13 Vorlesung vom 23. Mai 2000

13.1 Grammatiken

13.1.1 Beispiel zu Grammatiken

Als Beispiel für eine Grammatik diene hier die Menge der booleschen Ausdrücke[9]. Das Startsymbol ist ein Nicht-Terminalsymbol, das wir „BOOL“ nennen. Als Terminalsymbole lassen wir alle Zeichen zu, die man gemeinhin für boolesche Ausdrücke verwendet. Die Produktionsregeln sind (13.1) zu entnehmen, und sollten jeden booleschen Ausdruck produzieren können:

$$
\begin{aligned}
&G := (V, \Sigma, P, S)\\
&V := \{\mathrm{BOOL}, \mathrm{QUANTOR}, \mathrm{a} \ldots \mathrm{z}, \wedge, \vee, \neg, (,)\}\\
&\Sigma := \{\mathrm{a} \ldots \mathrm{z}, \wedge, \vee, \neg, (,)\}\\
&P := \left\{\begin{array}{ll}
\mathrm{BOOL} & \to (\mathrm{BOOL}),\\
\mathrm{BOOL} & \to \neg\mathrm{BOOL},\\
\mathrm{BOOL} & \to \mathrm{BOOL} \wedge \mathrm{BOOL},\\
\mathrm{BOOL} & \to \mathrm{BOOL} \vee \mathrm{BOOL},\\
\mathrm{BOOL} & \to \mathrm{QUANTOR},\\
\mathrm{QUANTOR} & \to \mathrm{a} \ldots \mathrm{z}
\end{array}\right\}\\
&S := \mathrm{BOOL}
\end{aligned}
\tag{13.1}
$$

Der Ableitungsbaum eines beispielhaften booleschen Ausdruckes gemäß obiger Grammatik könnte wie folgt aussehen:

$$
\begin{aligned}
&\left(\underbrace{x}_{\mathrm{QUANTOR}} \vee \neg \underbrace{y}_{\mathrm{QUANTOR}}\right) \wedge \neg \left(\underbrace{a}_{\mathrm{QUANTOR}} \wedge \underbrace{b}_{\mathrm{QUANTOR}}\right)\\
&\left(\underbrace{\mathrm{QUANTOR}}_{\mathrm{BOOL}} \vee \neg \underbrace{\mathrm{QUANTOR}}_{\mathrm{BOOL}}\right) \wedge \neg \left(\underbrace{\mathrm{QUANTOR}}_{\mathrm{BOOL}} \wedge \underbrace{\mathrm{QUANTOR}}_{\mathrm{BOOL}}\right)\\
&\left(\mathrm{BOOL} \vee \underbrace{\neg\mathrm{BOOL}}_{\mathrm{BOOL}}\right) \wedge \neg \left(\underbrace{\mathrm{BOOL} \wedge \mathrm{BOOL}}_{\mathrm{BOOL}}\right)\\
&\left(\underbrace{\mathrm{BOOL} \vee \mathrm{BOOL}}_{\mathrm{BOOL}}\right) \wedge \neg \underbrace{(\mathrm{BOOL})}_{\mathrm{BOOL}}\\
&\underbrace{(\mathrm{BOOL})}_{\mathrm{BOOL}} \wedge \underbrace{\neg\mathrm{BOOL}}_{\mathrm{BOOL}}\\
&\underbrace{\mathrm{BOOL} \wedge \mathrm{BOOL}}_{\mathrm{BOOL}}\\
&\mathrm{BOOL}
\end{aligned}
\tag{13.2}
$$

Wie man sieht, beginnt die Transformation bei einem gewöhnlichen booleschen Ausdruck, und endet bei dem Startsymbol S[10].

13.2 Die Äquivalenz von regulären Grammatiken und endlichen Automaten

Satz:

Sei $L \subseteq \Sigma^*$ eine Sprache, dann gilt:

[9] Für die Korrektheit übernehme ich keine Haftung ;-)
[10] Die Begriffe „starten“ und „enden“ dürfen in diesem Zusammenhang auch vertauscht werden.

$$\begin{array}{lll} 1) & \exists \text{ ein endlicher Automat } M: & L = T(M) \\ & \Updownarrow & \\ 2) & \exists \text{ eine reguläre Grammatik } G: & L = L(G) \end{array} \tag{13.3}$$

Beweis:

$1 \Rightarrow 2$:

Gegeben sei der endliche Automat

$$M = (Q, \Sigma, \delta, q_0, F) \tag{13.4}$$

Wir definieren eine neue Grammatik, die dem endlichen Automaten *M* entsprechen soll. Diese neue Grammatik benutzt das Alphabet des Automaten als Terminalalphabet, und die Zustandsmenge des Automaten als Nicht-Terminalalphabet:

$$G' := (Q \cup \Sigma, \Sigma, P, q_0) \tag{13.5}$$

Es bleibt die Frage nach der Produktionsmenge *P*.

$$P := \left\{ q \to aq' \middle| \delta(q,a) = q' \right\} \cup \left\{ q \to e \middle| q \in F \right\} \tag{13.6}$$

Wir zeigen nun, dass die durch G' definierte Menge tatsächlich die Menge ist, die von M akzeptiert wird. Für jedes Wort, das der Automat *M* akzeptiert, gilt:

$$\begin{array}{l} x \in T(M) \\ \Updownarrow \\ p := \delta(q_0, x) \in F \\ \Updownarrow \quad x = a_1 a_2 \ldots a_{|x|} \\ p = \delta\left(q_0, a_1 a_2 \ldots a_{|x|}\right) \in F \\ \Updownarrow \\ p = \delta\left(\ldots \delta\left(\delta(q_0, a_1), a_2\right) \ldots, a_{|x|}\right) \in F \\ \Updownarrow \\ \left(\delta(p_0 := q_0, a_1) = p_1\right) \wedge \left(\delta(p_1, a_2) = p_2\right) \wedge \ldots \wedge \left(\delta\left(p_{|x|-1}, a_{|x|}\right) = p\right) \wedge (p \in F) \end{array} \tag{13.7}$$

Aus (13.6) folgt:

$$\forall p_n \in Q, a_s \in \Sigma: \quad \left(\delta(p_n, a_s) = p_m\right) \Leftrightarrow \left(p_n \overset{1}{\underset{G'}{\Rightarrow}} a_s p_m\right) \tag{13.8}$$

Aus (13.7) und (13.8) folgt:

$$x \in T(M)$$

$$\Updownarrow$$

$$\left(p_0 := q_0 \underset{G'}{\overset{1}{\Rightarrow}} a_1 p_1\right) \wedge \left(p_1 \underset{G'}{\overset{1}{\Rightarrow}} a_2 p_2\right) \wedge \ldots \wedge \left(p_{|x|-1} \underset{G'}{\overset{1}{\Rightarrow}} a_{|x|} p\right) \wedge \left(p = \delta(q_0, x) \in F\right)$$

$$\Updownarrow$$

$$q_0 \underset{G'}{\overset{*}{\Rightarrow}} a_1 a_2 \ldots a_{|x|} p \tag{13.9}$$

$$\Updownarrow \quad (p \in F) \Leftrightarrow \left(p \underset{G'}{\overset{1}{\Rightarrow}} \varepsilon\right)$$

$$q_0 \underset{G'}{\overset{*}{\Rightarrow}} a_1 a_2 \ldots a_{|x|}$$

$$\Updownarrow \quad a_1, a_2, \ldots, a_{|x|} \in \Sigma \text{ (Terminalsymbole)}$$

$$x \in L(G')$$

Die Mengen sind also identisch:

$$T(M) = L(G') \tag{13.10}$$

„$\Leftarrow$":

Gegeben sei die reguläre Grammatik

$$G = (V, \Sigma, P, S) \tag{13.11}$$

Wir definieren einen nichtdeterministischen, endlichen Automaten *M'*, welcher der regulären Grammatik entsprechen soll. Dieser neue Automat benutzt die Nicht-Terminalsymbole der Grammatik als Zustandsmenge, und die Terminalsymbole der Grammatik als Eingabealphabet. Wir definieren weiterhin den ausgezeichneten (akzeptierenden) Zustand f:

$$M' := \left((V - \Sigma) \cup \{f\}, \Sigma, \delta, S, F\right) \tag{13.12}$$

Die Menge der akzeptierenden Zustände *F* seien alle jenen Zustände (bzw. Nicht-Terminalsymbole), welche in das leere Wort übergehen können, sowie der ausgezeichnete Zustand *f*[11]:

$$F := \left\{A \middle| (A \in V - \Sigma) \wedge (A \to \varepsilon \in P)\right\} \cup \{f\} \tag{13.13}$$

Die Übergangsfunktion ist beim endlichen Automaten definiert auf dem Kreuzprodukt von Zustandsmenge und Eingabesymbol, d.h. in unserem Fall, auf dem Kreuzprodukt der Nicht-Terminalsymbole (plus „*f*") und der Terminalsymbole. Die Übergangsfunktion liefert beim endlichen Automaten eine Teilmenge der Zustandsmenge zurück, d.h. in unserem Fall eine Teilmenge der Nicht-Terminalsymbole.

Für jeden Zustand, der einem Nicht-Terminalsymbol entspricht, welches direkt in ein Terminalsymbol übergehen kann ($A \in V - \Sigma, a \in \Sigma : A \to a \in P$), soll der neue Automat (unter anderem) in den ausgezeichnete Zustand *f* übergehen:

$$\delta(A, a) := \left\{B \middle| A \to aB \in P\right\} \cup \left\{f \middle| A \to a \in P\right\} \tag{13.14}$$

[11] Dieser, ausgezeichnete Zustand *f* ist das wichtigste Element der Menge der akzeptierenden Zustände...

Wir zeigen nun, äquivalent zum ersten Teil des Beweises, dass die von M’ akzeptierte Menge tatsächlich die Menge ist, die durch G definiert wird[12]. Für jedes Wort, das unser neuer Automat *M’* akzeptiert, gilt:

$$x \in T(M')$$
$$\Updownarrow$$
$$\delta(S,x) \cap F \neq \emptyset \tag{13.15}$$
$$\Updownarrow$$
$$\vdots$$

Falls $\delta(S,x)$ einen Zustand *Q’* beinhaltet, der akzeptierend ist, so gilt gemäß (13.13):

$$\vdots$$
$$\Updownarrow$$
$$\delta(S,x) \cap F \in \left\{A \middle| (A \in V - \Sigma) \wedge (A \rightarrow \varepsilon \in P)\right\} \cup \{f\} \tag{13.16}$$
$$\Updownarrow$$
$$\vdots$$

Bis hierher haben wir Umformungen vorgenommen, die eine Aussage darüber machen, wann ein Wort vom neuen Automaten akzeptiert wird, und wann nicht; in Abhängigkeit von dem Wort. Wie man an (13.16) sieht, gibt es genau zwei Möglichkeiten, warum ein Wort von dem neuen Automaten akzeptiert werden kann – von diesen zwei Möglichkeiten muss mindestens eine Möglichkeit eintreten.

Die Definition des neuen Automaten und seiner Übergangsfunktion beruht auf einzelnen Symbolen, nicht auf Wörtern. Daher zerpflücken wir an dieser Stelle die Bearbeitung eines Wortes in die Bearbeitung vieler Symbole:

$$\delta(S,x) = \bigcup_{\substack{\cdot^{\cdot^{\cdot}} \\ Q_2 \in \bigcup\limits_{Q_1 \in \delta(S,a_1)} \delta(Q_1,a_2)}} \delta\left(Q_{|x|-1}, a_{|x|}\right) \tag{13.17}$$

Mit diesem Wissen lässt sich (13.16) umschreiben zu:

$$\vdots$$
$$\Updownarrow$$
$$\delta\left(Q_{|x|-1}, a_{|x|}\right) \cap F \in \left\{A \middle| (A \in V - \Sigma) \wedge (A \rightarrow \varepsilon \in P)\right\} \cup \{f\} \tag{13.18}$$
$$\Updownarrow$$
$$\vdots$$

Auch bis hierher haben wir „nichts anderes“, als eine Gleichung, die beschreibt, wann ein Eingabewort von unserem neuen Automaten akzeptiert wird, und wann nicht – jetzt aber nur noch in Abhängigkeit von dem letzten Symbol des Eingabewortes, sowie dem Zustand des Automaten zu diesem Zeitpunkt.

Wir haben den ausgezeichneten Zustand f genau so definiert, dass ein Übergang f ergibt, falls der ursprüngliche Zustand gemäß Grammatik in das Eingabesymbol übergehen kann (siehe (13.14)):

$$\left(f \in \delta(A,a)\right) \Rightarrow (A \rightarrow a \in P) \tag{13.19}$$

[12] Dieser Beweis wurde im Skript komplett weggelassen. Warum, ist mir nicht klar.

Mit diesem Wissen folgt:

$$
\begin{aligned}
&\vdots \\
&\Updownarrow \\
&\underbrace{\left(\cancel{\delta\left(Q_{|x|-1}, a_{|x|}\right) \cap F \in \left(A \in V - \Sigma\right)} \wedge Q_{|x|-1} \to \varepsilon \in P\right)}_{\text{Möglichkeit 1}} \vee \underbrace{\left(Q_{|x|-1} \to a_{|x|} \in P\right)}_{\text{Möglichkeit 2}} \\
&\Updownarrow \\
&\underbrace{\left(Q_{|x|-1} \to \varepsilon \in P\right)}_{\text{Möglichkeit 1}} \vee \underbrace{\left(Q_{|x|-1} \to a_{|x|} \in P\right)}_{\text{Möglichkeit 2}} \\
&\Updownarrow \\
&\vdots
\end{aligned}
\tag{13.20}
$$

Die obige, boolesche „Kürzung“ durften wir vornehmen, da diese Bedingung gemäß der Definition der Übergangsfunktion immer erfüllt ist. Es gilt nun, irgendwie den „Weg zurück“ zu finden vom Zustand $Q_{|x|-1}$ zum Zustand *S*. Aus (13.14) folgt:

$$
\begin{aligned}
&x \in T\left(M'\right) \\
&\Updownarrow \\
&\delta\left(S, a_1\right) \ni Q_1 \wedge \\
&\wedge\, \delta\left(Q_1, a_2\right) \ni Q_2 \wedge \\
&\wedge \ldots \wedge \\
&\wedge\, \delta\left(Q_{|x|-3}, a_{|x|-2}\right) \ni Q_{|x|-2} \wedge \\
&\wedge\, \delta\left(Q_{|x|-2}, a_{|x|-1}\right) \ni Q_{|x|-1} \\
&\Updownarrow \\
&\left(S \to a_1 Q_1 \in P\right) \wedge \\
&\wedge \left(Q_1 \to a_2 Q_2 \in P\right) \wedge \\
&\wedge \ldots \wedge \\
&\wedge \left(Q_{|x|-3} \to a_{|x|-2} Q_{|x|-2} \in P\right) \wedge \\
&\wedge \left(Q_{|x|-2} \to a_{|x|-1} Q_{|x|-1} \in P\right) \\
&\Updownarrow \\
&S \overset{*}{\underset{G}{\Rightarrow}} a_1 \ldots a_{|x|-1} Q_{|x|-1}
\end{aligned}
\tag{13.21}
$$

Oho. Wir wissen nun also, dass – falls ein Wort von unserem neuen Automaten akzeptiert wird, das zugehörige Nicht-Terminalsymbol Bestandteil der durch die Grammatik definierten Sprache ist. Weiterhin wissen wir aus (13.20), dass der „letzte Zustand" des Automaten in ein Terminalsymbol übergeht. Daraus folgt:

$$\begin{aligned}
& x \in T(M') \\
& \Updownarrow \\
& \left(S \overset{*}{\underset{G}{\Rightarrow}} a_1 \dots a_{|x|-1} Q_{|x|-1} \right) \wedge \left(\underbrace{\left(Q_{|x|-1} \to \varepsilon \in P \right)}_{\text{Möglichkeit 1}} \vee \underbrace{\left(Q_{|x|-1} \to a_{|x|} \in P \right)}_{\text{Möglichkeit 2}} \right) \\
& \Updownarrow \\
& \left(S \overset{*}{\underset{G}{\Rightarrow}} a_1 \dots a_{|x|-1} \right) \wedge \left(S \overset{*}{\underset{G}{\Rightarrow}} a_1 \dots a_{|x|-1} a_{|x|} \right) \\
& \Updownarrow \\
& S \overset{*}{\underset{G}{\Rightarrow}} x
\end{aligned} \tag{13.22}$$

Die Mengen sind also identisch:

$$T(M') = L(G) \tag{13.23}$$

13.3 Die Sprache „$a^n b^n$"

Satz:

Die Sprache $L := \left\{ a^n b^n \,\middle|\, n \geq 1 \right\}$ ist nicht regulär!

Beweis:

Wir führen einen Widerspruchsbeweis durch, und behaupten, *L* ist regulär. Dann existiert auch ein endlicher Automat *M*, welcher L akzeptiert:

$$\begin{aligned}
& L \in \mathrm{REG} \\
& \Downarrow \\
& \exists \mathrm{DFA}\, M : \qquad M = \left(Q, \Sigma, \delta, q_0, F \right) \\
& \qquad\qquad\qquad\quad L = T(M)
\end{aligned} \tag{13.24}$$

Wir gehen davon aus, dass dieser Automat existiert und funktioniert. Wir betrachten diesen Automaten nach der „Hälfte" der Ausführungszeit, d.h. zu dem Zeitpunkt, an dem er die *a*'s abgearbeitet hat, und die *b*'s noch nicht. Wir wählen ein Eingabewort, dass genau so viele *a*'s enthält, wie der Automat Zustände besitzt. Nach der Verarbeitung aller *a*'s befindet sich der Automat in einem Zustand, den wir *q* nennen:

$$\begin{aligned}
& n := |Q| \\
& \delta\left(q_0, a^n \right) = q
\end{aligned} \tag{13.25}$$

Der Automat ist in dem Startzustand q_0 gestartet, hat n Eingabezeichen verarbeitet, und hat demzufolge n mal seinen Zustand gewechselt. Außer dem Startzustand q_0 existieren noch $(n-1)$ weitere Zustände. Daraus folgt, dass der Automat mindestens einen Zustand mehr als einmal angenommen hat:

$$\begin{aligned}
&\exists q' \in Q, \quad i,j,k \in \square : \\
&(j \geq 1) \quad \wedge \quad (i+j+k=n) \quad \wedge \\
&\delta\left(q_0, a^i\right) = q' \quad \wedge \\
&\underbrace{\delta\left(q', a^j\right) = q'}_{\text{"Schleife"}} \quad \wedge \\
&\delta\left(q', a^k\right) = q
\end{aligned} \tag{13.26}$$

Für den Automaten ist es logischerweise „egal“, ob er die Schleife ausführt, oder nicht. Daraus folgt, dass der Automat zwischen den beiden Eingabewörtern a^i und a^{i+j} indifferent ist:

$$\delta\left(q_0, a^{i+j}\right) = \delta\left(\underbrace{\delta\left(q_0, a^i\right)}_{q'}, a^j\right) = \delta\left(q', a^j\right) = q' = \delta\left(q_0, a^i\right) \tag{13.27}$$

Das bedeutet ganz allgemein, dass man die Schleife auch weglassen kann, d.h. die a^j’s aus dem Eingabewort weglassen kann, ohne dass sich am Verhalten des Automaten irgendetwas ändert:

$$\begin{aligned}
\delta\left(q_0, a^{i+k} b^{i+j+k}\right) &= \delta\left(\delta\left(q_0, a^i\right), a^k b^{i+j+k}\right) = \\
&= \delta\left(\delta\left(q_0, a^{i+j}\right), a^k b^{i+j+k}\right) = \\
&= \delta\left(q_0, a^{i+j+k} b^{i+j+k}\right) = \\
&= \delta\left(q_0, a^n b^n\right)
\end{aligned} \tag{13.28}$$

Wir haben durch (13.24) definiert, dass Eingabewörter der Form $a^n b^n$ von dem Automaten akzeptiert werden, d.h. den Automaten vom Startzustand aus in einen akzeptierenden Zustand versetzen. Das gleiche Verhalten gilt demnach für Eingabewörter der Form $a^{i+k} b^{i+j+k}$, mit $j>0$:

$$\begin{aligned}
&a^{i+j+k} b^{i+j+k} \in T(M) \\
&\Downarrow \\
&\delta\left(q_0, a^{i+j+k} b^{i+j+k}\right) \in F \\
&\Downarrow \\
&\delta\left(q_0, a^{i+k} b^{i+j+k}\right) \in F \\
&\Downarrow \\
&a^{i+k} b^{i+j+k} \in T(M) \qquad \text{Widerspruch!} \\
&\Downarrow \\
&\mathrm{L} \notin \mathrm{REG}
\end{aligned} \tag{13.29}$$

Unser Automat akzeptiert demzufolge auch Eingabewörter der Form $a^{i+k} b^{i+j+k}$, welche aber nicht Bestandteil der gegebenen Sprache L sind. Die Sprache L ist nicht regulär!

13.4 Das Pumping-Lemma

Satz:

„Wenn man einen Automaten hat, und ein akzeptiertes Wort kennt, dessen Länge größer ist als die Anzahl der Zustände des Automaten, dann ist die durch den Automaten akzeptierte Sprache unendlich.“

Wir haben im vorangegangenen Beispiel gesehen, dass es einen Zusammenhang gibt, zwischen der Anzahl von Zuständen eines Automaten, der Länge eines Wortes der erzeugten Sprache, und der Anzahl der Zustandsschleifen während der Verarbeitung dieses Wortes.

Das Pumping-Lemma generalisiert diese Erkenntnis: Nach dem Pumping-Lemma gibt es für jede Sprache L eine bestimmte Wortlänge n, ab welcher Wörter, die in der Sprache enthalten sind, zwangsweise solche „Schleifen“ enthalten:

$$\begin{aligned}
&\forall L \in \mathrm{REG} \quad \exists n \geq 1 \quad \forall x \in L: \\
&\left(|x| \geq n\right) \Rightarrow \exists u,v,w \in \Sigma^* : 1)\ x = uvw \\
&\qquad\qquad\qquad\qquad 2)\ 0 < |v| < n \\
&\qquad\qquad\qquad\qquad 3)\ u \underbrace{v^i}_{\text{Schleife}} w \in L \quad \forall i \geq 0
\end{aligned} \tag{13.30}$$

Beweis:

Der Beweis verläuft ähnlich wie im vorangegangenen Beispiel. Wenn L regulär ist, existiert auch ein endlicher Automat M, welcher L akzeptiert:

$$\begin{aligned}
&L \in \mathrm{REG} \\
&\Downarrow \\
&\exists \mathrm{DFA}\, M: \qquad M = \left(Q, \Sigma, \delta, q_0, F\right) \\
&\qquad\qquad\qquad\quad L = T\left(M\right)
\end{aligned} \tag{13.31}$$

Wir dürfen davon ausgehen, dass dieser Automat existiert, und dass er eine endlich Zustandsmenge besitzt:

$$n := |Q| \tag{13.32}$$

Angenommen, es existiert ein Eingabewort, welches von dem Automaten akzeptiert wird, und welches genauso viele Buchstaben besitzt, wie der Automat Zustände hat. Dann startet der Automat in dem Startzustand q_0, verarbeitet n Eingabezeichen, und wechselt demzufolge n mal seinen Zustand. Außer dem Startzustand q_0 existieren noch *(n-1)* weitere Zustände. Daraus folgt, dass der Automat mindestens einen Zustand mehr als einmal annimmt:

$$\begin{aligned}
&\exists x \in L: \quad |x| \geq n \\
&\Downarrow \\
&\delta\left(q_0, x\right) \in F \\
&\Downarrow \\
&\exists q' \in Q, \quad u,w \in \Sigma^*, \quad v \in \Sigma^+ : \\
&x = uvw \quad \wedge \quad \delta\left(q_0, u\right) = q' \quad \wedge \quad \underbrace{\delta\left(q', v\right) = q'}_{\text{"Schleife"}} \quad \wedge \quad \delta\left(q', w\right) \in F
\end{aligned} \tag{13.33}$$

Für den Automaten ist es logischerweise „egal“, ob, und wie oft er die Schleife ausführt. Daraus folgt, dass der Automat zwischen den Eingabewörtern der Form wv^iw für $i>0$ indifferent ist:

$$\begin{aligned}
\delta\left(q_0, uv^iw\right) &= \delta\left(\delta\left(\delta\left(q_0,u\right), v^i\right), w\right) = \\
&= \delta\left(\underbrace{\delta\left(\underbrace{\delta\left(q_0,u\right)}_{q'}, v^i\right)}_{q'}, w\right) = \\
&= \delta\left(q', w\right) \in F
\end{aligned} \tag{13.34}$$

Daraus folgt:

$$uv^i w \in L \tag{13.35}$$

13.5 Entscheidbarkeit

13.5.1 Prädikate

Der Begriff des „Prädikates“ ist der Prädikatenlogik als Teilgebiet der mathematischen Logik entnommen. Ein Prädikat ist eine Aussage, die entweder wahr oder falsch ist. Ein Prädikat wird zumeist als mathematische Funktion definiert.

13.5.2 Definition der Entscheidbarkeit

Sei S eine Menge, und $p : S \to \{0,1\}$ ein Prädikat mit

$$\forall s \in S:\ p(s) := \begin{cases} 1 & \text{falls } p(s) \text{ wahr} \\ 0 & \text{falls } p(s) \text{ falsch} \end{cases} \tag{13.36}$$

Es gilt:

$$\begin{gathered} \text{Problem } \langle \text{S,p} \rangle \text{ ist entscheidbar} \\ \Updownarrow_{\text{Df.}} \\ \exists \text{ Algorithmus/Programm } A:\quad \forall s \in S:\quad A(s) = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} \Leftrightarrow p(s) \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} \end{gathered} \tag{13.37}$$

Entscheidbar sind z.B. folgende Fragen:

1. T(M)=$\varnothing$?
2. T(M) endlich?
3. T(M) unendlich?

Die Beweise dafür erbringen wir im Folgenden.

13.5.3 Beispiel: Entscheidbarkeit der leeren Sprache

Behauptung:

Die Frage, ob *T(M)=*$\varnothing$, ist entscheidbar.

Beweis:

Sei $M = (Q, \Sigma, \delta, q_0, F)$, $n := |Q|$. Es gilt:

$$T(M) = \varnothing \Leftrightarrow \left(\underbrace{\forall x \in \Sigma^* : |x| < n}_{\text{Entscheidbares Problem}} \Rightarrow x \notin T(M) \right) \tag{13.38}$$

Wieso?

„Hinrichtung“:

Diese Richtung ist trivial. Wenn der Automat überhaupt kein einziges Wort akzeptiert, dann erst recht keines, das eine bestimmte Länge hat.

Rückrichtung:

Hierfür bemühen wir einen Widerspruchsbeweis. Wir nehmen an, dass der Automat alle Wörter der beschriebenen Länge nicht akzeptiert, aber es trotzdem Wörter gibt, die der Automat existiert:

Annahme:

$$\left(\forall x \in \Sigma^* : |x| < n \Rightarrow x \notin T(M)\right) \wedge \left(T(M) \neq \emptyset\right) \\ \Downarrow \\ \exists y : \left(y \in T(M)\right) \wedge \left(|y| \geq n\right) \tag{13.39}$$

Wir betrachten nun das kürzeste dieser akzeptierten Wörter, und nennen es *x*:

$$x := z \in L \quad \left| \forall z' \in L \Rightarrow \left(|z'| \geq |z|\right)\right. \tag{13.40}$$

Gemäß Annahme (13.39) gilt für die Länge dieses Wortes:

$$|x| \geq n \tag{13.41}$$

Nach dem Pumping-Lemma gilt, dass dieses Wort in dem endlichen Automaten „Schleifen“ verursachen muss:

$$\exists x_1, x_2, x_3 : \left(x = x_1 x_2 x_3\right) \wedge \left(1 \leq |x_2| \leq n\right) \wedge \left(x_1 x_3 \in L\right) \\ \Downarrow \\ |x_1 x_3| < |x| \\ \Downarrow \\ |x_1 x_3| \geq n-1 \qquad \text{Widerspruch!} \tag{13.42}$$

Das bedeutet, dass es automatisch auch ein Wort geben müsste, welches ebenfalls akzeptiert wird, aber gegen die Annahme (13.39) verstößt. Also gilt (13.38), und damit gilt:

$$T(M) \overset{?}{\neq} \emptyset \text{ ist entscheidbar!} \tag{13.43}$$

13.5.4 Beispiel: Entscheidbarkeit der unendlichen Sprache

Behauptung:

Die Frage, ob *T(M)* unendlich ist, ist entscheidbar.

Beweis:

Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA, $n := |Q|$. Es gilt:

$$T(M) \text{ unendlich} \Leftrightarrow \exists x \in T(M) : \underbrace{\left(n \leq |x| \leq 2n-1\right)}_{\text{Entscheidbares Problem}} \tag{13.44}$$

Wieso?

Rückrichtung:

Wenn ein Wort von dem Automaten akzeptiert wird, und genauso viele oder mehr Buchstaben besitzt, wie der Automat Zustände, dann ist gemäß dem Pumping-Lemma eine Schleife in dem Automatenablauf, die „aufgeblasen werden kann“:

$$\left(x \in T(M)\right) \wedge \left(|x| \geq n\right) \\ \Downarrow \\ \exists u, v, w : \left(uvw = x\right) \wedge \left(\left\{uv^i w \middle| i \geq 0\right\} \subseteq T(M)\right) \\ \Downarrow \\ T(M) \text{ unendlich} \tag{13.45}$$

„Hinrichtung":

Wir gehen von der Menge aller Wörter aus, die von dem Automaten akzeptiert werden, und länger sind als *2n-1*. Falls diese Menge leer ist, bedeutet das, dass die erzeugte Sprache ohnehin nicht unendlich sein kann. Wir gehen davon aus, dass diese Menge nicht leer ist. Aus dieser Menge greifen wir uns dann eines der kürzesten Wörter heraus, und nennen es *y*:

$$\begin{aligned} &S := \left\{ x \middle| \left(x \in T(M) \right) \wedge \left(|x| \geq 2n \right) \right\} \\ &S \neq \emptyset! \end{aligned} \tag{13.46}$$

$$y := z \in S \,\middle|\, \forall z' \in S : |z'| \geq |z| \tag{13.47}$$

Da dieses Wort immer noch mehr Buchstaben besitzt, als der Automat Zustände, schlägt das Pumping-Lemma zu, und wir können eine „Schleife" aus diesem Wort entfernen:

$$\begin{aligned} &|y| \geq n \\ &\Downarrow_{\text{Pumping-Lemma}} \\ &\exists \text{u,v,w:}(\text{uvw=y}) \wedge \left(1 \leq |v| \leq n \right) \wedge \left(uw \in T(M) \right) \\ &\Downarrow_{1 \leq |v|} \\ &|uw| < |y| = |uvw| \\ &\Downarrow \\ &|uw| \leq 2n - 1 \end{aligned} \tag{13.48}$$

Damit haben wir schon die halbe Miete. Wir wissen nun, dass es automatisch ein akzeptiertes Wort geben muss, welches eine Länge kleiner oder gleich *2n-1* besitzt. Nun gilt es noch zu zeigen, dass dieses Wort auch eine Länge von größer oder gleich *n* besitzt:

$$\begin{aligned} &|uw| = |uvw| - |v| \geq \min\left(|uvw| \right) - \max\left(|v| \right) \\ &\Downarrow_{|v| \leq n, |uvw| \geq 2n} \\ &|uw| \geq 2n - n \\ &\Downarrow \\ &|uw| \geq n \end{aligned} \tag{13.49}$$

Damit ist der Beweis vollbracht. Es gilt also:

$$\left(uw \in T(M) \right) \wedge \left(n \leq |uw| \leq 2n - 1 \right) \tag{13.50}$$

Also gilt (13.44), und daraus folgt:

$$\text{"}T(M) \text{ unendlich?" ist entscheidbar!} \tag{13.51}$$

13.5.5 Beispiel: Entscheidbarkeit der endlichen Sprache

Behauptung:

Die Frage, ob *T(M)* endlich ist, ist entscheidbar.

Beweis:

$$T(M) \text{ endlich} \quad \Leftrightarrow \quad \neg\left(T(M) \text{ unendlich} \right) \tag{13.52}$$

Somit gilt:

$$\text{"}T(M) \text{ endlich?" ist entscheidbar!} \tag{13.53}$$

13.5.6 Beispiel: Entscheidbarkeit der Schnittmenge

Behauptung:

Die Frage, ob für zwei beliebige, endliche Automaten M_1 und M_2 $T(M_1) \cap T(M_2) \neq \emptyset$ gilt, ist entscheidbar.

Beweis:

Es ist bekannt, dass die Schnittmenge zweier regulärer Automaten (bzw. Mengen) wieder regulär ist:

$$\begin{aligned} &T(M_1), T(M_2) \in \text{REG} \\ &\Downarrow \\ &T(M) := T(M_1) \cap T(M_2) \in \text{REG} \end{aligned} \tag{13.54}$$

Wir haben bereits bewiesen, dass es für einen einzelnen Automaten entscheidbar ist, ob er eine leere Sprache akzeptiert, oder nicht:

$$T(M) \overset{?}{\neq} \emptyset \text{ ist entscheidbar!} \tag{13.55}$$

13.5.7 Beispiel: Entscheidbarkeit der Teilmenge

Behauptung:

Die Frage, ob für zwei beliebige, endliche Automaten M_1 und M_2 $T(M_1) \subseteq T(M_2)$ gilt, ist entscheidbar.

Beweis:

Wir führen den Beweis über eine weitere Behauptung, die wir zuerst beweisen:

Behauptung α:

$$\left(T(M_1) \subseteq T(M_2)\right) \Leftrightarrow \left(T(M_1) \cap \overline{T(M_2)} = \emptyset\right) \tag{13.56}$$

Beweis α:

„Hinrichtung“:

$$\begin{aligned} &\forall x : \left(x \in T(M_1)\right) \Rightarrow \left(x \in T(M_2)\right) \\ &\Downarrow \\ &\forall x : \left(x \notin T(M_2)\right) \Rightarrow \left(x \notin T(M_1)\right) \\ &\Downarrow \\ &\forall x : \left(x \in \overline{T(M_2)}\right) \Rightarrow \left(x \notin T(M_1)\right) \\ &\Downarrow \\ &T(M_1) \cap \overline{T(M_2)} = \emptyset \end{aligned} \tag{13.57}$$

Rückrichtung:

Beweis über Contraposition ($\neg \Rightarrow \neg$):

$$\begin{aligned}
&T(M_1) \not\subseteq T(M_2) \\
&\Downarrow \\
&\exists x : \big(x \in T(M_1)\big) \wedge \big(x \notin T(M_2)\big) \\
&\Downarrow \\
&\exists x : \big(x \in T(M_1)\big) \wedge \big(x \in \overline{T(M_2)}\big) \\
&\Downarrow \\
&T(M_1) \cap T(M_2) \neq \emptyset
\end{aligned} \tag{13.58}$$

Also gilt die Behauptung α! Weiter im Beweis...

Wir konstruieren uns gedanklich einen dritten Automaten:

$$\begin{aligned}
&T(M) := \overline{T(M_2)} \\
&\Downarrow
\end{aligned} \tag{13.59}$$

Unter Zuhilfenahme von (13.56) folgt daraus:

$$\big(T(M_1) \subseteq T(M_2)\big) \Leftrightarrow \neg\big(T(M_1) \cap T(M) \neq \emptyset\big) \tag{13.60}$$

Wir haben aber bereits in „13.5.6 Beispiel: Entscheidbarkeit der Schnittmenge“ gezeigt, dass die Frage, ob die Schnittmenge der durch zwei endliche Automaten definierten Sprachen nicht leer ist, entscheidbar ist. Damit ist der Beweis erbracht:

$$T(M_1) \overset{?}{\subseteq} T(M_2) \text{ ist entscheidbar!} \tag{13.61}$$

13.5.8 Beispiel: Entscheidbarkeit der Äquivalenz

Behauptung:

Die Frage, ob für zwei beliebige, endliche Automaten M_1 und M_2 $T(M_1) = T(M_2)$ gilt, ist entscheidbar.

Beweis α:

Wir können den Beweis nun auf bereits Bewiesenes zurückführen:

$$\begin{aligned}
&T(M_1) = T(M_2) \\
&\Updownarrow \\
&\big(T(M_1) \subseteq T(M_2)\big) \wedge \big(T(M_2) \subseteq T(M_1)\big)
\end{aligned} \tag{13.62}$$

Den Beweis, dass dieser Zusammenhang entscheidbar ist, haben wir in „13.5.7 Beispiel: Entscheidbarkeit der Teilmenge“ erbracht. Daraus folgt:

$$T(M_1) \overset{?}{=} T(M_2) \text{ ist entscheidbar!} \tag{13.63}$$

Beweis β:

Wir konstruieren uns zwei minimale Automaten, die äquivalent zu den beiden gegebenen Automaten sind. Dann gilt:

$$\begin{aligned}
&M_1' := \text{Nerodeautomat}(\mathrm{M}_1), M_2' := \text{Nerodeautomat}(\mathrm{M}_2) \\
&\Downarrow \\
&T(M_1) = T(M_2) \Leftrightarrow T(M_1') = T(M_2')
\end{aligned} \tag{13.64}$$

Vorausgesetzt, dass die beiden Automaten wirklich die selbe Sprache definieren, müssen die dazugehörigen Nerode-Automaten isomorph sein, d.h.:

$$\exists \text{ Isomorphismus } h: \quad h: M_1' \rightarrow M_2' \tag{13.65}$$

Es ist nun möglich, alle denkbaren Abbildungen von M_1' nach M_2' auszuprobieren, und festzustellen, ob eine davon ein Isomorphismus ist. Es kann kein Isomorphismus dabei sein, wenn die Automaten nicht identisch sind... Auch hieraus folgt:

$$T(M_1) \stackrel{?}{=} T(M_2) \text{ ist entscheidbar!} \tag{13.66}$$

13.6 Kontext-freie Grammatiken

Definition:

$$\begin{aligned}
&\text{Eine Grammatik } G = (V, \Sigma, P, S) \text{ heißt "kontextfrei"} \\
&\Updownarrow \quad \text{Df.} \\
&\forall p \in P: \quad p \in (V - \Sigma) \times V^*
\end{aligned} \tag{13.67}$$

Im Gegensatz zur normalen Grammatik stehen hier auf „der linken Seite“ der Produktionen also lediglich Nicht-Terminalsymbole, und auch immer nur eines davon. Bei der normalen Grammatik standen hier auch Terminalsymbole, und dazu noch beliebig viele (konkateniert).

13.6.1 Beispiel zur Kontext-freien Grammatik

Gegeben sei die Grammatik:

$$\begin{aligned}
&G := (V, \Sigma, P, S) \\
&V := \{S, a, b\} \\
&\Sigma := \{a, b\} \\
&P := \{S \rightarrow aSa, S \rightarrow bSb, S \rightarrow \varepsilon\}
\end{aligned} \tag{13.68}$$

Behauptung:

$$L(G) = \left\{ ww^R \,\middle|\, w \in \{a, b\}^* \right\} \tag{13.69}$$

„*R*“ bedeutet in diesem Zusammenhang das „umgedrehte Wort“. Die Grammatik beschreibt also *Palyndrome*.

Beweis:

Wir führen den Beweis mittels vollständiger Induktion über die Länge der Produktionen durch.

Induktionsbehauptung:

$$\forall n \geq 1, x \in \Sigma^*: \quad S \stackrel{n}{\Rightarrow} x \quad \Leftrightarrow \quad \left(x = wSw^R \wedge |w| = n\right) \vee \left(x = ww^R \wedge |w| = n-1\right) \tag{13.70}$$

Induktionsverankerung (n=1):

„Hinrichtung“:

$$
\begin{gathered}
S \overset{1}{\Rightarrow} x \\
\Downarrow \\
(x = aSa) \vee (x = bSb) \vee (x = \varepsilon) \\
\Downarrow \\
\left(x = wSw^R \mid |w| = 1\right) \vee \left(x = ww^R \mid |w| = 0\right)
\end{gathered}
\tag{13.71}
$$

Rückrichtung:

Fallunterscheidung, Fall 1:

$$
\begin{gathered}
\text{Sei } \left(x = wSw^R\right) \wedge \left(|w| = 1\right) \\
\Downarrow \\
(x = aSa) \vee (x = bSb) \\
\Downarrow \\
\left(S \overset{1}{\Rightarrow} aSa = x\right) \vee \left(S \overset{1}{\Rightarrow} bSb = x\right)
\end{gathered}
\tag{13.72}
$$

Fallunterscheidung, Fall 2:

$$
\begin{gathered}
\text{Sei } \left(x = ww^R\right) \wedge \left(|w| = 0\right) \\
\Downarrow \\
x = \varepsilon \\
\Downarrow \\
S \overset{1}{\Rightarrow} \varepsilon = x
\end{gathered}
\tag{13.73}
$$

Induktionsschritt (n→n+1):

„Hinrichtung“:

$$
\begin{gathered}
S \overset{n+1}{\Rightarrow} x \\
\Downarrow \\
S \overset{n}{\Rightarrow} x' \overset{1}{\Rightarrow} x \\
\Downarrow \\
\left(\left(x' = wSw^R\right) \wedge \left(|w| = n\right)\right) \vee \xcancel{\left(\left(x' = ww^R\right) \wedge \left(|w| = n-1\right)\right)} \\
\Downarrow \\
\vdots
\end{gathered}
\tag{13.74}
$$

Die „Kürzung" in dem obigen Ausdruck ist erlaubt, bzw. muss sogar sein, weil in diesem Fall kein Nicht-Terminalsymbol (d.h. *S*) mehr in dem Wort enthalten wäre. Es gäbe keine Produktion, die es erlauben würde, das Wort noch weiter zu verändern.

$$\vdots$$

$$\Downarrow$$

$$\left(x = waSaw^R\right) \vee \left(x = wbSbw^R\right) \vee \left(x = ww^R\right) \tag{13.75}$$

$$\Downarrow$$

$$\left(\left(x = w'Sw'^R\right) \wedge \left(|w| = n+1\right)\right) \vee \left(\left(x = ww^R\right) \wedge \left(|w| = n\right)\right)$$

Rückrichtung:

Fallunterscheidung, Fall 1:

Sei $\left(x = wSw^R\right) \wedge \left(|w| = n+1\right)$

$$\Downarrow$$

$$\left(\left(x = w'aSaw'^R\right) \vee \left(x = w'bSbw'^R\right)\right) \wedge \left(|w'| = n\right)$$

$$\Downarrow$$

$$S \overset{n}{\Rightarrow} w'Sw'^R \wedge \left(\left(w'Sw'^R \overset{1}{\Rightarrow} w'aSaw'^R\right) \vee \left(w'Sw'^R \overset{1}{\Rightarrow} w'bSbw'^R\right)\right) \tag{13.76}$$

$$\Downarrow$$

$$\left(S \overset{n+1}{\Rightarrow} w'aSaw'^R\right) \vee \left(S \overset{n+1}{\Rightarrow} w'bSbw'^R\right)$$

$$\Downarrow$$

$$\left(S \overset{n+1}{\Rightarrow} wSw^R\right)$$

Fallunterscheidung, Fall 2:

Sei $\left(x = ww^R\right) \wedge \left(|w| = n\right)$

$$\Downarrow$$

$$\left(S \overset{n}{\Rightarrow} wSw^R\right) \wedge \left(wSw^R \overset{1}{\Rightarrow} ww^R\right) \tag{13.77}$$

$$\Downarrow$$

$$\left(S \overset{n+1}{\Rightarrow} ww^R\right)$$

14 Vorlesung vom 25. Mai 2000

14.1 Ableitungen von Wörtern in kontextfreien Grammatiken

14.1.1 Ableitungen

Durch Ableitungen auf einer kontextfreien Grammatik formt man Nichtterminalsymbole in einem oder mehreren Ableitungsschritten auf Konkatenationen von Terminal- und Nichtterminalsymbolen um. Diesen Vorgang wollen wir näher betrachten.

Sei $G = (V, \Sigma, P, S)$ eine Grammtik, dann wir eine Sprache L(G), die von dieser Grammatik generiert wird. Hier und im Folgenden sei V die Menge der Nichtterminalsymbole und Σ die Menge der Terminalsymbole. Die Produktionen P dieser Grammatik wandeln (auch resubstitiuieren, ersetzen) Nichtterminalsymbole A in Folgen von Nichtterminal- und Terminalsymbolen $(V \cup \Sigma)^*$ um.

$$A \to X^* \qquad \text{mit } A \in V, X \in (V \cup \Sigma) \tag{14.1}$$

Eine Produktion aus P: $A \to \beta$ angewandt auf $\alpha A \gamma$ mit $\alpha, \gamma \in (V \cup \Sigma)^*$ liefert

$$\alpha A \gamma \Rightarrow \alpha \beta \gamma \tag{14.2}$$

Damit ist $\Rightarrow$ eine Relation auf der Menge der vereingten Terminal- und Nichtterminalsymbole $(V \cup \Sigma)^*$ und bezeichnet die Anwendung einer einzelnen Produktion. Die Anzahl der angewandten Produktionen, d. h. die Ableitungsschritte, kann über den Doppelpfeil geschrieben werden. Erfolgt die Ableitung der linken auf die rechte Seite in n Schritten, schreibt man entsprechend ein n über den Doppelpfeil.

Bildet man den transitiven Abschluß (transitive Hülle) der Relation, so bedeutet das die Ableitung über alle möglichen Schritte bis keine weiteren Ableitungen mehr möglich sind.

$$\begin{gathered} \alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots \Rightarrow \alpha_m \\ \alpha_1 \overset{*}{\Rightarrow} \alpha_m \end{gathered} \tag{14.3}$$

Reflexivität gilt, da dies dem Anwenden keiner Produktion auf ein Nichtterminalsymbol entspricht. Es gilt somit immer

$$\alpha \overset{*}{\Rightarrow} \alpha \tag{14.4}$$

Mit dieser Relation als Hilfsmittel zur Beschreibung von Ableitungen läßt sich die von der Grammatik G generierte Sprache L(G) beschreiben

$$L(G) = \left\{ x \middle| x \in \Sigma^* \wedge S \overset{*}{\Rightarrow} x \right\} \tag{14.5}$$

Die Sprache L(G) besteht aus Zeichenketten, die nur Terminalsymbole ($x \in \Sigma^*$) enthalten und durch beliebig viele Ableitungsschritte ausgehend vom Startsymbol S erzeugt ($S \overset{*}{\Rightarrow} x$) werden. Hierzu drei Anmerkungen

1. Eine Sprache ist *kontextfrei*, wenn sie von einer *kontextfreien* Grammatik erzeugt wird.
2. Während Zeichenketten bestehend aus Terminalsymbolen (Alphabet) Wörter der Sprache genannt werden, heißen die Zwischenschritte der Ableitung bestehend aus Nichtterminal- und - wenn nötig - Terminalsymbolen *Satzform*.
3. Zwei Grammatiken G_1, G_2 sind äquivalent, wenn sie dieselbe Sprache generieren ($L(G_1) = L(G_2)$).

14.1.2 Ableitungsbäume

Ableitungen lassen sich mit Ableitungsbäumen (auch Parse-Bäume genannt) visualisieren. Jede kontextfreie Grammatik kann mit Ableitungsbäumen beschrieben werden. Die Knoten solcher Bäume sind Terminal- und Nichtterminalsymbole. Die Wurzel ist immer ein Nichtterminalsymbol, während Terminalsymbole nur in Blättern auftreten. Betrachtet man den gesamten Ableitungsbaum, ist die Wurzel das Startsymbol der Grammatik.

Teilbäume eines Ableitungsbaumes sind Bäume mit einem Nichtterminalsymbol A als Wurzel und allen sich daraus ergebenden Kinderknoten. Die Anzahl der Knoten ist kleiner als die des gesamten zugrundliegenden Baumes, erseidenn der Teilbaum ist gleich dem Ableitungsbaum. Solche Teilbäume werden nach ihrer Wurzel genannt. Heißt die Wurzel also A spricht man von einem A-Baum.

Damit läßt sich jede Produktion einer Grammatik als Teilbaum beginnend mit der linken Seite als Wurzel und den 1 bis n Symbolen der rechten Seite als Kinderknoten darstellen. Die Symbole der rechten Seite seien abgekürzt mit $X_1, X_2, \ldots, X_m \quad mit \quad X_i \in (V \cup \Sigma)$

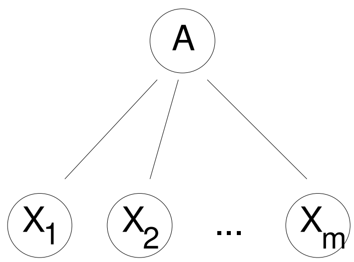

Abbildung 27 - Ableitungsbaum einer Produktion

Für einen Ableitungsbaum einer Grammatik $G = (V, \Sigma, P, S)$ gelten folgende Regeln

1. Jeder Knoten repräsentiert ein Symbol $X \in V \cup \Sigma \cup \{\varepsilon\}$.
2. Die Wurzel stellt das Startsymbol der Grammatik dar.
3. Alle inneren Knoten sind Nichtterminalsymbole, da kontexfreie Grammatiken nur solche auf der linken Seite der Produktionen stehen haben. Entsprechend sind alle Terminalsymbole Blätter, aber nicht notwendigerweise alle Blätter Terminalsymbole.
4. Wenn ein Knoten ein Nichtterminalsymbol A repräsentiert und die Kinderknoten X_1, X_2,...X_m hat, dann gibt es eine Produktion aus der Menge der Produktionen von G mit $A \to X_1 X_2 \ldots X_m$ (siehe Grafik vorangegangener Absatz).
5. Knoten, die das leere Wort ε repräsentieren, sind immer Blatt, da sie das einzige Kind eines Knoten mit Nichtterminalsymbol darstellen (sogenannte ε-Produktionen, wie wir später sehen werde). Aus dem leeren Wort können keine Abbildungen erfolgen und es kann ausschließlich auf das leere Wort und nicht auf Konkatenationen mit dem leeren Wort abgebildet werden, da sonst die restlichen Symbole der rechten Seite und nicht das leere Wort die Kinderknoten darstellen würden.

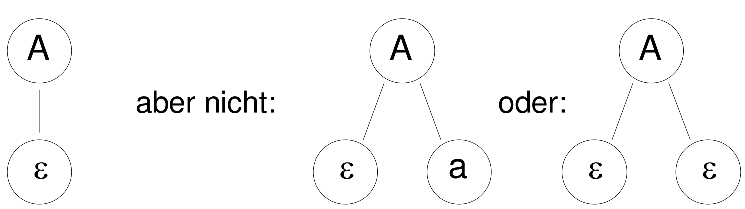

Abbildung 28 - richtige und falsche Ableitungsbäume von e-Produktionen

14.1.3 Beispiel für einen Ableitungsbaum

Gegeben sei eine Grammtik

$$\begin{aligned} G &= \left(\{S, A\}, \{a, b\}, P, S\right) \\ P &= \left\{\left(S \rightarrow aAS \mid a\right), \left(A \rightarrow SbA \mid ba \mid b\right)\right\} \end{aligned} \tag{14.6}$$

Das erste und letzte Zeichen der Wörter der zugehörigen Sprache sind also immer a. In der Mitte können b und a in mehrfacher Ausführung auftreten. Für das Wort

$$aabbaa \in L(G) \tag{14.7}$$

kann man folgende Ableitung mit dem entsprechenden Ableitungsbaum angeben.

$$S \rightarrow aAS \rightarrow aSbAS \rightarrow aabbaa \tag{14.8}$$

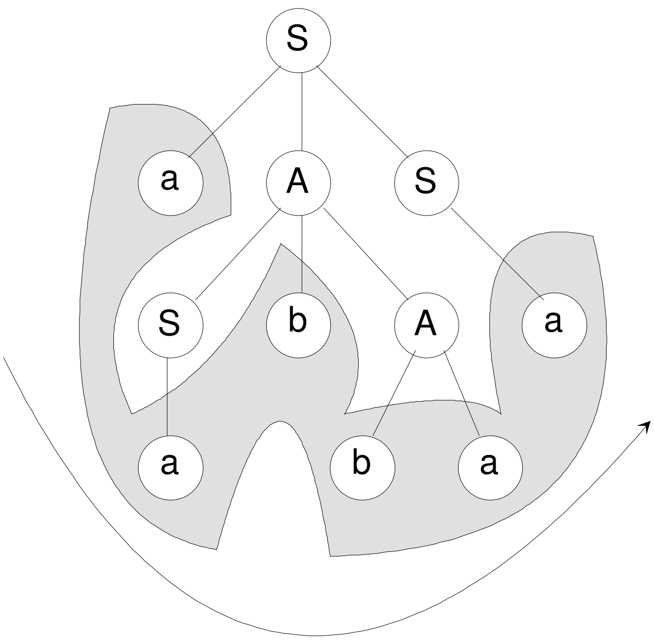

Abbildung 29 - Beispiel Ableitungsbaum für Wort aabbaa

Als „*yield*“ oder „*Front*“ bezeichnet man die Zeichenkette, die sich ergibt, wenn man die Blätter des Baumes perorder von links nach rechts abliest. In diesem Beispiel ist die Front das Wort aabbaa, es muß sich aber nicht ausschließlich um Terminalsymbole handeln, auch Nichtterminalsymbole sind je nach Grammatik möglich (bei Grammatiken, bei denen Blätter auch Nichtterminalsymbole sein können).

14.1.4 Die Beziehung zwischen Ableitungsbäumen und Grammatiken

Die Zusammenhänge zwischen Ableitungsbäumen und Ableitungen wurden im vorangegangenen Abschnitt nur exemplarisch und beschreibend dargelegt. Dies kann auch formal geschehen, wie im Folgenden gezeigt wird. Dazu wird wieder auf die Pfeilrelation zur Darstellung der Abbildungen und insbesondere auf die transitive Hülle dieser Relation zurückgegriffen.

Satz:

Für eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ gilt $S \overset{*}{\Rightarrow} \alpha$ genau dann, wenn ein Ableitungsbaum für G mit der Front (yield) α existiert.

$$\left(S \overset{*}{\Rightarrow} \alpha\right) \Leftrightarrow \exists \text{ Ableitungsbaum für } G \text{ mit Front } \alpha \tag{14.9}$$

Beweis:

Da eine ⇔ Beziehung vorliegt, muß in beide Richtungen bewiesen werden, was sich hier aber sehr ähnlich gestaltet. Zunächst die

„Hinrichtung“

Geht man von einem Baum aus und will damit die Ableitungsschritte zeigen, so beweist man per Induktion über die Anzahl der inneren Knoten. Anstatt des gesamten Ableitungsbaums vom Startsymbol S aus, wählen wir eine Ableitung der Form

$$A \overset{*}{\Rightarrow} \alpha \tag{14.10}$$

von einem beliebigen Nichtterminalsymbol A zur Front α aus.

Induktionsverankerung (Anzahl innerer Knoten = 1)

Es handelt sich um eine einzelne Produktion aus G. Aus dem inneren Knoten A folgen n verschiedene Terminal- und Nichtterminalsymbole $X_1,X_2,...,X_n$ als Front des Ableitungsbaumes. Die bereits weiter oben verwendete und hier nochmals abgebildete Grafik verdeutlich dies

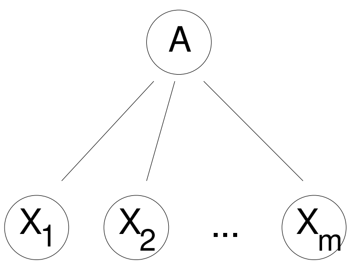

Abbildung 30 - Ableitung bei einem einzelnen inneren Knoten

Formal ausgedrückt ist der Satz für die Hinrichtung bei einem inneren Knoten bewiesen

$$\begin{aligned} &A \rightarrow X_1 X_2 \ldots X_n = \alpha \\ &A \overset{1}{\rightarrow} \alpha = A \overset{*}{\Rightarrow} \alpha \end{aligned} \tag{14.11}$$

Induktionsschritt:

Gelte die Behauptung nun für Teilbäume mit k - 1 inneren Knoten und sei α die Front eines A-Baums mit k inneren Knoten (k > 1). Die Kinderknoten eines solchen Wurzelementes A können nicht alle Blätter sein, da wegen k größer eins mindestens ein Kinderknoten innerer Knoten (bzw. Nichtterminalsymbol) ist.

Die Kinderknoten seien von 1 bis n durchnummeriert und mit $X_1,X_2,...,X_m$ bezeichnet. Dies entspricht wieder einer Produktion der Grammatik

$$A \rightarrow X_1 X_2 \ldots X_m \tag{14.12}$$

Wegen k größer eins ist eines der X_i Nichtterminalsymbol. Von diesen nichtterminalen X_i verzweigen die verbleibenden k - 1 inneren Knoten.

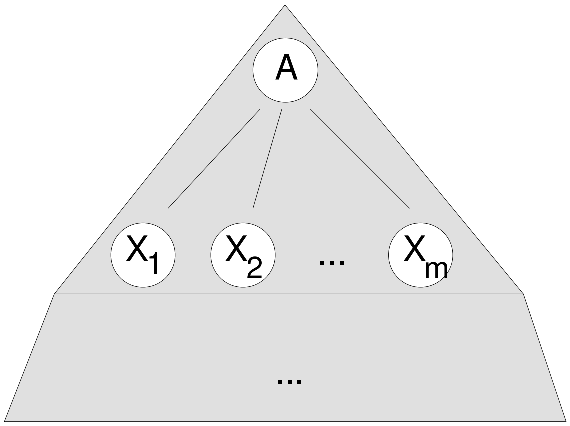

Abbildung 31 - Illustration eines A-Baum

Wenn X_i kein Blatt ist, dann ist X_i also Nichtterminalsymbol und Wurzel eines Unterbaums. Dieser X_i-Baum habe die Front α_i. Ist ein X_i allerdings Blatt und somit Nichtterminalsymbol, dann sei $X_i = \alpha_i$. Für j < i sind ein X_j sowie etwaige Kinderknoten in Front des A-Baums weiter links von einem X_i mit seinen Kinderknoten. Dann setzt sich die Front α des A-Baums folgendermaßen zusammen

$$\alpha = \alpha_2\alpha_1 \ldots \alpha_m \tag{14.13}$$

Die Grafik illustriert das mit der Darstellung von X_j- und X_i-Teilbäume nochmal

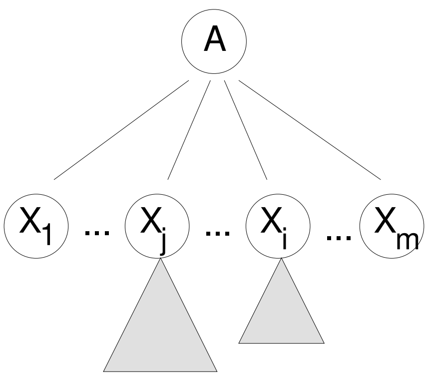

Abbildung 32 - A-Baum mit Kinderknoten und Xj-/Xi-Teilbäumen

Da A-Bäume k innere Knoten enthalten, ist die maximale Anzahl von inneren Knoten bei X_i-Bäumen entsprechend k - 1. Ausnahme ist natürlich der Fall, dass ein X_i-Baum genau der A-Baum ist, was aber hier unbeachtet bleiben soll. Gemäß Induktionsannahme, dass der Satz in Hinrichtung für Bäume mit inneren Knoten k < 1 bereits bewiesen sei, folgt

$$X_i \overset{*}{\Rightarrow} \alpha_i \qquad \forall i \mid X_i \neq Blatt \tag{14.14}$$

Faßt man diese Teilableitungen für jedes X_i zusammen, ergibt sich für die Front des A-Baums

$$A \overset{1}{\Rightarrow} X_1X_2 \ldots X_m \overset{*}{\Rightarrow} \alpha_1X_2 \ldots X_m \overset{*}{\Rightarrow} \alpha_1\alpha_2X_3 \ldots X_m \overset{*}{\Rightarrow} \ldots \overset{*}{\Rightarrow} \alpha_1\alpha_2 \ldots \alpha_m \tag{14.15}$$

Damit gilt $A \overset{*}{\Rightarrow} \alpha$. Wenn dies für alle Nichtterminalsymbole A gilt, dann auch für das Startsymbol S. Allerdings handelt es sich bei der durchgeführten Ab $G = (V, \Sigma, P, S)$ leitung nur um eine mögliche Ableitung des Ableitungsbaumes.

Die Rückrichtung des Beweises erfolgt analog. Hier wird von einem gegebenen $A \overset{*}{\Rightarrow} \alpha$ auf einen Ableitungsbaum geschlossen. Da sowohl Hin- als auch Rückrichtung in der Vorlesung ausgespart wurden, wird an dieser Stelle auf eine weitere Darlegung verzichtet, zumal die Ähnlichkeit von Rückrichtung und Hinrichtung sehr hoch ist. Stattdessen gehen wir lieber zum nächsten Thema.

14.1.5 Linksableitung

Werden in einer Ableitung die Produktionen immer zuerst auf die am weitesten links stehenden Nichtterminalsymbole angewandt, dann spricht man von einer Links- oder leftmost-Ableitung. Wendet man die Produktionen hingegen immer auf die rechte Seite an, so spricht man von Rechts- bzw. rightmost-Ableitungen.

Definition:

Für eine kontextfreie Grammatik ist eine Linksableitung (leftmost)

$$\begin{aligned} x \underset{lm}{\overset{1}{\Rightarrow}} y \quad \Leftrightarrow \quad & x = uA\alpha \quad \wedge \quad y = u\beta\alpha \\ & u \in \Sigma^*, a \in (V-\Sigma)^*, A \in V, (A \to \beta) \in P \end{aligned} \tag{14.16}$$

Die formale Definition sagt aus, dass in einem Ableitungsschritt (mit der Anwendung einer Produktion) immer das am weitesten links stehendene Terminalsymbol (das ist hier das A, gefolgt von einer belieben Kombination α aus Nichtterminal- und Terminalsymbolen) zuerst abgeleitet wird. Daraus folgen Ableitungen in n und beliebig vielen Schritten (transitive und reflexive Hülle).

$$\begin{aligned} & x \underset{lm}{\overset{n}{\Rightarrow}} y \quad \text{Ableitung in n Schritten} \\ & x \underset{lm}{\overset{*}{\Rightarrow}} y \quad \text{transitive Hülle} \\ & x \underset{lm}{\overset{+}{\Rightarrow}} y \quad \text{reflexive Hülle} \end{aligned} \tag{14.17}$$

14.1.6 Eindeutigkeit

Wenn es eine Ableitung

$$S \overset{1}{\Rightarrow} x_1 \overset{1}{\Rightarrow} x_2 \overset{1}{\Rightarrow} x_3 \overset{1}{\Rightarrow} \ldots \overset{1}{\Rightarrow} x_n \tag{14.18}$$

gibt, dann existiert genau ein entsprechender Ableitungsbaum. Existiert umgekehrt ein Ableitungsbaum B mit der Front α (wie in den vorangegangenen Ausführungen mehrfach aufgetreten), dann gibt es im allgemeinen mehrere Ableitungen, die diesem Baum entsprechen. Allerdings gibt es nur genau eine Linksableitung für diesen Ableitungsbaum B.

Damit lassen sich *Eindeutigkeit* und *Mehrdeutigkeit* von Grammatiken definieren.

Definition:

Für eine kontextfreie Grammatik G gilt

$$G \text{ eindeutig} \Leftrightarrow \forall x \in L(G): x \text{ hat nur einen Ableitungsbaum} \tag{14.19}$$

und analog

$$G \text{ eindeutig} \Leftrightarrow \forall x \in L(G): x \text{ hat nur eine Linksableitung} \tag{14.20}$$

Weiter ist eine kontextfreie Sprache L *eindeutig*, wenn es eine eindeutige kontextfreie Grammatik G gibt, für die L = L(G) gilt.

Umgekehrt heißt eine kontextfreie Sprache L inhärent mehrdeutig, wenn keine eindeutige kontextfreie Grammatik G mit L = L(G) existiert.

14.2 Vereinfachung kontextfreier Grammatiken

Grammatiken für kontextfreie Sprachen können überflüssige Produktionen und Nichtterminalsymbole enthalten. Mit verschiedenen Verfahren u. a. auch dem Umstellen auf Normalformen können Grammatiken auf einen einheitlichen und leichter zu untersuchenden Stand gebracht werden. In diesem Protokoll soll es um Entfernung von ε-Produktionen gehen. Das sind Produktionen, bei denen auf der rechten Seite das leere Wort steht.

14.2.1 ε-Produktionen

Solche Löschmöglichkeiten (auch Deletion genannt), sind für die Definition einer Sprache durch eine Grammatik nicht nötig. Eine Ausnahme muß gegeben sein, wenn die Sprache das leere Wort enthält. Dann kann das Startsymbol ins leere Wort übergehen. Enthält die Sprache das leere Wort nicht, so ist auch diese Produktion nicht nötig. Wir werden mit dieser Vorstellung zeigen, dass es zu beliebigen kontextfreien Sprachen L $L \subseteq \Sigma^*$ eine kontextfreie Grammatik gibt, die nur eine oder keine ε-Produktionen enthält.

Definition:

Zu einer kontextfreien Sprache $L \subseteq \Sigma^*$ existiert eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ mit den Eigenschaften

1. $L = L(G)$ (14.21)
2. $\forall A \in V - \{S\} : (A \to \varepsilon) \notin P$ (14.22)
3. $S \to \varepsilon \Leftrightarrow \varepsilon \in L$ (14.23)
4. S erscheint auf keiner rechten Seite einer Produktion (14.24)

Die Sprache L wird also von der Grammatik G erzeugt und verzichtet dabei auf ε-Produktionen. Lediglich der Übergang $S \to \varepsilon$ wird zugelassen, wenn das leere Wort Bestandteil von L ist.

Beweis:

Für jede kontextfreie Sprache L existiert eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ mit L = L(G').

Um auf die Sprache G zu gelangen, sind zwei Schritte zu gehen. Zunächst werden alle Nichtterminalsymbole aussortiert, die direkt oder über mehrere Produktionen zum leeren Wort ε führen.

$$\begin{aligned} &V_1 = \{A \mid (A \to \varepsilon) \in P'\} \\ &V_2 = \{A \mid (A \to \alpha) \in P' \wedge \alpha \in V_1^*\} \\ &\ldots \\ &V_n = \{A \mid (A \to \alpha) \in P' \wedge \alpha \in V_{n-1}^*\} \end{aligned} \tag{14.25}$$

Damit werden schrittweise alle Nichtterminalsymbole in V_i übernommen, die direkt oder über andere V_j mit j < i zum leeren Wort führen. Da die Anzahl der Symbole in V' nicht unendlich ist, sind nach N Läufen alle Nichtterminalsymbole gefunden, die direkt oder direkt aber nach maximal N Ableitungen zum leeren Wort kommen.

$$\begin{aligned} V_N &= V_N - 1 \\ V_{N+k} &= V_N \quad \forall k \end{aligned} \tag{14.26}$$

V_n sind dann die löschbaren Nichtterminalsymbole, d. h. Symbole, die mit endlich vielen Ableitungsschritten ins leere Wort ε überführt werden können. Das wird in bekannter Notation festgehalten:

$$A \overset{*}{\Rightarrow} \varepsilon \Leftrightarrow A \in V_N \tag{14.27}$$

Daraus folgt

$$\begin{aligned} &S \overset{*}{\Rightarrow} \varepsilon \Leftrightarrow S \in V_N \\ &\varepsilon \in L \Leftrightarrow S' \in V_N \end{aligned} \tag{14.28}$$

Nachdem die löschbaren Nichtterminalsymbole gefunden wurden, werden im zweiten Schritt die Produktionen auf Basis der verbliebenen Symbole umgebaut.

$$P_1 = P' \cup \left\{ \begin{aligned} &A \to x_1 Y_1 x_2 Y_2 \ldots x_n Y_n x_{n+1} \mid x_i \in ((V' - V_n) \cup \Sigma)^*, \\ &A \to x_1 A_1 x_2 A_2 \ldots x_n A_{n+1} \in P', Y_i \in \{A_i, \varepsilon\}, A_i \in V_n \end{aligned} \right\} \tag{14.29}$$

$$P_2 = P_1 - \{A \to \varepsilon \mid (A \to \varepsilon) \in P_1\} \tag{14.30}$$

$$P = P_2 \cup \{S \to S'\} \cup \{S' \to \varepsilon \mid \varepsilon \in L\} \tag{14.31}$$

Daraus wird anschaulich klar, dass die Sprache L mit der so entworfenen Grammatik $G = (V' \cup \{S\}, \Sigma, P, S)$ generiert werden kann und die Grammatik über die in der Definition vorgegebenen Eingenschaften verfügt.