# Einführung in die Theoretische Informatik

Johannes Köbler

Institut für Informatik
Humboldt-Universität zu Berlin

WS 2020/21

# Kontextsensitive Sprachen

## Definition

Sei $G = (V, \Sigma, P, S)$ eine Grammatik

1. $G$ heißt vom Typ 3 oder regulär, falls für alle Regeln $u \to v$ gilt:

   $$u \in V \text{ und } v \in \Sigma V \cup \Sigma \cup \{\varepsilon\}$$

   (d.h. alle Regeln haben die Form $A \to aB$, $A \to a$ oder $A \to \varepsilon$)

2. $G$ heißt vom Typ 2 oder kontextfrei, falls für alle Regeln $u \to v$ gilt:

   $u \in V$ (d.h. alle Regeln haben die Form $A \to \alpha$)

3. $G$ heißt vom Typ 1 oder kontextsensitiv, falls für alle Regeln $u \to v$ gilt:

   $|v| \geq |u|$ (mit Ausnahme der $\varepsilon$-Sonderregel, s. unten)

4. Jede Grammatik ist automatisch vom Typ 0

## Die $\varepsilon$-Sonderregel

In einer kontextsensitiven Grammatik ist auch die Regel $S \to \varepsilon$ zulässig, falls das Startsymbol $S$ nicht auf der rechten Seite einer Regel vorkommt

# CFL ist echt in CSL enthalten



### Bemerkung

- Wie wir gesehen haben, ist CFL in CSL enthalten
- Zudem ist die Sprache $L = \{a^n b^n c^n \mid n \geq 1\}$ nicht kontextfrei
- $L$ kann jedoch von einer kontextsensitiven Grammatik erzeugt werden (siehe nächste Folie)
- Daher ist CFL echt in CSL enthalten

# Eine kontextsensitive Grammatik für $\{a^n b^n c^n \mid n \geq 1\}$

## Beispiel

- Betrachte die kontextsensitive Grammatik $G = (V, \Sigma, P, S)$ mit $V = \{S, B\}$, $\Sigma = \{a, b, c\}$ und den Regeln

$$P\colon\ S \to aSBc, abc\ (1,2) \quad cB \to Bc\ (3) \quad bB \to bb\ (4)$$

- In $G$ lässt sich beispielsweise das Wort $w = aabbcc$ ableiten:

$$\underline{S} \underset{(1)}{\Rightarrow} a\underline{S}Bc \underset{(2)}{\Rightarrow} aa\underline{bcB}c \underset{(3)}{\Rightarrow} aa\underline{bB}cc \underset{(4)}{\Rightarrow} aabbcc$$

- Allgemein gilt für alle $n \geq 1$:

$$\underline{S} \underset{(1)}{\Rightarrow}^{n-1} a^{n-1}\underline{S}(Bc)^{n-1} \underset{(2)}{\Rightarrow} a^{n-1}\underline{abc(Bc)^{n-1}}$$
$$\underset{(3)}{\Rightarrow}^{\binom{n}{2}} a^n \underline{bB^{n-1}}c^n \underset{(4)}{\Rightarrow}^{n-1} a^n b^n c^n$$

- Also gilt $a^n b^n c^n \in L(G)$ für alle $n \geq 1$

# Eine kontextsensitive Grammatik für $\{a^n b^n c^n \mid n \geq 1\}$

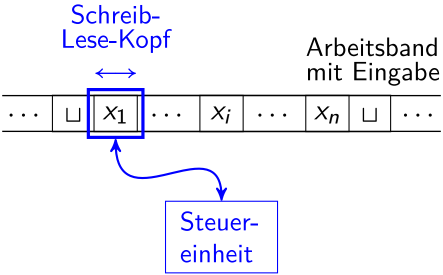
## Beispiel (Fortsetzung)

- Betrachte die kontextsensitive Grammatik $G = (V, \Sigma, P, S)$ mit $V = \{S, B\}$, $\Sigma = \{a, b, c\}$ und den Regeln

$$P\colon S \to aSBc, abc \; (1,2) \quad cB \to Bc \; (3) \quad bB \to bb \; (4)$$

- Umgekehrt folgt durch Induktion über die Ableitungslänge $m$, dass jede Satzform $\alpha \in (V \cup \Sigma)^*$ mit $S \Rightarrow^m \alpha$ die folgenden Bedingungen erfüllt:
  - $\#_a(\alpha) = \#_b(\alpha) + \#_B(\alpha) = \#_c(\alpha)$
  - links von $a$ und links von $S$ kommen nur $a$'s vor
  - links von $b$ kommen nur $a$'s oder $b$'s vor
- Daraus ergibt sich, dass in $G$ nur Wörter $w \in \Sigma^*$ der Form $w = a^n b^n c^n$ ableitbar sind, d.h. $L(G) = \{a^n b^n c^n \mid n \geq 1\} \in \text{CSL}$ $\triangleleft$

# Die Turingmaschine




- Um ein geeignetes Maschinenmodell für die kontextsensitiven Sprachen zu finden, führen wir zunächst das Rechenmodell der nichtdeterministischen Turingmaschine (NTM) ein
- Eine NTM erhält ihre Eingabe auf einem nach links und rechts unbegrenzten Band, das in Felder unterteilt ist; zudem kann sie weitere Bänder benutzen, die zu Beginn der Rechnung komplett leer sind
- In jedem Rechenschritt kann sie die aktuell besuchten Bandfelder lesen, die gelesenen Zeichen überschreiben und den Schreib-Lese-Kopf auf jedem Band um maximal ein Feld nach links oder rechts bewegen

## Definition

- Sei $k \geq 1$. Eine nichtdeterministische k-Band-Turingmaschine ($k$-NTM oder einfach NTM) wird durch ein 6-Tupel $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ beschrieben. Dabei ist
  - $Z$ eine endliche Menge von Zuständen
  - $\Sigma$ das Eingabealphabet (mit $\sqcup \notin \Sigma$; $\sqcup$ heißt Leerzeichen oder Blank)
  - $\Gamma$ das Arbeitsalphabet (mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$)
  - $\delta$: $Z \times \Gamma^k \to \mathcal{P}(Z \times \Gamma^k \times \{L, R, N\}^k)$ die Überführungsfunktion
  - $q_0$ der Startzustand und
  - $E \subseteq Z$ die Menge der Endzustände
- Eine $k$-NTM $M$ heißt deterministisch (kurz: $M$ ist eine $k$-DTM oder einfach DTM), falls für alle $(q, a_1, \ldots a_k) \in Z \times \Gamma^k$ gilt:

  $$\|\delta(q, a_1, \ldots a_k)\| \leq 1$$

# Das Rechenmodell der Turingmaschine

- Für $(q, b_1, \ldots, b_k, D_1, \ldots, D_k) \in \delta(p, a_1, \ldots a_k)$ schreiben wir auch

$$(p, a_1, \ldots, a_k) \rightarrow (q, b_1, \ldots, b_k, D_1, \ldots, D_k)$$

- Eine solche Anweisung ist ausführbar, falls
  - $p$ der aktuelle Zustand von $M$ ist und
  - sich für $i = 1, \ldots, k$ der Kopf des $i$-ten Bandes auf einem mit $a_i$ beschrifteten Feld befindet
- Bei ihrer Ausführung
  - geht $M$ vom Zustand $p$ in den Zustand $q$ über
  - ersetzt auf Band $i = 1, \ldots, k$ das Symbol $a_i$ durch $b_i$ und
  - bewegt den Kopf auf Band $i = 1, \ldots, k$ gemäß $D_i$
    (L: ein Feld nach links, R: ein Feld nach rechts, N: keine Bewegung)

# Das Rechenmodell der Turingmaschine



- Eine Konfiguration ist ein $(3k+1)$-Tupel

  $$K = (q, u_1, a_1, v_1, \ldots, u_k, a_k, v_k) \in Z \times (\Gamma^* \times \Gamma \times \Gamma^*)^k$$

  und besagt, dass
  - $q$ der momentane Zustand ist und
  - das $i$-te Band mit $\ldots \sqcup u_i a_i v_i \sqcup \ldots$ beschriftet ist, wobei sich der Kopf auf dem Zeichen $a_i$ befindet
- Im Fall $k = 1$ notieren wir eine Konfiguration $K = (q, u, a, v)$ auch in der Form $K = uqav$

# Das Rechenmodell der Turingmaschine

- Seien $K = (p, u_1, a_1, v_1, \ldots, u_k, a_k, v_k)$ und $K' = (q, u_1', a_1', v_1', \ldots, u_k', a_k', v_k')$ Konfigurationen
- $K'$ heißt Folgekonfiguration von $K$ (kurz $K \vdash K'$), falls eine Anweisung $(p, a_1, \ldots, a_k) \to (q, b_1, \ldots, b_k, D_1, \ldots, D_k)$ existiert, so dass für $i = 1, \ldots, k$ gilt:

| $D_i = \mathrm{N}$ | $D_i = \mathrm{R}$ | $D_i = \mathrm{L}$ |
|---|---|---|
| $K\colon\ u_i\ \boxed{a_i}\ v_i$ | $K\colon\ u_i\ \boxed{a_i}\ v_i$ | $K\colon\ u_i\ \boxed{a_i}\ v_i$ |
| $K'\colon\ u_i\ \boxed{b_i}\ v_i$ | $K'\colon\ u_i\ b_i\ \boxed{a_i'}\ v_i'$ | $K'\colon\ u_i'\ \boxed{a_i'}\ b_i\ v_i$ |
| $u_i' = u_i$ | $u_i' = u_i b_i$ | $u_i' a_i' = \begin{cases} u_i, & u_i \neq \varepsilon \\ \sqcup, & \text{sonst} \end{cases}$ |
| $a_i' = b_i$ | $a_i' v_i' = \begin{cases} v_i, & v_i \neq \varepsilon \\ \sqcup, & \text{sonst} \end{cases}$ | $v_i' = b_i v_i$ |
| $v_i' = v_i$ | | |

# Das Rechenmodell der Turingmaschine

- Die Startkonfiguration von $M$ bei Eingabe $x = x_1 \dots x_n \in \Sigma^*$ ist

$$K_x = \begin{cases} (q_0, \varepsilon, x_1, x_2 \dots x_n, \varepsilon, \sqcup, \varepsilon, \dots, \varepsilon, \sqcup, \varepsilon), & x \neq \varepsilon, \\ (q_0, \varepsilon, \sqcup, \varepsilon, \dots, \varepsilon \sqcup, \varepsilon), & x = \varepsilon \end{cases}$$

- Eine Rechnung von $M$ bei Eingabe $x$ ist eine (endliche oder unendliche) Folge von Konfigurationen $K_0, K_1, K_2 \dots$ mit $K_0 = K_x$ und $K_0 \vdash K_1 \vdash K_2 \dots$
- Die von $M$ akzeptierte oder erkannte Sprache ist

$$L(M) = \{x \in \Sigma^* \mid \exists\, K \in E \times (\Gamma^* \times \Gamma \times \Gamma^*)^k : K_x \vdash^* K\}$$

- Ein Wort $x$ wird also genau dann von $M$ akzeptiert (kurz: $M(x)$ akzeptiert), wenn es eine Rechnung von $M$ bei Eingabe $x$ gibt, bei der ein Endzustand erreicht wird