

Theorie der Informatik

12. Kontextsensitive und Typ-0-Sprachen

Malte Helmert Gabriele Röger

Universität Basel

8. April 2015

Theorie der Informatik

8. April 2015 — 12. Kontextsensitive und Typ-0-Sprachen

12.1 Kontextsensitive und allgemeine Grammatiken

12.2 Turingmaschinen

12.3 Abschlusseigenschaften und Entscheidbarkeit

12.4 Zusammenfassung

12.1 Kontextsensitive und allgemeine Grammatiken

Wiederholung: (kontextsensitive) Grammatiken

Definition (Grammatik)

Eine **Grammatik** ist ein 4-Tupel (Σ, V, P, S) mit

- ▶ Σ endliches Terminalalphabet,
- ▶ V endliche Menge von Variablen (mit $V \cap \Sigma = \emptyset$),
- ▶ $P \subseteq (V \cup \Sigma)^+ \times (V \cup \Sigma)^*$ endliche Menge von Regeln, und
- ▶ $S \in V$ Startvariable.

Typ 0 und Typ 1

- ▶ Jede **Grammatik** ist eine **Typ-0-Grammatik**.
- ▶ Bei **kontextsensitiven (Typ-1)** Grammatiken gilt für alle Regeln $w_1 \rightarrow w_2$, dass $|w_1| \leq |w_2|$.
Einzige Ausnahme: Regel $S \rightarrow \varepsilon$ ist für Startsymbol S erlaubt, falls S auf keiner rechten Regelseite vorkommt.

12.2 Turingmaschinen

Automatenmodell für Typ-1- und Typ-0-Sprachen?



Endliche Automaten akzeptieren genau die regulären Sprachen, Kellerautomaten genau die kontextfreien Sprachen. Gibt es auch ein Automatenmodell für kontextsensitive und Typ-0-Sprachen?

Ja! (Linear beschränkte) Turingmaschinen.

Foto mit freundlicher Genehmigung von imagerymajestic / FreeDigitalPhotos.net

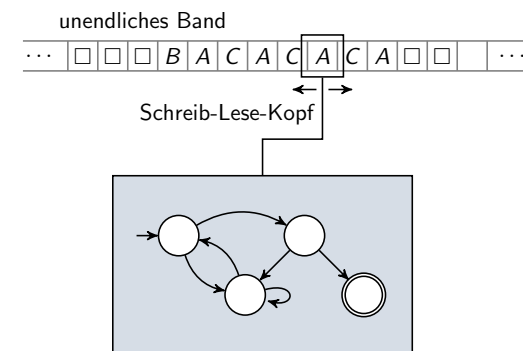
Alan Turing (1912-1954)



Foto mit freundlicher Genehmigung von Jon Callas / wikimedia commons

- ▶ Britischer Logiker, Mathematiker, Kryptoanalytiker und Informatiker
- ▶ Wichtigste Arbeit (für uns): On Computable Numbers, with an Application to the „Entscheidungsproblem“
→ **Turingmaschine**
- ▶ Mitarbeit an **Enigma-Entschlüsselung**
- ▶ Wegen Homosexualität verurteilt, wurde aber im Dezember 2013 von Elizabeth II begnadigt
- ▶ **Turing-Award** wichtigster Wissenschaftspreis in der Informatik

Turingmaschine: konzeptuell



Nichtdeterministische Turingmaschine: Definition

Definition (Nichtdeterministische Turingmaschine)

Eine (nichtdeterministische) **Turingmaschine (NTM)** ist gegeben durch ein 7-Tupel $M = \langle Z, \Sigma, \Gamma, \delta, z_0, \square, E \rangle$ mit:

- ▶ Z endliche, nicht-leere Menge von **Zuständen**
- ▶ $\Sigma \neq \emptyset$ endliches **Eingabealphabet**
- ▶ $\Gamma \supset \Sigma$ endliches **Bandalphabet**
- ▶ $\delta : (Z \setminus E) \times \Gamma \rightarrow \mathcal{P}(Z \times \Gamma \times \{L, R, N\})$ **Übergangsfunktion**
- ▶ $z_0 \in Z$ **Startzustand**
- ▶ $\square \in \Gamma \setminus \Sigma$ **Blank-Zeichen**
- ▶ $E \subseteq Z$ **Endzustände**

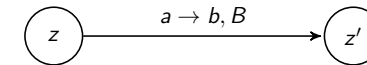
Turingmaschine: Übergangsfunktion

Sei $M = \langle Z, \Sigma, \Gamma, \delta, z_0, \square, E \rangle$ nichtdeterministische Turingmaschine.

Was bedeutet Übergangsfunktion δ intuitiv?

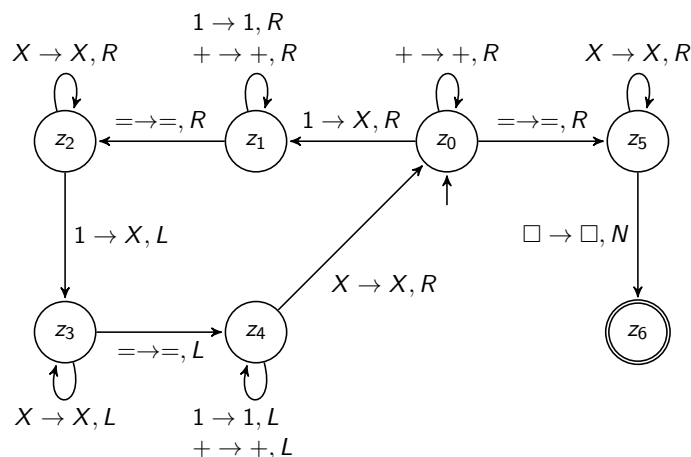
$(z', b, B) \in \delta(z, a)$: Wenn M im Zustand z das Zeichen a liest, dann kann M im nächsten Schritt in Zustand z' übergehen, a durch b ersetzen, und den Schreib-Lesekopf in Richtung $B \in \{L, N, R\}$ bewegen. Dabei bedeutet

- ▶ L einen Schritt nach **links**,
- ▶ N die **neutrale** Bewegung (also Stehenbleiben), und
- ▶ R einen Schritt nach **rechts**.



Nichtdeterministische Turingmaschine: Beispiel

$$M = (\{z_0, z_1, \dots, z_6\}, \{1, +, =\}, \{1, +, =, X, \square\}, \delta, z_0, \square, \{z_6\})$$



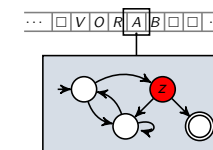
Turingmaschine: Konfiguration

Definition (Konfiguration einer Turingmaschine)

Eine **Konfiguration** einer Turingmaschine $M = \langle Z, \Sigma, \Gamma, \delta, z_0, \square, E \rangle$ ist gegeben durch ein Wort $k \in \Gamma^* Z \Gamma^+$.

Konfiguration $w_1 z w_2$ bedeutet, dass der nicht-leere bzw. bereits besuchte Teil des Bandes das Wort $w_1 w_2$ enthält, der Schreib-Lesekopf auf dem ersten Zeichen von w_2 steht und die TM sich in Zustand z befindet.

Beispiel



Konfiguration
 $\square V O R z A B \square \square$.

Turingmaschine: Berechnungsschritt

Definition (Berechnungsschritt einer Turingmaschine)

Eine NTM $M = \langle Z, \Sigma, \Gamma, \delta, z_0, \square, E \rangle$ kann nach den folgenden Regeln in einem Berechnungsschritt von Konfiguration k in Konfiguration k' übergehen ($k \vdash_M k'$):

$$a_1 \dots a_m z b_1 \dots b_n \vdash_M \begin{cases} a_1 \dots a_m z' c b_2 \dots b_n, & \text{falls } (z', c, N) \in \delta(z, b_1), m \geq 0, n \geq 1 \\ a_1 \dots a_m c z' b_2 \dots b_n, & \text{falls } (z', c, R) \in \delta(z, b_1), m \geq 0, n \geq 2 \\ a_1 \dots a_{m-1} z' a_m c b_2 \dots b_n, & \text{falls } (z', c, L) \in \delta(z, b_1), m \geq 1, n \geq 1 \\ a_1 \dots a_m c z' \square, & \text{falls } (z', c, R) \in \delta(z, b_1), m \geq 0, n = 1 \\ z' \square c b_2 \dots b_n, & \text{falls } (z', c, L) \in \delta(z, b_1), m = 0, n \geq 1 \end{cases}$$

Turingmaschine: Erreichbarkeit von Konfigurationen

Definition (Erreichbare Konfiguration)

Konfiguration k' ist mit NTM M von Konfiguration k aus **erreichbar** ($k \vdash_M^* k'$), falls $k = k'$ oder es gibt Konfigurationen $k_0, \dots, k_n$ ($n \geq 1$), so dass

- ▶ $k_0 = k$,
- ▶ $k_i \vdash_M k_{i+1}$ für $i \in \{0, \dots, n-1\}$, und
- ▶ $k_n = k'$.

Turingmaschine: Erkanntes Wort

Definition (erkanntes Wort bei Turingmaschinen)

NTM $M = \langle Z, \Sigma, \Gamma, \delta, z_0, \square, E \rangle$ **erkennt das Wort** w genau dann, wenn M von der **Startkonfiguration** $z_0 w$ mit endlich vielen Berechnungsschritten in eine Konfiguration mit einem **Endzustand** übergehen kann:

M erkennt w gdw. $z_0 w \vdash_M^* w_1 z w_2$ mit $z \in E, w_1 \in \Gamma^*, w_2 \in \Gamma^+$.

Spezialfall: Für $w = \varepsilon$ ist die **Startkonfiguration** $z_0 \square$.

Beispiel: Tafel

Turingmaschine: Akzeptierte Sprache

Definition (akzeptierte Sprache einer NTM)

Sei M eine nichtdeterministische Turingmaschine mit Eingabealphabet Σ . Die von M **akzeptierte Sprache** ist definiert durch

$$\mathcal{L}(M) = \{w \in \Sigma^* \mid w \text{ wird von } M \text{ erkannt}\}.$$

Beispiel: Tafel

Linear beschränkte Turingmaschinen

Brauchen wir
nicht unterschiedliche
Automatenmodelle für
kontextsensitive und
Typ-0-Sprachen?



Foto mit freundlicher Genehmigung von stockimages / FreeDigitalPhotos.net

Linear beschränkte Turingmaschinen: Idee

- ▶ **Linear beschränkte** Turingmaschinen dürfen nur den **Teil des Bandes verwenden, der durch das Eingabewort belegt wird.**
- ▶ Kann vorderes Ende der Eingabe im ersten Schritt markieren.
- ▶ Problem: Muss verhindern, über das hintere Bandende hinauszulaufen
- ▶ Für Sprache über Σ hat linear beschränkte Turingmaschine **modifiziertes Eingabealphabet** $\Sigma \cup \{\hat{a} \mid a \in \Sigma\}$.
- ▶ Eigentliche Eingabe $a_1 \dots a_n$ wird auf dem Band repräsentiert als $a_1 \dots a_{n-1} \hat{a}_n$.

Linear beschränkte Turingmaschinen: Definition

Definition (Linear beschränkte NTM)

Eine (nichtdeterministische) Turingmaschine heisst **linear beschränkt**, wenn für alle $a_1 \dots a_n \in \Sigma^+$ und alle Konfigurationen $w_1 z w_2$ mit $z_0 a_1 \dots a_{n-1} \hat{a}_n \vdash^* w_1 z w_2$ gilt $|w_1 w_2| = n$.

Linear beschränkte NTMs akzeptieren Typ-1-Sprachen

Satz

Die von linear beschränkten, nichtdeterministischen Turingmaschinen (LBAs) akzeptierbaren Sprachen sind genau die kontextsensitiven (Typ 1) Sprachen.

Ohne Beweis.

NTMs akzeptieren Typ-0-Sprachen

Satz

Die durch nichtdeterministische Turingmaschinen akzeptierbaren Sprachen sind genau die Typ-0-Sprachen.

Ohne Beweis.

Deterministische Turingmaschinen

Definition (Deterministische Turingmaschine)

Eine **deterministische Turingmaschine (DTM)** ist eine Turingmaschine $M = \langle Z, \Sigma, \Gamma, \delta, z_0, \square, E \rangle$ mit $\delta : (Z \setminus E) \times \Gamma \rightarrow Z \times \Gamma \times \{L, R, N\}$.

Deterministische Turingmaschinen

Satz

Für jede Typ-0-Sprache L gibt es eine deterministische Turingmaschine M mit $\mathcal{L}(M) = L$.

Beweisskizze.

Sei M' eine nichtdeterministische Turingmaschine mit $\mathcal{L}(M') = L$. Man konstruiert eine deterministische Turingmaschine, die systematisch den Berechnungsbaum von M' nach einer Konfiguration mit Endzustand untersucht.

Bemerkung: Es ist ein offenes Problem, ob der Satz analog für Typ-1-Sprachen und deterministische LBAs gilt.

12.3 Abschlusseigenschaften und Entscheidbarkeit

Abschlusseigenschaften

	Schnitt	Vereinigung	Komplement	Produkt	Stern
Typ 3	Ja	Ja	Ja	Ja	Ja
Typ 2	Nein	Ja	Nein	Ja	Ja
Typ 1	Ja ⁽¹⁾	Ja ⁽¹⁾	Ja ⁽¹⁾	Ja ⁽¹⁾	Ja ⁽¹⁾
Typ 0	Ja ⁽¹⁾	Ja ⁽¹⁾	Nein ⁽²⁾	Ja ⁽¹⁾	Ja ⁽¹⁾

Beweise?

(1) ohne Beweis

(2) Beweis in späteren Vorlesungskapiteln

Entscheidbarkeit

	Wort- problem	Leerheits- problem	Äquivalenz- problem	Schnitt- problem
Typ 3	Ja	Ja	Ja	Ja
Typ 2	Ja	Ja	Nein	Nein
Typ 1	Ja	Nein ⁽¹⁾	Nein ⁽¹⁾	Nein ⁽¹⁾
Typ 0	Nein ⁽²⁾	Nein ⁽²⁾	Nein ⁽²⁾	Nein ⁽²⁾

(1) ohne Beweis

(2) Beweis in Vorlesungsabschnitt 3

Wortproblem bei kontextsensitiven Sprachen:

Mit Typ-1-Grammatiken werden Wörter durch Ableitungen niemals kürzer, daher kann man alle Ableitungsmöglichkeiten bis zur Grösse des gegebenen Wortes ausprobieren.

12.4 Zusammenfassung

Zusammenfassung

- ▶ Turingmaschinen haben zwar nur endlich viele Zustände, aber ein **unendlich grosses Band** als „Speicher“.
- ▶ Turingmaschinen akzeptieren genau die **Typ-0-Sprachen**. Dies gilt auch für **deterministische Turingmaschinen**.
- ▶ **Linear beschränkte Turingmaschinen akzeptieren** genau die **kontextsensitiven Sprachen**.
- ▶ Die kontextsensitiven und Typ-0-Sprachen sind unter **fast allen gängigen Operationen abgeschlossen**. Ausnahme: **Typ-0 nicht unter Komplement abgeschlossen**
- ▶ Für kontextsensitive und Typ-0-Sprachen ist **fast kein Problem entscheidbar**. Ausnahme: **Wortproblem für Typ-1 ist entscheidbar**.