

Einführung in die Theoretische Informatik

Christian Dalvit Manuel Eberl

Samuel Frontull **Cezary Kaliszyk** Daniel Ranalter

Wintersemester 2022/23

Zusammenfassung

Wintersemester 2022/23

Definition (Deterministischer endlicher Automat (kurz: DEA))

Ein **DEA** ist ein 5-Tupel $A = (Q, \Sigma, \delta, q_0, F)$ sodass

- 1 Q eine endliche Menge von **Zuständen**
- 2 Σ eine endliche Menge von **Eingabesymbole**
- 3 $\delta: Q \times \Sigma \rightarrow Q$ die **Übergangsfunktion**
- 4 $q_0 \in Q$ der **Startzustand**
- 5 $F \subseteq Q$ eine endliche Menge von **akzeptierenden Zuständen**

Zu beachten: δ muss für alle möglichen Argumente definiert sein

Satz

Sei A ein DEA, dann ist $L(A)$ regulär und umgekehrt existiert zu jeder regulären Sprache L ein DEA A , sodass $L = L(A)$

Einführung in die Logik

Syntax & Semantik der Aussagenlogik, Formales Beweisen, Konjunktive und Disjunktive Normalformen

Einführung in die Algebra

Algebraische Strukturen, Boolesche Algebra, Universelle Algebra

Einführung in die Theorie der Formalen Sprachen

Grammatiken und Formale Sprachen, Chomsky-Hierarchie, Reguläre Sprachen, **Kontextfreie Sprachen, Anwendungen von formalen Sprachen**

Einführung in die Berechenbarkeitstheorie und Komplexitätstheorie

Algorithmisch unlösbare Probleme, Turing Maschinen, Registermaschinen, Komplexitätstheorie

Einführung in die Programmverifikation

Prinzipien der Analyse von Programmen, Verifikation nach Hoare

Linksableitung und Rechtsableitung

Wintersemester 2022/23

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird

$\Rightarrow_l, \overset{*}{\Rightarrow}_l$

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

 $\Rightarrow_l, \overset{*}{\Rightarrow}_l$ $\Rightarrow_r, \overset{*}{\Rightarrow}_r$

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

 $\Rightarrow_{\ell}, \Rightarrow_{\ell}^*$ $\Rightarrow_r, \Rightarrow_r^*$

Beispiel

Wir betrachten KFG $G = (\{S\}, \{(,)\}, R, S)$, wobei R :

$$S \rightarrow \epsilon \mid (S) \mid SS$$

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

$$\Rightarrow_{\ell}, \Rightarrow_{\ell}^*$$

$$\Rightarrow_{\text{r}}, \Rightarrow_{\text{r}}^*$$

Beispiel

Wir betrachten KFG $G = (\{S\}, \{(,)\}, R, S)$, wobei R :

$$S \rightarrow \epsilon \mid (S) \mid SS$$

dann gilt $S \Rightarrow_{\text{r}} SS$

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

$$\Rightarrow_{\ell}, \Rightarrow_{\ell}^*$$

$$\Rightarrow_r, \Rightarrow_r^*$$

Beispiel

Wir betrachten KFG $G = (\{S\}, \{(,)\}, R, S)$, wobei R :

$$S \rightarrow \epsilon \mid (S) \mid SS$$

dann gilt $S \Rightarrow_r SS \Rightarrow_r S(S)$

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

$$\Rightarrow_{\ell}, \Rightarrow_{\ell}^*$$

$$\Rightarrow_{\text{r}}, \Rightarrow_{\text{r}}^*$$

Beispiel

Wir betrachten KFG $G = (\{S\}, \{(,)\}, R, S)$, wobei R :

$$S \rightarrow \epsilon \mid (S) \mid SS$$

dann gilt $S \Rightarrow_{\text{r}} SS \Rightarrow_{\text{r}} S(S) \Rightarrow_{\text{r}} S()$

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

$$\Rightarrow_{\ell}, \Rightarrow_{\ell}^*$$

$$\Rightarrow_{\text{r}}, \Rightarrow_{\text{r}}^*$$

Beispiel

Wir betrachten KFG $G = (\{S\}, \{(,)\}, R, S)$, wobei R :

$$S \rightarrow \epsilon \mid (S) \mid SS$$

dann gilt $S \Rightarrow_{\text{r}} SS \Rightarrow_{\text{r}} S(S) \Rightarrow_{\text{r}} S() \Rightarrow_{\text{r}} (S)()$

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

 $\Rightarrow_l, \Rightarrow_l^*$ $\Rightarrow_r, \Rightarrow_r^*$

Beispiel

Wir betrachten KFG $G = (\{S\}, \{(,)\}, R, S)$, wobei R :

$$S \rightarrow \epsilon \mid (S) \mid SS$$

dann gilt $S \Rightarrow_r SS \Rightarrow_r S(S) \Rightarrow_r S() \Rightarrow_r (S)() \Rightarrow_r ()()$

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

$$\Rightarrow_l, \overset{*}{\Rightarrow}_l$$

$$\Rightarrow_r, \overset{*}{\Rightarrow}_r$$

Beispiel

Wir betrachten KFG $G = (\{S\}, \{(,)\}, R, S)$, wobei R :

$$S \rightarrow \epsilon \mid (S) \mid SS$$

dann gilt $S \underset{r}{\Rightarrow} SS \underset{r}{\Rightarrow} S(S) \underset{r}{\Rightarrow} S() \underset{r}{\Rightarrow} (S)() \underset{r}{\Rightarrow} ()()$

Definition (Eindeutigkeit einer Grammatik)

KFG G heißt **eindeutig**, wenn jedes Wort $x \in L(G)$ genau eine Linksableitung besitzt, ansonsten **mehrdeutig**

Definition

- Eine **Linksableitung** ist eine Ableitung sodass immer die am weitesten links stehende Variable ersetzt wird
- In einer **Rechtsableitung** wird immer die am weitesten rechts stehende Variable ersetzt

 $\Rightarrow_l, \overset{*}{\Rightarrow}_l$ $\Rightarrow_r, \overset{*}{\Rightarrow}_r$

Beispiel

Wir betrachten KFG $G = (\{S\}, \{(\, , \,)\}, R, S)$, wobei R :

$$S \rightarrow \epsilon \mid (S) \mid SS$$

dann gilt $S \underset{r}{\Rightarrow} SS \underset{r}{\Rightarrow} S(S) \underset{r}{\Rightarrow} S() \underset{r}{\Rightarrow} (S)() \underset{r}{\Rightarrow} ()()$

Definition (Eindeutigkeit einer Grammatik)

KFG G heißt **eindeutig**, wenn jedes Wort $x \in L(G)$ genau eine **Linksableitung** besitzt, ansonsten **mehrdeutig**

Rekursive Inferenz

Wintersemester 2022/23

Definition

Sei G eine KFG und sei $A \rightarrow x$ eine Regel in G , sodass $x \in (V \cup \Sigma)^*$.

Definition

Sei G eine KFG und sei $A \rightarrow x$ eine Regel in G , sodass $x \in (V \cup \Sigma)^*$. Die **Sprache von A** , bezeichnet als $L(A)$, kann mittels **rekursiver Inferenz** definiert werden:

Definition

Sei G eine KFG und sei $A \rightarrow x$ eine Regel in G , sodass $x \in (V \cup \Sigma)^*$. Die **Sprache von A** , bezeichnet als $L(A)$, kann mittels **rekursiver Inferenz** definiert werden:

- 1 Wenn $x \in \Sigma^*$, dann $x \in L(A)$

Definition

Sei G eine KFG und sei $A \rightarrow x$ eine Regel in G , sodass $x \in (V \cup \Sigma)^*$. Die **Sprache von A** , bezeichnet als $L(A)$, kann mittels **rekursiver Inferenz** definiert werden:

- 1 Wenn $x \in \Sigma^*$, dann $x \in L(A)$
- 2 Andererseits, sei $x = X_1 \cdots X_n$, wobei $X_i \in V \cup \Sigma$

Gelte außerdem $x_i \in L(X_i)$ oder $x_i = X_i \in \Sigma$

Dann gilt $x_1 x_2 \cdots x_n \in L(A)$

Definition

Sei G eine KFG und sei $A \rightarrow x$ eine Regel in G , sodass $x \in (V \cup \Sigma)^*$. Die **Sprache von A** , bezeichnet als $L(A)$, kann mittels **rekursiver Inferenz** definiert werden:

- 1 Wenn $x \in \Sigma^*$, dann $x \in L(A)$
- 2 Andererseits, sei $x = X_1 \cdots X_n$, wobei $X_i \in V \cup \Sigma$

Gelte außerdem $x_i \in L(X_i)$ oder $x_i = X_i \in \Sigma$

Dann gilt $x_1 x_2 \cdots x_n \in L(A)$

Bemerkung

- Die rekursive Inferenz entspricht dem **bottom-up parsing** eines Compilers: zuerst wird der Rumpf einer Regel gelesen, dann wird das Ergebnis der Variable im Kopf übergeben
- Der Parsergenerator yacc (siehe später) generiert einen bottom-up parser

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

und zeigen $(a+b10) \in L(E)$:

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

und zeigen $(a+b10) \in L(E)$:

	Wort x	Variable	Regel	Rekursion
1	a	I	$I \rightarrow a$	

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

und zeigen $(a+b10) \in L(E)$:

	Wort x	Variable	Regel	Rekursion
1	a	I	$I \rightarrow a$	
2	b	I	$I \rightarrow b$	

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

und zeigen $(a+b10) \in L(E)$:

	Wort x	Variable	Regel	Rekursion
1	a	I	$I \rightarrow a$	
2	b	I	$I \rightarrow b$	
3	b1	I	$I \rightarrow I1$	2

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

und zeigen $(a+b10) \in L(E)$:

	Wort x	Variable	Regel	Rekursion
1	a	I	$I \rightarrow a$	
2	b	I	$I \rightarrow b$	
3	b1	I	$I \rightarrow I1$	2
4	b10	I	$I \rightarrow I0$	3

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

und zeigen $(a+b10) \in L(E)$:

	Wort x	Variable	Regel	Rekursion
1	a	I	$I \rightarrow a$	
2	b	I	$I \rightarrow b$	
3	b1	I	$I \rightarrow I1$	2
4	b10	I	$I \rightarrow I0$	3
5	a	E	$E \rightarrow I$	1
6	b10	E	$E \rightarrow I$	4

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

und zeigen $(a+b10) \in L(E)$:

	Wort x	Variable	Regel	Rekursion
1	a	I	$I \rightarrow a$	
2	b	I	$I \rightarrow b$	
3	b1	I	$I \rightarrow I1$	2
4	b10	I	$I \rightarrow I0$	3
5	a	E	$E \rightarrow I$	1
6	b10	E	$E \rightarrow I$	4
7	a+b10	E	$E \rightarrow E+E$	5,6

Beispiel

Wir betrachten die KFG $G_1 = (\{E, I\}, \{+, \cdot, (,), a, b, 0, 1\}, R, E)$, wobei R wie folgt:

$$\begin{array}{llll} E \rightarrow I & E \rightarrow E \cdot E & I \rightarrow a & I \rightarrow I0 \\ E \rightarrow E+E & E \rightarrow (E) & I \rightarrow Ib & I \rightarrow I1 \\ & I \rightarrow Ia & I \rightarrow b & \end{array}$$

und zeigen $(a+b10) \in L(E)$:

	Wort x	Variable	Regel	Rekursion
1	a	I	$I \rightarrow a$	
2	b	I	$I \rightarrow b$	
3	b1	I	$I \rightarrow I1$	2
4	b10	I	$I \rightarrow I0$	3
5	a	E	$E \rightarrow I$	1
6	b10	E	$E \rightarrow I$	4
7	a+b10	E	$E \rightarrow E+E$	5, 6
8	$(a+b10)$	E	$E \rightarrow (E)$	7

Syntaxbaum

Wintersemester 2022/23

Wiederholung: Bäume

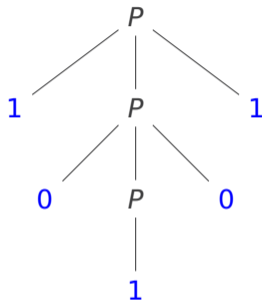


Das ist ein Baum

Wiederholung: Bäume



Das ist ein Baum



Und so sehen Bäume jetzt aus

Wiederholung: Bäume



Das ist ein Baum

- Ein einzelner Knoten ist ein **Baum**; dieser Knoten ist die **Wurzel**.
- Wenn T_1, T_2, \dots, T_k Bäume sind, bilden wir einen neuen **Baum** wie folgt:
 - 1 Man beginnt mit einem neuen Knoten N , der die Wurzel des Baumes darstellt.
 - 2 Schließlich fügt man k Kanten von N zu den Wurzeln der T_i hinzu.

Und so sehen Bäume jetzt aus

Definition (Syntaxbaum)

Ein **Syntaxbaum** für KFG $G = (V, \Sigma, R, S)$ ist ein Baum B mit:

Definition (Syntaxbaum)

Ein **Syntaxbaum** für KFG $G = (V, \Sigma, R, S)$ ist ein Baum B mit:

- 1 Jeder innere Knoten von B ist eine Variable in V

Definition (Syntaxbaum)

Ein **Syntaxbaum** für KFG $G = (V, \Sigma, R, S)$ ist ein Baum B mit:

- 1 Jeder innere Knoten von B ist eine Variable in V
- 2 Jedes Blatt in B ist entweder:
 - ein Terminal aus Σ
 - ein Nichtterminal aus V , oder
 - ϵ

Im letzten Fall ist das Blatt das einzige Kind seines Vorgängers

Definition (Syntaxbaum)

Ein **Syntaxbaum** für KFG $G = (V, \Sigma, R, S)$ ist ein Baum B mit:

- 1 Jeder innere Knoten von B ist eine Variable in V
- 2 Jedes Blatt in B ist entweder:
 - ein Terminal aus Σ
 - ein Nichtterminal aus V , oder
 - ϵ

Im letzten Fall ist das Blatt das einzige Kind seines Vorgängers

- 3 Sei A ein innerer Knoten, X_1, \dots, X_n seine Kinder ($X_i \in V \cup \Sigma$); dann gilt:

$$A \rightarrow X_1 \cdots X_n \in R$$

Definition (Syntaxbaum)

Ein **Syntaxbaum** für KFG $G = (V, \Sigma, R, S)$ ist ein Baum B mit:

- 1 Jeder innere Knoten von B ist eine Variable in V
- 2 Jedes Blatt in B ist entweder:
 - ein Terminal aus Σ
 - ein Nichtterminal aus V , oder
 - ϵ

Im letzten Fall ist das Blatt das einzige Kind seines Vorgängers

- 3 Sei A ein innerer Knoten, X_1, \dots, X_n seine Kinder ($X_i \in V \cup \Sigma$); dann gilt:

$$A \rightarrow X_1 \cdots X_n \in R$$

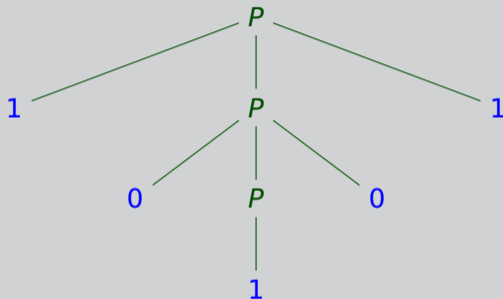
Das **Ergebnis** eines Syntaxbaums B für G ist das Wort über $(V \cup \Sigma)^*$, das wir erhalten, wenn wir die Blätter in B von links nach rechts lesen

Beispiel

Wir betrachten die KFG $G = (\{P\}, \Sigma, R, P)$ mit den Regeln $R: P \rightarrow \epsilon \mid 0 \mid 1 \mid 0P0 \mid 1P1$

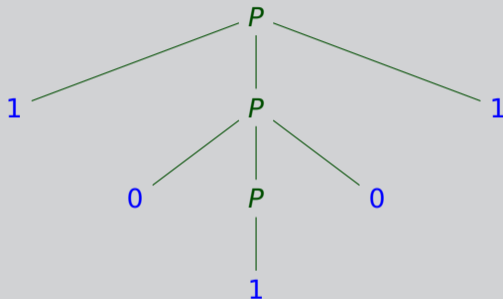
Beispiel

Wir betrachten die KFG $G = (\{P\}, \Sigma, R, P)$ mit den Regeln $R: P \rightarrow \epsilon \mid 0 \mid 1 \mid 0P0 \mid 1P1$
Dann ist der folgende Baum ein Syntaxbaum für G :



Beispiel

Wir betrachten die KFG $G = (\{P\}, \Sigma, R, P)$ mit den Regeln $R: P \rightarrow \epsilon \mid 0 \mid 1 \mid 0P0 \mid 1P1$
Dann ist der folgende Baum ein Syntaxbaum für G :



Bemerkung

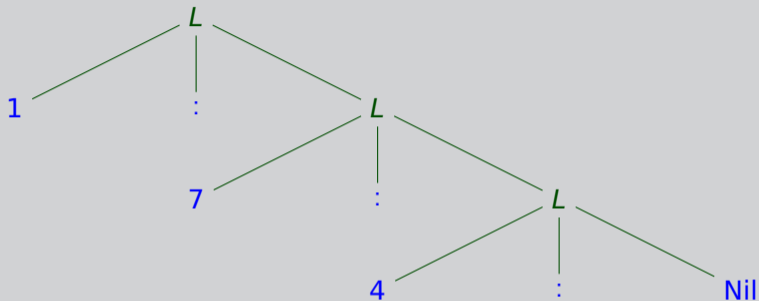
Vergleiche AST zum Parsen von Haskell-Ausdrücken in Funktionaler Programmierung

Beispiel

Sei $G = (\{L\}, \{0, 1, \dots, 9, Nil, :\}) R, P$ mit den Regeln $R: L \rightarrow Nil \mid 0 : L \mid 1 : L \mid \dots \mid 9 : L$

Beispiel

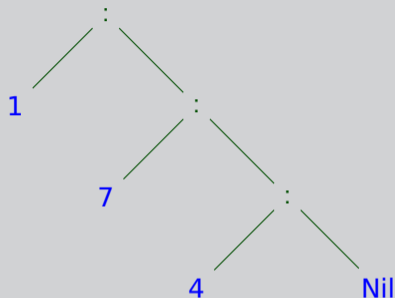
Sei $G = (\{L\}, \{0, 1, \dots, 9, \text{Nil}, :\}) R, P$ mit den Regeln $R: L \rightarrow \text{Nil} \mid 0 : L \mid 1 : L \mid \dots \mid 9 : L$



Das entspricht der Liste $1 : 7 : 4 : \text{Nil}$

Beispiel

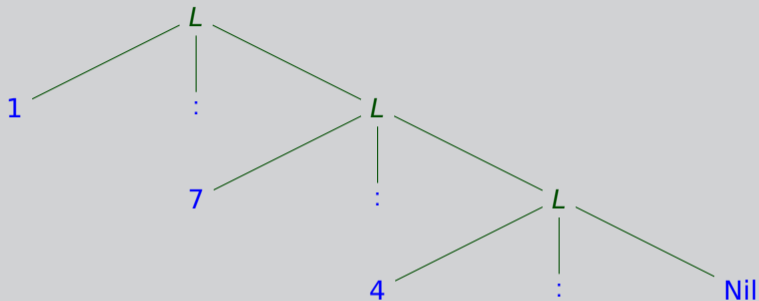
Sei $G = (\{L\}, \{0, 1, \dots, 9, \text{Nil}, :\}) R, P$ mit den Regeln $R: L \rightarrow \text{Nil} \mid 0 : L \mid 1 : L \mid \dots \mid 9 : L$



Das entspricht der Liste $1 : 7 : 4 : \text{Nil}$

Beispiel

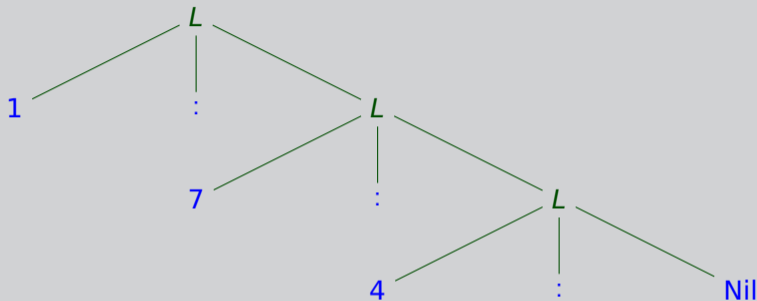
Sei $G = (\{L\}, \{0, 1, \dots, 9, \text{Nil}, :\}) R, P$ mit den Regeln $R: L \rightarrow \text{Nil} \mid 0 : L \mid 1 : L \mid \dots \mid 9 : L$



Das entspricht der Liste $1 : 7 : 4 : \text{Nil}$

Beispiel

Sei $G = (\{L\}, \{0, 1, \dots, 9, \text{Nil}, :\}) R, P$ mit den Regeln $R: L \rightarrow \text{Nil} \mid 0 : L \mid 1 : L \mid \dots \mid 9 : L$



Das entspricht der Liste $1 : 7 : 4 : \text{Nil}$

Bemerkung

Eigenschaften von Syntaxbäumen können mit Induktion über die **Höhe des Baumes** gezeigt werden.

Satz

Sei Σ ein Alphabet und sei $x \in \Sigma^*$. Die folgenden Beschreibungen kontextfreier Sprachen sind **äquivalent**:

1 $x \in L(A)$ nach dem rekursiven Inferenzverfahren

2 $A \xrightarrow{*} x$

3 $A \xrightarrow[\ell]{*} x$

4 $A \xrightarrow[r]{*} x$

5 Es existiert ein Syntaxbaum mit Wurzel A und Ergebnis x

Satz

Sei Σ ein Alphabet und sei $x \in \Sigma^*$. Die folgenden Beschreibungen kontextfreier Sprachen sind **äquivalent**:

1 $x \in L(A)$ nach dem rekursiven Inferenzverfahren

2 $A \xrightarrow{*} x$

3 $A \xrightarrow[\ell]{*} x$

4 $A \xrightarrow[r]{*} x$

5 Es existiert ein Syntaxbaum mit Wurzel A und Ergebnis x

$x \in L(A)$

\exists Syntaxbaum

$A \xrightarrow{*} x$

$A \xrightarrow[\ell]{*} x$

Satz

Sei Σ ein Alphabet und sei $x \in \Sigma^*$. Die folgenden Beschreibungen kontextfreier Sprachen sind **äquivalent**:

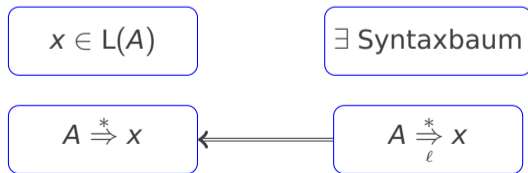
1 $x \in L(A)$ nach dem rekursiven Inferenzverfahren

2 $A \xrightarrow{*} x$

3 $A \xrightarrow[\ell]{*} x$

4 $A \xrightarrow[r]{*} x$

5 Es existiert ein Syntaxbaum mit Wurzel A und Ergebnis x



Satz

Sei Σ ein Alphabet und sei $x \in \Sigma^*$. Die folgenden Beschreibungen kontextfreier Sprachen sind **äquivalent**:

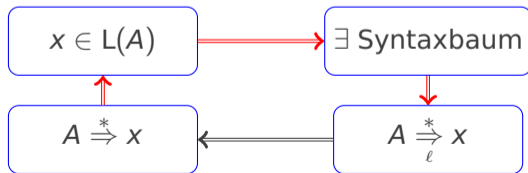
1 $x \in L(A)$ nach dem rekursiven Inferenzverfahren

2 $A \xrightarrow{*} x$

3 $A \xrightarrow[\ell]{*} x$

4 $A \xrightarrow[r]{*} x$

5 Es existiert ein Syntaxbaum mit Wurzel A und Ergebnis x



Satz

Angenommen $x \in L(A)$ nach dem rekursiven Inferenzverfahren, dann gibt es einen Syntaxbaum B mit Wurzel A und Ergebnis x

Satz

Angenommen $x \in L(A)$ nach dem rekursiven Inferenzverfahren, dann gibt es einen Syntaxbaum B mit Wurzel A und Ergebnis x

Beweis.

Mit Induktion nach der Anzahl der Rekursionen im Inferenzverfahren

Satz

Angenommen $x \in L(A)$ nach dem rekursiven Inferenzverfahren, dann gibt es einen Syntaxbaum B mit Wurzel A und Ergebnis x

Beweis.

Mit Induktion nach der Anzahl der Rekursionen im Inferenzverfahren

- **Basis** $x \in L(A)$, $x = a_1 \dots a_k$ benutzt genau die Regel

$$A \rightarrow a_1 \dots a_k$$

Satz

Angenommen $x \in L(A)$ nach dem rekursiven Inferenzverfahren, dann gibt es einen Syntaxbaum B mit Wurzel A und Ergebnis x

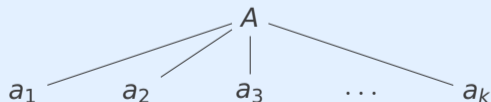
Beweis.

Mit Induktion nach der Anzahl der Rekursionen im Inferenzverfahren

- **Basis** $x \in L(A)$, $x = a_1 \dots a_k$ benutzt genau die Regel

$$A \rightarrow a_1 \dots a_k$$

Wir konstruieren einen Syntaxbaum B mit Wurzel A wie folgt



Satz

Angenommen $x \in L(A)$ nach dem rekursiven Inferenzverfahren, dann gibt es einen Syntaxbaum B mit Wurzel A und Ergebnis x

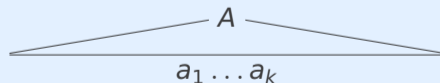
Beweis.

Mit Induktion nach der Anzahl der Rekursionen im Inferenzverfahren

- **Basis** $x \in L(A)$, $x = a_1 \dots a_k$ benutzt genau die Regel

$$A \rightarrow a_1 \dots a_k$$

Wir konstruieren einen Syntaxbaum B mit Wurzel A wie folgt



Beweis (Fortsetzung).

- **Schritt** $x \in L(A)$, $x = x_1 \dots x_n$ benutzt die Regel

$$A \rightarrow X_1 \cdots X_n$$

wobei $X_i \in V \cup \Sigma$ und $\forall i x_i \in L(X_i)$

Beweis (Fortsetzung).

- **Schritt** $x \in L(A)$, $x = x_1 \dots x_n$ benutzt die Regel

$$A \rightarrow X_1 \cdots X_n$$

wobei $X_i \in V \cup \Sigma$ und $\forall i x_i \in L(X_i)$

Nach IH \exists Syntaxbäume B_i gelten mit Wurzeln X_i und Ergebnis x_i

Beweis (Fortsetzung).

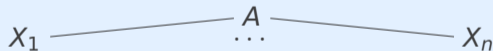
- **Schritt** $x \in L(A)$, $x = x_1 \dots x_n$ benutzt die Regel

$$A \rightarrow X_1 \cdots X_n$$

wobei $X_i \in V \cup \Sigma$ und $\forall i x_i \in L(X_i)$

Nach IH \exists Syntaxbäume B_i gelten mit Wurzeln X_i und Ergebnis x_i

Wir konstruieren einen neuen Syntaxbaum mit Wurzel A :



Beweis (Fortsetzung).

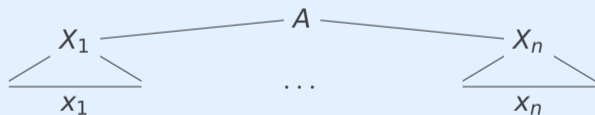
- **Schritt** $x \in L(A)$, $x = x_1 \dots x_n$ benutzt die Regel

$$A \rightarrow X_1 \cdots X_n$$

wobei $X_i \in V \cup \Sigma$ und $\forall i x_i \in L(X_i)$

Nach IH \exists Syntaxbäume B_i gelten mit Wurzeln X_i und Ergebnis x_i

Wir konstruieren einen neuen Syntaxbaum mit Wurzel A :



Beweis (Fortsetzung).

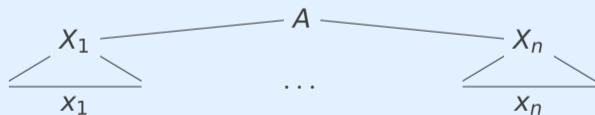
- **Schritt** $x \in L(A)$, $x = x_1 \dots x_n$ benutzt die Regel

$$A \rightarrow X_1 \cdots X_n$$

wobei $X_i \in V \cup \Sigma$ und $\forall i x_i \in L(X_i)$

Nach IH \exists Syntaxbäume B_i gelten mit Wurzeln X_i und Ergebnis x_i

Wir konstruieren einen neuen Syntaxbaum mit Wurzel A :



Satz

Sei B ein Syntaxbaum mit Wurzel A und Ergebnis $x \in \Sigma^$, dann gibt es eine Linksableitung (eine Rechtsableitung) von x aus A*

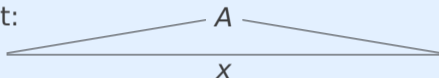
Satz

Sei B ein Syntaxbaum mit Wurzel A und Ergebnis $x \in \Sigma^*$, dann gibt es eine Linksableitung (eine Rechtsableitung) von x aus A

Beweis.

Mit Induktion nach der Höhe des Syntaxbaums B

- **Basis** Habe B die Gestalt:



dann existiert $A \rightarrow x \in R$, also $A \xRightarrow{\ell} x$

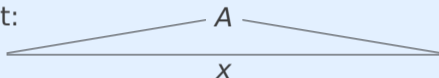
Satz

Sei B ein Syntaxbaum mit Wurzel A und Ergebnis $x \in \Sigma^*$, dann gibt es eine Linksableitung (eine Rechtsableitung) von x aus A

Beweis.

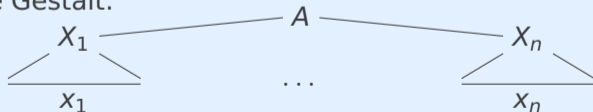
Mit Induktion nach der Höhe des Syntaxbaums B

- **Basis** Habe B die Gestalt:



dann existiert $A \rightarrow x \in R$, also $A \xRightarrow{\ell} x$

- **Schritt** Habe B die Gestalt:



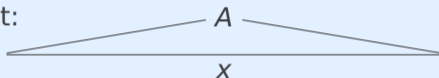
Satz

Sei B ein Syntaxbaum mit Wurzel A und Ergebnis $x \in \Sigma^*$, dann gibt es eine Linksableitung (eine Rechtsableitung) von x aus A

Beweis.

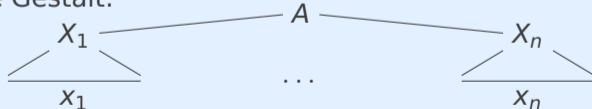
Mit Induktion nach der Höhe des Syntaxbaums B

- **Basis** Habe B die Gestalt:



dann existiert $A \rightarrow x \in R$, also $A \xRightarrow{\ell} x$

- **Schritt** Habe B die Gestalt:



dann $\exists A \rightarrow X_1 \cdots X_n \in R$, also:

$$A \xRightarrow{\ell} X_1 \cdots X_n \xRightarrow{\ell}^* x_1 x_2 \cdots x_n \xRightarrow{\ell}^* x_1 \cdots x_{n-1} x_n$$

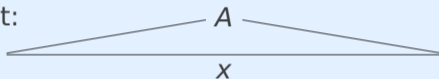
Satz

Sei B ein Syntaxbaum mit Wurzel A und Ergebnis $x \in \Sigma^*$, dann gibt es eine Linksableitung (eine Rechtsableitung) von x aus A

Beweis.

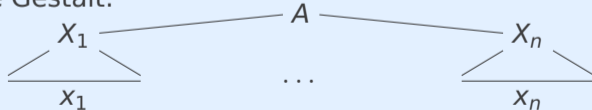
Mit Induktion nach der Höhe des Syntaxbaums B

- **Basis** Habe B die Gestalt:



dann existiert $A \rightarrow x \in R$, also $A \xRightarrow{\ell} x$

- **Schritt** Habe B die Gestalt:



dann $\exists A \rightarrow X_1 \cdots X_n \in R$, also:

$$A \xRightarrow{\ell} X_1 \cdots X_n \xRightarrow{\ell}^* x_1 x_2 \cdots x_n \xRightarrow{\ell}^* x_1 \cdots x_{n-1} x_n$$

Satz

Angenommen $A \stackrel{*}{\Rightarrow} x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Satz

Angenommen $A \xRightarrow{*} x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Beweis.

Mit Induktion nach der Länge ℓ der Ableitung $A \xRightarrow{*} x$:

Satz

Angenommen $A \Rightarrow^* x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Beweis.

Mit Induktion nach der Länge ℓ der Ableitung $A \Rightarrow^* x$:

- **Basis** Sei $\ell = 1$, dann gilt $x \in L(A)$

Satz

Angenommen $A \xRightarrow{*} x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Beweis.

Mit Induktion nach der Länge ℓ der Ableitung $A \xRightarrow{*} x$:

- **Basis** Sei $\ell = 1$, dann gilt $x \in L(A)$
- **Schritt** Angenommen $\ell = \ell' + 1$

Satz

Angenommen $A \Rightarrow^* x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Beweis.

Mit Induktion nach der Länge ℓ der Ableitung $A \Rightarrow^* x$:

- **Basis** Sei $\ell = 1$, dann gilt $x \in L(A)$
- **Schritt** Angenommen $\ell = \ell' + 1$

Zunächst können wir die Ableitung $A \Rightarrow^* x$ wie folgt schreiben:

$$A \Rightarrow X_1 X_2 \cdots X_n \Rightarrow^* x_1 x_2 \cdots x_n$$

Satz

Angenommen $A \Rightarrow^* x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Beweis.

Mit Induktion nach der Länge ℓ der Ableitung $A \Rightarrow^* x$:

- **Basis** Sei $\ell = 1$, dann gilt $x \in L(A)$
- **Schritt** Angenommen $\ell = \ell' + 1$

Zunächst können wir die Ableitung $A \Rightarrow^* x$ wie folgt schreiben:

$$A \Rightarrow X_1 X_2 \cdots X_n \Rightarrow^* x_1 x_2 \cdots x_n = x$$

Satz

Angenommen $A \Rightarrow^* x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Beweis.

Mit Induktion nach der Länge ℓ der Ableitung $A \Rightarrow^* x$:

- **Basis** Sei $\ell = 1$, dann gilt $x \in L(A)$
- **Schritt** Angenommen $\ell = \ell' + 1$

Zunächst können wir die Ableitung $A \Rightarrow^* x$ wie folgt schreiben:

$$A \Rightarrow X_1 X_2 \cdots X_n \Rightarrow^* x_1 x_2 \cdots x_n = x$$

Wir verwenden, dass wir die Ableitungen der Sätze x_i aufbrechen können,

Satz

Angenommen $A \xRightarrow{*} x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Beweis.

Mit Induktion nach der Länge ℓ der Ableitung $A \xRightarrow{*} x$:

- **Basis** Sei $\ell = 1$, dann gilt $x \in L(A)$
- **Schritt** Angenommen $\ell = \ell' + 1$

Zunächst können wir die Ableitung $A \xRightarrow{*} x$ wie folgt schreiben:

$$A \Rightarrow X_1 X_2 \cdots X_n \xRightarrow{*} x_1 x_2 \cdots x_n = x$$

Wir verwenden, dass wir die Ableitungen der Sätze x_i aufbrechen können, also gilt:

- Wenn $X_i \in \Sigma$, dann $X_i = x_i$
- Wenn $X_i \in V$, dann gilt $X_i \xRightarrow{*} x_i$ und somit $x_i \in L(X_i)$

Satz

Angenommen $A \xRightarrow{*} x$ mit $x \in \Sigma^*$, dann liefert das rekursive Inferenzverfahren, dass $x \in L(A)$

Beweis.

Mit Induktion nach der Länge ℓ der Ableitung $A \xRightarrow{*} x$:

- **Basis** Sei $\ell = 1$, dann gilt $x \in L(A)$
- **Schritt** Angenommen $\ell = \ell' + 1$

Zunächst können wir die Ableitung $A \xRightarrow{*} x$ wie folgt schreiben:

$$A \Rightarrow X_1 X_2 \cdots X_n \xRightarrow{*} x_1 x_2 \cdots x_n = x$$

Wir verwenden, dass wir die Ableitungen der Sätze x_i aufbrechen können, also gilt:

- Wenn $X_i \in \Sigma$, dann $X_i = x_i$
- Wenn $X_i \in V$, dann gilt $X_i \xRightarrow{*} x_i$ und somit $x_i \in L(X_i)$

Anwendung der Theorie der Kontextfreien Sprachen

Beispielanwendungen

- Parsergeneratoren, beziehungsweise im **Compilerbau**
 - 1 Parsergenerator verwandelt die Beschreibung einer Sprache in einen Parser für diese Sprache
 - 2 Parser werden zur syntaktischen Analyse von Programmen verwendet

Anwendung der Theorie der Kontextfreien Sprachen

Beispielanwendungen

- Parsergeneratoren, beziehungsweise im **Compilerbau**
 - 1 Parsergenerator verwandelt die Beschreibung einer Sprache in einen Parser für diese Sprache
 - 2 Parser werden zur syntaktischen Analyse von Programmen verwendet
- Anwendungen im Bereich der **Wissensrepräsentation**, etwa in XML-Dokumenten (siehe Skriptum)

Anwendung der Theorie der Kontextfreien Sprachen

Beispielanwendungen

- Parsergeneratoren, beziehungsweise im **Compilerbau**
 - 1 Parsergenerator verwandelt die Beschreibung einer Sprache in einen Parser für diese Sprache
 - 2 Parser werden zur syntaktischen Analyse von Programmen verwendet
- Anwendungen im Bereich der **Wissensrepräsentation**, etwa in XML-Dokumenten (siehe Skriptum)

lexikalische Analyse

- der Eingabestrom aus ASCII-Zeichen wird analysiert
- in terminale Symbole der formalen Sprache zerlegt, die man **Token** nennt

Demo: Lex & Yacc

Wintersemester 2022/23

Beispiel

KFG G_2 für arithmetische Ausdrücke

$$E \rightarrow T \mid +T \mid -T \mid E+T \mid E-T$$

$$T \rightarrow F \mid T \cdot F \mid T/F$$

$$F \rightarrow c \mid v \mid (E) .$$

Beispiel

KFG G_2 für arithmetische Ausdrücke

$$E \rightarrow T \mid +T \mid -T \mid E+T \mid E-T$$

$$T \rightarrow F \mid T \cdot F \mid T/F$$

$$F \rightarrow c \mid v \mid (E) .$$

in G_2 gilt etwa $-c * v \in L(E)$:

Beispiel

KFG G_2 für arithmetische Ausdrücke

$$E \rightarrow T \mid +T \mid -T \mid E+T \mid E-T$$

$$T \rightarrow F \mid T \cdot F \mid T/F$$

$$F \rightarrow c \mid v \mid (E) .$$

in G_2 gilt etwa $-c * v \in L(E)$:

	Wort x	Variable	Regel	Rekursion
1	c	F	$F \rightarrow c$	
2	v	F	$F \rightarrow v$	
3	c	T	$T \rightarrow F$	1
4	$c \cdot v$	T	$T \rightarrow T \cdot F$	3,2
5	$-c \cdot v$	E	$E \rightarrow -T$	4

Parsergenerierung in C mit yacc

Regelteil calc.y (gekürzt)

```
expression:      term          { $$ = $1; }
                | '+' term     { $$ = $2; }
                | '-' term     { $$ = -$2; }
                | expression '+' term { $$ = $1 + $3; }
                | expression '-' term { $$ = $1 - $3; }
                ;

term:            factor        { $$ = $1; }
                | term '*' factor { $$ = $1 * $3; }
                | term '/' factor { $$ = $1 / $3; }
                ;

factor:         NUMBER        { $$ = $1; }
                | '(' expression ')' { $$ = $2; }
                ;
```