

Theoretische Informatik I



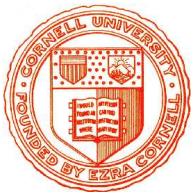
Einheit 3

Kontextfreie Sprachen



- 1. Kontextfreie Grammatiken**
- 2. Pushdown Automaten**
- 3. Eigenschaften kontextfreier Sprachen**

Theoretische Informatik I



Einheit 3.1

Kontextfreie Grammatiken



1. Grammatiken und Ableitungen
2. Ableitungsbäume
3. Mehrdeutigkeiten

● Grammatiken beschreiben Aufbau von Sprachen

- Produktionsregeln generieren nichtdeterministisch alle Worte der Sprache
- Grammatiken können sehr komplexe Sprachen beschreiben
- Chomsky Hierarchie klassifiziert Grammatiken nach “Freiheitsgraden”

- **Grammatiken beschreiben Aufbau von Sprachen**

- Produktionsregeln generieren nichtdeterministisch alle Worte der Sprache
- Grammatiken können sehr komplexe Sprachen beschreiben
- Chomsky Hierarchie klassifiziert Grammatiken nach “Freiheitsgraden”

- **Typ-3 Grammatiken sind einfach & effizient**

- Umwandelbar in endlichen Automaten und reguläre Ausdrücke
- Erkennung von Worten der Sprache in “Echtzeit”

- **Grammatiken beschreiben Aufbau von Sprachen**

- Produktionsregeln generieren nichtdeterministisch alle Worte der Sprache
- Grammatiken können sehr komplexe Sprachen beschreiben
- Chomsky Hierarchie klassifiziert Grammatiken nach “Freiheitsgraden”

- **Typ-3 Grammatiken sind einfach & effizient**

- Umwandelbar in endlichen Automaten und reguläre Ausdrücke
- Erkennung von Worten der Sprache in “Echtzeit”

- **Programmiersprachen können nicht regulär sein**

- Korrekte Klammerausdrücke / Schachtelungen sind nicht regulär
- Die meisten Programmstrukturen enthalten Schachtelungen
 - Blöcke, if-then-else, arithmetische Ausdrücke, ...

- **Grammatiken beschreiben Aufbau von Sprachen**

- Produktionsregeln generieren nichtdeterministisch alle Worte der Sprache
- Grammatiken können sehr komplexe Sprachen beschreiben
- Chomsky Hierarchie klassifiziert Grammatiken nach “Freiheitsgraden”

- **Typ-3 Grammatiken sind einfach & effizient**

- Umwandelbar in endlichen Automaten und reguläre Ausdrücke
- Erkennung von Worten der Sprache in “Echtzeit”

- **Programmiersprachen können nicht regulär sein**

- Korrekte Klammerausdrücke / Schachtelungen sind nicht regulär
- Die meisten Programmstrukturen enthalten Schachtelungen
 - Blöcke, if-then-else, arithmetische Ausdrücke, ...



Syntaxanalyse und Compilation von Programmiersprachen braucht kontextfreie Grammatiken

ANWENDUNGEN KONTEXTFREIER SPRACHEN

Alle bedeutenden Computersprachen sind kontextfrei

Alle bedeutenden Computersprachen sind kontextfrei

- **Programmiersprachen**

- Compiler kann kontextfreie Grammatiken effizient verarbeiten

Alle bedeutenden Computersprachen sind kontextfrei

● Programmiersprachen

- Compiler kann kontextfreie Grammatiken effizient verarbeiten
- Parser kann aus kontextfreier Grammatik automatisch erzeugt werden
 - Standard Unix tool YACC unterstützt schnellen Compilerentwurf

Alle bedeutenden Computersprachen sind kontextfrei

- **Programmiersprachen**

- Compiler kann kontextfreie Grammatiken effizient verarbeiten
- Parser kann aus kontextfreier Grammatik automatisch erzeugt werden
 - Standard Unix tool YACC unterstützt schnellen Compilerentwurf

- **Markup Sprachen**

Alle bedeutenden Computersprachen sind kontextfrei

● Programmiersprachen

- Compiler kann kontextfreie Grammatiken effizient verarbeiten
- Parser kann aus kontextfreier Grammatik automatisch erzeugt werden
 - Standard Unix tool YACC unterstützt schnellen Compilerentwurf

● Markup Sprachen

- HTML: Formattierung von Dokumenten mit Links zu Programmaufrufen
- XML: Einheitliche Beschreibung der Semantik von Dokumenten

Alle bedeutenden Computersprachen sind kontextfrei

● Programmiersprachen

- Compiler kann kontextfreie Grammatiken effizient verarbeiten
- Parser kann aus kontextfreier Grammatik automatisch erzeugt werden
 - Standard Unix tool YACC unterstützt schnellen Compilerentwurf

● Markup Sprachen

- HTML: Formattierung von Dokumenten mit Links zu Programmaufrufen
- XML: Einheitliche Beschreibung der Semantik von Dokumenten

Beide Sprachen erfordern die Mächtigkeit von kontextfreien Grammatiken

Mehr in HMU §5.3

- Eine kontextfreie Grammatik (kfG) ist ein 4-Tupel $G = (V, T, P, S)$ mit
 - T endliches Terminalalphabet
 - V endliches Hilfsalphabet mit $V \cap T = \emptyset$
 - $P \subseteq V \times \Gamma^*$ endliche Menge der Produktionen (wobei $\Gamma = V \cup T$)
 - $S \in V$ Startsymbol

Die übliche Schreibweise für Produktionen $(A, r) \in P$ ist $A \rightarrow r$

Eine kompakte Notation für $A \rightarrow r_1, A \rightarrow r_2 \dots A \rightarrow r_n$ ist $A \rightarrow r_1 | r_2 | \dots | r_n$

- Eine **kontextfreie Grammatik (kfG)** ist ein **4-Tupel** $G = (V, T, P, S)$ mit
 - T endliches **Terminalalphabet**
 - V endliches **Hilfsalphabet** mit $V \cap T = \emptyset$
 - $P \subseteq V \times \Gamma^*$ endliche Menge der **Produktionen** (wobei $\Gamma = V \cup T$)
 - $S \in V$ **Startsymbol**

Die übliche Schreibweise für Produktionen $(A, r) \in P$ ist $A \rightarrow r$

Eine kompakte Notation für $A \rightarrow r_1, A \rightarrow r_2 \dots A \rightarrow r_n$ ist $A \rightarrow r_1 | r_2 | \dots | r_n$

- **Ableitungbarkeit in G**

- $w \rightarrow z \equiv \exists x, y \in \Gamma^*. \exists A \rightarrow r \in P. w = x A y \wedge z = x r y$
- $w \xrightarrow{*} z \equiv \exists n \in \mathbb{N}. w \xrightarrow{n} z$
wobei $w \xrightarrow{0} z \equiv w = z$ und $w \xrightarrow{n+1} z \equiv \exists u \in \Gamma^*. w \rightarrow u \wedge u \xrightarrow{n} z$

- Eine **kontextfreie Grammatik (kfG)** ist ein **4-Tupel** $G = (V, T, P, S)$ mit
 - T endliches **Terminalalphabet**
 - V endliches **Hilfsalphabet** mit $V \cap T = \emptyset$
 - $P \subseteq V \times \Gamma^*$ endliche Menge der **Produktionen** (wobei $\Gamma = V \cup T$)
 - $S \in V$ **Startsymbol**

Die übliche Schreibweise für Produktionen $(A, r) \in P$ ist $A \rightarrow r$

Eine kompakte Notation für $A \rightarrow r_1, A \rightarrow r_2 \dots A \rightarrow r_n$ ist $A \rightarrow r_1 | r_2 | \dots | r_n$

- **Ableitungbarkeit in G**

- $w \rightarrow z \equiv \exists x, y \in \Gamma^*. \exists A \rightarrow r \in P. w = x A y \wedge z = x r y$
- $w \xrightarrow{*} z \equiv \exists n \in \mathbb{N}. w \xrightarrow{n} z$
wobei $w \xrightarrow{0} z \equiv w = z$ und $w \xrightarrow{n+1} z \equiv \exists u \in \Gamma^*. w \rightarrow u \wedge u \xrightarrow{n} z$

- **Von G erzeugte Sprache:**

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

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,)\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,)\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{ ({}^k)^k \mid k \in \mathbb{N} \}$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,)\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,)\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,)\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,)\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

- $S \xrightarrow{1} w$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,)\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,)\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P$$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,)\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,)\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon$$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,)\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,)\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon \Leftrightarrow w = (^0)^0$$



GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,),\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,),\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon \Leftrightarrow w = (^0)^0$$



Induktionsschritt

$$- \text{Es gelte } \forall v \in \{(,),\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = (^k)^k$$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,),\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,),\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon \Leftrightarrow w = (^0)^0$$



Induktionsschritt

$$- \text{Es gelte } \forall v \in \{(,),\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = (^k)^k$$

$$- S \xrightarrow{k+2} w$$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,),\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,),\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon \Leftrightarrow w = (^0)^0$$



Induktionsschritt

- Es gelte $\forall v \in \{(,),\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = (^k)^k$
- $S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow (S) \xrightarrow{k+1} w$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,),\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,),\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon \Leftrightarrow w = (^0)^0$$



Induktionsschritt

$$\begin{aligned} - \text{Es gelte } \forall v \in \{(,),\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = (^k)^k \\ - S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow (S) \xrightarrow{k+1} w \\ \Leftrightarrow \exists v \in \{(,),\}^*. S \xrightarrow{k+1} v \wedge w = (v) \end{aligned}$$

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,)\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,)\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon \Leftrightarrow w = (^0)^0$$



Induktionsschritt

$$\begin{aligned} - \text{Es gelte } \forall v \in \{(,)\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = (^k)^k \\ - S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow (S) \xrightarrow{k+1} w \\ \Leftrightarrow \exists v \in \{(,)\}^*. S \xrightarrow{k+1} v \wedge w = (v) \\ \Leftrightarrow \exists v \in \{(,)\}^*. v = (^k)^k \wedge w = (v) \end{aligned}$$

(Annahme)

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,),\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,),\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon \Leftrightarrow w = (^0)^0$$

✓

Induktionsschritt

$$\begin{aligned} - \text{Es gelte } \forall v \in \{(,),\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = (^k)^k \\ - S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow (S) \xrightarrow{k+1} w \\ \Leftrightarrow \exists v \in \{(,),\}^*. S \xrightarrow{k+1} v \wedge w = (v) \\ \Leftrightarrow \exists v \in \{(,),\}^*. v = (^k)^k \wedge w = (v) \quad (\text{Annahme}) \\ \Leftrightarrow w = (^{k+1})^{k+1} \end{aligned}$$

✓

GRAMMATIK FÜR GESCHACHTELTE KLAMMERAUSDRÜCKE

$$G_4 = (\{S\}, \{(,),\}, \{S \rightarrow (S), S \rightarrow \epsilon\}, S)$$

- Zeige $L(G_4) = \{(^k)^k \mid k \in \mathbb{N}\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{(,),\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = (^k)^k$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w = \epsilon \Leftrightarrow w = (^0)^0$$

✓

Induktionsschritt

$$\begin{aligned} - \text{Es gelte } \forall v \in \{(,),\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = (^k)^k \\ - S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow (S) \xrightarrow{k+1} w \\ \Leftrightarrow \exists v \in \{(,),\}^*. S \xrightarrow{k+1} v \wedge w = (v) \\ \Leftrightarrow \exists v \in \{(,),\}^*. v = (^k)^k \wedge w = (v) \quad (\text{Annahme}) \\ \Leftrightarrow w = (^{k+1})^{k+1} \end{aligned}$$

✓

$$\{(^k)^k \mid k \in \mathbb{N}\} \in \mathcal{L}_2 - \mathcal{L}_3$$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
 - Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$
- Basisfall
- $S \xrightarrow{1} w$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
 - Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$
- Basisfall
- $S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
 - Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$
- Basisfall
- $S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w \in \{0, 1, \epsilon\}$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
 - Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$
- Basisfall
- $S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w \in \{0, 1, \epsilon\} \Leftrightarrow w = w^R \wedge |w| \in \{0, 1\}$ ✓

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w \in \{0, 1, \epsilon\} \Leftrightarrow w = w^R \wedge |w| \in \{0, 1\} \quad \checkmark$$

Induktionsschritt

$$- \text{Es gelte } \forall v \in \{0, 1\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = v^R \wedge |v| \in \{2k, 2k+1\}$$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w \in \{0, 1, \epsilon\} \Leftrightarrow w = w^R \wedge |w| \in \{0, 1\} \quad \checkmark$$

Induktionsschritt

- Es gelte $\forall v \in \{0, 1\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = v^R \wedge |v| \in \{2k, 2k+1\}$
- $S \xrightarrow{k+2} w$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w \in \{0, 1, \epsilon\} \Leftrightarrow w = w^R \wedge |w| \in \{0, 1\} \quad \checkmark$$

Induktionsschritt

$$\begin{aligned} - & \text{Es gelte } \forall v \in \{0, 1\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = v^R \wedge |v| \in \{2k, 2k+1\} \\ - & S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow 0S0 \xrightarrow{k+1} w \vee S \rightarrow 1S1 \xrightarrow{k+1} w \end{aligned}$$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w \in \{0, 1, \epsilon\} \Leftrightarrow w = w^R \wedge |w| \in \{0, 1\} \quad \checkmark$$

Induktionsschritt

$$\begin{aligned} - \text{Es gelte } \forall v \in \{0, 1\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = v^R \wedge |v| \in \{2k, 2k+1\} \\ - S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow 0S0 \xrightarrow{k+1} w \vee S \rightarrow 1S1 \xrightarrow{k+1} w \\ \Leftrightarrow \exists v \in \{0, 1\}^*. S \xrightarrow{k+1} v \wedge w = 0v0 \vee w = 1v1 \end{aligned}$$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w \in \{0, 1, \epsilon\} \Leftrightarrow w = w^R \wedge |w| \in \{0, 1\} \quad \checkmark$$

Induktionsschritt

$$\begin{aligned} - \text{Es gelte } \forall v \in \{0, 1\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = v^R \wedge |v| \in \{2k, 2k+1\} \\ - S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow 0S0 \xrightarrow{k+1} w \vee S \rightarrow 1S1 \xrightarrow{k+1} w \\ \Leftrightarrow \exists v \in \{0, 1\}^*. S \xrightarrow{k+1} v \wedge w = 0v0 \vee w = 1v1 \\ \Leftrightarrow \exists v \in \{0, 1\}^*. v = v^R \wedge |v| \in \{2k, 2k+1\} \wedge w = 0v0 \vee w = 1v1 \end{aligned}$$

KONTEXTFREIE GRAMMATIK FÜR PALINDROME

$$L(G_5) = \{w \in \{0, 1\}^* \mid w = w^R\}$$

- $G_5 = (\{S\}, \{0, 1\}, P, S)$ mit
 $P = \{S \rightarrow \epsilon, S \rightarrow 0, S \rightarrow 1, S \rightarrow 0S0, S \rightarrow 1S1\}$
- Beweise durch Induktion über Länge der Ableitung
 - $\forall k \in \mathbb{N}. \forall w \in \{0, 1\}^*. S \xrightarrow{k+1} w \Leftrightarrow w = w^R \wedge |w| \in \{2k, 2k+1\}$

Basisfall

$$- S \xrightarrow{1} w \Leftrightarrow (S \rightarrow w) \in P \Leftrightarrow w \in \{0, 1, \epsilon\} \Leftrightarrow w = w^R \wedge |w| \in \{0, 1\} \quad \checkmark$$

Induktionsschritt

$$\begin{aligned} - \text{Es gelte } \forall v \in \{0, 1\}^*. S \xrightarrow{k+1} v \Leftrightarrow v = v^R \wedge |v| \in \{2k, 2k+1\} \\ - S \xrightarrow{k+2} w \Leftrightarrow S \rightarrow 0S0 \xrightarrow{k+1} w \vee S \rightarrow 1S1 \xrightarrow{k+1} w \\ \Leftrightarrow \exists v \in \{0, 1\}^*. S \xrightarrow{k+1} v \wedge w = 0v0 \vee w = 1v1 \\ \Leftrightarrow \exists v \in \{0, 1\}^*. v = v^R \wedge |v| \in \{2k, 2k+1\} \wedge w = 0v0 \vee w = 1v1 \\ \Leftrightarrow w = w^R \wedge |w| \in \{2k+2, 2k+3\} \quad \checkmark \end{aligned}$$

● Ausdrücke über Operatoren + und *

- Bezeichner (**I**entifier): Buchstabe gefolgt von Buchstaben/Ziffern
 - Buchstaben **a**, **b**, Ziffern **0**, **1**
- Ausdrücke (**E**xpressions): Schachtelung mit **+**, ***** und Klammern

- Ausdrücke über Operatoren + und *

- Bezeichner (**I**entifier): Buchstabe gefolgt von Buchstaben/Ziffern
 - Buchstaben **a**, **b**, Ziffern **0**, **1**
- Ausdrücke (**E**xpressions): Schachtelung mit **+**, ***** und Klammern

- $G_6 = (\{E, I\}, \{a, b, 0, 1, +, *, (,)\}, P, E)$

mit $P = \{ E \rightarrow I \mid E+E \mid E*E \mid (E) \mid I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$

LINKS- UND RECHTSSEITIGE ABLEITUNGEN

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$E \longrightarrow E * E$$

LINKS- UND RECHTSSEITIGE ABLEITUNGEN

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$E \longrightarrow E * E \longrightarrow I * E$

LINKS- UND RECHTSSEITIGE ABLEITUNGEN

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$E \longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E)$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \end{aligned}$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

- In w wird die am weitesten links stehende Variable ersetzt

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$E \longrightarrow_L E * E$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$E \longrightarrow_L E * E \longrightarrow_L I * E$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$E \longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$E \longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E)$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \longrightarrow_L a * (a + I) \end{aligned}$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \longrightarrow_L a * (a + I) \\ &\longrightarrow_L a * (a + I0) \longrightarrow_L a * (a + I00) \longrightarrow_L a * (a + b00) \end{aligned}$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \longrightarrow_L a * (a + I) \\ &\longrightarrow_L a * (a + I0) \longrightarrow_L a * (a + I00) \longrightarrow_L a * (a + b00) \end{aligned}$$

- Rechtsseitige Ableitung $w \xrightarrow{R} z$

– In w wird die am weitesten rechts stehende Variable ersetzt

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E * E \longrightarrow I * E \longrightarrow I * (E) \\ &\longrightarrow I * (E + E) \longrightarrow I * (I + E) \longrightarrow I * (I + I) \longrightarrow I * (a + I) \\ &\longrightarrow I * (a + I0) \longrightarrow I * (a + I00) \longrightarrow I * (a + b00) \longrightarrow a * (a + b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \longrightarrow_L a * (a + I) \\ &\longrightarrow_L a * (a + I0) \longrightarrow_L a * (a + I00) \longrightarrow_L a * (a + b00) \end{aligned}$$

- Rechtsseitige Ableitung $w \xrightarrow{R} z$

– In w wird die am weitesten rechts stehende Variable ersetzt

$$E \longrightarrow_R E * E$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned} E &\longrightarrow E*E \longrightarrow I*E \longrightarrow I*(E) \\ &\longrightarrow I*(E+E) \longrightarrow I*(I+E) \longrightarrow I*(I+I) \longrightarrow I*(a+I) \\ &\longrightarrow I*(a+I0) \longrightarrow I*(a+I00) \longrightarrow I*(a+b00) \longrightarrow a*(a+b00) \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned} E &\longrightarrow_L E*E \longrightarrow_L I*E \longrightarrow_L a*E \longrightarrow_L a*(E) \\ &\longrightarrow_L a*(E+E) \longrightarrow_L a*(I+E) \longrightarrow_L a*(a+E) \longrightarrow_L a*(a+I) \\ &\longrightarrow_L a*(a+I0) \longrightarrow_L a*(a+I00) \longrightarrow_L a*(a+b00) \end{aligned}$$

- Rechtsseitige Ableitung $w \xrightarrow{R} z$

– In w wird die am weitesten rechts stehende Variable ersetzt

$$E \longrightarrow_R E*E \longrightarrow_R E*(E)$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned}
 E &\longrightarrow E*E \longrightarrow I*E \longrightarrow I*(E) \\
 &\longrightarrow I*(E+E) \longrightarrow I*(I+E) \longrightarrow I*(I+I) \longrightarrow I*(a+I) \\
 &\longrightarrow I*(a+I0) \longrightarrow I*(a+I00) \longrightarrow I*(a+b00) \longrightarrow a*(a+b00)
 \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned}
 E &\longrightarrow_L E*E \longrightarrow_L I*E \longrightarrow_L a*E \longrightarrow_L a*(E) \\
 &\longrightarrow_L a*(E+E) \longrightarrow_L a*(I+E) \longrightarrow_L a*(a+E) \longrightarrow_L a*(a+I) \\
 &\longrightarrow_L a*(a+I0) \longrightarrow_L a*(a+I00) \longrightarrow_L a*(a+b00)
 \end{aligned}$$

- Rechtsseitige Ableitung $w \xrightarrow{R} z$

– In w wird die am weitesten rechts stehende Variable ersetzt

$$E \longrightarrow_R E*E \longrightarrow_R E*(E) \longrightarrow_R E*(E+E)$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned}
 E &\longrightarrow E*E \longrightarrow I*E \longrightarrow I*(E) \\
 &\longrightarrow I*(E+E) \longrightarrow I*(I+E) \longrightarrow I*(I+I) \longrightarrow I*(a+I) \\
 &\longrightarrow I*(a+I0) \longrightarrow I*(a+I00) \longrightarrow I*(a+b00) \longrightarrow a*(a+b00)
 \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned}
 E &\longrightarrow_L E*E \longrightarrow_L I*E \longrightarrow_L a*E \longrightarrow_L a*(E) \\
 &\longrightarrow_L a*(E+E) \longrightarrow_L a*(I+E) \longrightarrow_L a*(a+E) \longrightarrow_L a*(a+I) \\
 &\longrightarrow_L a*(a+I0) \longrightarrow_L a*(a+I00) \longrightarrow_L a*(a+b00)
 \end{aligned}$$

- Rechtsseitige Ableitung $w \xrightarrow{R} z$

– In w wird die am weitesten rechts stehende Variable ersetzt

$$E \longrightarrow_R E*E \longrightarrow_R E*(E) \longrightarrow_R E*(E+E) \longrightarrow_R E*(E+I)$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned}
 E &\longrightarrow E*E \longrightarrow I*E \longrightarrow I*(E) \\
 &\longrightarrow I*(E+E) \longrightarrow I*(I+E) \longrightarrow I*(I+I) \longrightarrow I*(a+I) \\
 &\longrightarrow I*(a+I0) \longrightarrow I*(a+I00) \longrightarrow I*(a+b00) \longrightarrow a*(a+b00)
 \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned}
 E &\longrightarrow_L E*E \longrightarrow_L I*E \longrightarrow_L a*E \longrightarrow_L a*(E) \\
 &\longrightarrow_L a*(E+E) \longrightarrow_L a*(I+E) \longrightarrow_L a*(a+E) \longrightarrow_L a*(a+I) \\
 &\longrightarrow_L a*(a+I0) \longrightarrow_L a*(a+I00) \longrightarrow_L a*(a+b00)
 \end{aligned}$$

- Rechtsseitige Ableitung $w \xrightarrow{R} z$

– In w wird die am weitesten rechts stehende Variable ersetzt

$$\begin{aligned}
 E &\longrightarrow_R E*E \longrightarrow_R E*(E) \longrightarrow_R E*(E+E) \longrightarrow_R E*(E+I) \\
 &\longrightarrow_R E*(E+I0) \longrightarrow_R E*(E+I00) \longrightarrow_R E*(E+b00)
 \end{aligned}$$

Strategie für Auswahl von Produktionen

- Beliebige Ableitung

$$\begin{aligned}
 E &\longrightarrow E*E \longrightarrow I*E \longrightarrow I*(E) \\
 &\longrightarrow I*(E+E) \longrightarrow I*(I+E) \longrightarrow I*(I+I) \longrightarrow I*(a+I) \\
 &\longrightarrow I*(a+I0) \longrightarrow I*(a+I00) \longrightarrow I*(a+b00) \longrightarrow a*(a+b00)
 \end{aligned}$$

- Linksseitige Ableitung $w \xrightarrow{L} z$

– In w wird die am weitesten links stehende Variable ersetzt

$$\begin{aligned}
 E &\longrightarrow_L E*E \longrightarrow_L I*E \longrightarrow_L a*E \longrightarrow_L a*(E) \\
 &\longrightarrow_L a*(E+E) \longrightarrow_L a*(I+E) \longrightarrow_L a*(a+E) \longrightarrow_L a*(a+I) \\
 &\longrightarrow_L a*(a+I0) \longrightarrow_L a*(a+I00) \longrightarrow_L a*(a+b00)
 \end{aligned}$$

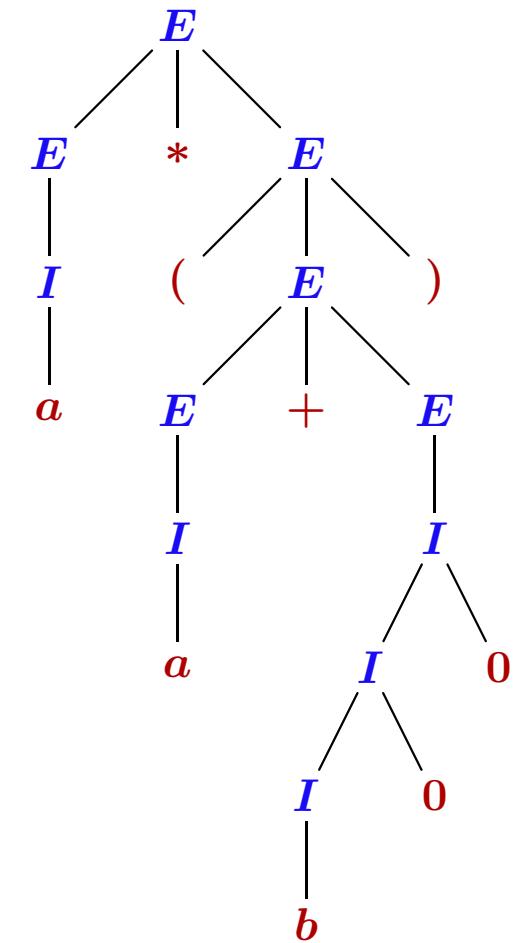
- Rechtsseitige Ableitung $w \xrightarrow{R} z$

– In w wird die am weitesten rechts stehende Variable ersetzt

$$\begin{aligned}
 E &\longrightarrow_R E*E \longrightarrow_R E*(E) \longrightarrow_R E*(E+E) \longrightarrow_R E*(E+I) \\
 &\longrightarrow_R E*(E+I0) \longrightarrow_R E*(E+I00) \longrightarrow_R E*(E+b00) \\
 &\longrightarrow_R E*(I+b00) \longrightarrow_R E*(a+b00) \longrightarrow_R I*(a+b00) \longrightarrow_R a*(a+b00)
 \end{aligned}$$

ABLEITUNGSBÄUME (PARSEBÄUME)

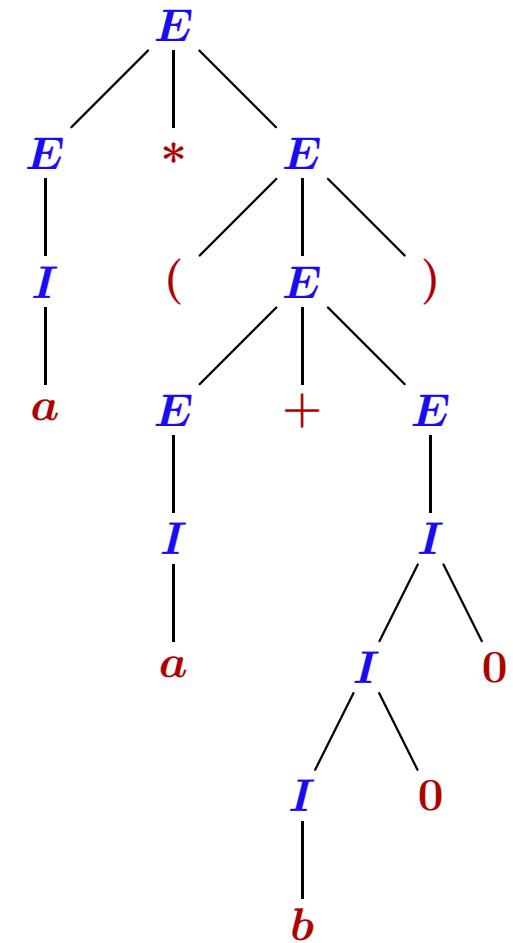
Baumdarstellung von Ableitungen



ABLEITUNGSBÄUME (PARSEBÄUME)

Baumdarstellung von Ableitungen

- Geordneter markierter Baum

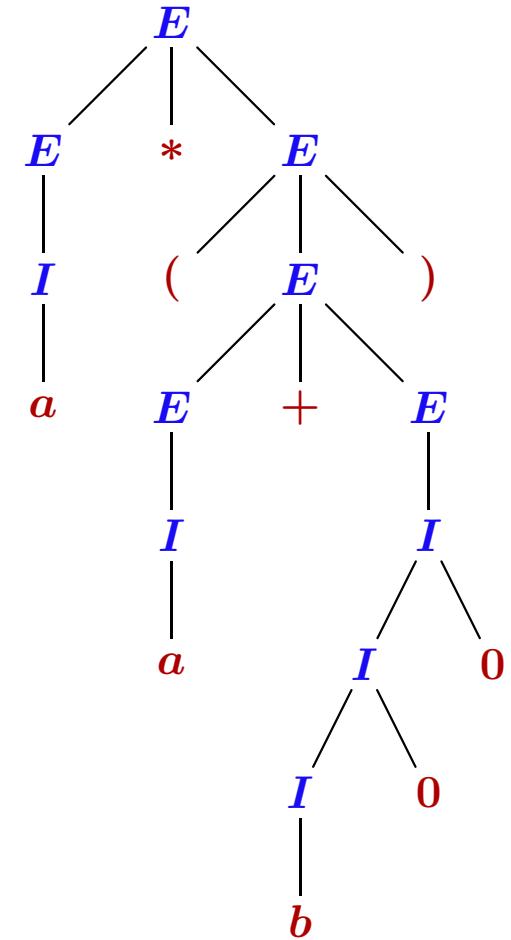


ABLEITUNGSBÄUME (PARSEBÄUME)

Baumdarstellung von Ableitungen

- Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert

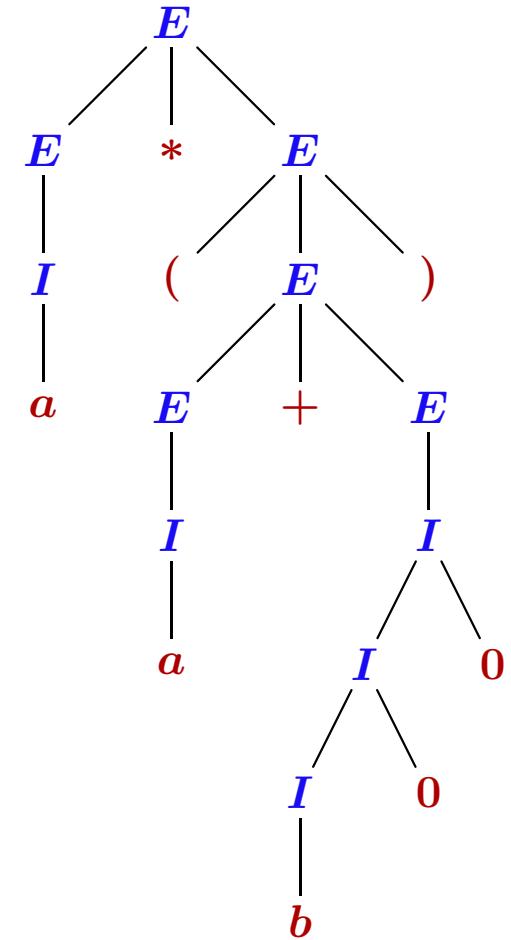


ABLEITUNGSBÄUME (PARSEBÄUME)

Baumdarstellung von Ableitungen

- Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert
- Wurzel markiert mit Startsymbol

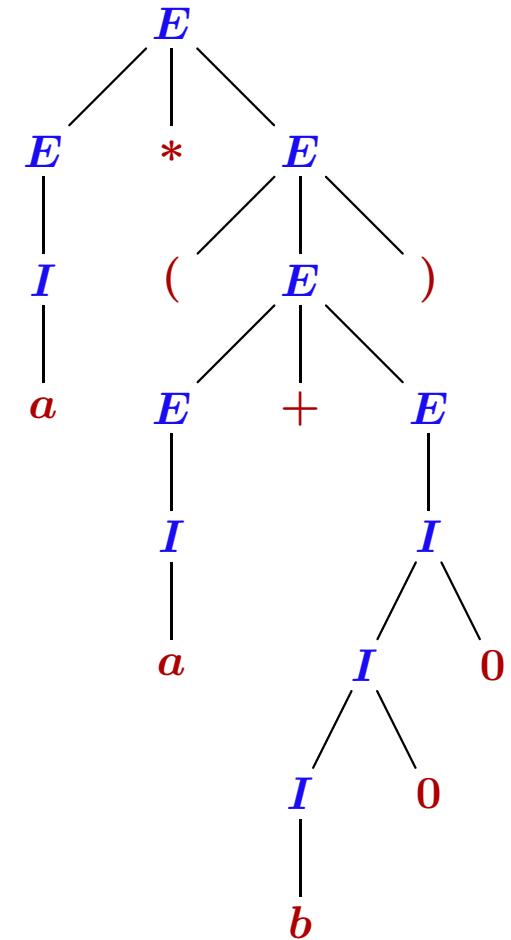


ABLEITUNGSBÄUME (PARSEBÄUME)

Baumdarstellung von Ableitungen

● Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert
- Wurzel markiert mit Startsymbol
- Blätter mit Terminalsymbolen $a \in T$ oder mit ϵ markiert

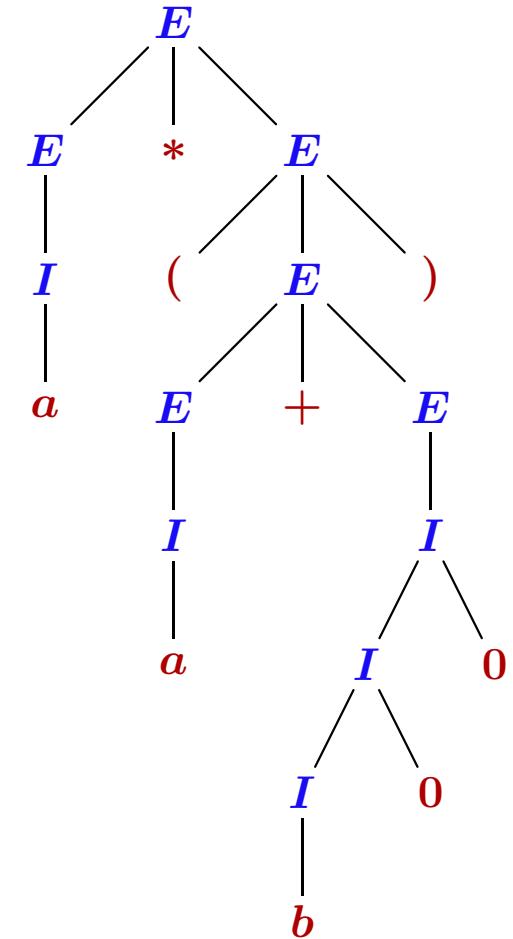


ABLEITUNGSBÄUME (PARSEBÄUME)

Baumdarstellung von Ableitungen

● Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert
- Wurzel markiert mit Startsymbol
- Blätter mit Terminalsymbolen $a \in T$ oder mit ϵ markiert
- Hat ein innerer Knoten Markierung A und Nachfolger mit Markierungen $v_1 \dots v_n$ so ist $A \rightarrow v_1 \dots v_n \in P$



ABLEITUNGSBÄUME (PARSEBÄUME)

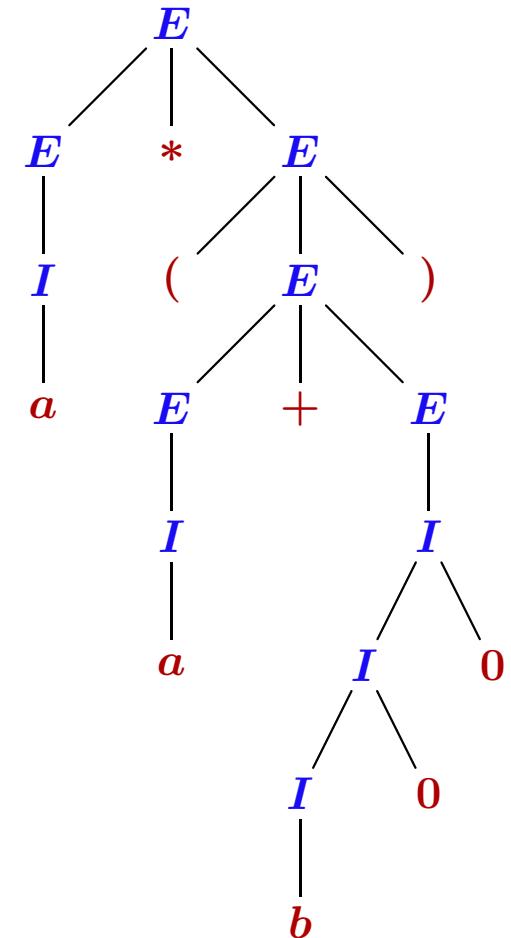
Baumdarstellung von Ableitungen

● Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert
- Wurzel markiert mit Startsymbol
- Blätter mit Terminalsymbolen $a \in T$ oder mit ϵ markiert
- Hat ein innerer Knoten Markierung A und Nachfolger mit Markierungen $v_1 \dots v_n$ so ist $A \rightarrow v_1 \dots v_n \in P$

● Exkurs: Notation für Bäume

- Baum: Sammlungen von Knoten mit Nachfolgerrelation



ABLEITUNGSBÄUME (PARSEBÄUME)

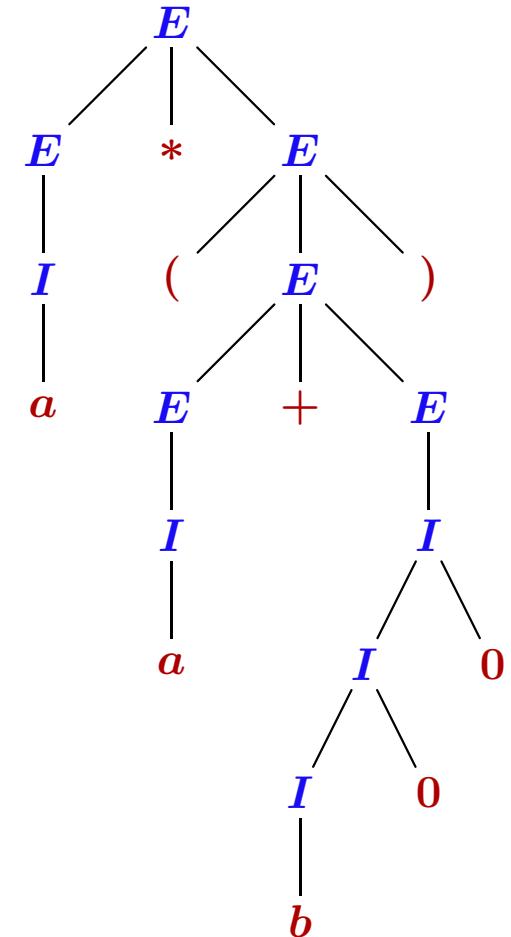
Baumdarstellung von Ableitungen

● Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert
- Wurzel markiert mit Startsymbol
- Blätter mit Terminalsymbolen $a \in T$ oder mit ϵ markiert
- Hat ein innerer Knoten Markierung A und Nachfolger mit Markierungen $v_1 \dots v_n$ so ist $A \rightarrow v_1 \dots v_n \in P$

● Exkurs: Notation für Bäume

- Baum: Sammlungen von Knoten mit Nachfolgerrelation
- Nachfolger sind geordnet



ABLEITUNGSBÄUME (PARSEBÄUME)

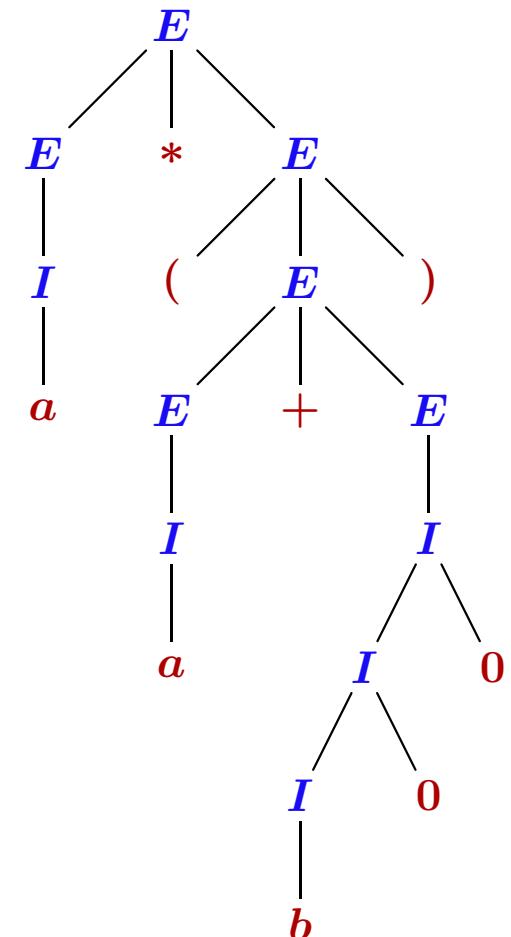
Baumdarstellung von Ableitungen

• Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert
 - Wurzel markiert mit Startsymbol
 - Blätter mit Terminalsymbolen $a \in T$ oder mit ϵ markiert
 - Hat ein innerer Knoten Markierung A und Nachfolger mit Markierungen $v_1 \dots v_n$ so ist $A \rightarrow v_1 \dots v_n \in P$

• Exkurs: Notation für Bäume

- Baum: Sammlungen von Knoten mit Nachfolgerrelation
 - Nachfolger sind geordnet
 - Ein Knoten hat maximal einen Vorgänger
 - Wurzel: Knoten ohne Vorgänger



ABLEITUNGSBÄUME (PARSEBÄUME)

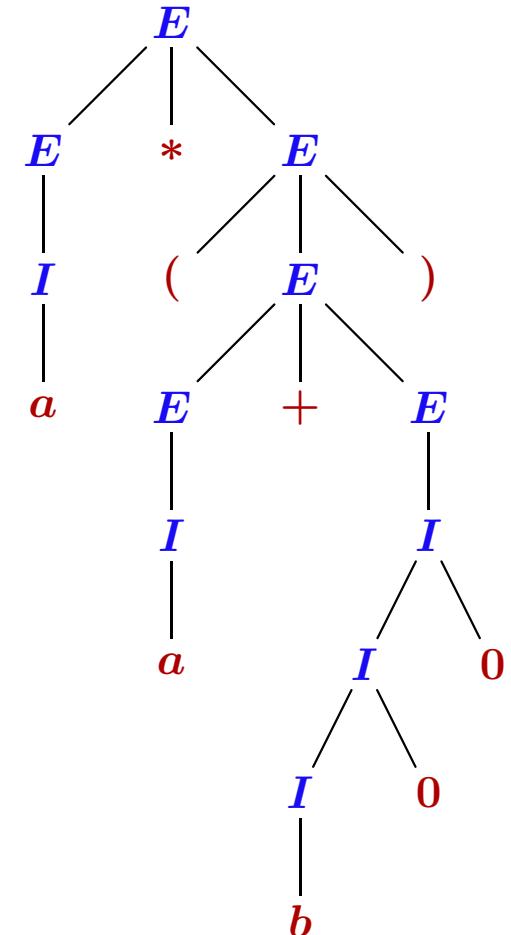
Baumdarstellung von Ableitungen

● Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert
- Wurzel markiert mit Startsymbol
- Blätter mit Terminalsymbolen $a \in T$ oder mit ϵ markiert
- Hat ein innerer Knoten Markierung A und Nachfolger mit Markierungen $v_1 \dots v_n$ so ist $A \rightarrow v_1 \dots v_n \in P$

● Exkurs: Notation für Bäume

- Baum: Sammlungen von Knoten mit Nachfolgerrelation
- Nachfolger sind geordnet
- Ein Knoten hat maximal einen Vorgänger
- Wurzel: Knoten ohne Vorgänger
- Blatt / Innerer Knoten: Knoten ohne/mit Nachfolger



ABLEITUNGSBÄUME (PARSEBÄUME)

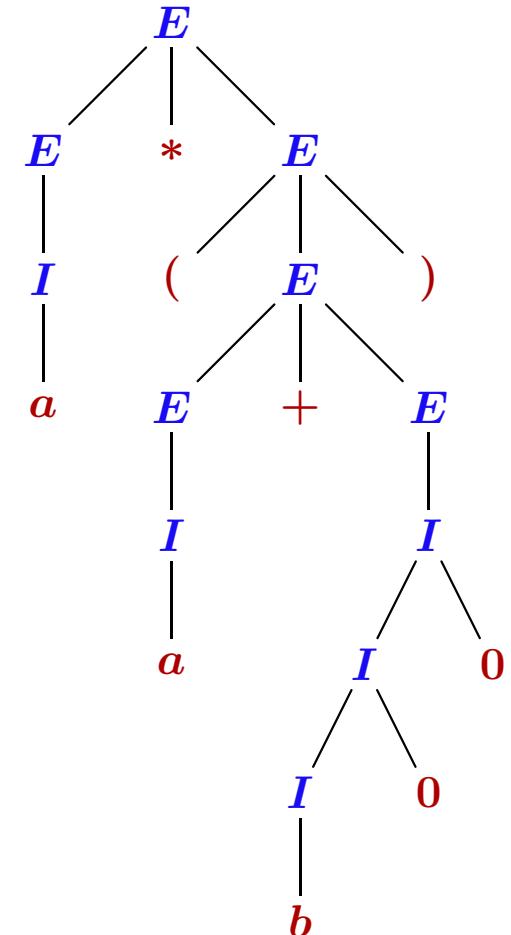
Baumdarstellung von Ableitungen

● Geordneter markierter Baum

- Innere Knoten mit Variablen $A \in V$ markiert
- Wurzel markiert mit Startsymbol
- Blätter mit Terminalsymbolen $a \in T$ oder mit ϵ markiert
- Hat ein innerer Knoten Markierung A und Nachfolger mit Markierungen $v_1 \dots v_n$ so ist $A \rightarrow v_1 \dots v_n \in P$

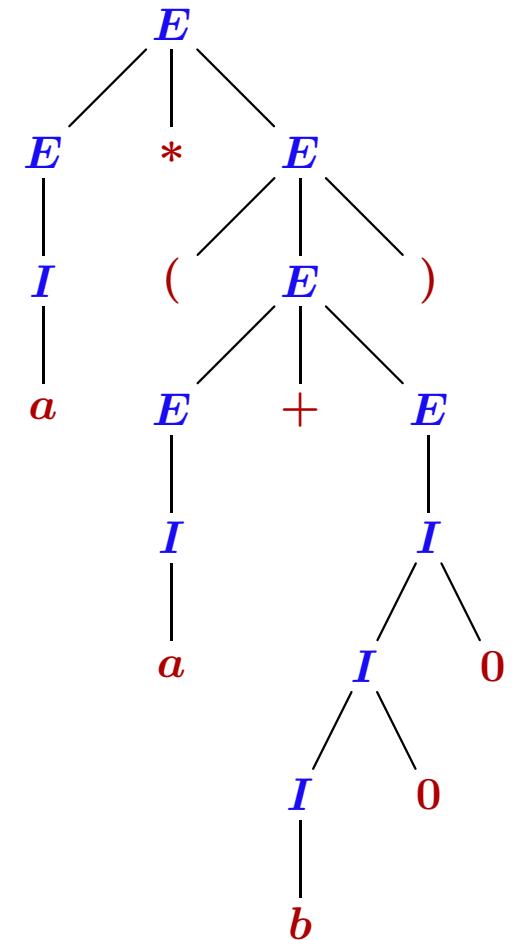
● Exkurs: Notation für Bäume

- Baum: Sammlungen von Knoten mit Nachfolgerrelation
- Nachfolger sind geordnet
- Ein Knoten hat maximal einen Vorgänger
- Wurzel: Knoten ohne Vorgänger
- Blatt / Innerer Knoten: Knoten ohne/mit Nachfolger
- Nachkommen: transitive Hülle der Nachfolgerrelation



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

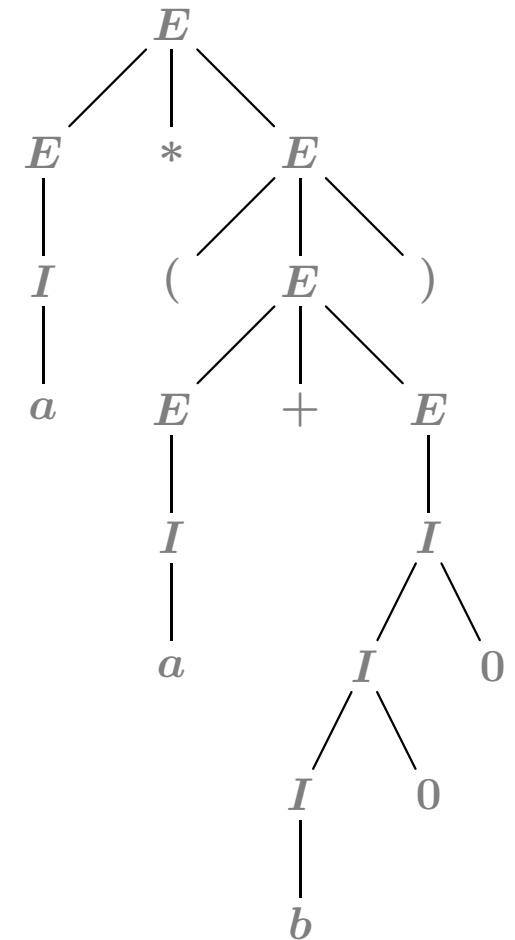
- Blätter repräsentieren Terminalwörter



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalwörter

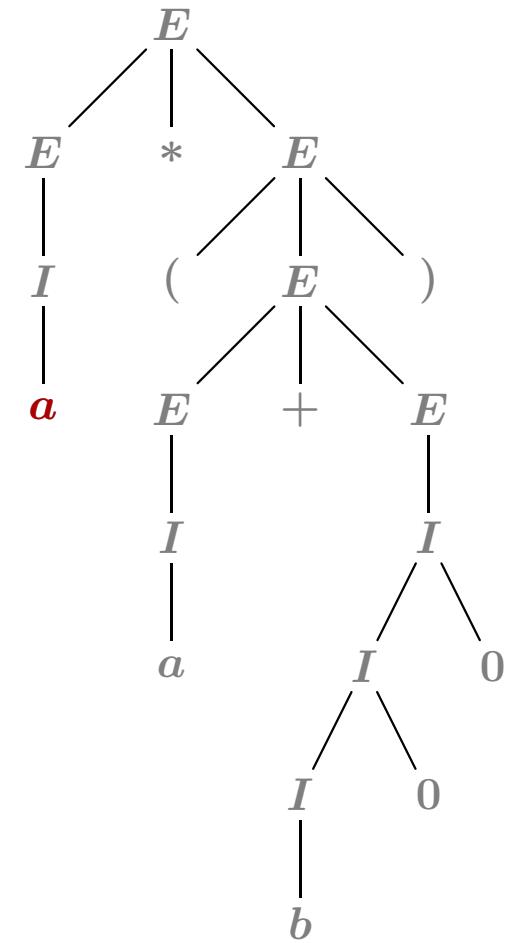
- Auslesen durch Tiefensuche von links nach rechts



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalwörter

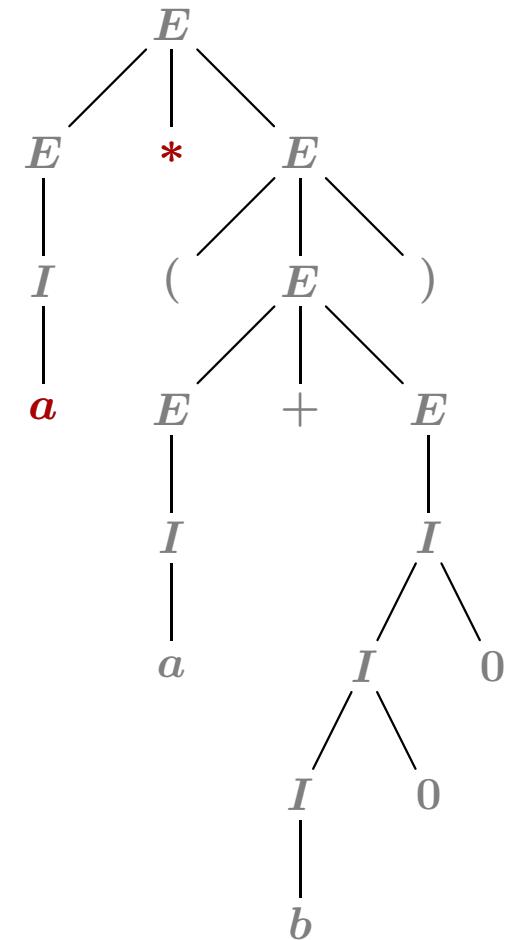
- Auslesen durch Tiefensuche von links nach rechts
- $\textcolor{red}{a}$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalwörter

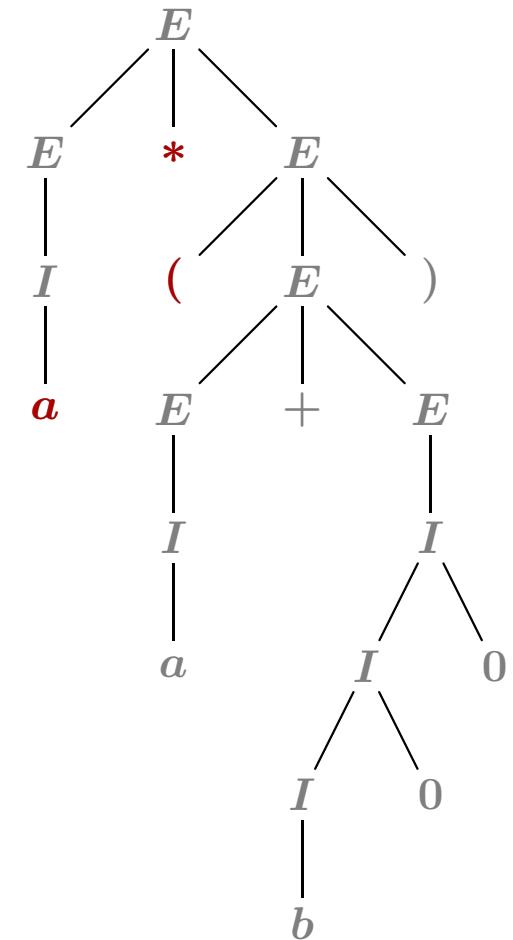
- Auslesen durch Tiefensuche von links nach rechts
- a^*



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalworte

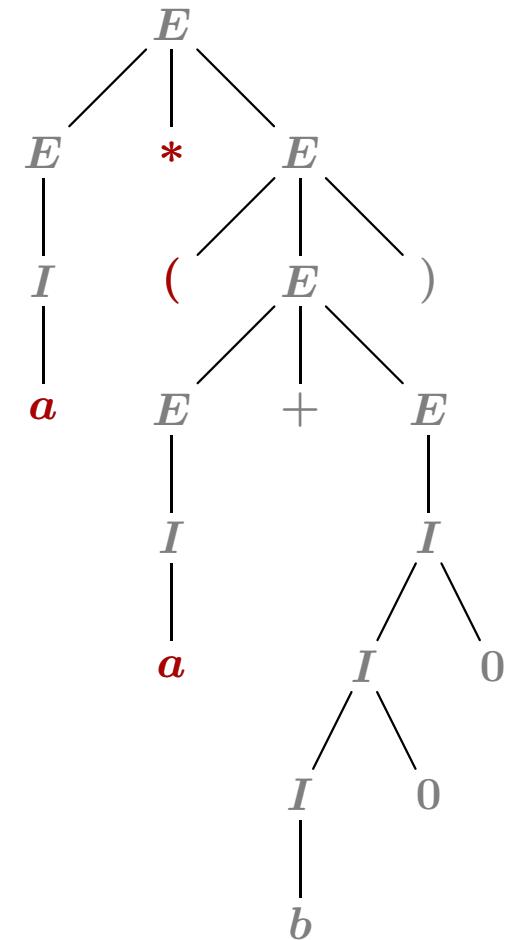
- Auslesen durch Tiefensuche von links nach rechts
- $a * ($



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalworte

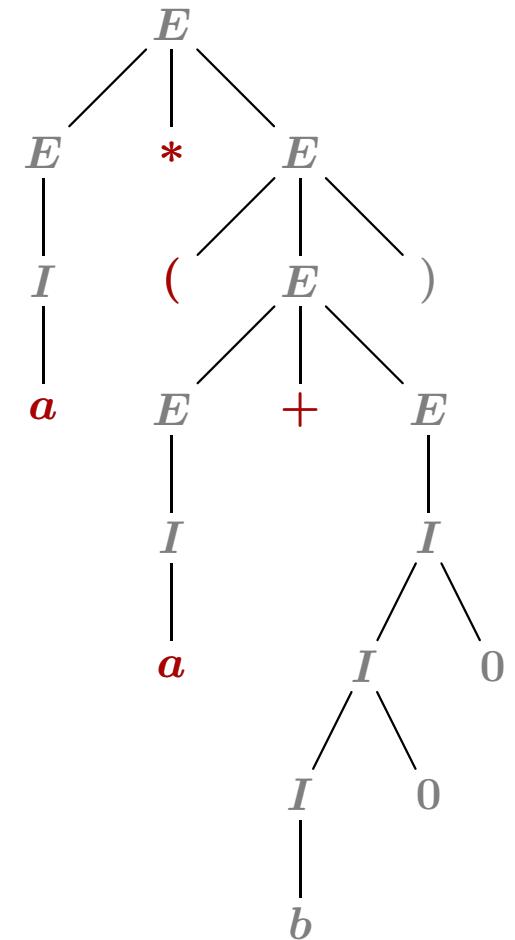
- Auslesen durch Tiefensuche von links nach rechts
- $a * (a$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalworte

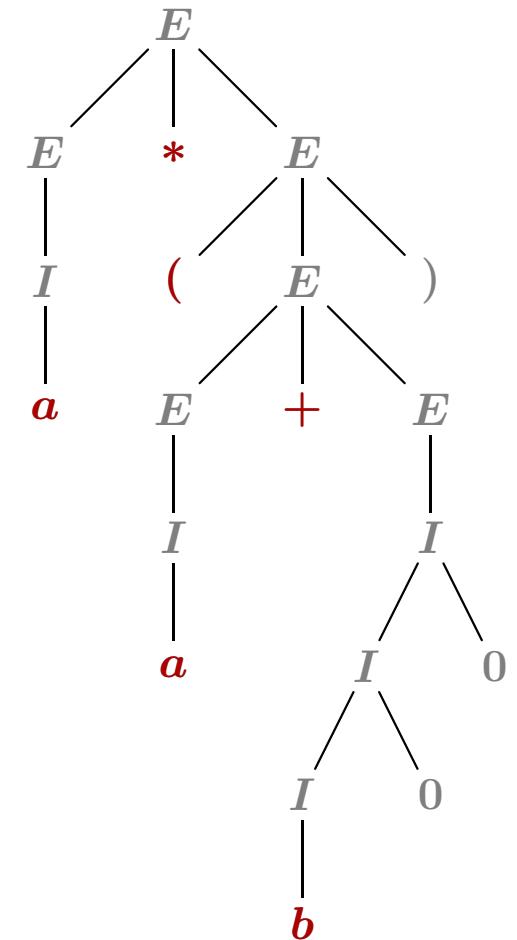
- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b)^*$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalwörter

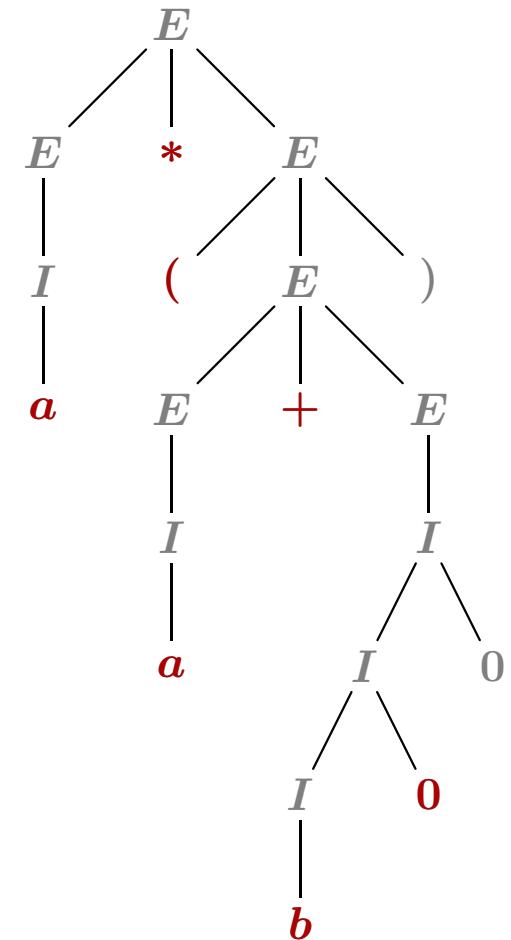
- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b)$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalwörter

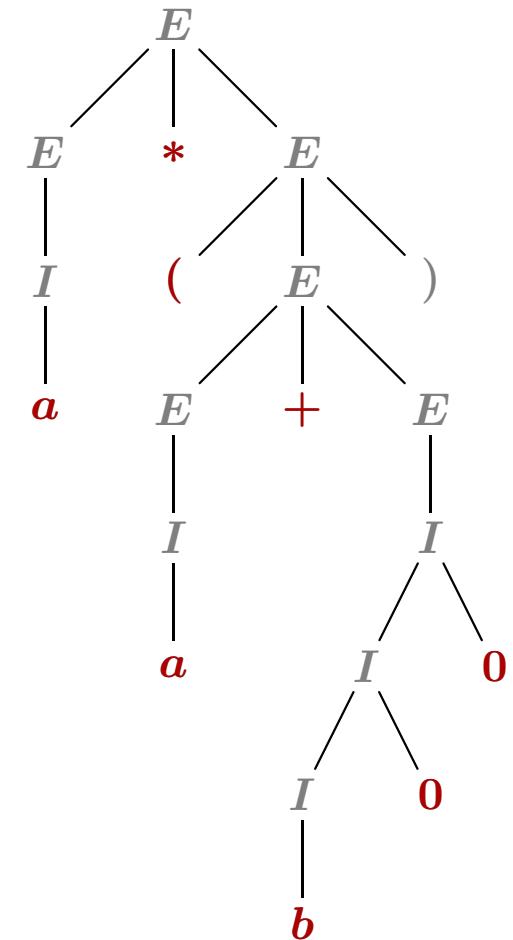
- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b0)$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalworte

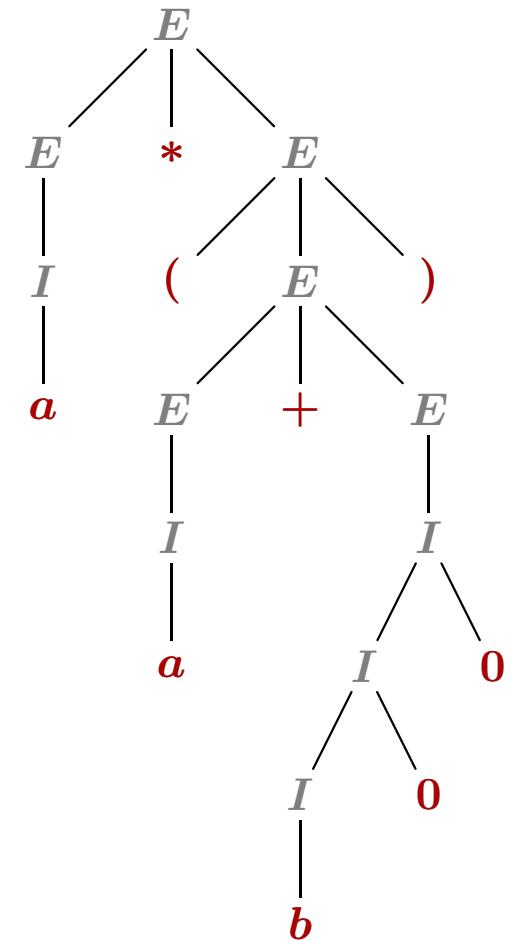
- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- Blätter repräsentieren Terminalwörter

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$



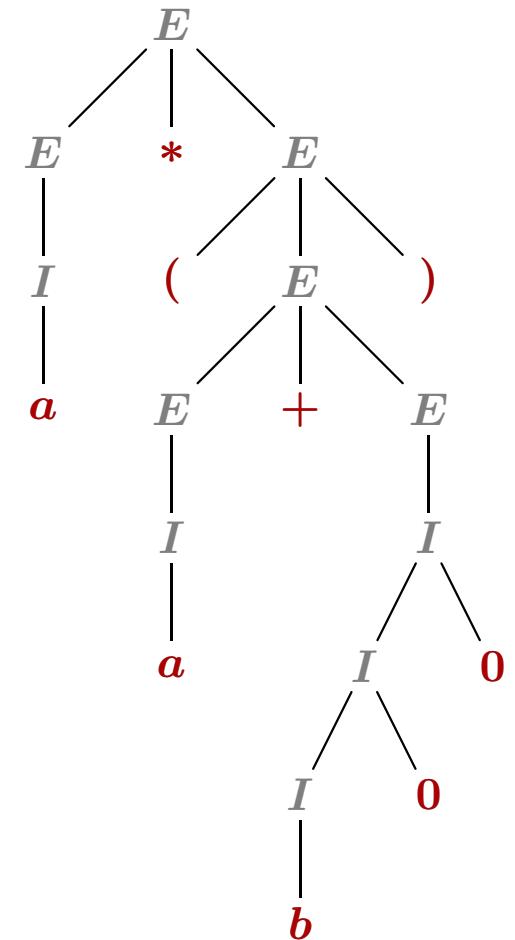
ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- **Blätter repräsentieren Terminalworte**

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- **Baum repräsentiert Ableitungen**

- Rekursive Erzeugung beginnend mit Wurzel



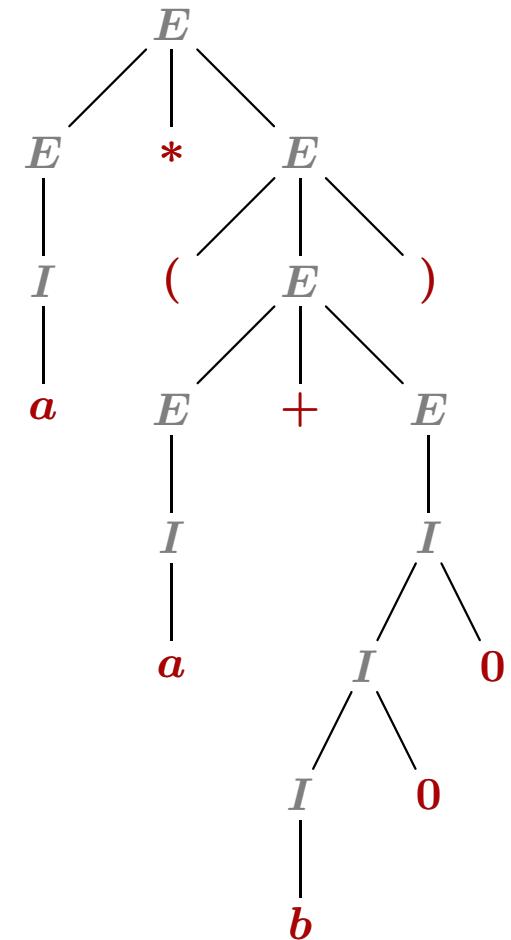
ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- **Blätter repräsentieren Terminalworte**

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- **Baum repräsentiert Ableitungen**

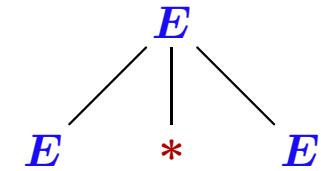
- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- **Blätter repräsentieren Terminalworte**

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$



- **Baum repräsentiert Ableitungen**

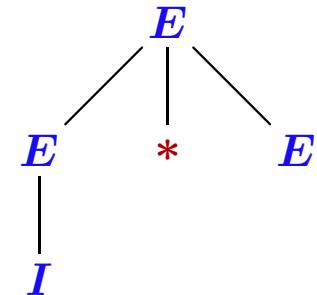
- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$E \xrightarrow{L} E * E$$

ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- **Blätter repräsentieren Terminalworte**

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$



- **Baum repräsentiert Ableitungen**

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$E \xrightarrow{L} E * E \xrightarrow{L} I * E$$

ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

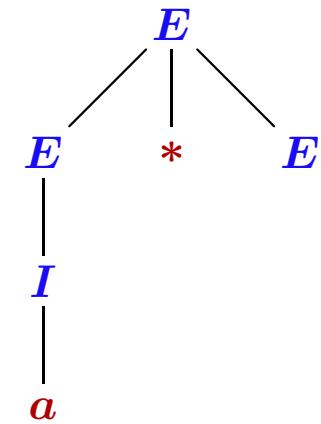
- **Blätter repräsentieren Terminalworte**

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- **Baum repräsentiert Ableitungen**

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$E \xrightarrow{L} E * E \xrightarrow{L} I * E \xrightarrow{L} a * E$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

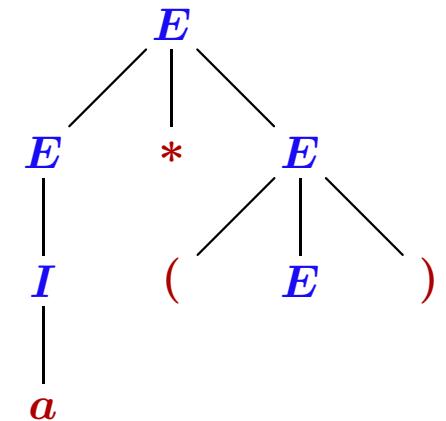
- **Blätter repräsentieren Terminalworte**

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- **Baum repräsentiert Ableitungen**

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$E \xrightarrow{L} E * E \xrightarrow{L} I * E \xrightarrow{L} a * E \xrightarrow{L} a * (E)$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

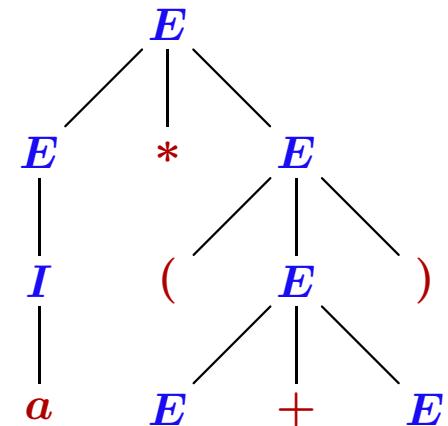
- Blätter repräsentieren Terminalworte

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- Baum repräsentiert Ableitungen

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \end{aligned}$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

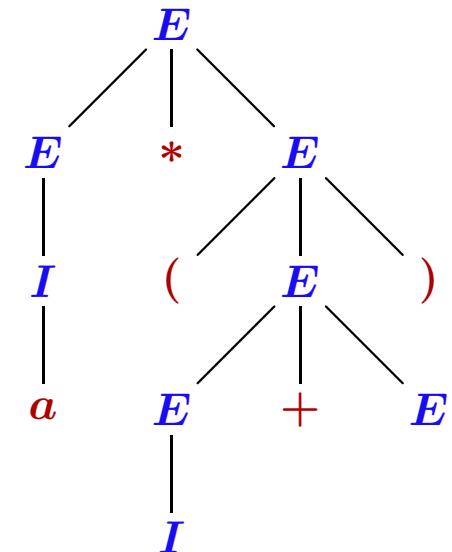
- Blätter repräsentieren Terminalworte

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- Baum repräsentiert Ableitungen

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \end{aligned}$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

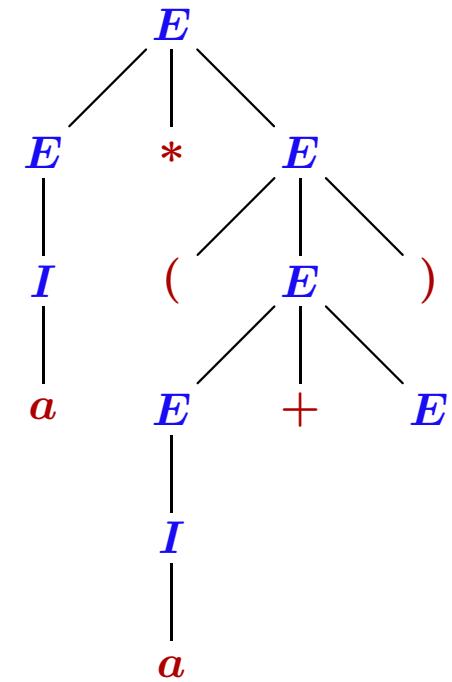
- **Blätter repräsentieren Terminalworte**

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- **Baum repräsentiert Ableitungen**

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \end{aligned}$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

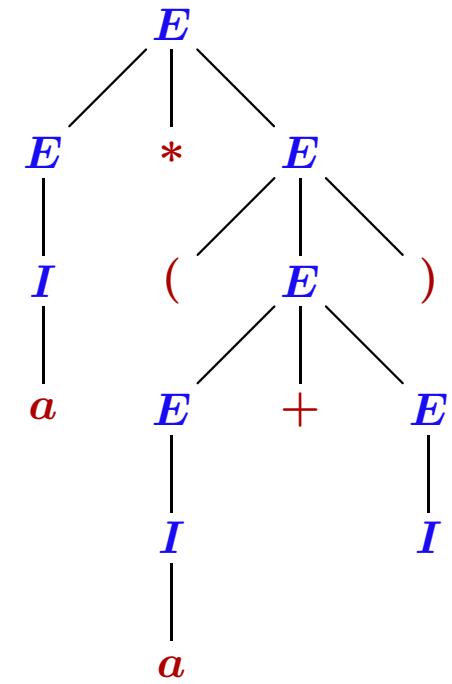
- Blätter repräsentieren Terminalworte

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- Baum repräsentiert Ableitungen

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \\ &\longrightarrow_L a * (a + I) \end{aligned}$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

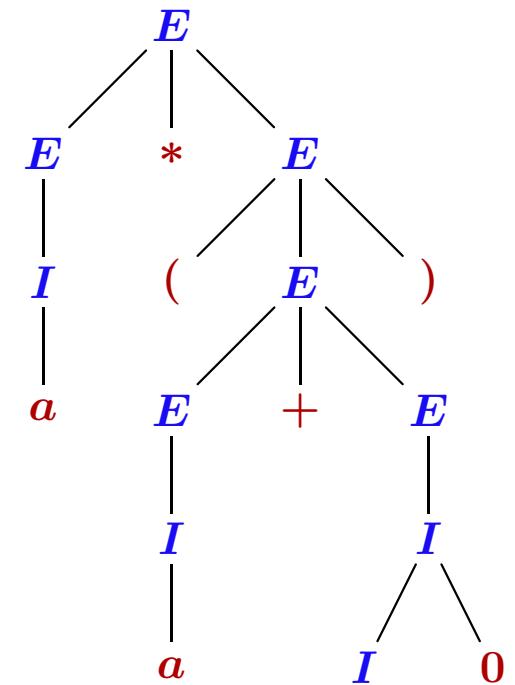
- Blätter repräsentieren Terminalworte

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- Baum repräsentiert Ableitungen

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \\ &\longrightarrow_L a * (a + I) \longrightarrow_L a * (a + I0) \end{aligned}$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

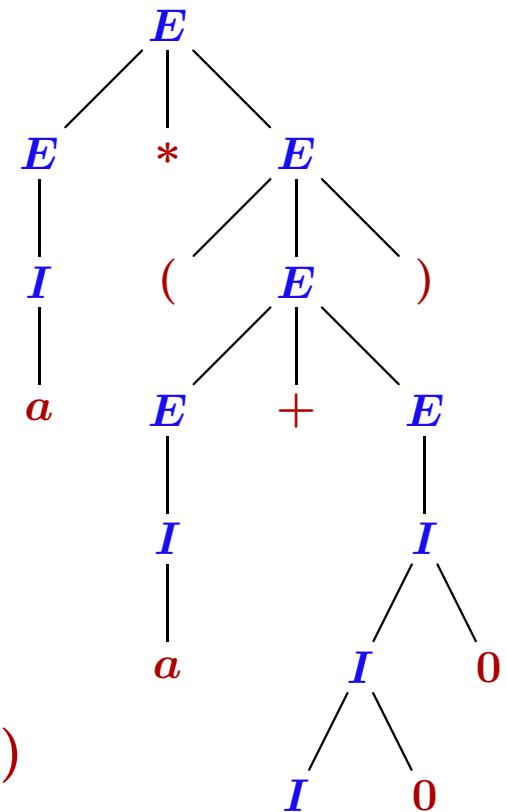
- Blätter repräsentieren Terminalworte

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- Baum repräsentiert Ableitungen

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \\ &\longrightarrow_L a * (a + I) \longrightarrow_L a * (a + I0) \longrightarrow_L a * (a + I00) \end{aligned}$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

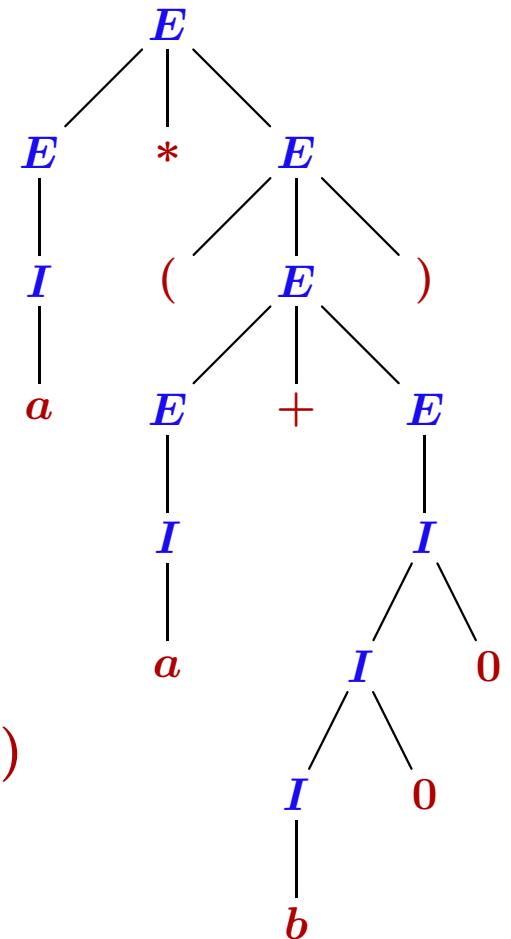
- Blätter repräsentieren Terminalworte

- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- Baum repräsentiert Ableitungen

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \\ &\longrightarrow_L a * (a + I) \longrightarrow_L a * (a + I0) \longrightarrow_L a * (a + I00) \\ &\longrightarrow_L a * (a + b00) \end{aligned}$$



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- **Blätter repräsentieren Terminalworte**

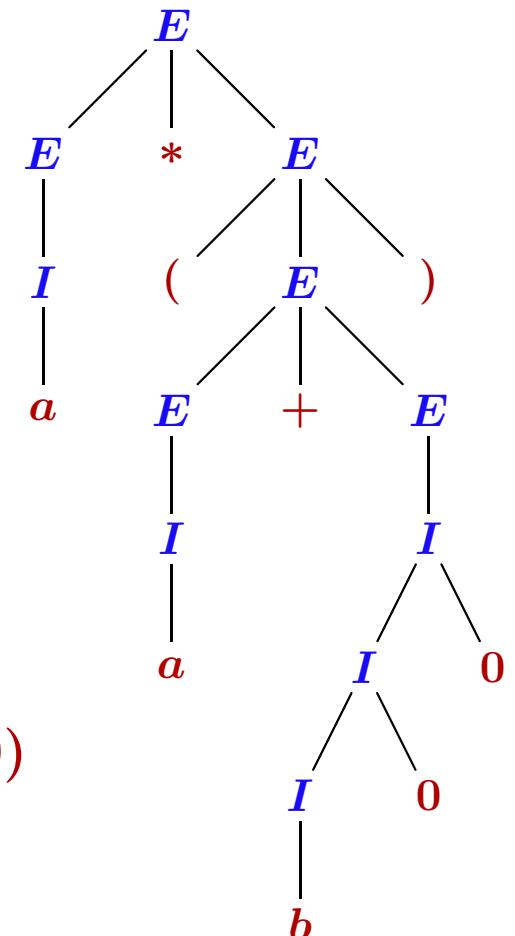
- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- **Baum repräsentiert Ableitungen**

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned} E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\ &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \\ &\longrightarrow_L a * (a + I) \longrightarrow_L a * (a + I0) \longrightarrow_L a * (a + I00) \\ &\longrightarrow_L a * (a + b00) \end{aligned}$$

- Vorrang für tiefe rechte Knoten ergibt Rechtssableitung



ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- **Blätter repräsentieren Terminalworte**

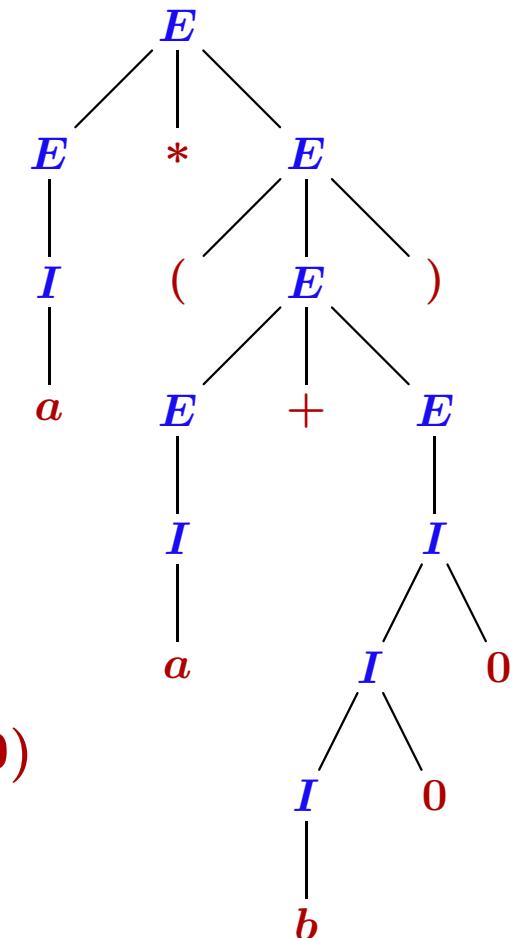
- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- **Baum repräsentiert Ableitungen**

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned}
 E &\longrightarrow_L E * E \longrightarrow_L I * E \longrightarrow_L a * E \longrightarrow_L a * (E) \\
 &\longrightarrow_L a * (E + E) \longrightarrow_L a * (I + E) \longrightarrow_L a * (a + E) \\
 &\longrightarrow_L a * (a + I) \longrightarrow_L a * (a + I0) \longrightarrow_L a * (a + I00) \\
 &\longrightarrow_L a * (a + b00)
 \end{aligned}$$

- Vorrang für tiefe rechte Knoten ergibt Rechtssableitung



$S \xrightarrow{*} w \Leftrightarrow \text{es gibt einen Ableitungsbäume mit Blattmarkierung } w$

ABLEITUNGSBÄUME REPRÄSENTIEREN ABLEITUNGEN

- **Blätter repräsentieren Terminalworte**

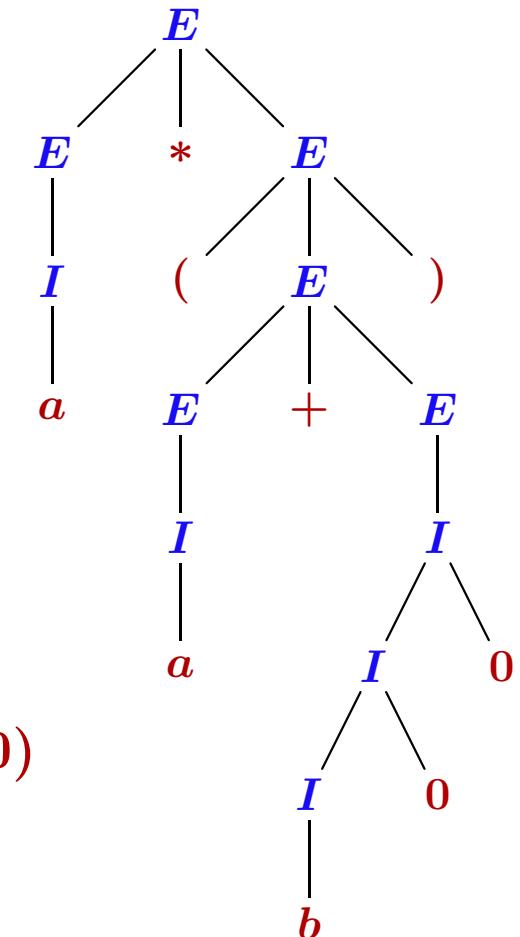
- Auslesen durch Tiefensuche von links nach rechts
- $a * (a + b00)$

- **Baum repräsentiert Ableitungen**

- Rekursive Erzeugung beginnend mit Wurzel
- Vorrang für tiefe linke Knoten ergibt Linksableitung

$$\begin{aligned}
 E &\xrightarrow{L} E * E \xrightarrow{L} I * E \xrightarrow{L} a * E \xrightarrow{L} a * (E) \\
 &\quad \xrightarrow{L} a * (E + E) \xrightarrow{L} a * (I + E) \xrightarrow{L} a * (a + E) \\
 &\quad \xrightarrow{L} a * (a + I) \xrightarrow{L} a * (a + I0) \xrightarrow{L} a * (a + I00) \\
 &\quad \xrightarrow{L} a * (a + b00)
 \end{aligned}$$

- Vorrang für tiefe rechte Knoten ergibt Rechtssableitung



$S \xrightarrow{*} w \Leftrightarrow$ es gibt einen Ableitungsbaum mit Blattmarkierung w

\Rightarrow : Konstruiere Baum induktiv aus Linksableitung von w

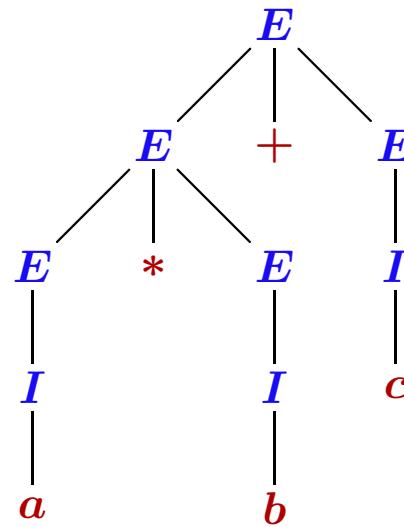
\Leftarrow : Extrahiere Linksableitung von w induktiv aus Baum

HMU §5.2

WANN IST DER ABLEITUNGSBAUM EINDEUTIG?

- $a * b + c$ hat zwei Ableitungen in G_6

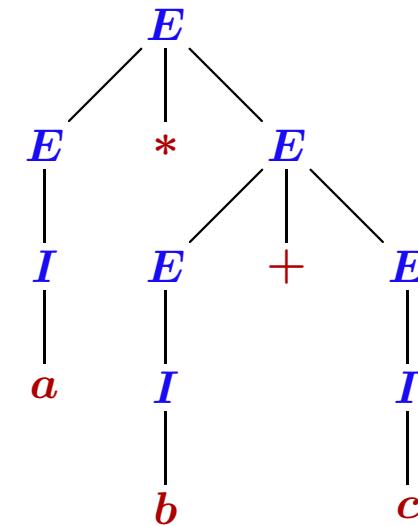
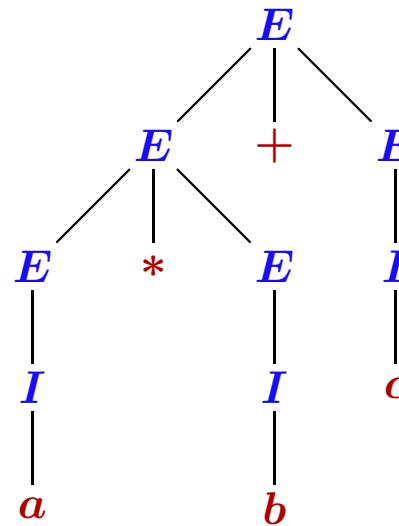
WANN IST DER ABLEITUNGSBAUM EINDEUTIG?



- $a * b + c$ hat zwei Ableitungen in G_6

$$\begin{aligned} E &\longrightarrow E+E \longrightarrow E*E+E \longrightarrow I*E+E \longrightarrow a*E+E \longrightarrow a*I+E \\ &\longrightarrow a*b+E \longrightarrow a*b+I \longrightarrow a*b+c \end{aligned}$$

WANN IST DER ABLEITUNGSBAUM EINDEUTIG?

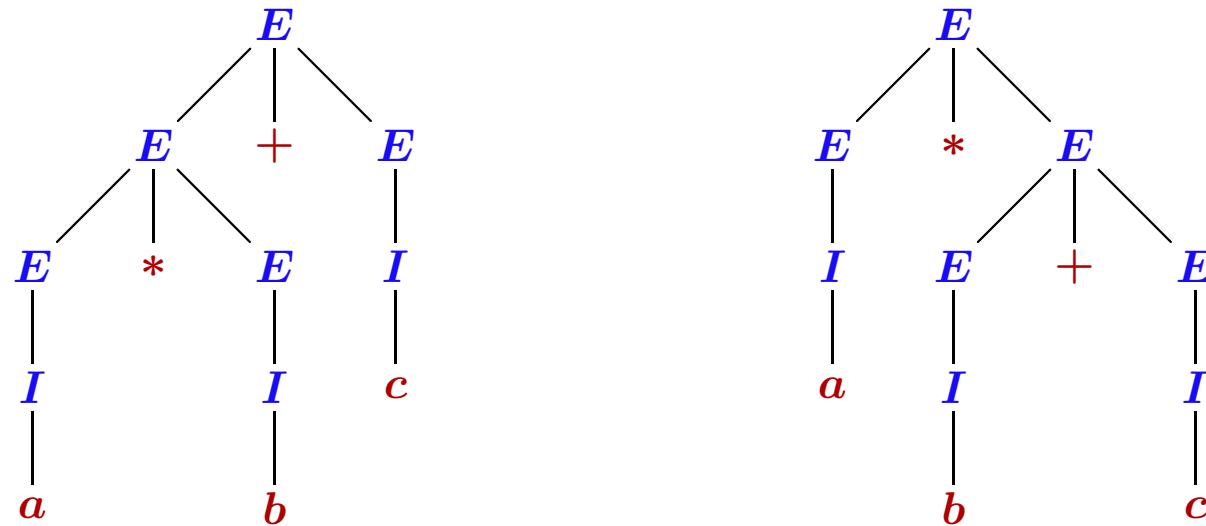


- $a * b + c$ hat zwei Ableitungen in G_6

$E \rightarrow E+E \rightarrow E*E+E \rightarrow I*E+E \rightarrow a*E+E \rightarrow a*I+E$
 $\rightarrow a*b+E \rightarrow a*b+I \rightarrow a*b+c$

$E \rightarrow E*E \rightarrow I*E \rightarrow a*E \rightarrow a*E+E \rightarrow a*I+E$
 $\rightarrow a*b+E \rightarrow a*b+I \rightarrow a*b+c$

WANN IST DER ABLEITUNGSBAUM EINDEUTIG?



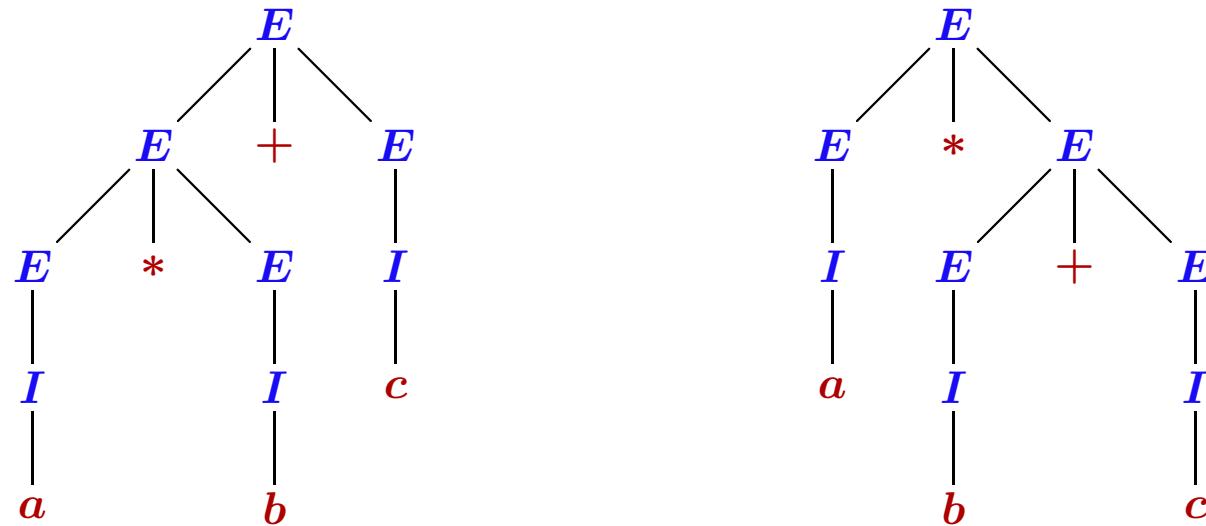
- $a * b + c$ hat zwei Ableitungen in G_6

$$\begin{aligned} E &\longrightarrow E+E \longrightarrow E*E+E \longrightarrow I*E+E \longrightarrow a*E+E \longrightarrow a*I+E \\ &\longrightarrow a*b+E \longrightarrow a*b+I \longrightarrow a*b+c \end{aligned}$$

$$\begin{aligned} E &\longrightarrow E*E \longrightarrow I*E \longrightarrow a*E \longrightarrow a*E+E \longrightarrow a*I+E \\ &\longrightarrow a*b+E \longrightarrow a*b+I \longrightarrow a*b+c \end{aligned}$$

Beide Ableitungen sind Linksableitungen

WANN IST DER ABLEITUNGSBAUM EINDEUTIG?



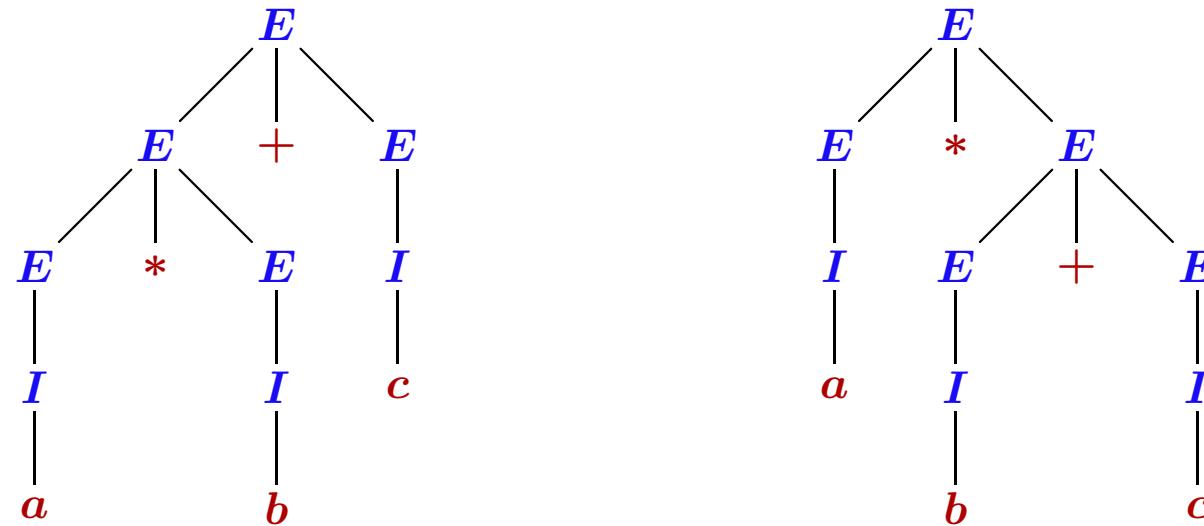
- $a * b + c$ hat zwei Ableitungen in G_6

$$\begin{aligned} E &\longrightarrow E+E \longrightarrow E*E+E \longrightarrow I*E+E \longrightarrow a*E+E \longrightarrow a*I+E \\ &\longrightarrow a*b+E \longrightarrow a*b+I \longrightarrow a*b+c \end{aligned}$$

$$\begin{aligned} E &\longrightarrow E*E \longrightarrow I*E \longrightarrow a*E \longrightarrow a*E+E \longrightarrow a*I+E \\ &\longrightarrow a*b+E \longrightarrow a*b+I \longrightarrow a*b+c \end{aligned}$$

Beide Ableitungen sind Linksableitungen

WANN IST DER ABLEITUNGSBAUM EINDEUTIG?



- $a * b + c$ hat zwei Ableitungen in G_6

$$\begin{aligned} E &\longrightarrow E+E \longrightarrow E*E+E \longrightarrow I*E+E \longrightarrow a*E+E \longrightarrow a*I+E \\ &\longrightarrow a*b+E \longrightarrow a*b+I \longrightarrow a*b+c \end{aligned}$$
$$\begin{aligned} E &\longrightarrow E*E \longrightarrow I*E \longrightarrow a*E \longrightarrow a*E+E \longrightarrow a*I+E \\ &\longrightarrow a*b+E \longrightarrow a*b+I \longrightarrow a*b+c \end{aligned}$$

Beide Ableitungen sind Linksableitungen

- Grammatik G_6 ist mehrdeutig

– Worte der Sprache können nicht eindeutig analysiert werden

MEHRDEUTIGKEIT

- Eindeutige Grammatik $G = (V, T, P, S)$
 - Jedes Wort $w \in L(G)$ hat genau einen Ableitungsbaum

MEHRDEUTIGKEIT

- Eindeutige Grammatik $G = (V, T, P, S)$
 - Jedes Wort $w \in L(G)$ hat genau einen Ableitungsbaum
 - Andernfalls ist G **mehrdeutig**
(ein $w \in L(G)$ hat mindestens zwei verschiedene Ableitungsbäume)

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

- Jedes Wort $w \in L(G)$ hat genau einen Ableitungsbaum
- Andernfalls ist G **mehrdeutig**
(ein $w \in L(G)$ hat mindestens zwei verschiedene Ableitungsbäume)
- G_6 ist mehrdeutig

- **Eindeutige Grammatik $G = (V, T, P, S)$**
 - Jedes Wort $w \in L(G)$ hat genau einen Ableitungsbaum
 - Andernfalls ist G **mehrdeutig**
(ein $w \in L(G)$ hat mindestens zwei verschiedene Ableitungsbäume)
 - G_6 ist mehrdeutig
- **Eindeutige Sprache L**
 - Es gibt eine eindeutige Grammatik G mit $L = L(G)$

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

- Jedes Wort $w \in L(G)$ hat genau einen Ableitungsbaum
- Andernfalls ist G **mehrdeutig**
(ein $w \in L(G)$ hat mindestens zwei verschiedene Ableitungsbäume)
- G_6 ist mehrdeutig

- **Eindeutige Sprache L**

- Es gibt eine eindeutige Grammatik G mit $L = L(G)$
- Andernfalls ist L **inhärent mehrdeutig**
(eine eindeutige Grammatik kann nicht angegeben werden)

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

- Jedes Wort $w \in L(G)$ hat genau einen Ableitungsbaum
- Andernfalls ist G **mehrdeutig**
(ein $w \in L(G)$ hat mindestens zwei verschiedene Ableitungsbäume)
- G_6 ist mehrdeutig

- **Eindeutige Sprache L**

- Es gibt eine eindeutige Grammatik G mit $L = L(G)$
- Andernfalls ist L **inhärent mehrdeutig**
(eine eindeutige Grammatik kann nicht angegeben werden)
- Die Sprache von G_6 ist eindeutig

- **Eindeutige Grammatik $G = (V, T, P, S)$**
 - Jedes Wort $w \in L(G)$ hat genau einen Ableitungsbaum
 - Andernfalls ist G **mehrdeutig**
(ein $w \in L(G)$ hat mindestens zwei verschiedene Ableitungsbäume)
 - G_6 ist mehrdeutig
- **Eindeutige Sprache L**
 - Es gibt eine eindeutige Grammatik G mit $L = L(G)$
 - Andernfalls ist L **inhärent mehrdeutig**
(eine eindeutige Grammatik kann nicht angegeben werden)
 - Die Sprache von G_6 ist eindeutig
 - $\{0^i 1^j 2^k \mid i=j \vee j=k\}$ ist inhärent mehrdeutig \mapsto Buch von Wegener

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

- Jedes Wort $w \in L(G)$ hat genau einen Ableitungsbaum
- Andernfalls ist G **mehrdeutig**
(ein $w \in L(G)$ hat mindestens zwei verschiedene Ableitungsbäume)
- G_6 ist mehrdeutig

- **Eindeutige Sprache L**

- Es gibt eine eindeutige Grammatik G mit $L = L(G)$
- Andernfalls ist L **inhärent mehrdeutig**
(eine eindeutige Grammatik kann nicht angegeben werden)
- Die Sprache von G_6 ist eindeutig
- $\{0^i 1^j 2^k \mid i=j \vee j=k\}$ ist inhärent mehrdeutig \mapsto Buch von Wegener

Programmiersprachen müssen eindeutig sein

AUFLÖSUNG VON MEHRDEUTIGKEITEN

- $G_6 = (\{E, I\}, \{a, b, 0, 1, +, *, (,)\}, P, E)$

mit $P = \{ E \rightarrow I \mid E+E \mid E*E \mid (E), I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$

– G_6 beinhaltet keine Konventionen für $*$ und $+$

AUFLÖSUNG VON MEHRDEUTIGKEITEN

- $G_6 = (\{E, I\}, \{a, b, 0, 1, +, *, (,)\}, P, E)$

mit $P = \{ E \rightarrow I \mid E+E \mid E*E \mid (E), I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$

- G_6 beinhaltet keine Konventionen für $*$ und $+$
 - $*$ bindet stärker als $+$
 - $*$ und $+$ werden als linkssassoziativ angesehen
 - Alle anderen Lesarten benötigen Klammern

AUFLÖSUNG VON MEHRDEUTIGKEITEN

- $G_6 = (\{E, I\}, \{a, b, 0, 1, +, *, (,)\}, P, E)$

mit $P = \{ E \rightarrow I \mid E+E \mid E*E \mid (E), I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$

- G_6 beinhaltet keine Konventionen für $*$ und $+$
 - $*$ bindet stärker als $+$
 - $*$ und $+$ werden als linkssassoziativ angesehen
 - Alle anderen Lesarten benötigen Klammern

- **Prioritätsregeln können Eindeutigkeit erzeugen**

- Niedrigste Priorität $+$ steht linkssassoziativ außen $\mapsto T$ erme
- Höhere Priorität $*$ steht linkssassoziativ innen $\mapsto F$ aktoren
- Faktoren können Bezeichner oder Ausdrücke in Klammern sein

AUFLÖSUNG VON MEHRDEUTIGKEITEN

- $G_6 = (\{E, I\}, \{a, b, 0, 1, +, *, (,)\}, P, E)$

mit $P = \{ E \rightarrow I \mid E+E \mid E*E \mid (E), I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$

- G_6 beinhaltet keine Konventionen für $*$ und $+$
 - $*$ bindet stärker als $+$
 - $*$ und $+$ werden als linkssassoziativ angesehen
 - Alle anderen Lesarten benötigen Klammern

- **Prioritätsregeln können Eindeutigkeit erzeugen**

- Niedrigste Priorität $+$ steht linkssassoziativ außen $\mapsto T$ erme
- Höhere Priorität $*$ steht linkssassoziativ innen $\mapsto F$ aktoren
- Faktoren können Bezeichner oder Ausdrücke in Klammern sein

$$G'_6 = (\{E, T, F, I\}, \{a, b, 0, 1, +, *, (,)\}, P', E)$$

mit $P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \}$
 $I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$

AUFLÖSUNG VON MEHRDEUTIGKEITEN

- $G_6 = (\{E, I\}, \{a, b, 0, 1, +, *, (,)\}, P, E)$

mit $P = \{ E \rightarrow I \mid E+E \mid E*E \mid (E), I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$

- G_6 beinhaltet keine Konventionen für $*$ und $+$
 - $*$ bindet stärker als $+$
 - $*$ und $+$ werden als linkssassoziativ angesehen
 - Alle anderen Lesarten benötigen Klammern

- **Prioritätsregeln können Eindeutigkeit erzeugen**

- Niedrigste Priorität $+$ steht linkssassoziativ außen $\mapsto T$ erme
- Höhere Priorität $*$ steht linkssassoziativ innen $\mapsto F$ aktoren
- Faktoren können Bezeichner oder Ausdrücke in Klammern sein

$$G'_6 = (\{E, T, F, I\}, \{a, b, 0, 1, +, *, (,)\}, P', E)$$

mit $P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \}$
 $I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$

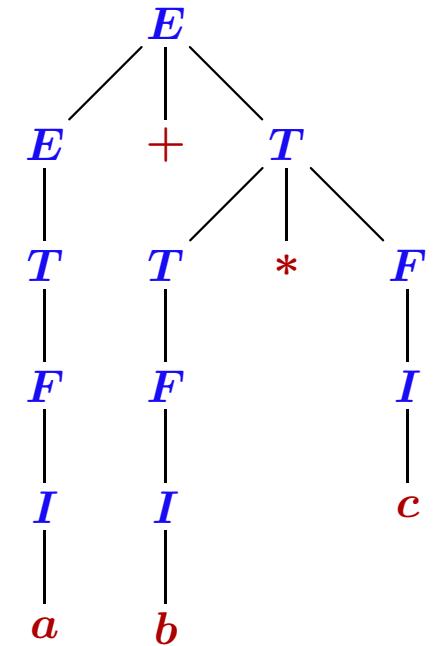
G'_6 ist äquivalent zu G_6 und eindeutig

EINDEUTIGKEIT VON G'_6

$$\begin{aligned} P' = \{ & E \rightarrow T \mid E+T, \quad T \rightarrow F \mid T*F, \quad F \rightarrow I \mid (E) \\ & I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \} \end{aligned}$$

EINDEUTIGKEIT VON G'_6

$$P' = \{ E \rightarrow T \mid E+T, \quad T \rightarrow F \mid T*T, \quad F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$$



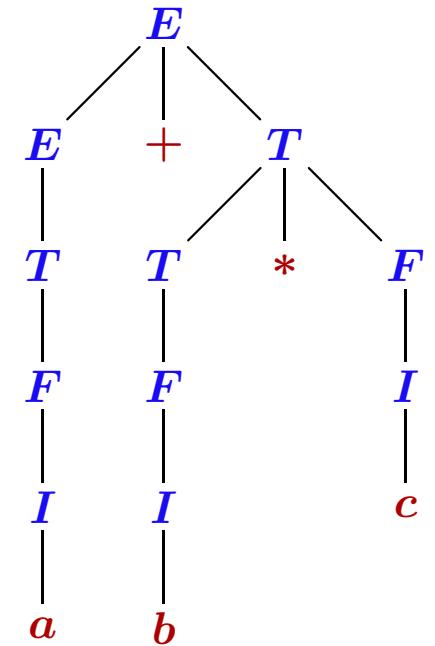
Einziger Ableitungsbaum
für $a * b + c$

EINDEUTIGKEIT VON G'_6

$$P' = \{ E \rightarrow T \mid E+T, \quad T \rightarrow F \mid T*T, \quad F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \}$$

- Jeder Term muß aus einer **Faktorenfolge** bestehen

- Faktorenfolge muß von rechts nach links erzeugt werden
- Faktoren sind nur Bezeichner oder Ausdrücke in Klammern
- Es gibt nur einen Parsebaum für $f_1 * f_2 * \dots * f_n$



Einziger Ableitungsbaum
für $a * b + c$

EINDEUTIGKEIT VON G'_6

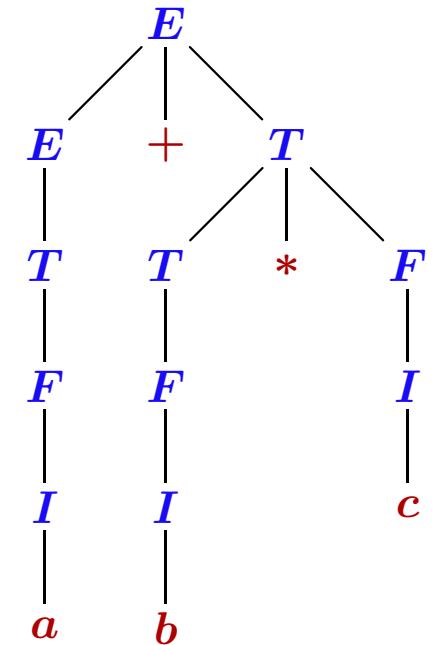
$$P' = \{ E \rightarrow T \mid E+T, \quad T \rightarrow F \mid T*T, \quad F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid Ia \mid Ib \mid Io \mid I1 \}$$

- Jeder Term muß aus einer **Faktorenfolge** bestehen

- Faktorenfolge muß von rechts nach links erzeugt werden
- Faktoren sind nur Bezeichner oder Ausdrücke in Klammern
- Es gibt nur einen Parsebaum für $f_1 * f_2 * \dots * f_n$

- Jeder Ausdruck muß aus einer **Termfolge** bestehen

- Termefolge muß von rechts nach links erzeugt werden
- Terme haben keine Ausdrücke als direkte Teile
- Es gibt nur einen Parsebaum für $t_1 + t_2 + \dots + t_k$



Einziger Ableitungsbaum
für $a * b + c$