Theoretische Informatik I

Einheit 2.4

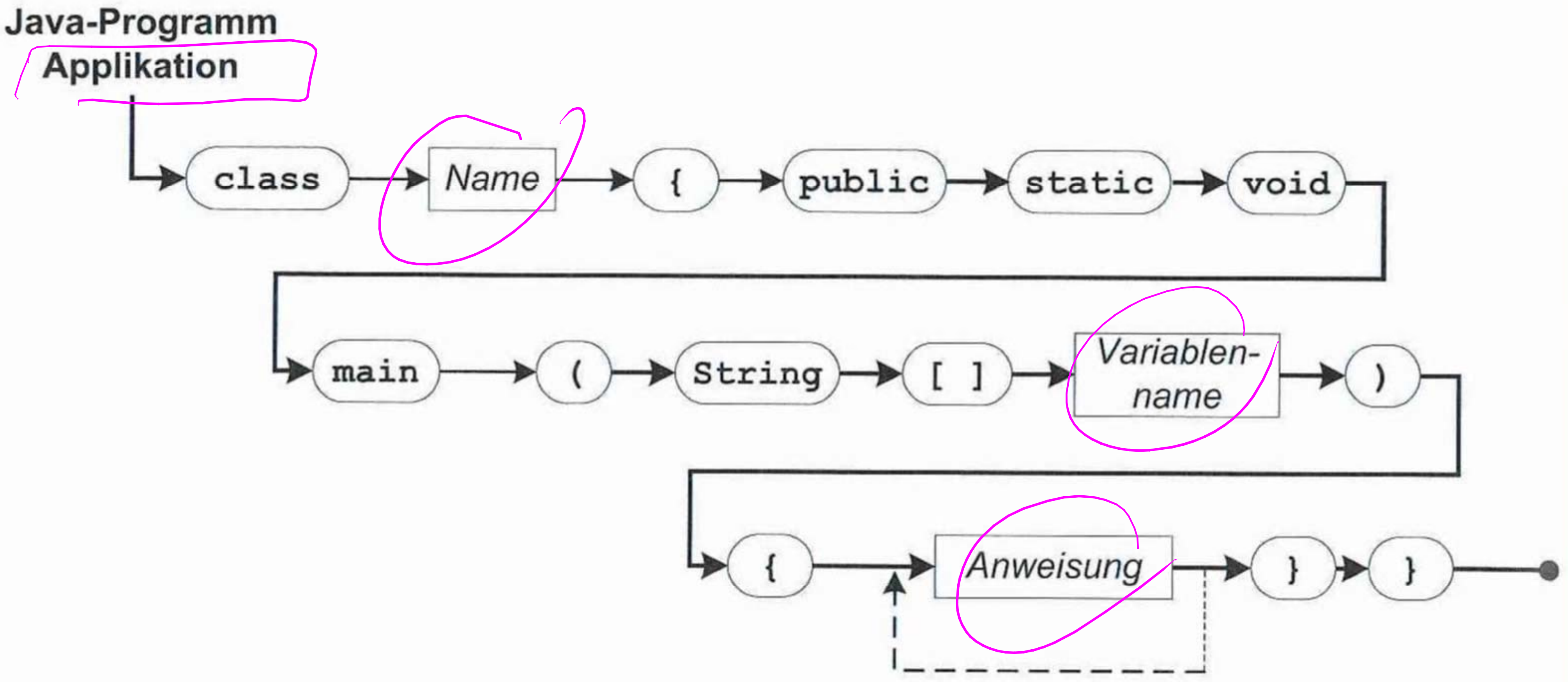
Grammatiken

1. Arbeitsweise
2. Klassifizierung
3. Beziehung zu Automaten

BESCHREIBUNGSFORMEN FÜR SPRACHEN

- **Mathematische Mengennotation**
  - Prädikate beschreiben Eigenschaften der Wörter
  - Extrem flexibel, nicht notwendig “berechenbar”
- **Endliche Automaten**
  - Beschreibung der Verarbeitung von Sprachen
  - Schwerpunkt ist Erkennen korrekter Wörter
- **Reguläre Ausdrücke**
  - Beschreibung der Struktur der Sprache
- **Grammatiken**
  - Produktionsregeln beschreiben Aufbau der Wörter
  - Auch für komplexere Strukturen als reguläre Sprachen
  - Gängig für die Beschreibung von Programmiersprachen

Beispiel: Auszug der Grammatik von JAVA



KOMPONENTEN VON GRAMMATIKEN

- **Terminalsymbole: Alphabet der Sprache**
  - Symbole, aus denen die erzeugten Wörter bestehen sollen
  - Bei Programmiersprachen meist ASCII-Symbole ohne Kontrollzeichen

KOMPONENTEN VON GRAMMATIKEN

- **Terminalsymbole: Alphabet der Sprache**
  - Symbole, aus denen die erzeugten Wörter bestehen sollen
  - Bei Programmiersprachen meist ASCII-Symbole ohne Kontrollzeichen
- **Variablen: Hilfsalphabet für Verarbeitung**
  - Beschreiben die syntaktischen Kategorien der Sprache
  - Bei JAVA z.B. Applikation, Name, Variablenname, Anweisung, ...
  - Andere Bezeichnung: Nichtterminale Symbole

KOMPONENTEN VON GRAMMATIKEN

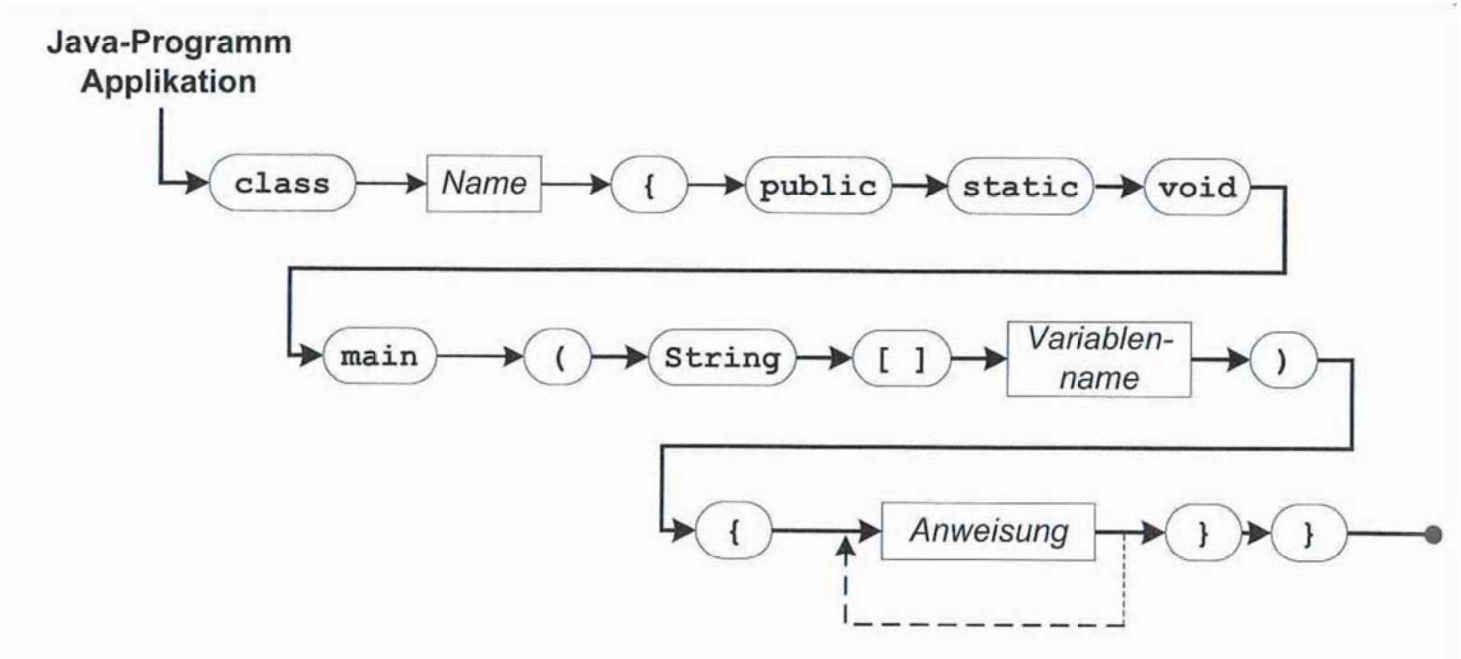
- **Terminalsymbole: Alphabet der Sprache**
  - Symbole, aus denen die erzeugten Wörter bestehen sollen
  - Bei Programmiersprachen meist ASCII-Symbole ohne Kontrollzeichen
- **Variablen: Hilfsalphabet für Verarbeitung**
  - Beschreiben die syntaktischen Kategorien der Sprache
  - Bei JAVA z.B. Applikation, Name, Variablenname, Anweisung, . . .
  - Andere Bezeichnung: Nichtterminale Symbole
- **Produktionen: Regeln zur Erzeugung von Wörtern**
  - Erklären wie syntaktischen Kategorien aufgebaut sind
  - Erklären Erzeugung von Wörtern der Sprache in den einzelnen Kategorien
  - z.B. “Eine Applikation beginnt mit `class` gefolgt von einem Namen, ...”

Applikation → class Name ...

KOMPONENTEN VON GRAMMATIKEN

- **Terminalsymbole: Alphabet der Sprache**
  - Symbole, aus denen die erzeugten Wörter bestehen sollen
  - Bei Programmiersprachen meist ASCII-Symbole ohne Kontrollzeichen
- **Variablen: Hilfsalphabet für Verarbeitung**
  - Beschreiben die syntaktischen Kategorien der Sprache
  - Bei JAVA z.B. Applikation, Name, Variablenname, Anweisung, . . .
  - Andere Bezeichnung: Nichtterminale Symbole
- **Produktionen: Regeln zur Erzeugung von Wörtern**
  - Erklären wie syntaktischen Kategorien aufgebaut sind
  - Erklären Erzeugung von Wörtern der Sprache in den einzelnen Kategorien
  - z.B. “Eine Applikation beginnt mit `class` gefolgt von einem Namen, ...”
- **Startsymbol**
  - Erklärt welche syntaktische Kategorie beschrieben werden soll

GRAMMATIKEN – MATHEMATISCH PRÄZISIERT



Eine Grammatik ist ein 4-Tupel $G = (\underline{V}, T, P, S)$ mit

- V endliches **Hilfsalphabet**
- T endliches **Terminalalphabet** mit $V \cap T = \emptyset$
- $P \subseteq \Gamma^{+} \times \Gamma^{*}$ endliche Menge der **Produktionen** (wobei $\underline{\Gamma = V \cup T}$)

  Schreibweise für Produktionen: $\underline{\boldsymbol{l \rightarrow r}} \in P$ statt $(l, r) \in P$
- $S \in V$ **Startsymbol**

ARBEITSWEISE: PRODUKTION VON WÖRTERN DER ZIELSPRACHE

- $G_1 = (\underline{\{S\}},\ \underline{\{0,1\}},\ P,\ \underline{S})$ **mit** $P = \{\underline{S\to S1},\ S\to S0,\ \underline{S\to\epsilon}\}$

ARBEITSWEISE: PRODUKTION VON WÖRTERN DER ZIELSPRACHE

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \rightarrow S1,\ S \rightarrow S0,\ S \rightarrow \epsilon\}$

  Erzeugung von Wörtern:

  $S \rightarrow \epsilon$

  $S \rightarrow S1 \rightarrow S11 \rightarrow S011 \rightarrow 011$

ARBEITSWEISE: PRODUKTION VON WÖRTERN DER ZIELSPRACHE

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S{\rightarrow}S1,\ S{\rightarrow}S0,\ S{\rightarrow}\epsilon\}$

  Erzeugung von Wörtern:

  $$S \rightarrow \epsilon$$

  $$S \rightarrow S0 \rightarrow \mathbf{0}$$

ARBEITSWEISE: PRODUKTION VON WÖRTERN DER ZIELSPRACHE

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S{\rightarrow}S1,\ S{\rightarrow}S0,\ S{\rightarrow}\epsilon\}$

  Erzeugung von Wörtern:

  $$S \rightarrow \boxed{\epsilon}$$

  $$S \rightarrow S0 \rightarrow \boxed{\mathbf{0}}$$

  $$S \rightarrow S0 \rightarrow S10 \rightarrow S010 \rightarrow S0010 \rightarrow \boxed{\mathbf{0010}}$$

ARBEITSWEISE: PRODUKTION VON WÖRTERN DER ZIELSPRACHE

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S{\rightarrow}S1,\ S{\rightarrow}S0,\ S{\rightarrow}\epsilon\}$

  Erzeugung von Wörtern:

  $$S \rightarrow \epsilon$$

  $$S \rightarrow S0 \rightarrow \mathbf{0}$$

  $$S \rightarrow S0 \rightarrow S10 \rightarrow S010 \rightarrow S0010 \rightarrow \mathbf{0010}$$

  – Nur Wörter über dem Terminalalphabet sind von Interesse

  – ϵ, 0, 0010 gehören zur erzeugten Sprache

  – S, $S0$, $S10$, $S010$, $S0010$ sind nur “Zwischenschritte”

ARBEITSWEISE: PRODUKTION VON WÖRTERN DER ZIELSPRACHE

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S\to S1,\ S\to S0,\ S\to\epsilon\}$

  Erzeugung von Wörtern:

  $S \longrightarrow ?$

  $S \to \epsilon$

  $S \to S0 \to \mathbf{0}$

  $S \to S0 \to S10 \to S010 \to S0010 \to \mathbf{0010}$

  – Nur Wörter über dem Terminalalphabet sind von Interesse

  – ϵ, 0, 0010 gehören zur erzeugten Sprache

  – S, $S0$, $S10$, $S010$, $S0010$ sind nur “Zwischenschritte”

- $G_2 = (\{S, A, B, C\},\ \{0,1\},\ P,\ S)$ **mit**

  $P = \{S\to B,\ S\to CA0,\ A\to BBB,\ B\to C1,\ B\to 0,\ CC1\to\epsilon\}$

  Ableitungen:

  $S \longrightarrow B \longrightarrow \mathbf{0}$ ✓

  $S \longrightarrow B \longrightarrow C1$ Erfolglos, kein Wort der Zielsprache erreichbar

  $S \longrightarrow CA0 \longrightarrow CBBB0 \longrightarrow CC1BB0 \longrightarrow BB0 \longrightarrow 0B0 \longrightarrow \mathbf{000}$ ✓

ARBEITSWEISE VON GRAMMATIKEN – PRÄZISIERT

- **Ableitungsrelation** $\longrightarrow \; \subseteq \Gamma^+ \times \Gamma^*$
  - $\underline{w \longrightarrow z} \;\equiv\; \exists x, y \in \Gamma^*.\ \exists l \to r \in P.\ w = \boxed{x\,\enclose{circle}{l}\,y} \wedge z = x\,\enclose{circle}{r}\,y$

    Anwendung von Produktionen auf Wörter

ARBEITSWEISE VON GRAMMATIKEN – PRÄZISIERT

- **Ableitungsrelation** $\longrightarrow\ \subseteq \Gamma^{+} \times \Gamma^{*}$
  - $w \longrightarrow z \;\equiv\; \exists x, y \in \Gamma^{*}.\ \exists l \to r \in P.\ w{=}x\,l\,y \wedge z{=}x\,r\,y$

    Anwendung von Produktionen auf Wörter

- **Erweiterte Ableitungsrelation** $\xrightarrow{*}\ \subseteq \Gamma^{+} \times \Gamma^{*}$
  - $w \xrightarrow{0} z \;\equiv\; w{=}z$
  - $w \xrightarrow{n+1} z \;\equiv\; \exists u \in \Gamma^{*}.\ w \longrightarrow u \wedge u \xrightarrow{n} z$
  - $w \xrightarrow{*} z \;\equiv\; \exists n \in \mathbb{N}.\ w \xrightarrow{n} z$
  - Grammatik durch optionalen Index G ($\xrightarrow{*}_{G}$) spezifizierbar

ARBEITSWEISE VON GRAMMATIKEN – PRÄZISIERT

- **Ableitungsrelation** $\longrightarrow\ \subseteq \Gamma^{+} \times \Gamma^{*}$
  - $w \longrightarrow z \ \equiv\ \exists x, y \in \Gamma^{*}.\ \exists l \to r \in P.\ w = x\,l\,y \wedge z = x\,r\,y$

    Anwendung von Produktionen auf Wörter

- **Erweiterte Ableitungsrelation** $\xrightarrow{*}\ \subseteq \Gamma^{+} \times \Gamma^{*}$
  - $w \xrightarrow{0} z \ \equiv\ w = z$
  - $w \xrightarrow{n+1} z \ \equiv\ \exists u \in \Gamma^{*}.\ w \longrightarrow u \wedge u \xrightarrow{n} z$
  - $w \xrightarrow{*} z \ \equiv\ \exists n \in \mathbb{N}.\ w \xrightarrow{n} z$
  - Grammatik durch optionalen Index G ($\xrightarrow{*}_{G}$) spezifizierbar

- **Von G erzeugte Sprache**
  - Menge der Terminalwörter, die aus S abgeleitet werden können

$$L(G) \ \equiv\ \{w \in T^{*} \mid S \xrightarrow{*} w\}$$

$S \rightarrow \rightarrow \rightarrow \rightarrow w$

GRAMMATIK FÜR $L = \{\underline{0}^k\underline{1}^l \mid k \leq l\}$

$S \rightarrow 0S1$

$S \rightarrow S1$

$S \rightarrow \varepsilon$

GRAMMATIK FÜR $L = \{0^k1^l \mid k \leq l\}$

- $G_3 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \to S1,\ S \to 0S1,\ S \to \epsilon\}$

GRAMMATIK FÜR $L = \{0^k1^l \mid k \leq l\}$

- $G_3 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \to S1,\ S \to 0S1,\ S \to \epsilon\}$
- **Zeige** $L(G_3) = L$ **per Induktion über Länge der Ableitung**
  - Ableitungen der Länge 0 liefern keine Terminalwörter
  - Zeige: $\forall l \in \mathbb{N}.\ \forall w \in \{0,1\}^*.\ S \xrightarrow{l+1} w \Leftrightarrow (\exists k \leq l.\ w = 0^k1^l)$

GRAMMATIK FÜR $L = \{0^k 1^l \mid k \leq l\}$

- $G_3 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \to S1,\ S \to 0S1,\ S \to \underline{\epsilon}\}$
- **Zeige** $L(G_3) = L$ **per Induktion über Länge der Ableitung**
  - Ableitungen der Länge 0 liefern keine Terminalwörter
  - Zeige: $\forall l \in \mathbb{N}.\ \forall w \in \{0,1\}^*.\ S \xrightarrow{l+1} w \Leftrightarrow (\exists k \leq l.\ w = \underline{0^k 1^l})$
- **Basisfall**
  - $S \xrightarrow{1} w \ \Leftrightarrow\ (S \to w) \in P \ \Leftrightarrow\ w = \epsilon \ \Leftrightarrow\ \exists k \leq 0.\ w = \underline{0^k 1^0}$ ✓

GRAMMATIK FÜR $L = \{0^k1^l \mid k \leq l\}$

- $G_3 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \to S1,\ S \to 0S1,\ S \to \epsilon\}$
- **Zeige** $L(G_3) = L$ **per Induktion über Länge der Ableitung**
  - Ableitungen der Länge 0 liefern keine Terminalwörter
  - Zeige: $\forall l \in \mathbb{N}.\ \forall w \in \{0,1\}^*.\ S \xrightarrow{l+1} w \Leftrightarrow (\exists k \leq l.\ w = 0^k1^l)$
- **Basisfall**
  - $S \xrightarrow{1} w \ \Leftrightarrow\ (S \to w) \in P \ \Leftrightarrow\ w = \epsilon \ \Leftrightarrow\ \exists k \leq 0.\ w = 0^k1^0$ ✓
- **Induktionsschritt**
  - Es gelte $\forall w \in \{0,1\}^*.\ S \xrightarrow{l+1} w \Leftrightarrow (\exists k \leq l.\ w = 0^k1^l)$

$$
\begin{aligned}
& S \xrightarrow{l+2} v \\
\Leftrightarrow\ & S \to S1 \xrightarrow{l+1} v \ \vee\ S \to 0S1 \xrightarrow{l+1} v \\
\Leftrightarrow\ & \exists w \in \{0,1\}^*.\ \ S \xrightarrow{l+1} w \ \wedge\ (v = w1 \vee v = 0w1) \\
\Leftrightarrow\ & \exists w \in \{0,1\}^*.\exists k \leq l.\ \ w = 0^k1^l \ \wedge\ (v = w1 \vee v = 0w1) \qquad \text{(Annahme)} \\
\Leftrightarrow\ & \exists k \leq l.\ \ v = 0^k1^{l+1} \ \vee\ v = 0^{k+1}1^{l+1} \\
\Leftrightarrow\ & \exists k \leq (l+1).\ \ v = 0^k1^{l+1}
\end{aligned}
$$

✓ ✓

KLASSIFIZIERUNG VON GRAMMATIKEN

- **allgemein (Typ 0):** keine Einschränkung an die Produktionen
- **kontextsensitiv (Typ 1)**
  - nur Regeln der Form $x\,A\,y \rightarrow x\,z\,y$ oder $S \rightarrow \epsilon$ $\qquad (x, y, z \in \Gamma^*,\ A \in V,\ z \neq \epsilon)$

    ($S \rightarrow \epsilon$ nur erlaubt, wenn S nicht rechts in einer anderen Regel auftaucht)
- **expansiv**
  - nur Regeln der Form $x \rightarrow z$ mit $|x| \leq |z|$, oder $S \rightarrow \epsilon$ $\qquad (x \in \Gamma^+,\ z \in (\Gamma - \{S\})^+)$
- **kontextfrei (Typ 2)**
  - nur Regeln der Form $A \rightarrow z$ $\qquad (z \in \Gamma^*,\ A \in V)$
- **linear**
  - nur Regeln der Form $A \rightarrow \epsilon$ oder $A \rightarrow u\,B\,v$ $\qquad (A, B \in V,\ u, v \in T^*)$
- **rechtslinear (Typ 3)**
  - nur Regeln der Form $A \rightarrow \epsilon$ oder $A \rightarrow a\,B$ $\qquad (A, B \in V,\ a \in T)$

    Manche Bücher: nur Regeln der Form $A \rightarrow \epsilon$ oder $A \rightarrow v\,B$ $\qquad (A, B \in V,\ v \in T^*)$
- **linkslinear**
  - nur Regeln der Form $A \rightarrow \epsilon$ oder $A \rightarrow B\,a$ $\qquad (A, B \in V,\ a \in T)$

BEISPIELE FÜR GRAMMATIKKLASSEN

- **kontextsensitiv**: Regeln $x\,A\,y{\rightarrow}x\,z\,y$ oder $S{\rightarrow}\epsilon$
- **expansiv**: Regeln $x{\rightarrow}z$ mit $|x|{\le}|z|$, oder $S{\rightarrow}\epsilon$
- **kontextfrei**: Regeln $A{\rightarrow}z$
- **linear**: Regeln $A{\rightarrow}\epsilon$ oder $A{\rightarrow}u\,B\,v$
- **rechtslinear**: Regeln $A{\rightarrow}\epsilon$ oder $A{\rightarrow}a\,B$
- **linkslinear**: Regeln $A{\rightarrow}\epsilon$ oder $A{\rightarrow}B\,a$

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S{\rightarrow}S1,\ S{\rightarrow}S0,\ S{\rightarrow}\epsilon\}$

BEISPIELE FÜR GRAMMATIKKLASSEN

- **kontextsensitiv**: Regeln $x\,A\,y \rightarrow x\,z\,y$ oder $S \rightarrow \epsilon$
- **expansiv**: Regeln $x \rightarrow z$ mit $|x| \leq |z|$, oder $S \rightarrow \epsilon$
- **kontextfrei**: Regeln $A \rightarrow z$
- **linear**: Regeln $A \rightarrow \epsilon$ oder $A \rightarrow u\,B\,v$
- **rechtslinear**: Regeln $A \rightarrow \epsilon$ oder $A \rightarrow a\,B$
- **linkslinear**: Regeln $A \rightarrow \epsilon$ oder $A \rightarrow B\,a$

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \rightarrow S1,\ S \rightarrow S0,\ S \rightarrow \epsilon\}$
  - linkslinear, kontextfrei, nicht expansiv, nicht kontextsensitiv (S rechts, $S \rightarrow \epsilon$)

BEISPIELE FÜR GRAMMATIKKLASSEN

- **kontextsensitiv**: Regeln $x\,A\,y \to x\,z\,y$ oder $S \to \epsilon$
- **expansiv**: Regeln $x \to z$ mit $|x| \leq |z|$, oder $S \to \epsilon$
- **kontextfrei**: Regeln $A \to z$
- **linear**: Regeln $A \to \epsilon$ oder $A \to u\,B\,v$
- **rechtslinear**: Regeln $A \to \epsilon$ oder $A \to a\,B$
- **linkslinear**: Regeln $A \to \epsilon$ oder $A \to B\,a$

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \to S1,\ S \to S0,\ S \to \epsilon\}$
  - linkslinear, kontextfrei, nicht expansiv, nicht kontextsensitiv (S rechts, $S \to \epsilon$)

- $G_2 = (\{S, A, B, C\},\ \{0,1\},\ P,\ S)$ **mit**

  $P = \{S \to B,\ S \to CA0,\ A \to BBB,\ B \to C1,\ B \to 0,\ \underline{CC1} \to \epsilon\}$

BEISPIELE FÜR GRAMMATIKKLASSEN

- **kontextsensitiv**: Regeln $x\,A\,y \to x\,z\,y$ oder $S \to \epsilon$
- **expansiv**: Regeln $x \to z$ mit $|x| \leq |z|$, oder $S \to \epsilon$
- **kontextfrei**: Regeln $A \to z$
- **linear**: Regeln $A \to \epsilon$ oder $A \to u\,B\,v$
- **rechtslinear**: Regeln $A \to \epsilon$ oder $A \to a\,B$
- **linkslinear**: Regeln $A \to \epsilon$ oder $A \to B\,a$

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \to S1,\ S \to S0,\ S \to \epsilon\}$
  – linkslinear, kontextfrei, nicht expansiv, nicht kontextsensitiv (S rechts, $S \to \epsilon$)

- $G_2 = (\{S, A, B, C\},\ \{0,1\},\ P,\ S)$ **mit**
  $P = \{S \to B,\ S \to CA0,\ A \to BBB,\ B \to C1,\ B \to 0,\ CC1 \to \epsilon\}$
  – allgemein (keine anderen Bedingungen erfüllt)

Beispiele für Grammatikklassen

- **kontextsensitiv**: Regeln $x\,A\,y \rightarrow x\,z\,y$ oder $S \rightarrow \epsilon$
- **expansiv**: Regeln $x \rightarrow z$ mit $|x| \leq |z|$, oder $S \rightarrow \epsilon$
- **kontextfrei**: Regeln $A \rightarrow z$
- **linear**: Regeln $A \rightarrow \epsilon$ oder $A \rightarrow u\,B\,v$
- **rechtslinear**: Regeln $A \rightarrow \epsilon$ oder $A \rightarrow a\,B$
- **linkslinear**: Regeln $A \rightarrow \epsilon$ oder $A \rightarrow B\,a$

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \rightarrow S1,\ S \rightarrow S0,\ S \rightarrow \epsilon\}$
  - linkslinear, kontextfrei, nicht expansiv, nicht kontextsensitiv (S rechts, $S \rightarrow \epsilon$)

- $G_2 = (\{S, A, B, C\},\ \{0,1\},\ P,\ S)$ **mit**
  $P = \{S \rightarrow B,\ S \rightarrow CA0,\ A \rightarrow BBB,\ B \rightarrow C1,\ B \rightarrow 0,\ CC1 \rightarrow \epsilon\}$
  - allgemein (keine anderen Bedingungen erfüllt)

- $G_3 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \rightarrow S1,\ \underline{S \rightarrow 0S1},\ S \rightarrow \epsilon\}$
  - linear ✓, kontextfrei ✓, nicht expansiv ✓, nicht kontextsensitiv ✓

BEISPIELE FÜR GRAMMATIKKLASSEN

- **kontextsensitiv**: Regeln $x\,A\,y \to x\,z\,y$ oder $S \to \epsilon$
- **expansiv**: Regeln $x \to z$ mit $|x| \leq |z|$, oder $S \to \epsilon$
- **kontextfrei**: Regeln $A \to z$
- **linear**: Regeln $A \to \epsilon$ oder $A \to u\,B\,v$
- **rechtslinear**: Regeln $A \to \epsilon$ oder $A \to a\,B$
- **linkslinear**: Regeln $A \to \epsilon$ oder $A \to B\,a$

- $G_1 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \to S1,\ S \to S0,\ S \to \epsilon\}$
  – linkslinear, kontextfrei, nicht expansiv, nicht kontextsensitiv (S rechts, $S \to \epsilon$)
- $G_2 = (\{S, A, B, C\},\ \{0,1\},\ P,\ S)$ **mit**
  $P = \{S \to B,\ S \to CA0,\ A \to BBB,\ B \to C1,\ B \to 0,\ CC1 \to \epsilon\}$
  – allgemein (keine anderen Bedingungen erfüllt)
- $G_3 = (\{S\},\ \{0,1\},\ P,\ S)$ **mit** $P = \{S \to S1,\ S \to 0S1,\ S \to \epsilon\}$
  – linear, kontextfrei, nicht expansiv, nicht kontextsensitiv
- $G_4 = (\{S, A, B, C\},\ \{a, b, c\},\ P,\ S)$ **mit** $P = \{S \to aCBC,$
  $S \to aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,\ bC \to bc,\ cC \to cc\}$
  – expansiv, nicht kontextfrei, nicht kontextsensitiv

SPRACHKLASSEN

0A → A0

- **Typ-0 Sprachen**
  - Sprachen der Form $L = L(G)$ für eine beliebige Grammatik G
- **Typ-1 Sprachen (kontextsensitive Sprachen)**
  - Sprachen der Form $L = L(G)$ für eine kontextsensitive Grammatik G
  - L ist kontextsensitiv g.d.w. $L = L(G)$ für eine expansive Grammatik G
- **Typ-2 Sprachen (kontextfreie Sprachen)**
  - Sprachen der Form $L = L(G)$ für eine kontextfreie Grammatik G
- **Lineare Sprachen**
  - Sprachen der Form $L = L(G)$ für eine lineare Grammatik G
- **Typ-3 Sprachen (reguläre Sprachen)** $\mathcal{L}_3$
  - Sprachen der Form $L = L(G)$ für eine rechtslineare Grammatik G
  - L ist regulär g.d.w. $L = L(G)$ für eine linkslineare Grammatik G

$$\mathcal{L}_i \equiv \{\, L \mid L \text{ ist Sprache vom Typ } i \}$$

TYP-3 SPRACHEN VS. REGULÄRE SPRACHEN

Wie hängen Grammatiken und Automaten zusammen?

- **Automaten verarbeiten Eingabewörter**
  - Jedes Symbol wird in einem Schritt abgearbeitet
  - Symbol bestimmt, ob Automat im Zustand bleibt oder wechselt
- **Grammatiken erzeugen Wörter**
  - Hilfssymbole werden im Endeffekt in Terminalwörter umgewandelt
  - Nichtlineare Grammatiken erzeugen mehrere Symbole gleichzeitig
  - Ableitungen in rechts-/linkslinearen Grammatiken erzeugen pro Schritt ein Terminalsymbol und verwenden jeweils nur ein Hilfssymbol
- **Wie kann man umwandeln?**
  - Konstruiere zu jedem DEA eine äquivalente rechtslineare Grammatik
  - Konstruiere zu jeder rechtslinearen Grammatik einen äquivalenten DEA

$\mapsto$ **$\mathcal{L}_3 = \{\, L \mid L$ ist regulär $\}$**

UMWANDLUNG VON DEAS IN TYP-3 GRAMMATIKEN

Für jeden DEA A gibt es eine Typ-3 Grammatik G mit $L(G) = L(A)$

- **Gegeben DEA** $A = (Q, \Sigma, \delta, q_0, F)$

$$\delta(\boxed{q}, \boxed{a}) = \boxed{q'}$$

$$\boxed{A} \rightarrow \boxed{a}\boxed{B}$$

$$q \rightarrow a\, q'$$

UMWANDLUNG VON DEAS IN TYP-3 GRAMMATIKEN

Für jeden DEA A gibt es eine Typ-3 Grammatik G mit $L(G) = L(A)$

- **Gegeben DEA $A = (Q, \Sigma, \delta, q_0, F)$**
  - Wandle Abarbeitung von Symbolen in Erzeugung durch Grammatik um
  - Setze $G := (Q, \Sigma, P, q_0)$ mit $P = \{q \to aq' \mid \delta(q, a) = q'\} \cup \{q \to \epsilon \mid q \in F\}$
  - G ist per Konstruktion rechtslinear, also vom Typ 3

~~$q \to \varepsilon$~~

UMWANDLUNG VON DEAS IN TYP-3 GRAMMATIKEN

Für jeden DEA A gibt es eine
Typ-3 Grammatik G mit $L(G) = L(A)$

- **Gegeben DEA $A = (Q, \Sigma, \delta, q_0, F)$**
  - Wandle Abarbeitung von Symbolen in Erzeugung durch Grammatik um
  - Setze $G := (Q, \Sigma, P, q_0)$ mit $P = \{q \to aq' \mid \delta(q,a) = q'\} \cup \{q \to \epsilon \mid q \in F\}$
  - G ist per Konstruktion rechtslinear, also vom Typ 3

- **Zeige $L(G) = L(A)$**

$$
\begin{array}{lll}
 & & w = w_1..w_n \in L(G) \\
\Leftrightarrow & & q_0 \xrightarrow{*} w_1..w_n \\
\Leftrightarrow & \exists q_1,..,q_n \in Q. & q_0 \longrightarrow w_1 q_1 \longrightarrow w_1 w_2 q_2 \longrightarrow \ldots \longrightarrow w_1..w_n q_n \longrightarrow w_1..w_n \\
\Leftrightarrow & \exists q_1,..,q_n \in Q. & q_0, w_1..w_n \vdash q_1, w_2..w_n \vdash \ldots \vdash q_{n-1}, w_n \vdash q_n, \epsilon \wedge q_n \in F \\
\Leftrightarrow & \exists q_n \in F. & q_0, w_1..w_n \vdash^* q_n, \epsilon \\
\Leftrightarrow & w \in L(A) &
\end{array}
$$

$q_n \to \varepsilon$

✓

Umwandlung von Typ-3 Grammatiken in NEAs

Für jede Typ-3 Grammatik G gibt es einen NEA A mit $L(A) = L(G)$

- **Gegeben Grammatik $G = (V, T, P, S)$**

$$\boxed{\begin{array}{l} A \to aB \\ A \to aC \end{array}}$$

$\delta(A, a) \overset{?}{=} B$

$\delta(A, a) \overset{?}{=} C$??

$\to \{B, C\}$

NEA

$\{X \mid A \to aX \in P\}$

UMWANDLUNG VON TYP-3 GRAMMATIKEN IN NEAS

Für jede Typ-3 Grammatik G gibt es einen NEA A mit $L(A) = L(G)$

- **Gegeben Grammatik $G = (V, T, P, S)$**
  - Wandle Erzeugung von Symbolen in Abarbeitung durch NEA um
  - Setze $A := (V, T, \delta, S, F)$ mit $\underline{\delta(X, a)} = \{X' \mid \underline{X \to a\boxed{X'}} \in P\}$

    und $F = \{X \in V \mid \underline{X \to \epsilon} \in P\}$

Umwandlung von Typ-3 Grammatiken in NEAs

Für jede Typ-3 Grammatik G gibt es einen NEA A mit $L(A) = L(G)$

- **Gegeben Grammatik $G = (V, T, P, S)$**
  - Wandle Erzeugung von Symbolen in Abarbeitung durch NEA um
  - Setze $A := (V, T, \delta, S, F)$ mit $\delta(X, a) = \{X' \mid X \rightarrow aX' \in P\}$ und $F = \{X \in V \mid X \rightarrow \epsilon \in P\}$

- **Zeige $L(A) = L(G)$**

$\boxed{X_n \rightarrow \varepsilon}$

$$\underline{w = w_1..w_n \in L(A)}$$

$$\Leftrightarrow\ \underline{\exists X_n \in F}. \qquad \underline{S}, \underline{w_1..w_n} \vdash^* \underline{X_n}, \boxed{\epsilon}$$

$$\Leftrightarrow\ \exists X_1, .., X_n \in V.\ \underline{S}, \underline{w_1}..w_n \vdash \underline{X_1}, w_2..w_n \vdash \ldots \vdash \underline{\boxed{X_n}, \epsilon} \ \wedge\ \boxed{X_n \in F}$$

$$\Leftrightarrow\ \exists X_1, .., X_n \in V.\ \underline{S \longrightarrow w_1X_1} \longrightarrow \ldots \longrightarrow \underline{w_1..w_n}\cancel{X_n} \longrightarrow \underline{w_1..w_n}$$

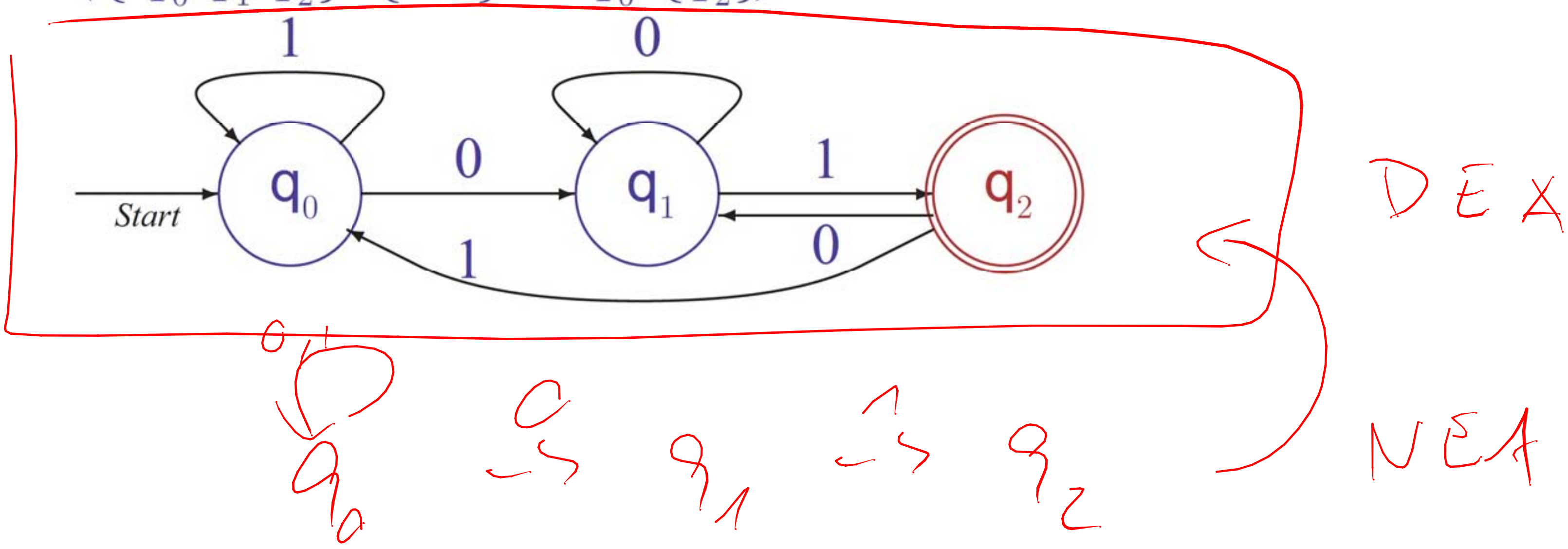
$$\Leftrightarrow\ S \xrightarrow{*} w$$

$$\Leftrightarrow\ w \in L(G) \ ✓$$

√

UMWANDLUNGEN AM BEISPIEL

- **Konvertiere DEA für** `(0+1)*01`
  - $A = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$ mit



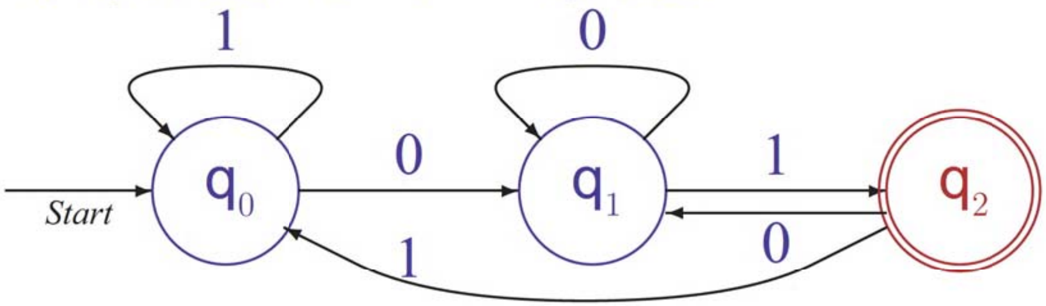
$q_0 \rightarrow 1q_0 \mid 0q_1$ $\quad (q_0 \rightarrow 1q_0, q_0 \rightarrow 0q_1)$

$q_1 \rightarrow 0q_1 \mid 1q_2$

$q_2 \rightarrow 0q_1 \mid 1q_0 \mid \varepsilon$

UMWANDLUNGEN AM BEISPIEL

- **Konvertiere DEA für** `(0+1)*01`
  - $A = (\{q_0,q_1,q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$ mit

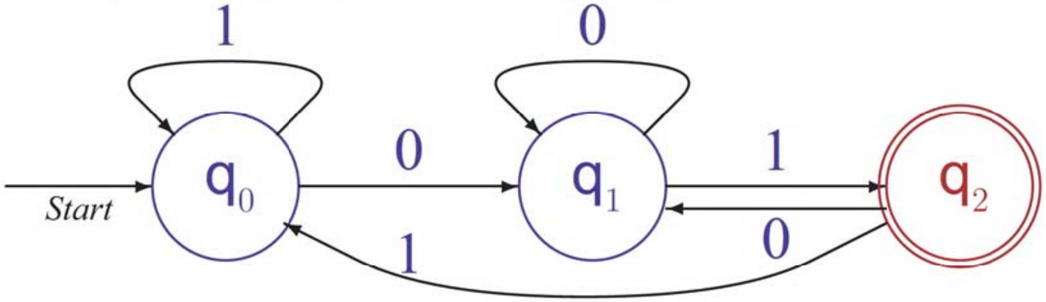


- **Erzeugte Grammatik**
  - $G = (\{q_0,q_1,q_2\}, \{0,1\}, P, q_0)$ mit

    $P = \{q_0 \rightarrow 1q_0,\ q_0 \rightarrow 0q_1,\ q_1 \rightarrow 1q_2,\ q_1 \rightarrow 0q_1,\ q_2 \rightarrow 1q_0,\ q_2 \rightarrow 0q_1,\ q_2 \rightarrow \epsilon\}$

UMWANDLUNGEN AM BEISPIEL

- **Konvertiere DEA für** `(0+1)*01`
  - $A = (\{ q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$ mit



- **Erzeugte Grammatik**
  - $G = (\{ q_0, q_1, q_2\}, \{0,1\}, P, q_0)$ mit

    $P = \{q_0 \to 1q_0,\ q_0 \to 0q_1,\ q_1 \to 1q_2,\ q_1 \to 0q_1,\ q_2 \to 1q_0,\ q_2 \to 0q_1,\ q_2 \to \epsilon\}$

- **Umwandlung von *G* in einen NEA**
  - Transformation erzeugt ursprünglichen Automaten

DIE CHOMSKY HIERARCHIE

$$\mathcal{L}_3 \subset \mathcal{L}_2 \subset \mathcal{L}_1 \subset \mathcal{L}_0$$

- **Wichtige Vertreter der Klassen**
  - $\mathcal{L}_2 - \mathcal{L}_3$: $\{0^n 1^n \mid n \in \mathbb{N}\}$
  - $\mathcal{L}_1 - \mathcal{L}_2$: $\{0^n 1^n 2^n \mid n \in \mathbb{N}\}$
  - $\mathcal{L}_0 - \mathcal{L}_1$: $\{w_i \in \{0,1\}^* \mid$ Das Programm mit Codierung w_i hält bei Eingabe w_i $\}$

TI-2

- **Zugehörige Automatenmodelle**
  - $\mathcal{L}_0$: Turingmaschine
  - $\mathcal{L}_1$: linear platzbeschränkte nichtdeterministische Turingmaschine
  - $\mathcal{L}_2$: nichtdeterministischer endlicher Automat mit Kellerspeicher
  - $\mathcal{L}_3$: endlicher Automat

Mehr in zukünftigen Vorlesungen