Grundlagen der theoretischen Informatik

Kurt Sieber

Fakultät IV, Department ETI
Universität Siegen

SS 2013

Vorlesung vom 28.05.2013

Um zu testen, ob w in $L(G)$ liegt, brauchen wir also nur alle Ableitungen aus S bis zur Länge $2|w| - 1$ zu betrachten und nachzusehen, ob eine von ihnen das Wort w liefert.

2. Um zu testen, ob $L(G) = \emptyset$ ist, müssen wir systematisch nach einer Ableitung eines *beliebigen* Wortes $z \in \Sigma^*$ suchen.

   Auch hier stellt sich die Frage, ob man die Suche irgendwann abbrechen kann, d.h. ob man irgendwann sicher sein kann, dass kein Wort mehr gefunden wird.

   Wir zeigen dazu:

   > Wenn $G = (\Sigma, N, S, P)$ und $L(G) \neq \emptyset$, dann existiert ein Ableitungsbaum für ein Wort $z \in L(G)$ der höchstens die Höhe $|N|$ hat. (*)

   Um zu testen, ob $L(G) = \emptyset$ ist, brauchen wir also nur alle Ableitungsbäume (mit Wurzel S) bis zur Höhe $|N|$ zu erzeugen und nachzusehen, ob einer von ihnen ein Blattwort $z \in \Sigma^*$ hat.

Es bleibt (*) zu zeigen.

> Wenn $G = (\Sigma, N, S, P)$ und $L(G) \neq \emptyset$, dann existiert ein Ableitungsbaum für ein Wort $z \in L(G)$ der höchstens die Höhe $|N|$ hat. (*)

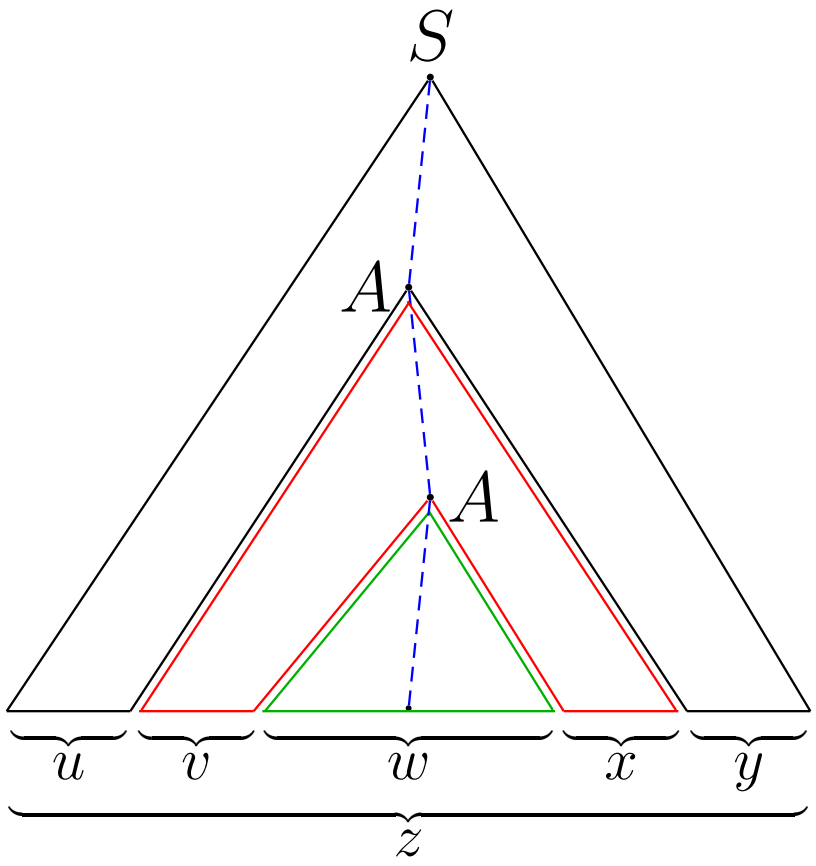
Wenn $L(G) \neq \emptyset$, so existiert zunächst *irgendein* Ableitungsbaum für ein Wort $z \in L(G)$.

Wenn dieser Ableitungsbaum noch nicht die gewünschte Eigenschaft hat, d.h. wenn seine Höhe größer als $|N|$ ist, so existiert ein Pfad von der Wurzel S zu einem Blatt, dessen Länge größer als $|N|$ ist.

Dieser Pfad enthält also mindestens $|N| + 2$ Knoten, von denen nur der letzte (das Blatt) mit einem Terminalzeichen markiert ist.

Also sind mindestens $|N| + 1$ Knoten mit Nichtterminalzeichen markiert, d.h. mindestens ein Nichtterminalzeichen A muss mehrmals auf diesem Pfad vorkommen.

Deshalb sieht der Ableitungsbaum für $S \stackrel{*}{\Rightarrow} z$ so aus:



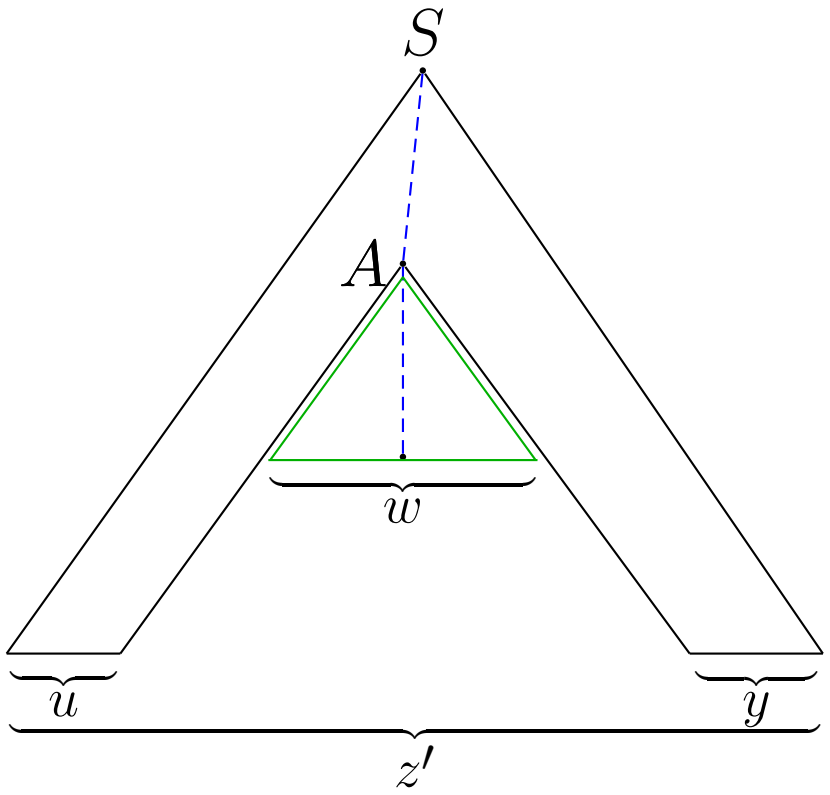
Auf dem *blauen Pfad* kommt A mindestens zweimal vor. u, v, x, y sind die Teilwörter von z, die links bzw. rechts des ersten bzw. zweiten A entstehen. w ist das Wort, das aus dem zweiten A entsteht. Es gilt also

$$\begin{aligned} S &\stackrel{*}{\Rightarrow} uAy \\ A &\stackrel{+}{\Rightarrow} vAx \\ A &\stackrel{*}{\Rightarrow} w \end{aligned}$$

und damit insgesamt

$$S \stackrel{*}{\Rightarrow} uAy \stackrel{+}{\Rightarrow} uvAxy \stackrel{*}{\Rightarrow} uvwxy = z$$

Jetzt entfernen wir das rote Stück aus dem Ableitungsbaum, und erhalten einen Ableitungsbaum für ein neues Wort z', aus dem mindestens eine Wiederholung eines Nichtterminalzeichens entfernt ist.



Eine zugehörige Ableitung ist $S \stackrel{*}{\Rightarrow} uAy \stackrel{*}{\Rightarrow} uwy = z'$.

Indem man dieses Verfahren wiederholt, erhält man irgendwann einen Ableitungsbaum, dessen Pfade *keine* Wiederholungen von Nichtterminalzeichen mehr enthalten.

Der hat dann höchstens die Höhe $|N|$. □

Kontextfreie Sprachen

Für einige andere Fragestellungen über kontextfreie Grammatiken gibt es *keine* Entscheidungsalgorithmen, z.B. für das *Äquivalenzproblem*:

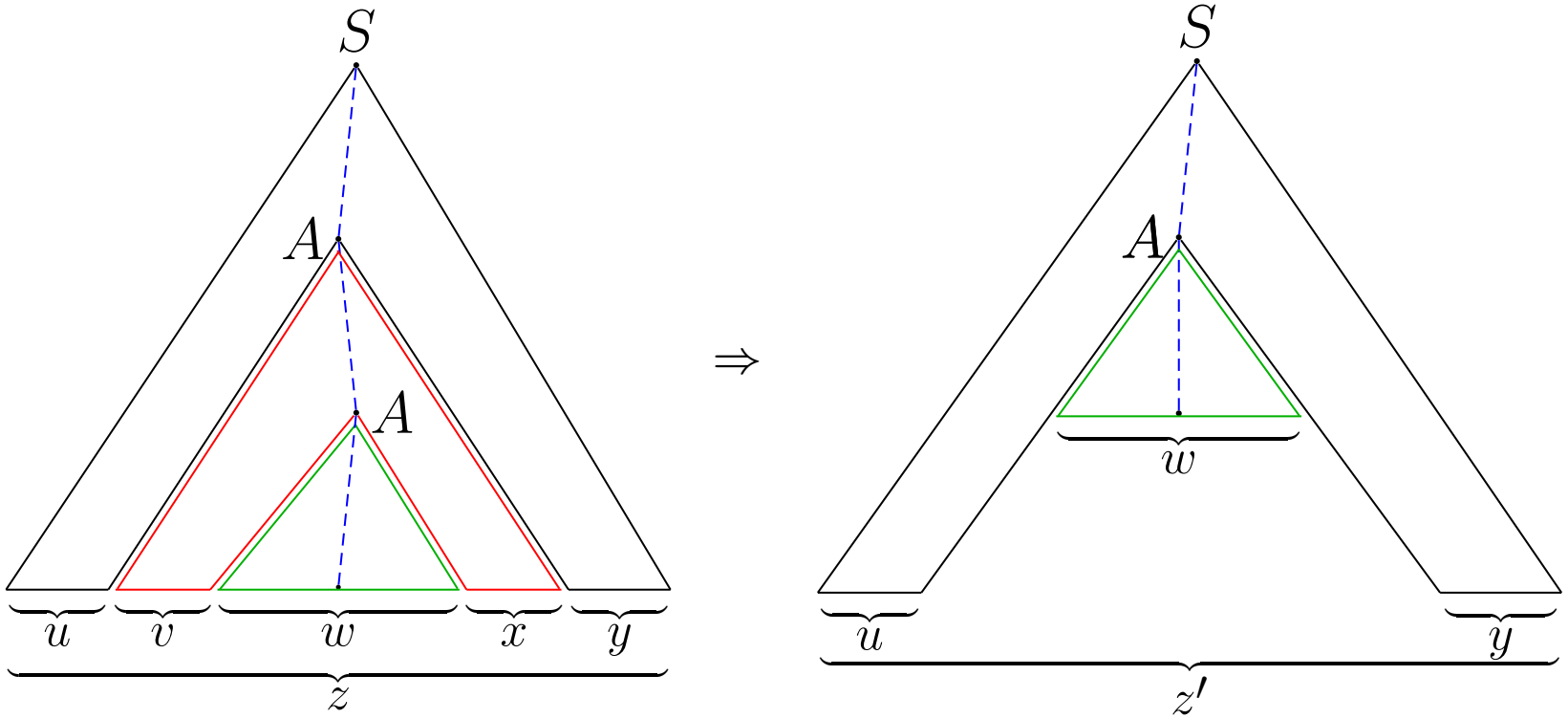
Eingabe: Zwei kontextfreie Grammatiken G_1 und G_2.
Frage: Ist $L(G_1) = L(G_2)$?

Es gibt *keinen* Algorithmus, der diese Frage stets korrekt beantwortet.

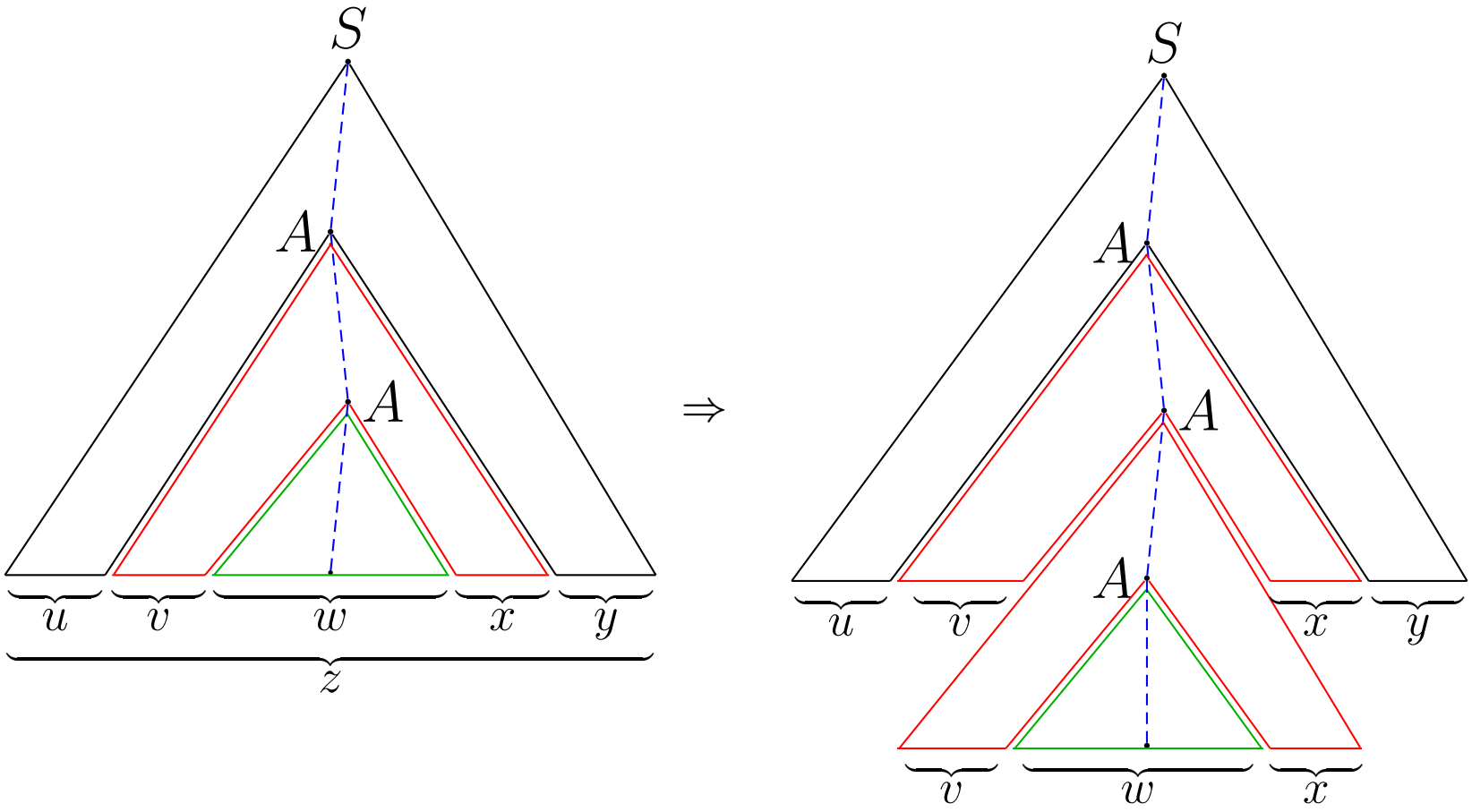
Solche Ergebnisse können wir zur Zeit noch nicht beweisen.
Sie gehören zur *Berechenbarkeitstheorie* (Teil II der Vorlesung).

Grenzen kontextfreier Sprachen

Im Beweis zu Satz 2.66 haben wir aus einem ‘hinreichend großen’ Ableitungsbaum ein Stück herausgeschnitten.



Stattdessen kann man das Stück auch wiederholen:



So erhält man aus der ursprünglichen Ableitung

$$S \overset{*}{\Rightarrow} uAy \overset{+}{\Rightarrow} uvAxy \overset{*}{\Rightarrow} uvwxy$$

neue Ableitungen der Form

$$S \overset{*}{\Rightarrow} uAy \overset{+}{\Rightarrow} \ldots \overset{+}{\Rightarrow} uv^iAx^iy \overset{*}{\Rightarrow} uv^iwx^iy$$

indem man i-mal die Ableitung $A \overset{+}{\Rightarrow} vAx$ wiederholt.

Das ist die Beweisidee für

Satz 2.67 (Pumping Lemma für kontextfreie Sprachen) *Sei $L \subseteq \Sigma^*$ kontextfrei. Dann existiert eine Zahl $n \geq 1$, so dass für jedes $z \in L$ mit $|z| \geq n$ gilt: Es gibt eine Zerlegung $z = uvwxy$ mit $vx \neq \varepsilon$ und $uv^iwx^iy \in L$ für alle $i \geq 0$.*

Beweis:

Sei $L = L(G)$, wobei $G = (\Sigma, N, S, P)$ eine Grammatik ist, die mit den Konstruktionen aus Satz 2.64 und Satz 2.65 entstanden ist, und sei $p = \max\{\,|\gamma| \mid (A \to \gamma) \in P\}$.

Dann hat das Blattwort eines Ableitungsbaums der Höhe m höchstens die Länge p^m (weil der Ableitungsbaum von einer Ebene zur nächsten höchstens um den *Faktor* p breiter werden kann).

Wir wählen $n = p^{|N|} + 1 > p^{|N|}$.

Dann hat jeder Ableitungsbaum für ein Wort z mit $|z| \geq n$ mindestens die Höhe $|N| + 1$, also ist die Ableitung für z von der Form

$$S \overset{*}{\Rightarrow} uAy \overset{+}{\Rightarrow} uvAxy \overset{*}{\Rightarrow} uvwxy$$

und wir erhalten daraus Ableitungen

$$S \overset{*}{\Rightarrow} uAy \overset{+}{\Rightarrow} \ldots \overset{+}{\Rightarrow} uv^iAx^iy \overset{*}{\Rightarrow} uv^iwx^iy$$

für alle Wörter uv^iwx^iy mit $i \geq 0$. Dabei gilt $vx \neq \varepsilon$, weil $A \overset{+}{\Rightarrow} \varepsilon A\varepsilon = A$ ohne Einheits- und ε-Produktionen nicht möglich ist. $\square$

Korollar 2.68 *Zum Nachweis, dass L nicht kontextfrei ist, genügt es zu zeigen: Für jedes $n \geq 1$ existiert ein $z \in L$ mit $|z| \geq n$, so dass für alle Zerlegungen $z = uvwxy$ mit $vx \neq \varepsilon$ gilt: Es gibt ein $i \geq 0$ mit $uv^iwx^iy \notin L$.*

Der Beweis, dass eine Sprache L nicht kontextfrei ist, läuft also ähnlich ab wie beim Pumping Lemma für reguläre Sprachen.

Für *jedes* $n \geq 1$ gibt man ein *passendes* Wort z mit $|z| \geq n$ an. Dann muss man für *jede* Zerlegung $z = uvwxy$ mit $vx \neq \varepsilon$ einen *passenden* Pumpfaktor i wählen mit $uv^iwx^iy \notin L$.

Merke:
z und i darf man *passend wählen*, aber man muss *alle* Zerlegungen des Wortes z betrachten. Dabei ist oft eine aufwändige Fallunterscheidung nötig, weil man *alle* Möglichkeiten für die Positionen der Teilwörter v und x im Wort z durchspielen muss.

Beispiel:

Die Sprache $L = \{a^n b^n c^n \mid n \geq 0\}$ ist nicht kontextfrei, denn:

Sei $n \geq 1$.

Wir wählen $z = a^n b^n c^n$. Dann ist $z \in L$ und $|z| \geq n$.

Sei nun $z = uvwxy$ eine beliebige Zerlegung von z mit $vx \neq \varepsilon$.

1. Fall: Wenn v oder x mehr als eines der Zeichen a, b, c enthält, dann ist $uv^2wx^2y \notin L$, weil in diesem Wort die Zeichen a, b, c nicht mehr in der richtigen Reihenfolge stehen.

2. Fall: Wenn $v = a^k$ und $x = b^l$, dann sind k und l nicht beide 0, also ist $uv^2wx^2y \notin L$, weil dieses Wort mehr as als cs oder mehr bs als cs (oder beides) enthält.

Alle übrigen Fälle (z.B. $v = a^k$ und $x = a^l$) verlaufen analog, weil mindestens eines der Zeichen a, b, c in vx fehlt. Beim Pumpen bleibt die Anzahl dieses Zeichens gleich, während sich die Anzahl eines der anderen beiden Zeichen erhöht.

Korollar 2.69 *Der Durchschnitt zweier kontextfreier Sprachen ist im allgemeinen nicht kontextfrei.*

Beweis:

Sei $L_1 = \{a^n b^n \mid n \geq 0\} \circ \{c\}^* = \{a^n b^n c^m \mid m, n \geq 0\}$

und $L_2 = \{a\}^* \circ \{b^n c^n \mid n \geq 0\} = \{a^m b^n c^n \mid m, n \geq 0\}$.

L_1 und L_2 sind kontextfrei, weil sie beide durch Konkatenation aus kontextfreien Sprachen entstehen,

aber $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 0\}$ ist *nicht* kontextfrei. □

Auch das Pumping Lemma für kontextfreie Sprachen lässt sich (wie das Pumping Lemma für reguläre Sprachen) so verstärken, dass es besser einsetzbar ist.

Satz 2.70 (Starkes Pumping Lemma für kontextfreie Sprachen)
Sei $L \subseteq \Sigma^*$ kontextfrei. Dann existiert eine Zahl $n \geq 1$, so dass für jedes $z \in L$ mit $|z| \geq n$ gilt: Es gibt eine Zerlegung $z = uvwxy$ mit $vx \neq \varepsilon$, $|vwx| \leq n$ und $uv^iwx^iy \in L$ für alle $i \geq 0$.

Beweis:

Um die Abschätzung $|vwx| \leq n$ zu erhalten, muss man im Beweis von Satz 2.67

- die Zahl n etwas größer wählen,

- und das Nichtterminalzeichen A sorgfältiger auswählen.

Sei $n = p^{|N|+1}$ und $z \in \Sigma^*$ mit $|z| \geq n$.

Dann wählen wir im Ableitungsbaum für z einen Pfad *maximaler Länge* (von der Wurzel S zu einem Blatt).

Da n sogar größer gewählt ist als im Beweis von Satz 2.67, muss auf diesem Pfad mindestens ein Nichtterminalzeichen mehrmals vorkommen.

Wir wählen das Nichtterminalzeichen A aus, das sich vom Blatt aus gesehen als *erstes* wiederholt.

Diese erste Wiederholung passiert (wenn man wieder vom Blatt ausgeht) nach spätestens $|N|+1$ Schritten, d.h. der Teilpfad vom Blatt zu diesem Zeichen A hat höchstens die Länge $|N|+1$.

Da der Pfad von S zum Blatt maximal gewählt war, kann es von diesem A aus keinen Pfad zu einem anderen Blatt geben, der länger ist als $|N|+1$.

Also hat das Wort vwx, das ja aus diesem Zeichen A entsteht, höchstens die Länge $p^{|N|+1} = n$. □

Beispiel:

Die Sprache $L = \{ww \mid w \in \{a,b\}^*\}$ ist nicht kontextfrei (im Gegensatz zur Sprache $L' = \{ww^R \mid w \in \{a,b\}^*\}$).

Sei $n \geq 1$. Wir wählen $z = a^n b^n a^n b^n$, also $z \in L$ und $|z| = 4n \geq n$.

Mit dem schwachen Pumping Lemma (Satz 2.67) kann man hier nicht argumentieren. Sei nämlich $z = uvwxy$ mit $u = \varepsilon$, $v = x = a^n$ und $w = y = b^n$. Dann gilt $uv^i wx^i y = a^{in} b^n a^{in} b^n \in L$ für *jedes* $i \geq 0$, d.h. für diese Zerlegung von z erhält man *keinen* Widerspruch.

Aber mit dem starken Pumping Lemma kommt man zum Ziel:

Wenn $z = uvwxy$ mit $|vwx| \leq n$ und $vx \neq \varepsilon$, so unterscheiden wir die folgenden drei Fälle:

1. vwx beginnt im linken a^n: Dann liegt vwx in der linken Worthälfte $a^n b^n$.
2. vwx beginnt im linken b^n: Dann liegt vwx im Mittelstück $b^n a^n$.
3. vwx liegt in der rechten Worthälfte $a^n b^n$.

In allen drei Fällen betrachten wir $z_0 = uv^0wx^0y$. Dieses Wort entsteht aus $z = uvwxy$, indem man v und x (also einen Teil von vwx) entfernt. Also hat z_0 in den drei oben genannten Fällen die Form $a^kb^la^nb^n$ oder $a^nb^ka^lb^n$ oder $a^nb^na^kb^l$, wobei $k < n$ oder $l < n$ (oder beides) gilt. All diese Wörter sind offensichtlich *nicht* von der Form ww, also gilt in jedem Fall $z_0 \notin L$, d.h. L ist nicht kontextfrei. □

Das Beispiel erklärt, warum (die meisten) Programmiersprachen nicht kontextfrei sind. Üblicherweise müssen die Programme nämlich sogenannte 'Kontextbedingungen' erfüllen, z.B.: "Jede Funktion muss deklariert sein, bevor sie aufgerufen wird."
Um eine solche Bedingung zu überprüfen, muss man den Funktionsnamen an der Deklarationsstelle mit dem Funktionsnamen an der Aufrufstelle vergleichen (so wie man in L die erste mit der zweiten Worthälfte vergleichen muss). Solche Vergleiche sind—wie wir an der Sprache L gesehen haben—mit einer kontextfreien Grammatik nicht möglich.

Kontextfreie Sprachen

Trotzdem spielen kontextfreie Grammatiken in der Praxis eine wichtige Rolle. Bei der Definition einer Programmiersprache L geht man nämlich so vor:

Zunächst definiert man die sogenannte 'kontextfreie Syntax' von L, d.h. man definiert eine kontextfreie Sprache $L' \supseteq L$. Dann definiert man L als die Menge aller Wörter aus L', die gewisse 'Kontextbedingungen' (wie die oben genannte Bedingung, dass jede Funktion deklariert sein muss, bevor man sie aufruft) erfüllen.

Ganz analog geht man bei der Implementierung der Programmiersprache vor: Durch die *syntaktische Analyse* (die vom *Parser* durchgeführt wird) wird überprüft, ob die vom Programmierer eingegebene Zeichenreihe der kontextfreien Syntax genügt, d.h. ob sie in L' liegt. Wenn ja, so erzeugt der Parser den Syntaxbaum für die Zeichenreihe. Bei der anschließenden *semantischen Analyse* werden dann am Syntaxbaum die *Kontextbedingungen* überprüft.

Weitere Beispiele für die Anwendung des Pumping Lemmas:

Sei $\Sigma = \{a\}$. Dann sind die früher betrachteten Sprachen

- $\{a^{n^2} \mid n \geq 0\}$
- $\{a^{2^n} \mid n \geq 0\}$
- $\{a^p \mid p \text{ Primzahl}\}$

nicht kontextfrei.

Beweis:
Mit Hilfe des Pumping Lemmas für reguläre Sprachen haben wir bewiesen, dass diese Sprachen nicht regulär sind. Da es bei einem einelementigen Alphabet keinen Unterschied macht, ob man an *einer* oder an *zwei* Stellen pumpt, lassen sich diese Beweise leicht so abändern, dass man in ihnen das Pumping Lemma für kontextfreie Sprachen benutzt. Also sind diese Sprachen auch nicht kontextfrei. □

In der Tat gilt über jedem *einelementigen* Alphabet:

$$\mathcal{L}_{reg} = \mathcal{L}_{kf}$$

Das lässt sich aber nicht mit dem Pumping Lemma beweisen, denn in beiden Fällen (sowohl bei regulären, als auch bei kontextfreien Sprachen) gibt das Pumping Lemma nur eine notwendige und keine hinreichende Bedingung an.

Wir verzichten auf den Beweis dieser Gleichheit (und verwenden sie auch nicht).

Über jedem Alphabet mit mindestens zwei Zeichen gilt natürlich

$$\mathcal{L}_{reg} \subsetneq \mathcal{L}_{kf}$$

wie wir bereits bewiesen haben.

Deterministisch kontextfreie Sprachen

Die syntaktische Analyse von Programmen wird (im Prinzip) mit Kellerautomaten durchgeführt. Wir haben bereits gesehen, wie man aus einer kontextfreien Grammatik G einen Kellerautomaten M mit $L(M) = L(G)$ erhält, aber dieser Kellerautomat war (in hohem Maße) nichtdeterministisch. Für die Praxis benötigt man natürlich "deterministische" Kellerautomaten. Dieser Begriff muss nun erst einmal definiert werden. Es genügt sicherlich *nicht*, zu fordern, dass die Übergangsrelation Δ des Kellerautomaten M eine (partielle) Funktion ist. Man betrachte z.B.

$$\Delta = \{((p, a, \varepsilon), (p, \varepsilon)), ((p, \varepsilon, b), (p, \varepsilon)\}$$

Diese Relation Δ ist eine partielle Funktion, aber der zugehörige Kellerautomat hat trotzdem die Auswahl zwischen zwei Transitionen, wenn die aktuelle Eingabe mit a beginnt und der aktuelle Kellerinhalt mit b.

Solche "Konflikte" zwischen zwei Transitionen sollten in einem deterministischen Kellerautomaten ausgeschlossen sein. Das erreicht man durch die folgende

Definition 2.71 *Sei $M = (\Sigma, \Gamma, Q, s, F, \Delta)$ ein Kellerautomat.*

1. *Zwei Wörter α_1, α_2 heißen* konsistent*, wenn α_1 Präfix von α_2 oder α_2 Präfix von α_1 ist.*
2. *Zwei Transitionen $((p_1, u_1, \beta_1), (q_1, \gamma_1))$ und $((p_2, u_2, \beta_2), (q_2, \gamma_2))$ heißen* kompatibel*, wenn gilt:*

   - $p_1 = p_2$,
   - *u_1, u_2 sind konsistent und*
   - *β_1, β_2 sind konsistent*

   Ansonsten heißen sie inkompatibel*.*
3. *M heißt* deterministisch*, wenn die Transitionen in Δ paarweise inkompatibel sind.*

Kontextfreie Sprachen

Man beachte, dass nur kompatible Transitionen $((p_1, u_1, \beta_1), (q_1, \gamma_1))$ und $((p_2, u_2, \beta_2), (q_2, \gamma_2))$ auf die gleiche Konfiguration (p, u, β) anwendbar sein können, denn dazu muss gelten:

- $p_1 = p_2 = p$,
- u_1 und u_2 konsistent, weil beides Präfixe von u sind,
- β_1 und β_2 konsistent, weil beides Präfixe von β sind.

Also sind zwei Transitionen eines deterministischen Kellerautomaten niemals auf die gleiche Konfiguration anwendbar, d.h. für einen deterministischen Kellerautomaten M ist die Relation $\vdash_M$ eine partielle Funktion.

Im Prinzip definieren wir jetzt die deterministisch kontextfreien Sprachen als diejenigen, die von deterministischen Kellerautomaten erkannt werden, wobei wir den Automaten aber noch die Möglichkeit geben, das Ende des Eingabeworts zu erkennen.

Definition 2.72 *Eine Sprache $L \subseteq \Sigma^*$ heißt* deterministisch kontextfrei*, wenn es einen deterministischen Kellerautomaten M gibt, der die Sprache*

$$L\$ = \{w\$ \mid w \in L\}$$

erkennt, wobei \$ ein neues Zeichen ist, d.h. $\$ \notin \Sigma$.

(In der Praxis, d.h. bei einem Parser, erkennt man ebenfalls das Ende der Eingabe, z.B. indem man auf 'end of file' testet.)

Beispiel:

Die Sprache $L = \{a^n b^n \mid n \geq 0\}$ ist deterministisch kontextfrei.

Ein deterministischer Kellerautomat, der $L\$$ erkennt, arbeitet wie folgt:

Wenn er im Startzustand schon das Zeichen \$ liest, geht er sofort in den Endzustand.

Wenn er im Startzustand ein a liest, geht er in einen Zustand q_a, in dem weitere as gelesen werden können. Diese as werden in den Keller verschoben.

Sobald er das erste b liest, wechselt er in einen Zustand q_b, in dem nur noch bs (oder \$) gelesen werden dürfen.

Die bs werden gegen die as im Keller aufgehoben.

Sobald \$ erscheint, wechselt der Automat in den Endzustand.

Wenn gleich viele as und bs gelesen wurden, ist der Keller am Ende leer, also wird das Wort akzeptiert.

Andernfalls bleibt der Automat entweder stecken (d.h. das Eingabeband ist nicht leer) oder der Keller ist am Ende nicht leer, also wird das Wort nicht akzeptiert.

Formale Definition dieses Kellerautomaten:

$M = (\{a, b, \$\}, \{a\}, \{s, q_a, q_b, f\}, s, \{f\}, \Delta)$ mit:

$$
\begin{aligned}
\Delta = \{\ & ((s, \$, \varepsilon), (f, \varepsilon)), && (1)\\
& ((s, a, \varepsilon), (q_a, \varepsilon)), && (2)\\
& ((q_a, a, \varepsilon), (q_a, a)), && (3)\\
& ((q_a, b, \varepsilon), (q_b, \varepsilon)), && (4)\\
& ((q_b, b, a), (q_b, \varepsilon)), && (5)\\
& ((q_b, \$, \varepsilon), (f, \varepsilon))\ \} && (6)
\end{aligned}
$$

Kontextfreie Sprachen

Wir zeigen, dass tatsächlich $L(M) = L\$ = \{a^n b^n \$ \mid n \geq 0\}$ gilt.

'$\supseteq$': Sei $w = a^n b^n \$$.

Im Falle $n = 0$ gilt $(s, w, \varepsilon) = (s, \$, \varepsilon) \vdash_{(1)} (f, \varepsilon, \varepsilon)$, also $w \in L(M)$.

Im Falle $n > 0$ gilt:

$$
\begin{aligned}
(s, w, \varepsilon) \;=\; (s, a^n b^n \$, \varepsilon) \;&\vdash_{(2)} \; (q_a, a^{n-1} b^n \$, \varepsilon) \\
&\vdash_{(3)}^{n-1} \; (q_a, b^n \$, a^{n-1}) \\
&\vdash_{(4)} \; (q_b, b^{n-1} \$, a^{n-1}) \\
&\vdash_{(5)}^{n-1} \; (q_b, \$, \varepsilon) \\
&\vdash_{(6)} \; (f, \varepsilon, \varepsilon)
\end{aligned}
$$

'$\subseteq$': Sei $w \in L(M)$, d.h. $(s, w, \varepsilon) \vdash_M^* (f, \varepsilon, \varepsilon)$.

Wie kann diese Folge von Übergangsschritten aussehen?