Gliederung des Kapitels 1.1

1.1 Algorithmen und Sprachen

~~1.1.1 Darstellung von Algorithmen~~
~~1.1.2 Grundlegende Datenbereiche~~
~~1.1.3 Realisierte Abbildung~~
~~1.1.4 (Künstliche) Sprachen~~
1.1.5 Grammatiken
1.1.6 BNF, Syntaxdiagramme
1.1.7 Sprachen zur Beschreibung von Sprachen
1.1.8 Übungsaufgaben

Abschnitt 1.1.5 soll Ihnen folgende Inhalte vermitteln:

Um Sprachen, insbesondere Programmiersprachen beschreiben zu können, lernen Sie kontextfreie Grammatiken kennen, mit denen die zulässigen Wörter der Sprache aus einem Startsymbol abgeleitet werden. Sie werden vor allem mit dem Begriff der Ableitung vertraut gemacht.

Als Beispiel lernen Sie, wie man Bezeichner und einfache arithmetische Ausdrücke mit Grammatiken exakt beschreibt. Zugleich wird der Begriff der Eindeutigkeit herausgearbeitet.

Schließlich wird der allgemeine Grammatikbegriff vorgestellt. Am Ende des Abschnitts sollten Sie in der Lage sein, zu sehr einfachen Sprachen eine Grammatik anzugeben.

1.1.5 Grammatiken

Definition 1.1.5.1: Eine kontextfreie Grammatik ist ein Viertupel $G = (V, \Sigma, P, S)$ mit

(1) V ist eine nicht-leere endliche Menge (die Menge der Nichtterminalzeichen oder Variablen),

(2) Σ ist eine nicht-leere endliche Menge (die Menge der Terminalzeichen) mit $V \cap \Sigma = \emptyset$,

(3) $S \in V$ ist ein Nichtterminalzeichen (das Startsymbol),

(4) $P \subset V \times (V \cup \Sigma)^*$ ist eine endliche Menge (die Menge der Regeln oder Produktionen).

(Englisch: contextfree grammar.)

Beispiel: Betrachte die Grammatik $G_1 = (V_1, \Sigma_1, P_1, S_1)$ mit

$V_1 = \{S_1\}$, $\Sigma_1 = \{0, 1\}$, $P_1 = \{(S_1, 1), (S_1, S_1 0), (S_1, S_1 1)\}$.

Die Idee ist nun, ausgehend von dem Startsymbol S_1 schrittweise alle Nichtterminalzeichen (hier ist dies nur das Zeichen S_1) durch ihre rechten Seiten in P_1 zu ersetzen. Alle Wörter, die man auf diese Weise erhält und die kein Nichtterminalzeichen mehr enthalten, bilden die Sprache, die von dieser Grammatik erzeugt wird. Bei der Ersetzung ("Ableitung" oder "Herleitung") hat man Freiheiten, da man sich willkürlich für eine Regel entscheiden kann, sofern P_1 mehrere Regeln mit der gleichen rechten Seite besitzt.

Also: S_1 ersetzen durch 1 (fertig, denn das Wort 1 enthält kein Nichtterminalzeichen mehr).

Grammatik $G_1 = (V_1, \Sigma_1, P_1, S_1)$ mit
$V_1 = \{S_1\}$, $\Sigma_1 = \{0, 1\}$, $P_1 = \{(S_1, 1), (S_1, S_1 0), (S_1, S_1 1)\}$.

S_1 ersetzen durch 1 (fertig, denn das Wort 1 enthält kein Nichtterminalzeichen mehr). Andere Möglichkeiten:

S_1 ersetzen durch $S_1 0$, hierin S_1 wieder ersetzen durch $S_1 0$, so dass $S_1 00$ entsteht; hierin S_1 ersetzen durch 1; Ergebnis ist das Wort 100 (fertig, da 100 kein Nichtterminalzeichen enthält).

S_1 ersetzen durch $S_1 1$, hierin S_1 ersetzen durch $S_1 0$, so dass $S_1 01$ entsteht; hierin S_1 ersetzen durch $S_1 1$, wobei $S_1 101$ entsteht; hierin S_1 ersetzen durch 1; Ergebnis ist das Wort 1101 (fertig, da dieses Wort kein Nichtterminalzeichen mehr enthält).

Welche Wörter entstehen auf diese Weise aus S_1? ■

Um die Ersetzung der linken Seite A durch die rechte Seite w eines Paares $(A,w) \in P$ deutlicher hervor zu heben, schreibt man statt (A,w) im Allgemeinen $A \to w$.

$P_1 = \{(S_1, 1), (S_1, S_1 0), (S_1, S_1 1)\}$ schreibt man also folgendermaßen:

$$P_1 = \{S_1 \to 1,\ S_1 \to S_1 0,\ S_1 \to S_1 1\}.$$

Definition 1.1.5.2: Gegeben sei eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$. Die Regelmenge P definiert auf der Menge $(V \cup \Sigma)^*$ die "Ableitungsrelationen" $\Rightarrow$ und $\Rightarrow^*$:

(1) Es gilt $u \Rightarrow v$ genau dann, wenn man die Wörter u und v in der Form $u = xAy$, $v = xwy$ mit $x, y \in (V \cup \Sigma)^*$ und $(A,w) \in P$ schreiben kann. Man sagt: y ist aus x in einem Schritt herleitbar oder y lässt sich aus x in einem Schritt ableiten.

(2) Es gilt $u \Rightarrow^* v$ genau dann, wenn entweder $u = v$ ist oder wenn es Wörter $z_0, z_1, ..., z_k \in (V \cup \Sigma)^*$ für ein $k \geq 1$ gibt mit $u = z_0$, $v = z_k$, $z_i \Rightarrow z_{i+1}$ für alle $i = 0, 1, ..., k-1$. Man sagt dann, v ist aus u herleitbar oder ableitbar. (Die Zahl k heißt Ableitungslänge.)

Hinweis: $\Rightarrow^*$ ist der so genannte "reflexive und transitive Abschluss" von $\Rightarrow$. (Englisch: Ableitung = derivation.)

Definition 1.1.5.3:
Die von einer kontextfreien Grammatik $G = (V, \Sigma, P, S)$ erzeugte Sprache (engl.: generated language) ist die Menge $L(G) = \{ w \in \Sigma^* \mid S \Rightarrow^* w \} \subseteq \Sigma^*$.
Eine Sprache $L \subseteq \Sigma^*$ heißt kontextfreie Sprache, wenn es eine kontextfreie Grammatik G mit $L = L(G)$ gibt.

Die Nichtterminalzeichen dienen also nur zwischenzeitlich dem Ableitungsprozess; zur erzeugten Sprache zählen dagegen nur die Wörter, die ausschließlich aus Terminalzeichen bestehen.

Wir werden im Abschnitt 1.1.6 sehen, wie man Programmiersprachen mit Hilfe von kontextfreien Grammatiken erzeugt. Hier betrachten wir zunächst Bezeichner und Ausdrücke.

Beispiel 1.1.5.4: Betrachte erneut die Grammatik
$G_1 = (V_1, \Sigma_1, P_1, S_1)$ mit $V_1 = \{S_1\}$, $\Sigma_1 = \{0, 1\}$ und
$P_1 = \{S_1 \to 1,\ S_1 \to S_1 0,\ S_1 \to S_1 1\}$.

Aus dem Startsymbol S_1 kann man mit der zweiten und der dritten Regel $S_1 0$ und $S_1 1$ ableiten, hieraus mit den gleichen Regeln $S_1 00$, $S_1 10$, $S_1 01$ und $S_1 11$, hieraus mit den gleichen Regeln $S_1 000$, $S_1 100$, $S_1 010$, $S_1 110$, $S_1 001$, $S_1 101$, $S_1 011$ und $S_1 111$ usw., also alle Wörter der Form $S_1 x$ für ein beliebiges Wort $x \in \Sigma^*$. Um das Nichtterminalzeichen zu entfernen, muss irgendwann die erste Regel verwendet werden, welche hieraus das Wort $1x$ für ein beliebiges Wort $x \in \Sigma^*$ herleitet.

Da andere Wörter nicht herleitbar sind, gilt:
$L(G_1) = \{1x \mid x \in \Sigma^*\} = \{1\}\Sigma^*$. ■

Beispiel 1.1.5.5: Menge der Bezeichner, vgl. Definition 1.1.4.1. und Regeln 1.1.4.4.

Erinnerung: Seien $\Phi_Z = \{0,1,2,3,4,5,6,7,8,9\}$,
$\Phi_B = \{A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,$
$X,Y,Z,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z\}$,
$\Phi = \Phi_B \cup \Phi_Z \cup \{_\}$, dann ist die Menge der Bezeichner

$\underline{Bez} = \{w \in \Phi^* \mid w = bv,\ b \in \Phi_B \text{ und } w \in \Phi^*\} = \Phi_B \Phi^*$.

Definiere die kontextfreie Grammatik $G_2 = (V_2, \Sigma_2, P_2, S_2)$ mit $V_2 = \{S_2, S_2', S_2''\}$ und $\Sigma_2 = \Phi$, und P_2 bestehe aus folgenden 66 Regeln:

P_2 =

{ $S_2' \to A$, $S_2' \to B$, $S_2' \to C$, $S_2' \to D$, $S_2' \to E$, $S_2' \to F$, $S_2' \to G$,
$S_2' \to H$, $S_2' \to I$, $S_2' \to J$, $S_2' \to K$, $S_2' \to L$, $S_2' \to M$, $S_2' \to N$,
$S_2' \to O$, $S_2' \to P$, $S_2' \to Q$, $S_2' \to R$, $S_2' \to S$, $S_2' \to T$, $S_2' \to U$,
$S_2' \to V$, $S_2' \to W$, $S_2' \to X$, $S_2' \to Y$, $S_2' \to Z$, $S_2' \to a$, $S_2' \to b$,
$S_2' \to c$, $S_2' \to d$, $S_2' \to e$, $S_2' \to f$, $S_2' \to g$, $S_2' \to h$, $S_2' \to i$,
$S_2' \to j$, $S_2' \to k$, $S_2' \to l$, $S_2' \to m$, $S_2' \to n$, $S_2' \to o$, $S_2' \to p$,
$S_2' \to q$, $S_2' \to r$, $S_2' \to s$, $S_2' \to t$, $S_2' \to u$, $S_2' \to v$, $S_2' \to w$,
$S_2' \to x$, $S_2' \to y$, $S_2' \to z$,
$S_2'' \to 0$, $S_2'' \to 1$, $S_2'' \to 2$, $S_2'' \to 3$, $S_2'' \to 4$, $S_2'' \to 5$,
$S_2'' \to 6$, $S_2'' \to 7$, $S_2'' \to 8$, $S_2'' \to 9$, $S_2'' \to _$
$S_2 \to S_2'$, $S_2 \to S_2S_2'$, $S_2 \to S_2S_2''$ }

Es gilt dann:

Aus S_2' sind in einem Schritt genau alle Buchstaben ableitbar, d.h.: $S_2' \Rightarrow y$ genau dann, wenn $y \in \Phi_B$.
Aus S_2'' sind in einem Schritt genau alle Ziffern und der Unterstrich ableitbar, d.h.: $S_2' \Rightarrow y$ genau dann, wenn $y \in \Phi_Z \cup \{ _ \}$.

Mit den letzten beiden Regeln sind aus S_2 genau alle Wörter der Form S_2x mit $x \in \{S_2', S_2''\}^*$ ableitbar.

Aus S_2 sind mit den letzten drei Regeln genau alle Wörter der Form $S_2'x$ mit $x \in \{S_2', S_2''\}^*$ ableitbar.

Da aus S_2' nur Buchstaben und aus S_2'' nur die Ziffern und der Unterstrich ableitbar sind, so sind folglich aus S_2 genau alle Wörter der Form zx mit $z \in \Phi_B$ und $x \in \Phi^*$ ableitbar, d.h.:
$L(G_2) = \{zx \mid z \in \Phi_B \text{ und } x \in \Phi^*\} = \Phi_B\Phi^* =$ Bez. ■

Beispiel 1.1.5.6: Einfache arithmetische Ausdrücke

Definiere die kontextfreie Grammatik $G_3 = (V_3, \Sigma_3, P_3, S_3)$ mit $V_3 = \{S_3\}$ und $\Sigma_3 = \{ (,), \mathbf{a}, +, -, * \}$, und P_3 bestehe aus folgenden vier Regeln:

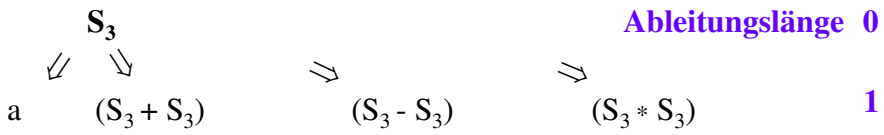
$\mathbf{S_3 \to a,\ S_3 \to (S_3 + S_3),\ S_3 \to (S_3 - S_3),\ S_3 \to (S_3 * S_3)}$

Die Sprache $L(G_3)$ lässt sich nun nicht mehr in einfacher Form angeben. Man erkennt jedoch, dass sich alle *vollständig geklammerten arithmetischen Ausdrücke, die nur den Operanden "a" enthalten*, aus S_3 ableiten lassen, z.B.:

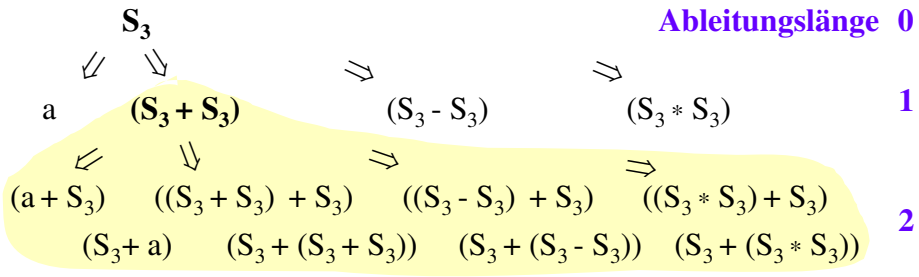
$S_3 \Rightarrow \mathbf{a}$, $\quad S_3 \Rightarrow (S_3 + S_3) \Rightarrow (a + S_3) \Rightarrow \mathbf{(a + a)}$,

$S_3 \Rightarrow (S_3 + S_3) \Rightarrow (a + S_3) \Rightarrow (a + (S_3 * S_3)) \Rightarrow (a + (S_3 * a))$

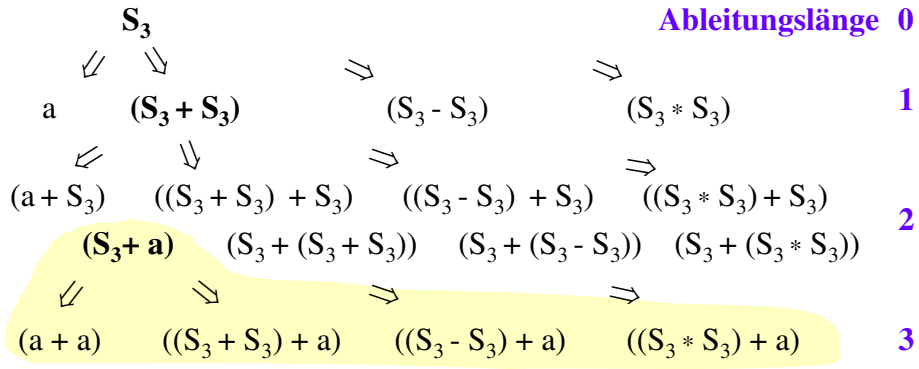
$\Rightarrow (a + ((S_3 - S_3) * a)) \Rightarrow (a + ((a - S_3) * a)) \Rightarrow \mathbf{(a + ((a - a) * a))}$.



Regeln: $S_3 \to a,\ S_3 \to (S_3 + S_3),\ S_3 \to (S_3 - S_3),\ S_3 \to (S_3 * S_3)$



Regeln: $S_3 \to a$, $S_3 \to (S_3 + S_3)$, $S_3 \to (S_3 - S_3)$, $S_3 \to (S_3 * S_3)$

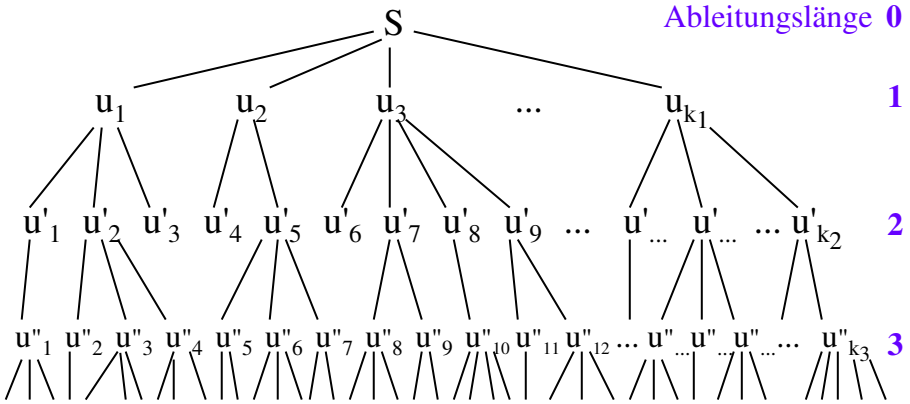


Überprüfen Sie:

4 verschiedene Wörter werden aus S_3 in genau einem Schritt abgleitet,
24 verschiedene Wörter werden aus S_3 in genau 2 Schritten abgleitet,
230 verschiedene Wörter werden aus S_3 in genau 3 Schritten abgleitet,
mehr als 1000 verschiedene Wörter werden aus S_3 in 4 Schritten abgeleitet.
Viele Wörter lassen sich auf verschiedene Art herleiten (?). ■

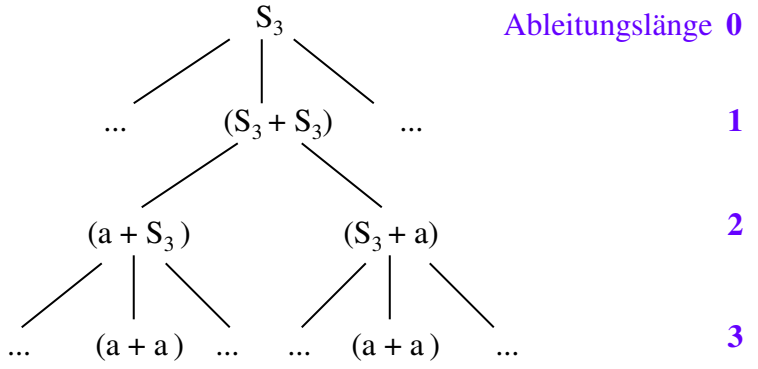
In diesem Beispiel haben wir zwei Darstellungen kennen gelernt:

(1.) Alle Ableitungen aus dem Startsymbol S einer kontextfreien Grammatik, d.h., die Herleitung aller möglicher Wörter, meist in folgender Form:

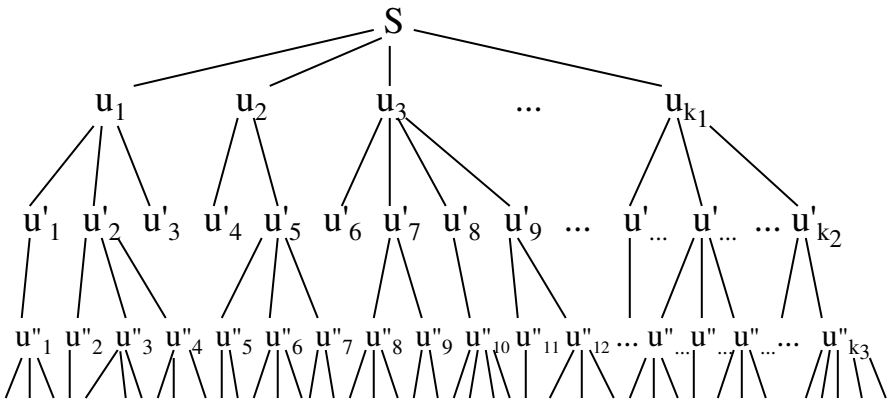


Die Menge der hergeleiteten Wörter, die nur Terminalzeichen enthalten, bilden die erzeugte Sprache $L(G_3)$.

In diesen Ableitungen können Wörter mehrfach (prinzipiell sogar unendlich oft) vorkommen. Zum Beispiel in obiger Grammatik G_3:

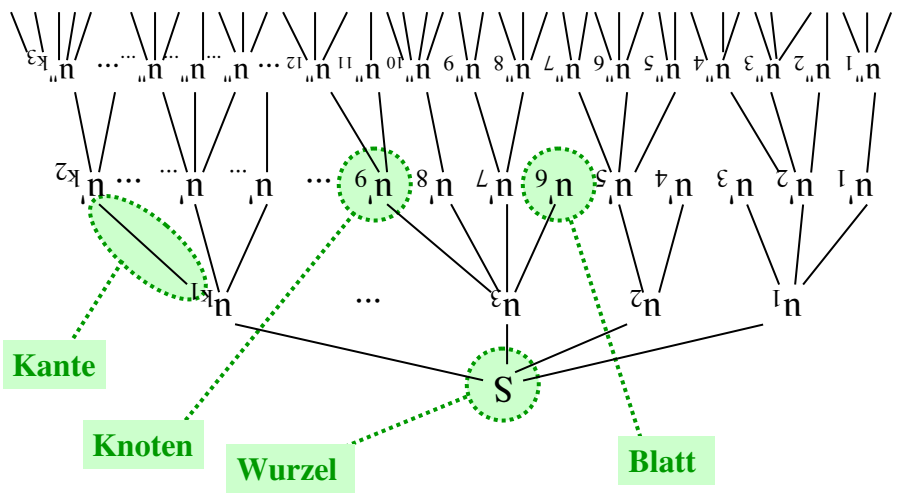


Bemerkung: Solch ein Gebilde, bei dem von jedem Punkt aus mehrere Linien abgehen, nirgends aber Linien zusammenlaufen können, nennt man einen Baum (engl.: tree). Das Gebilde sieht ja auch baumartig aus:



Allerdings wächst dieser Baum nach unten statt nach oben.

Einige Bezeichnungen bei Bäumen:



Bezeichnungen 1.1.5.7 bei "Bäumen":

Bei den Linien spricht auch von Zweigen, meist aber von Kanten (engl.: edges),

die Stellen, an denen die Wörter stehen und von denen Kanten ausgehen, heißen Knoten (engl.: nodes);

ein Knoten, von denen keine Zweige mehr weiterführen, heißt Blatt (engl.: leaf),

der unterste (bzw. oberste) Knoten heißt die Wurzel des Baums (engl.: root).

Feststellung 1.1.5.8: Alle Ableitungen einer kontextfreien Grammatik zusammen bilden einen (i.A. unendlich großen) Baum.

In der Wurzel des Baumes steht das Startsymbol S.

Jeder Ableitung

$$u = z_0 \Rightarrow z_1 \Rightarrow z_2 \Rightarrow z_{3i} \Rightarrow \ldots \Rightarrow z_{k-1} \Rightarrow z_k = v$$

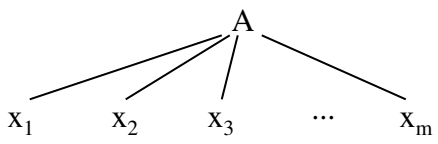
entspricht in diesem Baum ein Pfad (= eine Folge von Kanten) beginnend an einem Knoten, in dem u steht, und endend an einem Knoten, in dem v steht.

Knoten, in denen Wörter aus Terminalzeichen stehen, bilden stets Blätter; genau solche Wörter sind in der erzeugten Sprache L(G) enthalten.

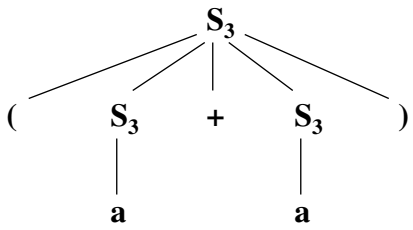
Die Ableitungen von Wörtern der erzeugten Sprache beginnen (als Pfad) stets in der Wurzel S und enden bei dem jeweiligen Wort in einem Blatt.

(2.) Die Ableitung eines speziellen Wortes aus dem Startsymbol einer Grammatik haben wir wie folgt dargestellt:
$S_3 \Rightarrow (S_3 + S_3) \Rightarrow (a + S_3) \Rightarrow$ **(a + a)**
Diese Ableitung kann man umstellen, indem man zuerst das zweite S_3 durch a ersetzt und danach erst das erste S_3:
$S_3 \Rightarrow (S_3 + S_3) \Rightarrow (S_3 + a) \Rightarrow$ **(a + a)**
Dies ist im Wesentlichen *die gleiche Ableitung*.

Um dies präzise zu definieren, schreiben wir die einzelnen Ableitungen baumartig auf, wobei in jedem Knoten genau ein Terminal- bzw. Nichtterminalzeichen steht. Genauer: Eine Regel $A \rightarrow x_1x_2x_3 \ldots x_m$ mit $x_i \in (V \cup \Sigma)$ notieren wir in der Form



Die Ableitung $S_3 \Rightarrow (S_3 + S_3) \Rightarrow (S_3 + a) \Rightarrow (a + a)$ wird dann durch folgenden Baum dargestellt:



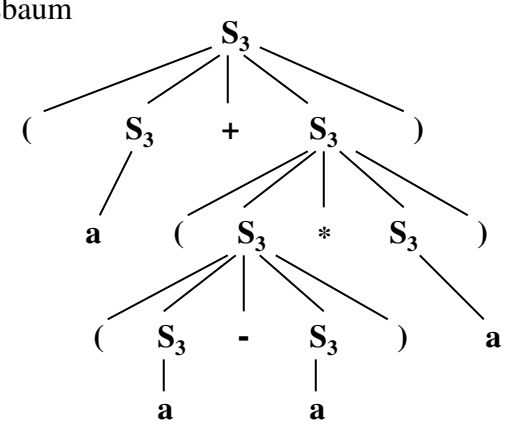
1.1.5.9:
Dieses Gebilde heißt Ableitungsbaum (engl.: derivation tree) zu obiger Ableitung für das Wort $(a + a)$ in der Grammatik G_3.

Dieser Baum ist zugleich der Ableitungsbaum zur Ableitung
$S_3 \Rightarrow (S_3 + S_3) \Rightarrow (a + S_3) \Rightarrow (a + a)$

Als weiteres Beispiel betrachten wir (aus 1.1.5.6) die Ableitung

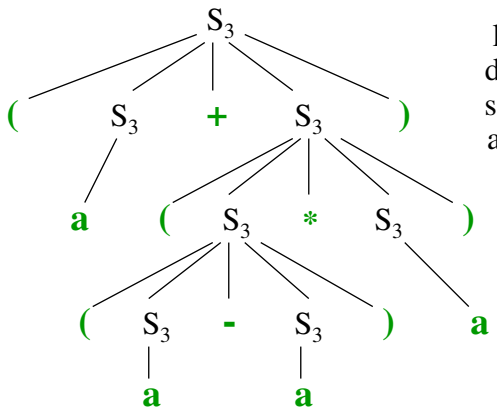
$S_3 \Rightarrow (S_3 + S_3) \Rightarrow (a + S_3) \Rightarrow (a + (S_3 * S_3)) \Rightarrow (a + (S_3 * a))$
$\Rightarrow (a + ((S_3 - S_3) * a)) \Rightarrow (a + ((a - S_3) * a)) \Rightarrow (a + ((a - a) * a))$.

Ableitungsbaum hierzu:



Als weiteres Beispiel betrachten wir (aus 1.1.5.6) die Ableitung

$S_3 \Rightarrow (S_3 + S_3) \Rightarrow (a + S_3) \Rightarrow (a + (S_3 * S_3)) \Rightarrow (a + (S_3 * a))$
$\Rightarrow (a + ((S_3 - S_3) * a)) \Rightarrow (a + ((a - S_3) * a)) \Rightarrow$ **(a + ((a - a) * a))** .



Geht man von links nach rechts die Blätter durch, so erhält man das abgeleitete Wort.

Diesen Ableitungsbaum besitzen auch andere Ableitungen, z.B.:

$S_3 \Rightarrow (S_3 + S_3) \Rightarrow (S_3 + (S_3 * S_3)) \Rightarrow (S_3 + (S_3 * a))$
$\Rightarrow (S_3 + ((S_3 - S_3) * a)) \Rightarrow (S_3 + ((a - S_3) * a))$
$\Rightarrow (S_3 + ((a - a) * a)) \Rightarrow (a + ((a - a) * a))$

oder:

$S_3 \Rightarrow (S_3 + S_3) \Rightarrow (S_3 + (S_3 * S_3)) \Rightarrow (a + (S_3 * S_3))$
$\Rightarrow (a + ((S_3 - S_3) * S_3)) \Rightarrow (a + ((a - S_3) * S_3))$
$\Rightarrow (a + ((a - S_3) * a)) \Rightarrow (a + ((a - a) * a))$.

1.1.5.10: *Ableitungen, die den gleichen Ableitungsbaum besitzen, sehen wir als gleich an.* Sie unterscheiden sich nur in der Reihenfolge, in der Regeln auf Nichtterminalzeichen angewendet werden, die an voneinander unabhängigen Stellen im Wort stehen.

Das Wort, das sich aus einem Ableitungsbaum ergibt, wenn man die Blätter von links nach rechts durchläuft, heißt das mit diesem Ableitungsbaum abgeleitete Wort.

Definition 1.1.5.11: Es sei $G = (V, \Sigma, P, S)$ eine kontextfreie Grammatik und $w \in L(G)$ ein Wort der erzeugten Sprache.

(1) w heißt eindeutig (bzgl. G), wenn es nur genau einen Ableitungsbaum gibt, dessen abgeleitetes Wort w ist.

(2) Gibt es mindestens zwei verschiedene Ableitungsbäume für w, so heißt w mehrdeutig (bzgl. G).

(3) G heißt eindeutig, wenn alle Wörter $w \in L(G)$ eindeutig sind.

(4) G heißt mehrdeutig, wenn mindestens ein Wort $w \in L(G)$ mehrdeutig ist.

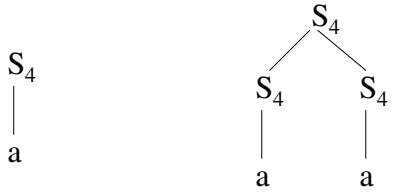
(Englisch: mehrdeutig = ambiguous, eindeutig = unambiguous.)

Beispiel 1.1.5.12: Wir untersuchen die folgende kontextfreie Grammatik $G_4 = (V_4, \Sigma_4, P_4, S_4)$ mit $V_4 = \{S_4\}$, $\Sigma_4 = \{a\}$ und $P_4 = \{S_4 \to S_4S_4,\ S_4 \to a\}$.

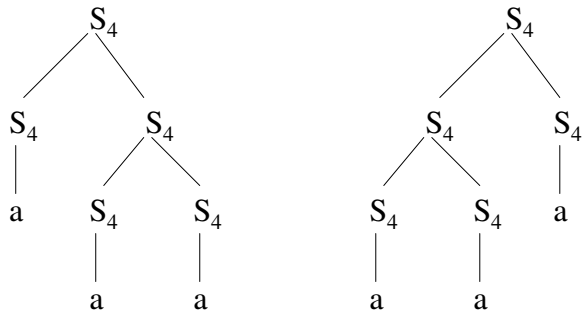
Offenbar kann man mit der ersten Regel aus S_4 jede nicht-leere Folge des Zeichens S_4 erzeugen; mit der zweiten Regel lässt sich hieraus dann jedes nicht-leere Wort über Σ_4 ableiten, d.h.,

$L(G) = \Sigma_4^+ = \{a^n \mid n \geq 1\}$.

Die Wörter a und aa sind eindeutig bzgl. G_4. Sie besitzen nur die Ableitungsbäume:



Jedes Wort über Σ_4, das mindestens die Länge 3 besitzt, ist mehrdeutig. aaa besitzt z.B. die beiden Ableitungsbäume



Beachten Sie: Die Reihenfolge der ausgehenden Kanten darf nicht vertauscht werden! Daher sind diese beiden Ableitungsbäume verschieden. ■

Die Anwendung kontextfreier Grammatiken auf Programmiersprachen, erfolgt im nächsten Abschnitt. Hier wollen wir abschließend den allgemeinen Grammatikbegriff vorstellen.

Die bisher eingeführten Grammatiken heißen "kontextfrei", weil die Ableitung eines Nichtterminalzeichens "ohne Beachtung des Kontexts" erfolgt, d.h., eine Regel $A \to w$ kann auf das Nichtterminalzeichen A in einem Wort angewendet werden, ohne die links und rechts von A stehenden Zeichen zu beachten.

Darf man eine Regel $A \to w$ aber nur anwenden, wenn links von A das (Teil-) Wort x und rechts das (Teil-) Wort y stehen, dann würde man die Regel $xAy \to xwy$ verwenden müssen. Dies führt zu den "kontextsensitiven" Grammatiken. Wir verallgemeinern diesen Gedanken noch einmal und erhalten:

Definition 1.1.5.13:

G = (V, Σ, P, S) heißt (Chomsky-) Grammatik genau dann, wenn:

(1) V ist eine nicht-leere endliche Menge (die Menge der Nichtterminalzeichen oder Variablen),

(2) Σ ist eine nicht-leere endliche Menge (die Menge der Terminalzeichen),

(3) $S \in V$ ist ein Nichtterminalzeichen (das Startsymbol),

(4) $P \subset V^+ \times (V \cup \Sigma)^*$ ist eine endliche Menge (die Menge der Regeln oder Produktionen).

Wir haben an Definition 1.1.5.1 also nur eine Kleinigkeit geändert, indem wir in (4) die Menge V zur Menge V^+ abgeändert haben. Wir passen nun den Begriff der Ableitung hierauf an.

Definition 1.1.5.14: Gegeben sei eine Grammatik G = (V, Σ, P, S). Die Regelmenge P definiert auf der Menge $(V \cup \Sigma)^*$ die "Ableitungsrelationen" $\Rightarrow$ und $\Rightarrow^*$:

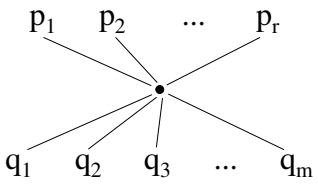
(1) Es gilt $u \Rightarrow v$ genau dann, wenn man die Wörter u und v in der Form $u = xpy$, $v = xqy$ mit $x, y \in (V \cup \Sigma)^*$ und $p \rightarrow q \in P$ schreiben kann. Man sagt: y ist aus x in einem Schritt herleitbar oder y lässt sich aus x in einem Schritt ableiten.

(2) Es gilt $u \Rightarrow^* v$ genau dann, wenn entweder $u = v$ ist oder wenn es Wörter $z_0, z_1, ..., z_k \in (V \cup \Sigma)^*$ für ein $k \geq 1$ gibt mit $u = z_0$, $v = z_k$, $z_i \Rightarrow z_{i+1}$ für alle $i = 0, 1, ..., k-1$. Man sagt dann, v ist aus u herleitbar oder ableitbar. (Die Zahl k heißt Ableitungslänge.)

Hinweis: $\Rightarrow^*$ ist der so genannte "reflexive und transitive Abschluss" von $\Rightarrow$. (Englisch: Ableitung = derivation.)

Definition 1.1.5.15:
Die von einer Grammatik $G = (V, \Sigma, P, S)$ erzeugte Sprache (engl.: generated language) ist die Menge
$L(G) = \{ w \in \Sigma^* \mid S \Rightarrow^* w \} \subseteq \Sigma^*$.

Ableitungen kann man nun nicht mehr als Baum darstellen, vielmehr bilden sie ein "Netz", das aus Gebilden der Form



für eine Regel $p_1p_2p_3 \ldots p_r \rightarrow q_1q_2q_3 \ldots q_m$ mit $p_i \in V$, $q_j \in (V \cup \Sigma)$, $r \geq 1$, $m \geq 0$, aufgebaut wird. Wir betrachten ein Beispiel.

Beispiel 1.1.5.16: Betrachte folgende Grammatik $G_5 = (V_5, \Sigma_5, P_5, S_5)$ mit $V_5 = \{S_5, A, B, C\}$, $\Sigma_5 = \{a, b\}$ und

$P_5 = \{S_5 \rightarrow AS_5B,\ S_5 \rightarrow CC,\ AC \rightarrow BA,\ CB \rightarrow BC,$
$BAB \rightarrow C,\ AB \rightarrow a,\ CC \rightarrow b\}$.

Um die "Arbeitsweise" der Grammatik zu verstehen, versucht man zunächst, einige Wörter herzuleiten:

$S_5 \Rightarrow CC \Rightarrow b$

$S_5 \Rightarrow AS_5B \Rightarrow ACCB \Rightarrow BACB \Rightarrow BABC \Rightarrow CC \Rightarrow b$

Hinweis: Dies sind offensichtlich zwei wesentlich verschiedene Ableitungen, d.h., die Grammatik G_5 ist mehrdeutig. Man kann jetzt aber nicht mehr mit Bäumen argumentieren, sondern man muss "Ableitungsnetze" betrachten.

Zunächst stellen wir die einzelnen Regeln dar:

$S_5 \rightarrow AS_5B$

$S_5 \rightarrow CC$

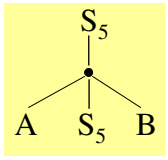
$AC \rightarrow BA$

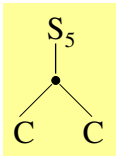
$CB \rightarrow BC$

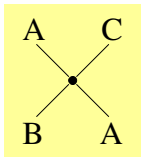
$BAB \rightarrow C$

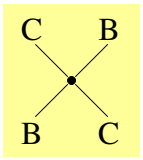
$AB \rightarrow a$

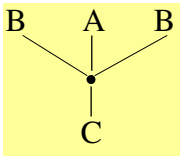
$CC \rightarrow b$

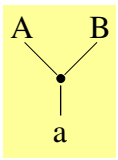


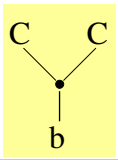












Nun kleben wir die einzelnen Regeln zu Ableitungen zusammen:

$S_5 \rightarrow AS_5B$

$S_5 \rightarrow CC$

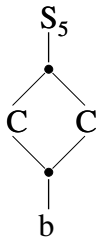
$AC \rightarrow BA$

$CB \rightarrow BC$

$BAB \rightarrow C$

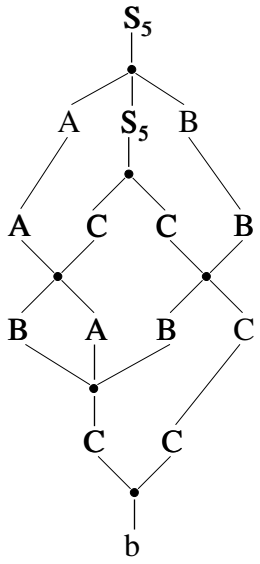
$AB \rightarrow a$

$CC \rightarrow b$



Nun kleben wir die einzelnen Regeln zu Ableitungen zusammen:

$S_5 \to AS_5B$	S_5
$S_5 \to CC$	$\Rightarrow AS_5B$
$AC \to BA$	$\Rightarrow ACCB$
$CB \to BC$	$\Rightarrow BACB$
$BAB \to C$	$\Rightarrow BABC$
$AB \to a$	$\Rightarrow CC$
$CC \to b$	$\Rightarrow b$



Suche nach weiteren Ableitungen:

$S_5 \Rightarrow AS_5B \Rightarrow ACCB \Rightarrow ACBC \Rightarrow ABCC \Rightarrow aCC \Rightarrow ab$

$S_5 \Rightarrow AS_5B \Rightarrow AAS_5BB \Rightarrow AACCBB \Rightarrow ABACBB$
$\Rightarrow ABABCB \Rightarrow ACCB \Rightarrow ACBC \Rightarrow ABCC \Rightarrow aCC \Rightarrow ab$

Es gibt noch viele weitere Ableitungen, aber sie alle erzeugen eines der beiden Wörter b oder ab, d.h., $L(G_5) = \{b, ab\}$.

Stimmt das wirklich? Wie kann man so etwas beweisen? Wie kann überhaupt die Menge L(G) für eine Grammatik bestimmen bzw. wie kann man zu einem Wort w und einer Grammatik feststellen, ob diese Grammatik das Wort erzeugt oder nicht (sog. Wortproblem)? *Siehe künftige Theorievorlesungen ...*

■

Definition 1.1.5.17: Gegeben sei eine Grammatik $G = (V, \Sigma, P, S)$. Die Grammatik G heißt

(1) vom Typ 1 oder kontextsensitiv genau dann, wenn alle Regeln von der Form $xAy \rightarrow xwy$ sind mit $A \in V$, $x, y \in V^*$ und $w \in (V \cup \Sigma)^+$; insbesondere muss $w \neq \varepsilon$ gelten.

(2) vom Typ 2 oder kontextfrei genau dann, wenn alle Regeln von der Form $A \rightarrow w$ sind mit $A \in V$ und $w \in (V \cup \Sigma)^*$ ($w = \varepsilon$ ist hier erlaubt).

(3) vom Typ 3 oder rechtslinear genau dann, wenn alle Regeln von der Form $A \rightarrow uB$ oder $A \rightarrow u$ sind mit $A, B \in V$ und $u \in \Sigma^*$.

(4) linkslinear genau dann, wenn alle Regeln von der Form $A \rightarrow Bu$ oder $A \rightarrow u$ sind mit $A, B \in V$ und $u \in \Sigma^*$.

Definition 1.1.5.18: Eine Sprache heißt "xxx", wenn es eine "xxx" Grammatik G mit L = L(G) gibt für "xxx" ∈ {kontextsensitiv, kontextfrei, rechtslinear, linkslinear}.

Die Einteilung in Grammatiken vom Typ 0 (= keine Einschränkung), Typ 1 (kontextsensitiv), Typ 2 (kontextfrei) und Typ 3 (rechtslinear) stammt aus der Originalarbeit von Noam Chomsky 1959 und hat sich bis heute gehalten. Es gibt aber mittlerweile viele weitere Grammatikklassen.

Hinweise: Die Sprache der Bezeichner Bez und die Sprache der korrekten Zahldarstellungen sind rechtslinear. Die Sprache der korrekt geklammerten arithmetischen oder Booleschen Ausdrücke ist kontextfrei, aber nicht rechtslinear.

Einige Aussagen (ohne Beweis):

Fügt man zu einer "xxx" Sprache eine endliche Sprache hinzu oder zieht man eine endliche Sprache ab, so bleibt das Ergebnis eine "xxx" Sprache.

Eine Sprache genau dann rechtslinear, wenn sie linkslinear ist.

Die Sprache
$\{a^n b^n \mid n \geq 0\}$ ist kontextfrei, aber nicht rechtslinear.

Die Sprache
$\{a^n b^n a^n \mid n \geq 0\}$ ist kontextsensitiv, aber nicht kontextfrei.

Es gibt Sprachen vom Typ 0, die nicht kontextsensitiv sind
(leider lassen sie sich nicht mit einer kurzen Formel beschreiben).

Alle in der Praxis verwendeten Programmiersprachen sind kontextsensitive Sprachen; sie lassen sich als kontextfreie Sprachen mit zusätzlichen Einschränkungen beschreiben.