow \mathbb{N}$ wird $g\colon \{0, 1, \#\}^* \to \{0,1\}^*$ mit
    $g(\mathrm{bin}(n_1) \# \mathrm{bin}(n_2) \# \cdots \# \mathrm{bin}(n_k)) = \mathrm{bin}(f(n_1, \ldots, n_k))$

Kodierungen

Kodierung von $f(3,4) = 7$

- Unäre Kodierung

$$g(\underbrace{aaa}_{3} \# \underbrace{aaaa}_{4}) = \underbrace{aaaaaaa}_{7}$$

- Binäre Kodierung

$$g(\underbrace{11}_{2+1} \# \underbrace{100}_{4+0+0}) = \underbrace{111}_{4+2+1}$$

- Andere berechenbare Kodierungen auch möglich

  Dezimalkodierung: $g\colon \{0, 1, \ldots, 9, \#\}^* \dashrightarrow \{0, 1, \ldots, 9\}^*$

Kodierung von Sprachen

§2.2 Definition (Sprachenkodierung)

Für jede Sprache $L \subseteq \Sigma^*$ ist $\mathrm{id}_L\colon \Sigma^* \dashrightarrow \Sigma^*$ gegeben durch

$$\mathrm{id}_L = \{(w, w) \mid w \in L\}$$

Notizen

- 'undef' (oder $\perp$) steht für nicht definierte Funktionswerte
- Alternative Definition

$$\mathrm{id}_L(w) = \begin{cases} w & \text{falls } w \in L \\ \text{undef} & \text{sonst} \end{cases}$$

- Also $\mathrm{id}_L^{-1}(\Sigma^*) = L$

Intuitive Berechenbarkeit

Algorithmus = endliche & eindeutige Handlungsbeschreibung

§2.3 Definition (intuitive Berechenbarkeit; *computability*)

Funktion $f : \Sigma^* \dashrightarrow \Delta^*$ **intuitiv berechenbar** *(computable)*,
falls Algorithmus A_f existiert, so dass für jede Eingabe $w \in \Sigma^*$

- A_f produziert Ergebnis nach endlicher Zeit gdw. $w \in f^{-1}(\Delta^*)$
- A_f produziert Ergebnis $f(w)$ falls $w \in f^{-1}(\Delta^*)$

Notizen

- $w \in f^{-1}(\Delta^*)$ bedeutet "$f(w)$ definiert"
- A_f muss bei Eingabe $w \in f^{-1}(\Delta^*)$ Ergebnis $f(w)$ liefern
- A_f darf bei Eingabe $w \in \Sigma^* \setminus f^{-1}(\Delta^*)$ kein Ergebnis liefern
  (Endlosschleife, Absturz, Exception, etc.)

Intuitive Berechenbarkeit

Weitere Notizen

- Mathematische Existenz ausreichend
  (kann Funktion 2 Formen annehmen, also $f = f_1$ oder $f = f_2$,
  dann reicht intuitive Berechenbarkeit von f_1 und f_2)
- Beschreibungssprache beliebig (C++, Java, Pseudocode, etc.)
- Hardware irrelevant (Architektur, Ablaufmechanismus, etc.)
- Keine Zeit- oder Speicherbeschränkung
  (aber A_f muss bei Eingabe $w \in f^{-1}(\Delta^*)$ letztlich terminieren)

Intuitive Berechenbarkeit

Erklärungsversuch

- E sei Eigenschaft der Welt und $f : \Sigma^* \dashrightarrow \Delta^*$
  (z.B. E = Gültigkeit der Goldbachschen Vermutung)
- Weiterhin gelten $E \to \text{Berechenbar}(f)$ und $\neg E \to \text{Berechenbar}(f)$

$$\begin{aligned} & (E \to \text{Berechenbar}(f)) \wedge (\neg E \to \text{Berechenbar}(f)) \\ \equiv\ & (\neg E \vee \text{Berechenbar}(f)) \wedge (E \vee \text{Berechenbar}(f)) \\ \equiv\ & (\neg E \wedge E) \vee \text{Berechenbar}(f) \\ \equiv\ & \text{Berechenbar}(f) \end{aligned}$$

- Also gilt $\text{Berechenbar}(f)$

Intuitive Berechenbarkeit

- Addition: Funktion $+ : \mathbb{N}^2 \to \mathbb{N}$ intuitiv berechenbar
  - Schulmethode
  - x_1 mal Erhöhung von x_2 für $x_1 + x_2$

- Format-Prüfung: Funktion $\text{id}_L : \{0,1,\#\}^* \dashrightarrow \{0,1,\#\}^*$ mit

$$L = \underbrace{1\,(0 \mid 1)^*\,(\#\,1\,(0 \mid 1)^*)^*}$$

(1, beliebig viele 0 und 1, # und weitere solche Blöcke)

intuitiv berechenbar (L regulär)

Intuitive Berechenbarkeit

$\pi[n]$ = erste n Stellen in Dezimalbruchdarstellung von π für alle $n \in \mathbb{N}$

$$\pi[3] = 314 \qquad \pi[6] = 314159 \qquad \pi[1] = 3$$

- Approximation π: Funktion $\pi : \{a\}^* \to \{0,1,\ldots,9\}^*$ mit

$$\pi(a^n) = \pi[n] \qquad \text{für alle } n \in \mathbb{N}$$

intuitiv berechenbar

- Approximationsalgorithmus für π
- Ausgabe erste n Stellen sobald ausreichende Genauigkeit

Intuitive Berechenbarkeit

- Teilstrings von π: Funktion $\text{sub}_\pi : \{0,1,\ldots,9\}^* \longrightarrow \{0,1\}^*$ mit

$$\text{sub}_\pi(w) = \begin{cases} 1 & \text{falls } w \text{ in } \pi \text{ vorkommt} \\ 0 & \text{sonst} \end{cases}$$

für alle $w \in \{0,\ldots,9\}^*$

Intuitive Berechenbarkeit unklar

Beispiele

$$\text{sub}_\pi(314) = 1 \qquad \text{sub}_\pi(15) = 1 \qquad \text{sub}_\pi(41) = 1$$

Intuitive Berechenbarkeit

- Teilstrings von π: Funktion $\text{sub}_\pi : \{0,1,\ldots,9\}^* \dashrightarrow \{0,1\}^*$ mit

$$\text{sub}_\pi(w) = \begin{cases} 1 & \text{falls } w \text{ in } \pi \text{ vorkommt} \\ \text{undef} & \text{sonst} \end{cases}$$

für alle $w \in \{0,\ldots,9\}^*$

intuitive Berechenbarkeit: intuitiv berechenbar

Beispiele:

$$\text{sub}_\pi(314) = 1 \qquad \text{sub}_\pi(15) = 1 \qquad \text{sub}_\pi(41) = 1$$

Intuitive Berechenbarkeit

- Länge von Nichtteilstrings von π: Funktion $\ell_\pi : \mathbb{N} \dashrightarrow \mathbb{N}$ mit

$$\ell_\pi(n) = \begin{cases} n & \text{falls Sequenz der Länge } n \text{ existiert,} \\ & \text{die nicht in } \pi \text{ vorkommt} \\ \text{undef} & \text{sonst} \end{cases}$$

für alle $n \in \mathbb{N}$

Intuitive Berechenbarkeit intuitiv berechenbar

- Falls alle Sequenzen in π vorkommen, (Eigenschaft E)
  dann ℓ_π überall undefiniert & intuitiv berechenbar
- Sonst existiert kürzeste Sequenz der Länge k, die nicht in π vorkommt & ℓ_π intuitiv berechenbar, da

$$\ell_\pi(n) = f_k(n) = \begin{cases} n & \text{falls } n \geq k \\ \text{undef} & \text{sonst} \end{cases}$$

$(\neg E \to \exists k((\ell_\pi = f_k) \wedge \text{Berechenbar}(f_k))$ also
$\neg E \to \text{Berechenbar}(\ell_\pi))$

Intuitive Berechenbarkeit

- Wortproblem Sprache $L \subseteq \Sigma^*$: Funktion $\chi_L \colon \Sigma^* \to \{0,1\}^*$ mit

$$\chi_L(w) = \begin{cases} 1 & \text{falls } w \in L \\ 0 & \text{sonst} \end{cases} \qquad \text{für alle } w \in \Sigma^*$$

Intuitive Berechenbarkeit

- L kontextsensitiv: intuitiv berechenbar
- Typ-0-Sprache L: unklar/nicht intuitiv berechenbar

Intuitive Berechenbarkeit

- Aufzählung einer Sprache $L \subseteq \Sigma^*$: Funktion $\rho_L \colon \Sigma^* \dashrightarrow \{0,1\}^*$ mit

$$\rho_L(w) = \begin{cases} 1 & \text{falls } w \in L \\ \text{undef} & \text{sonst} \end{cases} \qquad \text{für alle } w \in \Sigma^*$$

intuitive Berechenbarkeit:

- für kontextsensitive Sprache L: intuitiv berechenbar
- für Typ-0-Sprache L: intuitiv berechenbar

Intuitive Berechenbarkeit

Problem

- Wie argumentiert man "nicht intuitiv berechenbar"?
  (muss für beliebige Algorithmen funktionieren)

Ansatz der modellbezogenen Berechenbarkeit

- Festlegung Berechnungsmodell (Grammatik, Turingmaschine, etc.)
- Klärt Begriff 'Algorithmus'

Wiederholung: Chomsky-Grammatik

Beispiel (§1.4)

Grammatik $G = (\{S, S', A, B, E\}, \{a, b\}, S, P)$ mit Produktionen P

$$\begin{array}{llll} S \to S'E & S' \to aS'a & S' \to bS'b & S' \to E \\ Ea \to EA & Aa \to aA & Ab \to bA & AE \to Ea \\ Eb \to EB & Ba \to aB & Bb \to bB & BE \to Eb \\ EE \to \varepsilon & & & \end{array}$$

Ableitungsschritte

$$\begin{aligned} S &\Rightarrow_G S'E \Rightarrow_G aS'aE \Rightarrow_G abS'baE \Rightarrow_G abEbaE \\ &\Rightarrow_G abEBaE \Rightarrow_G abEaBE \Rightarrow_G abEaEb \Rightarrow_G abEAEb \\ &\Rightarrow_G abEEab \Rightarrow_G ab\varepsilon ab = abab \end{aligned}$$

Wiederholung: Chomsky-Grammatik

Analyse der Funktionsweise

- Ziel ww mit $w \in \{a, b\}^*$
- Erzeuge zunächst wEw^RE

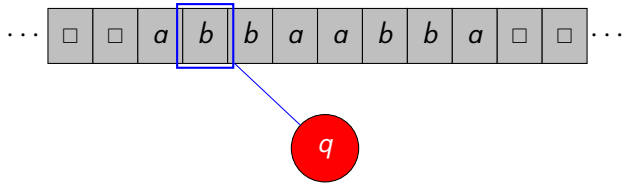
$$S \to S'E \qquad S' \to aS'a \qquad S' \to bS'b \qquad S' \to E$$

- Symbol hinter linkem E direkt hinter rechtes E bewegen

$$\begin{array}{llll} Ea \to EA & Aa \to aA & Ab \to bA & AE \to Ea \\ Eb \to EB & Ba \to aB & Bb \to bB & BE \to Eb \end{array}$$

- Invertiert w^R; liefert w und Satzform $wEEw$
- Löschen Begrenzer EE mit Produktion $EE \to \varepsilon$

Turingmaschine

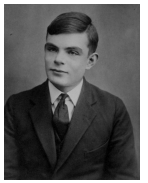


Notizen

- Beidseitig unbeschränktes Arbeitsband
- Endliche Kontrolle (zustandsgesteuert)
- Mobiler Lese- & Schreibkopf
- Eingabe auf Band; Symbole überschreibbar (Speicher)

Alan Turing (* 1912; † 1954)

- Engl. Informatiker
- Brach dtsch. Enigma-Verschlüsselung
- Verurteilt wegen Homosexualität; akzeptierte Kastration; 2013 offiziell rehabilitiert



Turingmaschine

§2.4 Definition (Turingmaschine; *Turing machine*)

Turingmaschine ist Tupel $M = (Q, \Sigma, \Gamma, \Delta, \square, q_0, q_+, q_-)$

- endl. Menge Q von Zuständen *(states)* mit $Q \cap \Gamma = \emptyset$
- endl. Menge Σ von Eingabesymbolen *(input symbols)*
- endl. Menge Γ von Arbeitssymbolen *(work symbols)* mit $\Sigma \subseteq \Gamma$
- Übergangsrelation *(transition relation)*
  $\Delta \subseteq \Big((Q \setminus \{q_+, q_-\}) \times \Gamma\Big) \times \Big(Q \times \Gamma \times \{\triangleleft, \triangleright, \diamond\}\Big)$
- Leersymbol *(blank)* $\square \in \Gamma \setminus \Sigma$ $(\Gamma_M = \Gamma \setminus \{\square\})$
- Startzustand *(initial state)* $q_0 \in Q$
- Akzeptierender Zustand *(accepting state)* $q_+ \in Q$
- Ablehnender Zustand *(rejecting state)* $q_- \in Q$

$\triangleleft$ = gehe nach links; $\triangleright$ = gehe nach rechts; $\diamond$ = keine Bewegung

Turingmaschine

Damit programmieren?

- Einfaches Modell (vereinfacht Beweise Nichtberechenbarkeit)
- Gleichmächtig wie gebräuchliche Programmiersprachen (C++, Java, Perl, Python, etc.)
- Nicht komfortabel
- Übergangsrelation $\widehat{=}$ Programm
- Arbeitsband $\widehat{=}$ Speicher (kein Direktzugriff)

Turingmaschine

Notation: $(q, \gamma) \to (q', \gamma', d) \in \Delta$ statt $((q, \gamma), (q', \gamma', d)) \in \Delta$

§2.5 Beispiel (Turingmaschine = TM)

TM $M = (\{q_0, q, q_a, q'_a, q_b, q'_b, f, \bot\}, \{a, b\}, \{a, b, \square\}, \Delta, \square, q_0, f, \bot)$
mit den Übergängen Δ

$$(q_0, a) \to (q_a, \square, \triangleright) \quad (q_0, b) \to (q_b, \square, \triangleright) \quad (q_0, \square) \to (f, \square, \diamond)$$
$$(q_a, a) \to (q_a, a, \triangleright) \quad (q_a, b) \to (q_a, b, \triangleright) \quad (q_a, \square) \to (q'_a, \square, \triangleleft)$$
$$(q_b, a) \to (q_b, a, \triangleright) \quad (q_b, b) \to (q_b, b, \triangleright) \quad (q_b, \square) \to (q'_b, \square, \triangleleft)$$
$$(q'_a, a) \to (q, \square, \triangleleft) \quad (q'_b, b) \to (q, \square, \triangleleft)$$
$$(q, a) \to (q, a, \triangleleft) \quad (q, b) \to (q, b, \triangleleft) \quad (q, \square) \to (q_0, \square, \triangleright)$$

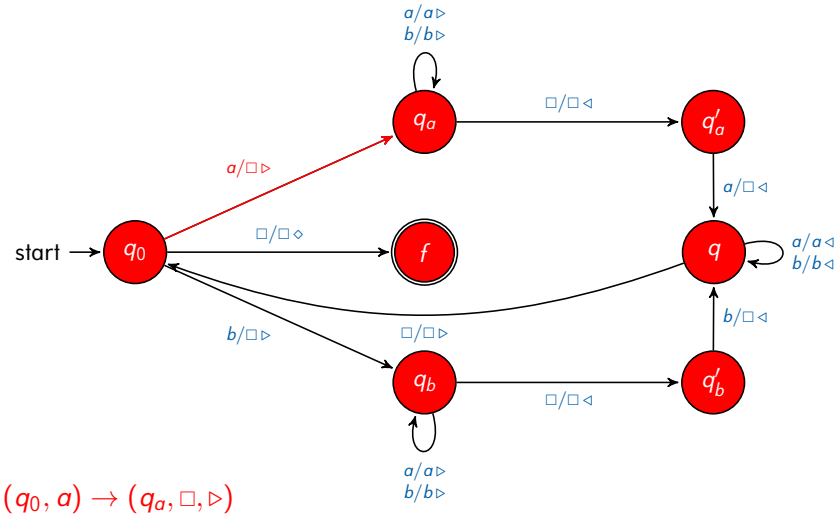
Turingmaschine

Notizen

- Übergang $(q, \gamma) \to (q', \gamma', d)$
  - Vorbedingungen
    1. Aktueller Zustand q
    2. Zeichen γ in Bandzelle, auf der der Kopf steht
  - Konsequenzen
    1. TM wechselt in Zustand q'
    2. γ' überschreibt Inhalt aktueller Bandzelle (ersetzt γ)
    3. Kopf bewegt sich Richtung $d \in \{\triangleleft, \triangleright, \diamond\}$
- Übergänge mit aktuellem Zustand $q \in \{q_+, q_-\}$ verboten
  (Übergänge aus Finalzustand heraus nicht erlaubt)

$\triangleleft$ = gehe nach links; $\triangleright$ = gehe nach rechts; $\diamond$ = keine Bewegung

Turingmaschine



$(q_0, a) \to (q_a, \square, \triangleright)$

Turingmaschine

1. Ausgangssituation
   - Eingabe auf Band (andere Zellen □)
   - TM in Startzustand q_0
   - Kopf auf erstem Symbol der Eingabe (auf □ falls Eingabe leer)
2. Übergänge gemäß Δ
3. Haltebedingung
   - Aktueller Zustand final; akzeptierend q_+ oder ablehnend q_-
   - Kein passender Übergang → TM hält nicht ordnungsgemäß (Ausnahme)

Akzeptanz Eingabe

Existenz Übergänge von Ausgangssituation in akzeptierenden Zustand

Turingmaschine

Beispiel (§2.5)

TM $M = (\{q_0, q, q_a, q'_a, q_b, q'_b, f, \bot\}, \{a, b\}, \{a, b, \Box\}, \Delta, \Box, q_0, f, \bot)$

$$(q_0, a) \to (q_a, \Box, \triangleright) \quad (q_0, b) \to (q_b, \Box, \triangleright) \quad (q_0, \Box) \to (f, \Box, \diamond)$$

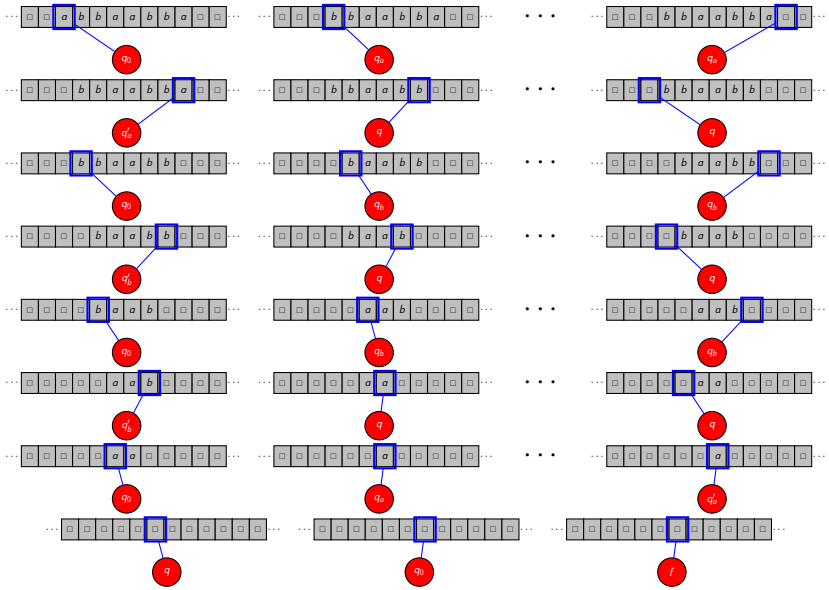
$$(q_a, a) \to (q_a, a, \triangleright) \quad (q_a, b) \to (q_a, b, \triangleright) \quad (q_a, \Box) \to (q'_a, \Box, \triangleleft)$$

$$(q_b, a) \to (q_b, a, \triangleright) \quad (q_b, b) \to (q_b, b, \triangleright) \quad (q_b, \Box) \to (q'_b, \Box, \triangleleft)$$

$$(q'_a, a) \to (q, \Box, \triangleleft) \quad (q'_b, b) \to (q, \Box, \triangleleft)$$

$$(q, a) \to (q, a, \triangleleft) \quad (q, b) \to (q, b, \triangleleft) \quad (q, \Box) \to (q_0, \Box, \triangleright)$$

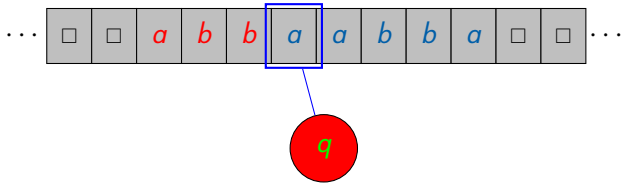
Turingmaschine



Turingmaschine

Satzform

- Globale Systemsituation als Wort
  (Arbeitsband, Position des Kopfes und interner Zustand)
- Kürzen von $\Box$ vom linken und rechten Rand, aber nicht unter Kopf
- Satzform ist $u\,q\,w$
  1. Arbeitsbandbereich $u \in \Gamma^*$ links des Kopfes
  2. Zustand $q \in Q$
  3. Arbeitsbandbereich $w \in \Gamma^+$ unter und rechts des Kopfes
- Situation $abb\ q\ aabba$

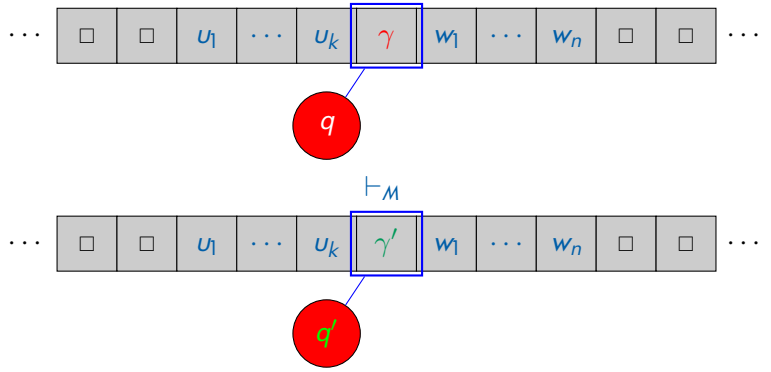


Turingmaschine

§2.6 Definition (Ableitungsrelation — keine Bewegung)

$$u\,q\,\gamma w \vdash_M u\,q'\,\gamma' w$$

falls $(q, \gamma) \to (q', \gamma', \diamond) \in \Delta$

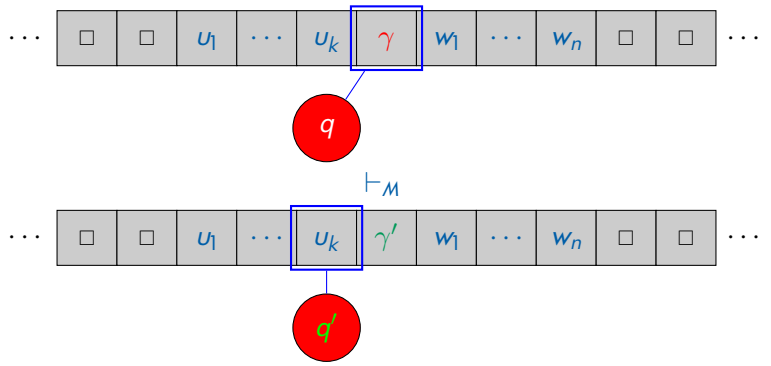


Turingmaschine

§2.6 Definition (Ableitungsrelation — Schritt nach links)

$$u\ q\ \gamma w \vdash_M \begin{cases} \varepsilon\ q'\ \square\gamma' w & \text{falls } u = \varepsilon \\ u'\ q'\ \gamma''\gamma' w & \text{falls } u = u'\gamma'' \text{ mit } \gamma'' \in \Gamma \end{cases}$$

falls $(q, \gamma) \to (q', \gamma', \triangleleft) \in \Delta$

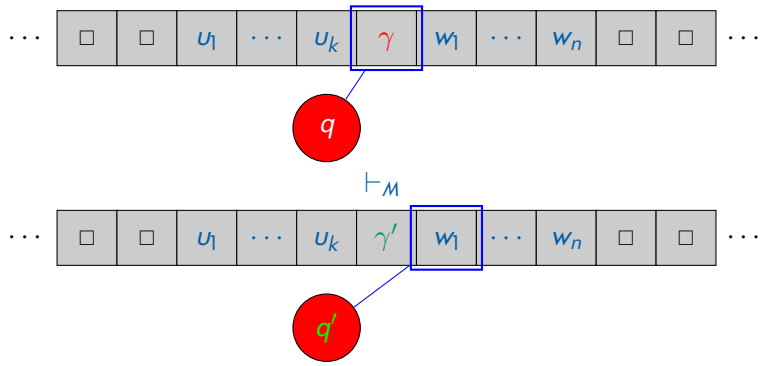


Turingmaschine

§2.6 Definition (Ableitungsrelation — Schritt nach rechts)

$$u\ q\ \gamma w \vdash_M \begin{cases} u\gamma'\ q'\ \square & \text{falls } w = \varepsilon \\ u\gamma'\ q'\ w & \text{sonst} \end{cases}$$

falls $(q, \gamma) \to (q', \gamma', \triangleright) \in \Delta$



Turingmaschine

§2.7 Definition (akzeptierte Sprache; *accepted language*)

Akzeptierte Sprache von TM $M = (Q, \Sigma, \Gamma, \Delta, \square, q_0, q_+, q_-)$ ist

$$L(M) = \{w \in \Sigma^* \mid \exists u, v \in \Gamma^* : \varepsilon\ q_0\ w\square \vdash_M^*\ u\ q_+\ v\}$$

Akzeptanz Eingabe

- Ausgangssituation $\varepsilon\ q_0\ w$ für Eingabe w
- TM **akzeptiert** Eingabe w falls Übergänge von Ausgangssituation $\varepsilon\ q_0\ w$ in akzeptierenden Zustand q_+ existieren

Turingmaschine

Beispiel (§2.5)

TM $M = (\{q_0, q, q_a, q'_a, q_b, q'_b, f, \perp\}, \{a, b\}, \{a, b, \square\}, \Delta, \square, q_0, f, \perp)$

$$(q_0, a) \to (q_a, \square, \triangleright) \quad (q_0, b) \to (q_b, \square, \triangleright) \quad (q_0, \square) \to (f, \square, \diamond)$$

$$(q_a, a) \to (q_a, a, \triangleright) \quad (q_a, b) \to (q_a, b, \triangleright) \quad (q_a, \square) \to (q'_a, \square, \triangleleft)$$

$$(q_b, a) \to (q_b, a, \triangleright) \quad (q_b, b) \to (q_b, b, \triangleright) \quad (q_b, \square) \to (q'_b, \square, \triangleleft)$$

$$(q'_a, a) \to (q, \square, \triangleleft) \quad (q'_b, b) \to (q, \square, \triangleleft)$$

$$(q, a) \to (q, a, \triangleleft) \quad (q, b) \to (q, b, \triangleleft) \quad (q, \square) \to (q_0, \square, \triangleright)$$