8 KONTEXTFREIE GRAMMATIKEN

8.1 REKURSIVE DEFINITION SYNTAKTISCHER STRUKTUREN

Wir hatten in Einheit 7 über Dokumente schon darauf hingewiesen, dass die Beschreibung formaler Sprachen nur mit Hilfe einzelner Symbole und der Operation Vereinigung, Konkatenation und Konkatenationsabschluss manchmal möglich ist, aber manchmal anscheinend auch nicht.

Tatsächlich ist es manchmal unmöglich. Wie man zu einen harten Beweis für diese Aussage kommen kann, werden wir in einer späteren Einheit sehen.

Als typisches Beispiel eines Problemfalles sehen wir uns einen Ausschnitt der Definition der Syntax von Java an (siehe die Seite über Anweisungen unter `http://java.sun.com/docs/books/jls/third_edition/html/statements.html`, 13.11.08). Dort steht im Zusammenhang mit der Festlegung, was in Java eine Anweisung sei, unter anderem folgende fünf Teile:

1	Block:
	{ $\text{BlockStatements}_{\text{opt}}$ }
2	BlockStatements:
	BlockStatement
	BlockStatements BlockStatement
3	BlockStatement:
	Statement
	.....
4	Statement:
	StatementWithoutTrailingSubstatement
	.....
5	StatementWithoutTrailingSubstatement:
	Block
	.....

Tabelle 8.1: Auszug aus der Spezifikation der Syntax von Java

Wir werden die Wörter aus obiger Spezifikation von normalem Text unterscheiden, indem wir sie in spitzen Klammern und kursiv setzen, wie z. B. bei $\langle \textit{Block} \rangle$, um das Lesen zu vereinfachen (und um schon mal deutlich zu machen, dass es sich z. B. bei $\langle \textit{Block} \rangle$ um etwas handelt, was wir als *ein* Ding ansehen wollen).

Man macht keinen Fehler, wenn man das zum Beispiel erst mal so liest:

1. Ein $\langle \textit{Block} \rangle$ hat die Struktur: Ein Zeichen { , eventuell gefolgt von $\langle \textit{BlockStatements} \rangle$

(das tiefgestellte Suffix „opt“ besagt, dass das optional ist), gefolgt von einem Zeichen `}` .

2. ⟨*BlockStatements*⟩ sind von einer der Strukturen
   - ein einzelnes ⟨*BlockStatement*⟩ oder
   - ⟨*BlockStatements*⟩ gefolgt von einem ⟨*BlockStatement*⟩
3. Ein ⟨*BlockStatement*⟩ kann ein ⟨*Statement*⟩ sein (oder anderes . . .).
4. Ein ⟨*Statement*⟩ kann ein ⟨*StatementWithoutTrailingSubstatement*⟩ sein (oder anderes, zum Beispiel etwas „ganz einfaches“ . . .).
5. Ein ⟨*StatementWithoutTrailingSubstatement*⟩ kann ein ⟨*Block*⟩ sein (oder anderes . . .).

Diese Formulierungen beinhalten Rekursion: Bei der Beschreibung der Struktur von ⟨*BlockStatements*⟩ wird direkt auf ⟨*BlockStatements*⟩ Bezug genommen.

Außerdem wird bei der Definition von ⟨*Block*⟩ (indirekt) auf ⟨*Statement*⟩ verwiesen und bei der Definition von ⟨*Statement*⟩ (indirekt) wieder auf ⟨*Block*⟩.

Wir werden uns der Antwort auf die Frage, wie man diese Rekursionen eine vernünftige Bedeutung zuordnen kann, in mehreren Schritten nähern.

Als erstes schälen wir den für uns gerade wesentlichen Aspekt noch besser heraus. Schreiben wir einfach X statt ⟨*Block*⟩, ⟨*Statement*⟩ o.ä., und schreiben wir lieber runde Klammern `(` und `)` statt der geschweiften, weil wir die die ganze Zeit als Mengenklammern benutzen wollen. (Sie fragen sich, warum nicht einen Ausschnitt aus der Grammatik für arithmetische Ausdrücke genommen haben, in denen sowieso schon runde Klammern benutzt werden? Die Antwort ist: Der Ausschnitt aus der Syntaxbeschreibung wäre viel länger und unübersichtlicher geworden.)

Dann besagt die Definition unter anderem:

K1 Ein X kann etwas “ganz einfaches“ sein. Im aktuellen Zusammenhang abstrahieren wir mal ganz stark und schreiben für dieses Einfache einfach das leere Wort ε.

K2 Ein X kann ein Y sein oder auch ein X gefolgt von einem Y; also kann ein X von der Form YY sein. Jedes Y seinerseits kann wieder ein X sein. Also kann ein X auch von der Form XX sein.

K3 Wenn man ein X hat, dann ist auch (X) wieder ein X.

K4 Schließlich tun wir so, als dürfe man in die Definition auch hineininterpretieren: Es ist nichts ein X, was man nicht auf Grund der obigen Festlegungen als solches identifizieren kann.

Und nun? Wir könnten jetzt zum Beispiel versuchen, mit X eine formale Sprache L zu assoziieren, und folgende Gleichung hinschreiben:

$$L = \{\varepsilon\} \cup LL \cup \{(\}L\{)\} \tag{8.1}$$

Dabei wäre die Hoffnung, dass die Inklusion $L \supseteq \ldots$ die ersten drei aufgezählten Punkte widerspiegelt, und die Inklusion $L \subseteq \ldots$ den letzten Punkt. Wir werden sehen, dass diese Hoffnung zum Teil leider trügt.

Die entscheidenden Fragen sind nun natürlich:

1. Gibt es überhaupt eine formale Sprache, die Gleichung 8.1 erfüllt?

   Das hätten wir gerne, und wir werden sehen, dass es tatsächlich so ist, indem wir eine Lösung der Gleichung konstruieren werden.
2. Und falls ja: Ist die formale Sprache, die Gleichung 8.1 erfüllt, nur durch die Gleichung eindeutig festgelegt?

   Das hätten wir auch gerne, wir werden aber sehen, dass das *nicht* so ist. Das bedeutet für uns die zusätzliche Arbeit, dass wir „irgendwie“ eine der Lösungen als die uns interessierende herausfinden und charakterisieren müssen. Das werden wir im nächsten Unterabschnitt machen.

Zum Abschluss dieses Unterabschnittes wollen wir eine Lösung von Gleichung 8.1 konstruieren. Wie könnte man das tun? Wir „tasten uns an eine Lösung heran“.

Das machen wir, indem wir

- erstens eine ganze Folge $L_0, L_1, \ldots$ formaler Sprachen L_i für $i \in \mathbb{N}_0$ definieren, der Art, dass jedes L_i offensichtlich jedenfalls einige der gewünschten Wörter über dem Alphabet $\{(,)\}$ enthält, und
- zweitens dann zeigen, dass die Vereinigung aller dieser L_i eine Lösung von Gleichung 8.1 ist.

Also:

- Der Anfang ist ganz leicht: $L_0 = \{\varepsilon\}$.
- und weiter machen kann man so: für $i \in \mathbb{N}_0$ sei $L_{i+1} = L_i L_i \cup \{(\}L_i\{)\}$.

Wir behaupten, dass $L = \bigcup_{i=0}^{\infty} L_i$ Gleichung 8.1 erfüllt.

Zum Beweis der Gleichheit überzeugen wir uns davon, dass beide Inklusionen erfüllt sind.

Zunächst halten wir fest: $\varepsilon \in L_0$. Außerdem ist für alle $i \in \mathbb{N}_0$ auch $L_i L_i \subseteq L_{i+1}$, wenn also $\varepsilon \in L_i$, dann auch $\varepsilon = \varepsilon\varepsilon \in L_i L_i$, also $\varepsilon \in L_{i+1}$. Also gilt für alle $i \in \mathbb{N}_0$: $\varepsilon \in L_i$. Und folglich gilt für alle $i \in \mathbb{N}_0$: $L_i = L_i\{\varepsilon\} \subseteq L_i L_i$, also ist für alle $i \in \mathbb{N}_0$: $L_i \subseteq L_{i+1}$.

$L \subseteq \{\varepsilon\} \cup LL \cup \{(\}L\{)\}$**:** Da $\varepsilon \in L_0 \subseteq L$ ist, ist $L = L\{\varepsilon\} \subseteq LL$.
$L \supseteq \{\varepsilon\} \cup LL \cup \{(\}L\{)\}$**:** sei $w \in \{\varepsilon\} \cup LL \cup \{(\}L\{)\}$.

1. Fall: $w = \varepsilon$**:** $w = \varepsilon \in L_0 \subseteq L$.

2. Fall: $w \in LL$**:** Dann ist $w = w_1 w_2$ mit $w_1 \in L$ und $w_2 \in L$. Es gibt also Indizes i_1 und i_2 mit $w_1 \in L_{i_1}$ und $w_2 \in L_{i_2}$. Für $i = \max(i_1, i_2)$ ist also $w_1 \in L_i$ und $w_2 \in L_i$, also $w = w_1 w_2 \in L_i L_i \subseteq L_{i+1} \subseteq L$.

3. Fall: $w \in \{(\}L\{)\}$**:** Für ein $i \in \mathbb{N}_0$ ist dann $w \in \{(\}L_i\{)\} \subseteq L_{i+1} \subseteq L$.

Damit haben wir gezeigt, dass Gleichung 8.1 (mindestens) eine Lösung hat.

Um sehen, dass die konstruierte Lösung keineswegs die einzige ist, muss man sich nur klar machen, dass $\{(,)\}^*$ auch eine Lösung der Gleichung ist, aber eine andere.

- Dass es sich um eine Lösung handelt, sieht man so: „$\subseteq$“ zeigt man wie oben; „$\supseteq$“ ist trivial, da $\{(,)\}^*$ eben *alle* Wörter sind.
- Dass es eine andere Lösung ist, sieht man daran, dass z. B. das Wort `(((` zwar in $\{(,)\}^*$ liegt, aber *nicht* in der oben konstruierten Lösung. Die enthält nämlich jedenfalls nur Wörter, in denen gleich viele `(` und `)` vorkommen. (Diese Behauptung gilt nämlich für alle L_i, wie man durch vollständige Induktion zeigen kann.)

Kommen wir zurück zu unserer zuerst konstruierten Lösung $L = \bigcup_{i=0}^{\infty} L_i$. Zählen wir für die ersten vier L_i explizit auf, welche Wörter jeweils neu hinzukommen:

$$
\begin{aligned}
L_0 &= \{\varepsilon\} \\
L_1 \setminus L_0 &= \{\ \texttt{()}\ \} \\
L_2 \setminus L_1 &= \{\ \texttt{()()}\ ,\ \texttt{(())}\ \} \\
L_3 \setminus L_2 &= \{\ \texttt{()()()}\ ,\ \texttt{(())()}\ ,\ \texttt{()(())}\ , \\
&\qquad \texttt{()()()()}\ ,\ \texttt{()()(())}\ ,\ \texttt{(())()()}\ ,\ \texttt{(())(())}\ , \\
&\qquad \texttt{(()())}\ ,\ \texttt{((()))}\ \}
\end{aligned}
$$

Dabei gilt zum Beispiel:

- Die Erklärung für `(())()()` $\in L_3$ ist, dass `(())` $\in L_2$ und `()()` $\in L_2$ und Regel K2.
  - Die Erklärung für `(())` $\in L_2$ ist, dass `()` $\in L_1$ und Regel K3.
    - Die Erklärung für `()` $\in L_1$ ist, dass $\varepsilon \in L_0$ und Regel K3.
  - Die Erklärung für `()()` $\in L_2$ ist, dass `()` $\in L_1$ und `()` $\in L_1$ ist und Regel K2
    - Die Erklärung für `()` $\in L_1$ ist, dass $\varepsilon \in L_0$ und Regel K3.
    - Die Erklärung für `()` $\in L_1$ ist, dass $\varepsilon \in L_0$ und Regel K3.

Das kann man auch in graphischer Form darstellen: siehe Abbildung 8.1. Ersetzt man dann noch überall die $w \in L_i$ durch „X“, ergibt sich Abbildung 8.2.

Eine solche Darstellung nennt man in der Informatik einen Baum (kurioserweise wird die „Wurzel“ üblicherweise oben dargestellt und die „Blätter“ unten). Auf Bäume und andere Graphen als Untersuchungsgegenstand werden wir in einer späteren Einheit zu sprechen kommen. Vorläufig genügt es uns, dass wir solche Bilder malen können.

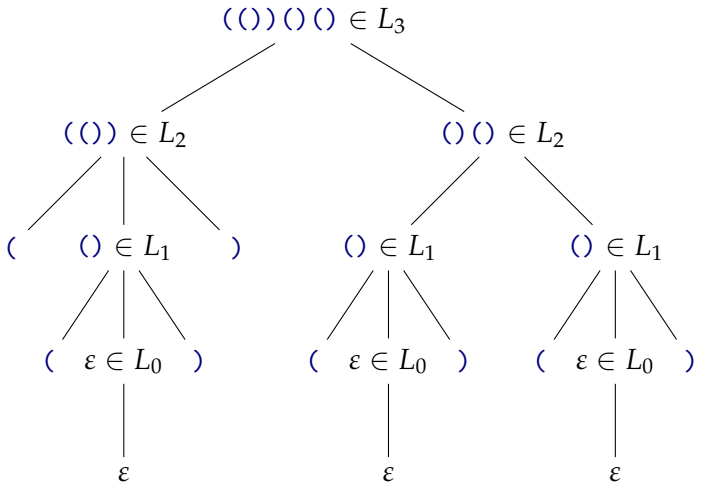

Abbildung 8.1: Eine „Begründung“, warum $(())()() \in L_3$ ist.

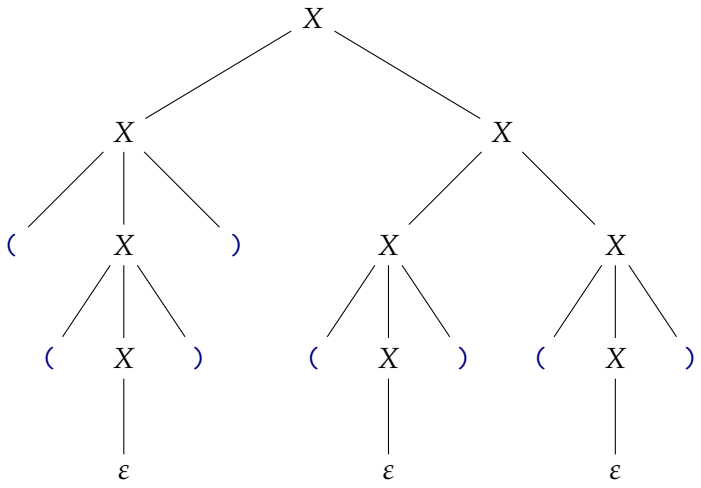

Abbildung 8.2: Eine abstraktere Darstellung der „Begründung“, warum $(())()() \in L_3$ ist.

Zu kontextfreien Grammatiken ist es nun nur noch ein kleiner Schritt: Statt wie eben im Baum Verzweigungen von unten nach oben zu als Rechtfertigungen zu interpretieren, geht man umgekehrt vor und interpretiert Verzweigungen als Stellen, an denen man „Verfeinerungen“ vornimmt, indem man rein syntaktisch zum Beispiel ein X *ersetzt* durch (X) oder XX.

8.2 KONTEXTFREIE GRAMMATIKEN

kontextfreie Grammatik

Eine *kontextfreie Grammatik* $G = (N, T, S, P)$ ist durch vier Bestandteile gekennzeichnet, die folgende Bedeutung haben:

Nichtterminalsymbole
- N ist ein Alphabet, dessen Elemente *Nichtterminalsymbole* heißen.

Terminalsymbole
- T ist ein Alphabet, dessen Elemente *Terminalsymbole* heißen. Kein Zeichen darf in beiden Alphabeten vorkommen; es muss stets $N \cap T = \emptyset$ sein.

Startsymbol
- $S \in N$ ist das sogenannte *Startsymbol* (also ein Nichtterminalsymbol).

Produktionen
- $P \subseteq N \times V^*$ ist eine *endliche* Menge sogenannter *Produktionen*. Dabei ist $V = N \cup T$ die Menge aller Symbole überhaupt.

$X \to w$

Gilt für ein Nichtterminalsymbol X und ein Wort w, dass $(X, w) \in P$ ist, dann schreibt man diese Produktion üblicherweise in der Form $X \to w$ auf. Eine solche Produktion besagt, dass man ein Vorkommen des Zeichens X durch das Wort w ersetzen darf, und zwar „egal wo es steht“, d. h. ohne auf den Kontext zu achten.

Ableitungsschritt

Das ist dann das, was man einen *Ableitungsschritt* gemäß einer Grammatik nennt. Formal definiert man das so: Aus einem Wort $u \in V^*$ ist in einem Schritt ein Wort $v \in V^*$ ableitbar, in Zeichen $u \Rightarrow v$, wenn es Wörter $w_1, w_2 \in V^*$ und eine Produktion $X \to w$ in P gibt, so dass $u = w_1 X w_2$ und $v = w_1 w w_2$.

$u \Rightarrow v$

Betrachten wir als Beispiel die Grammatik $G = (\{X\}, \{\mathtt{a}, \mathtt{b}\}, X, P)$ mit der Produktionenmenge $P = \{X \to \varepsilon, X \to \mathtt{a}X\mathtt{b}\}$. Dann gilt zum Beispiel $\mathtt{aba}X\mathtt{ba}XXXX \Rightarrow \mathtt{aba}X\mathtt{baa}X\mathtt{b}XXX$, wie man an der folgenden Aufteilung sieht:

$$\underbrace{\mathtt{aba}X\mathtt{ba}}_{w_1} X \underbrace{XXX}_{w_2} \Rightarrow \underbrace{\mathtt{aba}X\mathtt{ba}}_{w_1} \mathtt{a}X\mathtt{b} \underbrace{XXX}_{w_2}$$

Ebenso gilt $\mathtt{aba}X\mathtt{ba}XXXX \Rightarrow \mathtt{abaa}X\mathtt{bba}XXXX$:

$$\underbrace{\mathtt{aba}}_{w_1} X \underbrace{\mathtt{ba}XXXX}_{w_2} \Rightarrow \underbrace{\mathtt{aba}}_{w_1} \mathtt{a}X\mathtt{b} \underbrace{\mathtt{ba}XXXX}_{w_2}$$

Man kann also unter Umständen aus dem gleichen Wort verschiedene Wörter in einem Schritt ableiten.

Die Definition von $\Rightarrow$ legt eine Relation zwischen Wörtern über dem Alphabet $V = N \cup T$ fest. Man könnte also auch schreiben: $R_\Rightarrow \subseteq V^* \times V^*$ oder gar $\Rightarrow \subseteq V^* \times V^*$. In diesem Fall, wie z. B. auch bei der $\leq$-Relation, ist es aber so, dass man die sogenannte *Infixschreibweise* bevorzugt: Man schreibt $5 \leq 7$ und nicht $(5, 7) \in R_\leq$ o. ä.

Infixschreibweise

Im allgemeinen ist $\Rightarrow$ weder links- noch rechtstotal und weder links- noch rechtseindeutig.

Da die linke Seite einer Produktion immer ein Nichtterminalsymbol ist, wird bei einem Ableitungsschritt nie ein Terminalsymbol ersetzt. Wo sie stehen, ist „die

Ableitung zu Ende“ (daher der Name *Terminal*symbol). Eine *Ableitung* (oder auch *Ableitungsfolge*) ist eine Folge von Ableitungsschritten, deren Anzahl irrelevant ist. Die folgenden Definitionen kommen Ihnen vermutlich schon vertrauter vor als vor einigen Wochen. Für alle $u, v \in V^*$ gelte

Ableitung

$$\begin{aligned} u \Rightarrow^0 v \text{ genau dann, wenn } u = v \\ \forall i \in \mathbb{N}_0 : (u \Rightarrow^{i+1} v \text{ genau dann, wenn } \exists w \in V^* : u \Rightarrow w \Rightarrow^i v) \\ u \Rightarrow^* v \text{ genau dann, wenn } \exists i \in \mathbb{N}_0 : u \Rightarrow^i v \end{aligned}$$

Bei unserer Beispielgrammatik gilt zum Beispiel

$$X \Rightarrow \mathtt{a}X\mathtt{b} \Rightarrow \mathtt{aa}X\mathtt{bb} \Rightarrow \mathtt{aaa}X\mathtt{bbb} \Rightarrow \mathtt{aaabbb}$$

Also hat man z. B. die Beziehungen $X \Rightarrow^* \mathtt{aa}X\mathtt{bb}$, $\mathtt{a}X\mathtt{b} \Rightarrow^* \mathtt{aaa}X\mathtt{bbb}$, $X \Rightarrow^* \mathtt{aaabbb}$ und viele andere. Weiter geht es in obiger Ableitung nicht, weil überhaupt keine Nichtterminalsymbole mehr vorhanden sind.

Sozusagen das Hauptinteresse bei einer Grammatik besteht in der Frage, welche Wörter aus Terminalsymbolen aus dem Startsymbol abgeleitet werden können. Ist $G = (N, T, S, P)$, so ist die *von einer Grammatik erzeugte formale Sprache*

von einer Grammatik erzeugte formale Sprache

$$L(G) = \{w \in T^* \mid S \Rightarrow^* w\} .$$

Eine formale Sprache, die man mit einer kontextfreien Grammatik erzeugen kann, heißt auch *kontextfreie Sprache*.

kontextfreie Sprache

Was ist bei unserer Beispielgrammatik $G = (\{X\}, \{\mathtt{a}, \mathtt{b}\}, X, \{X \to \varepsilon, X \to \mathtt{a}X\mathtt{b}\}$? Eben haben wir gesehen: $\mathtt{aaabbb} \in L(G)$. Die Ableitung ist so strukturiert, dass man als Verallgemeinerung leicht sieht, dass für alle $i \in \mathbb{N}_0$ gilt: $\mathtt{a}^i\mathtt{b}^i \in L(G)$. Der Beweis (Sie wissen schon wie ...) wird leichter, wenn man mit der allgemeineren Behauptung beginnt:

$$\forall i \in \mathbb{N}_0 : (X \Rightarrow^* \mathtt{a}^i\mathtt{b}^i \wedge X \Rightarrow^* \mathtt{a}^i X \mathtt{b}^i)$$

Daraus folgt, dass $\{\mathtt{a}^i\mathtt{b}^i \mid i \in \mathbb{N}_0\} \subseteq L(G)$ ist. Umgekehrt kann man zeigen, dass gilt:

$$\forall i \in \mathbb{N}_0 : \text{ wenn } X \Rightarrow^{i+1} w, \text{ dann } w = \mathtt{a}^i\mathtt{b}^i \vee w = \mathtt{a}^{i+1}X\mathtt{b}^{i+1}$$

Daraus folgt $L(G) \subseteq \{\mathtt{a}^i\mathtt{b}^i \mid i \in \mathbb{N}_0\}$. Insgesamt ergibt sich also:

$$L(G) = \{\mathtt{a}^i\mathtt{b}^i \mid i \in \mathbb{N}_0\} .$$

Um eine etwas kompaktere Notation zu haben, fasst man manchmal mehrere Produktionen mit gleicher linker Seite zusammen, indem man das Nichtterminalsymbol und den Pfeil nur einmal hinschreibt, und die rechten Seiten durch senkrechte Striche getrennt aufführt. Für unsere Beispielgrammatik sieht die Produktionenmenge dann so aus:

$$P = \{\ X \to \texttt{a}X\texttt{b} \mid \varepsilon\ \}$$

Und damit können wir nun auch die richtige Interpretation für die Fragmente aus Tabelle 8.1 angeben: Es handelt sich um Produktionen einer kontextfreien Grammatik.

- Jedes der Wörter „Block“ ist *ein* Nichtterminalsymbol. Deswegen haben wir Schreibweisen wie $\langle \mathit{Block} \rangle$ vorgezogen, um deutlicher zu machen, dass wir so etwas als *ein* nicht weiter zerlegbares Objekt betrachten wollen.
- Der Doppelpunkt entspricht unserem Pfeil $\to$ und trennt linke Seite einer Produktion von rechten Seiten.
- Jede eingerückte Zeile ist eine rechte Seite einer Produktion.
- Aufeinander folgende Zeilen muss man sich durch senkrechte Striche | getrennt denken.

Der zweite Teil

2	BlockStatements: BlockStatement BlockStatements BlockStatement

der Tabelle aus dem ersten Unterabschnitt repräsentiert also z. B. die Produktionen

$$\begin{aligned}\langle \mathit{BlockStatements} \rangle \to\ & \langle \mathit{BlockStatement} \rangle \\ \mid\ & \langle \mathit{BlockStatements} \rangle \langle \mathit{BlockStatement} \rangle\end{aligned}$$

Und der erste Teil

1	Block: { BlockStatements$_{\text{opt}}$ }

der Tabelle repräsentiert die Produktionen

$$\langle \mathit{Block} \rangle \to \texttt{\{}\ \langle \mathit{BlockStatements} \rangle\ \texttt{\}} \mid \texttt{\{\ \}}$$

Wie man sieht, stehen (jedenfalls manche) Nichtterminalsymbole für strukturelle Konzepte der Programmiersprache. Im Idealfall wäre es dann so, dass man

mit der kontextfreie Grammatik für Java alle syntaktisch korrekten Javaprogramme ableiten kann, aber auch nur diese und nichts anderes. Die Realität ist etwas komplizierter: Was man mit der kontextfreie Grammatik nicht ableiten kann, ist bestimmt kein Javaprogramm. Aber man kann Dinge ableiten, die keine korrekten Programme sind (weil man manche Forderungen, wie z. B. „alle vorkommenden Variablen müssen vorher deklariert worden sein“, überhaupt nicht mit Hilfe kontextfreier Grammatiken ausdrücken kann).

Wenn man sich davon überzeugen will, dass ein Wort w aus dem Startsymbol ableitbar ist, ist das Aufschreiben einer langen Ableitungsfolge ist manchmal sehr mühsam aber nicht sonderlich erhellend. Die Grammatik für unser Klammerproblem ist ja sehr übersichtlich: $(\{X\}, \{(,)\}, X, \{X \to XX|(X)|\varepsilon\})$. Aber

$$\begin{aligned} X &\Rightarrow XX \Rightarrow (X)X \Rightarrow (X)XX \Rightarrow (X)X(X) \Rightarrow ((X))X(X) \\ &\Rightarrow ((X))X() \Rightarrow ((X))(X)() \Rightarrow (())(X)() \Rightarrow (())()() \end{aligned}$$

findet jedenfalls der Autor dieser Zeilen wenig hilfreich. Das kommt natürlich zum Teil daher, dass hier mal vorne und mal weiter hinten ein Ableitungsschritt gemacht wurde. Aber da bei kontextfreien Grammatiken eben Ersetzungen vom Kontext unabhängig sind, kann man umsortieren und zum Beispiel immer möglichst weit links abzuleiten. Dann erhält man eine sogenannte *Linksableitung*; in unserem Beispiel:

$$\begin{aligned} X &\Rightarrow XX \Rightarrow (X)X \Rightarrow ((X))X \Rightarrow (())X \Rightarrow (())XX \\ &\Rightarrow (())(X)X \Rightarrow (())()X \Rightarrow (())()(X) \Rightarrow (())()() \end{aligned}$$

Noch nützlicher ist es manchmal, stattdessen den zu dieser Ableitung gehörenden sogenannten *Ableitungsbaum* aufzumalen. Im vorliegenden Fall erhält man gerade die Darstellung aus Abbildung 8.2. Machen Sie sich den Zusammenhang klar! (An dieser Stelle verzichten wir noch bewusst darauf, Ableitungsbäumen und ihren Zusammenhang mit Ableitungen zu formalisieren, weil es unseres Erachtens mehr verwirrt als erleuchtet.)

Ableitungsbaum

Zum Abschluss dieses Unterabschnittes sei noch erwähnt, dass Kontextfreie Grammatiken auch *Typ-2-Grammatiken* heißen. Daneben gibt es auch noch:

Typ-2-Grammatiken

- Typ-3-Grammatiken: Auf sie werden wir später in der Vorlesung noch eingehen.
- Typ-1-Grammatiken und Typ-0-Grammatiken: Was es damit auf sich hat, werden Sie in anderen Vorlesungen kennenlernen.

8.3 RELATIONEN (TEIL 2)

Einige formale Aspekte dessen, was wir im vorangegangenen Unterabschnitt im Zusammenhang mit der Ableitungsrelation $\Rightarrow$ und mit $\Rightarrow^*$ getan haben, sind im Folgenden noch einmal allgemein aufgeschrieben, ergänzt um weitere Erläuterungen. Weil wir schon ausführliche Beispiele gesehen haben, und weil Sie nicht mehr ganz an Anfang des Semesters stehen und sich auch schon an einiges gewöhnt haben, beginnen wir, an manchen Stellen etwas kompakter zu schreiben. Das bedeutet insbesondere, dass Sie an einigen Stellen etwas mehr selbstständig mitdenken müssen. Tun Sie das! Und seien Sie ruhig sehr sorgfältig; mehr Übung im Umgang mit Formalismen kann wohl nicht schaden.

Sind $R \subseteq M_1 \times M_2$ und $S \subseteq M_2 \times M_3$ zwei Relationen, dann heißt

$$S \circ R = \{(x,z) \in M_1 \times M_3 \mid \exists y \in M_2 : (x,y) \in R \wedge (y,z) \in S\}$$

Produkt von Relationen

das *Produkt der Relationen* R und S. Machen Sie sich bitte klar, dass diese Schreibweise mit der Komposition von Funktionen kompatibel ist, die Sie kennen. Machen Sie sich bitte auch klar, dass das Relationenprodukt eine assoziative Operation ist.

Mit Id_M bezeichnen wir die Relation

$$\mathrm{Id}_M = \{(x,x) \mid x \in M\}$$

Wie man schnell sieht, ist das die identische Abbildung auf der Menge M. Deswegen macht man sich auch leicht klar, dass für jede binäre Relation R auf einer Menge M, also für $R \subseteq M \times M$ gilt:

$$R \circ \mathrm{Id}_M = R = \mathrm{Id}_M \circ R$$

Potenzen einer Relation

Ist $R \subseteq M \times M$ binäre Relation auf einer Menge M, dann definiert man *Potenzen* R^i wie folgt:

$$\begin{aligned} R^0 &= \mathrm{Id}_M \\ \forall i \in \mathbb{N}_0 : R^{i+1} &= R^i \circ R \end{aligned}$$

reflexiv-transitive Hülle

Die sogenannte *reflexiv-transitive Hülle* einer Relation R ist

$$R^* = \bigcup_{i=0}^{\infty} R^i$$

Die reflexiv-transitive Hülle R^* einer Relation R hat folgende Eigenschaften:

- R^* ist reflexiv. Was das bedeutet, werden wir gleich sehen.
- R^* ist transitiv. Was das bedeutet, werden wir gleich sehen.
- R^* ist die kleinste Relation, die R enthält und reflexiv und transitiv ist.

Eine Relation heißt *reflexiv*, wenn $\mathrm{Id}_M \subseteq R$ ist. *reflexive Relation*

Eine Relation heißt *transitiv*, wenn gilt: *transitive Relation*

$$\forall x \in M : \forall y \in M : \forall z \in M : xRy \wedge yRz \Longrightarrow xRz$$

Bitte bringen Sie den logischen Implikationspfeil $\Longrightarrow$ in dieser Formel nicht mit dem Ableitungspfeil $\Rightarrow$ durcheinander. Wir werden versuchen, die Pfeile immer unterschiedlich lang zu machen, aber das ist natürlich ein nur bedingt tauglicher Versuch besserer Lesbarkeit. Was mit einem Doppelpfeil gemeint ist, muss man immer aus dem aktuellen Kontext herleiten.

Dass R^* stets eine reflexive Relation ist, ergibt sich daraus, dass ja $\mathrm{Id}_M = R^0 \subseteq R^*$ ist.

Man kann sich überlegen, dass für alle $i, j \in \mathbb{N}_0$ gilt: $R^i \circ R^j = R^{i+j}$. Wenn man das weiß, dann kann man auch leicht beweisen, dass R^* stets eine transitive Relation ist. Denn sind $(x, y) \in R^*$ und $(y, z) \in R^*$, dann gibt es i und $j \in \mathbb{N}_0$ mit $(x, y) \in R^i$ und $(y, z) \in R^j$. Also ist dann $(x, z) \in R^i \circ R^j = R^{i+j} \subseteq R^*$.

Hinter der Formulierung, R^* sei die *kleinste* Relation, die R umfasst und reflexiv und transitiv ist, steckt folgende Beobachtung: Wenn S eine beliebige Relation ist, die reflexiv und transitiv ist und R umfasst, also $R \subseteq S$, dann ist sogar $R^* \subseteq S$.

8.4 EIN NACHTRAG ZU WÖRTERN

Gelegentlich ist es hilfreich, eine kompakte Notation für die Zahl der Vorkommen eines Zeichens $x \in A$ in einem Wort $w \in A^*$ zu haben. Wir definieren daher für alle Alphabete A und alle $x \in A$ Funktionen $N_x : A^* \to \mathbb{N}_0$, die wie folgt festgelegt sind:

$$N_x(\varepsilon) = 0$$

$$\forall y \in A : \forall w \in A^* : N_x(yw) = \begin{cases} 1 + N_x(w) & \text{falls } y = x \\ N_x(w) & \text{falls } y \neq x \end{cases}$$

Dann ist zum Beispiel

$$\begin{aligned} N_{\mathtt{a}}(\mathtt{abbab}) &= 1 + N_{\mathtt{a}}(\mathtt{bbab}) = 1 + N_{\mathtt{a}}(\mathtt{bab}) = 1 + N_{\mathtt{a}}(\mathtt{ab}) \\ &= 1 + 1 + N_{\mathtt{a}}(\mathtt{b}) = 1 + 1 + N_{\mathtt{a}}(\varepsilon) = 1 + 1 + 0 = 2 \end{aligned}$$

8.5 AUSBLICK

Es sollte klar sein, dass kontextfreie Grammatiken im Zusammenhang mit der Syntaxanalyse von Programmiersprachen eine wichtige Rolle spielen. Allgemein bieten Grammatiken eine Möglichkeit zu spezifizieren, wann etwas syntaktisch korrekt ist.

Die Relation $\Rightarrow^*$ ist ein klassisches Beispiel der reflexiv-transitiven Hülle einer Relation. Eine (etwas?) andere Interpretation von Relationen, Transitivität, usw. werden wir in der Einheit über sogenannte Graphen kennenlernen.