Eine kontextfreie Grammatik ist in Greibachnormalform, wenn die rechte Seite einer jeden Regel mit einem Terminalsymbol beginnt.

Theorem 1 *Für jede kontextfreie Sprache L mit $\varepsilon \notin L$ gibt es eine Grammatik in Greibachnormalform.*

Wir brauchen den Begriff der links-rekursiven Variable. Eine Variable A heißt *links-rekursiv*, wenn es eine A-Regel gibt, deren rechte Seite mit A beginnt. Eine A-Regel ist eine Regel $A \to \alpha$.

Sei G eine Grammatik in Chomskynormalform für L. Betrachte folgenden Graph (wir nennen ihn Abhängigkeitsgraph): Die Knoten sind die Variablen der Grammatik und wir haben eine Kante von A nach B, wenn es eine A-Regel gibt, deren rechte Seite mit B beginnt. Wir betrachten zunächst einen einfachen Fall.

Der azyklische Fall: Der Abhängigkeitsgraph ist azyklisch, d.h., wir können die Variablen als $A_1, A_2, \ldots, A_m$ durchnummerieren, so dass es nur Kanten (A_i, A_j) mit $i < j$ gibt.

Für A_m bedeutet das, dass alle rechten Seiten mit einem Terminal beginnen müssen. Für A_{m-1} beginnen sie entweder mit einem Terminal oder mit A_m. Wir können den letzteren Fall eliminieren durch Anwendung einer Operation $subst(A_{m-1}, A_m)$. Dabei ist:

$subst(A,B)$: Die Operation ist nur anwendbar, wenn B nicht links-rekursiv ist. Seien $B \to \beta_1, \ldots, B \to \beta_k$ die B-Regeln. Für jede Regel $A \to B\alpha$ tue: ersetze die Regel durch die Regeln $A \to \beta_i\alpha$, $1 \le i \le k$. Die Voraussetzung, dass B nicht links-rekursiv ist garantiert, dass keine neuen Regeln erzeugt werden, die mit B beginnen und so der Prozeß unendlich wird.

Für A_{m-2} ersetzen wir durch $subst(A_{m-2}, A_{m-1})$ zunächst die führenden Vorkommen von A_{m-1} und dann durch $subst(A_{m-2}, A_m)$ die führenden Vorkommen von A_m auf der rechten Seite von A_{m-2}-Regeln. Allgemein erhalten wir im azyklischen Fall die Greibachnormalform durch:

for $i = m-1$ downto 1 **do**
  for $j = i+1$ to m **do**
    $subst(A_i, A_j)$;
  end for
end for

Der allgemeine Fall: Wir werden den allgemeinen Fall auf den azyklischen Fall zurückführen. Wir starten wieder mit einer Grammatik in Chomsky Normalform und nummerieren die Variablen beliebig durch, etwas A_1 bis A_m. Im Laufe der Konstruktion, werden wir für jede Variable A_i eine neue Variable B_i zu unserer Grammatik hinzufügen. Wir nennen die Variablen $\{A_1, \ldots, A_m\}$ die A-Variablen und die Variablen $\{B_1, \ldots, B_m\}$ die B-Variablen. Wir brauchen eine weitere Technik zur Umformung von Grammatiken.

eliminate_left_recursion(A): Seien $A \to A\alpha_1$ bis $A \to A\alpha_t$ die A-Regeln, deren rechte Seite mit A beginnt und seinen $A \to \beta_1$ bis $A \to \beta_l$ die A-Regeln, deren rechte Seite nicht mit einem A beginnt. Wir führen eine neue Variable B ein, und ersetzen die A-Regeln durch die Regeln $A \to \beta_j$, $A \to \beta_j B$, $B \to \alpha_s$ und $B \to \alpha_s B$, $1 \le j \le l$ und $1 \le s \le t$ hinzu. Beachte, dass beide Regelsätze

die gleiche Sprache erzeugen: nämlich ein β gefolgt von einer Folge (unter Umständen leer) von α's.

Die folgende Operationenfolge stellt den azyklischen Fall her.

for $i = 1$ to m **do**
  for $j = 1$ to $i-1$ **do**
    $subst(A_i, A_j)$;
  end for
  $eliminate\_left\_recursion(A_i)$;
end for

Vor Ausführen der äußeren Schleife gilt:

1. die A_r-Regeln mit $r \geq i$ sind noch wie in der Ausgangsgrammatik
2. die A_r-Regeln mit $r < i$, beginnen entweder mit einem Terminal oder mit einem A_s mit $s > r$. Ferner ist im zweiten Fall auch das zweite Symbol der rechten Seite eine A-Variable.
3. die rechten Seiten von Regeln für B-Variablen beginnen mit einem Terminal oder einer A-Variable. Ferner hat im zweiten Fall die rechte Seite mindestens die Länge zwei.

Das ist sicher alles richtig vor dem ersten Ausführen äußeren Schleife, da die Behauptung leer ist ($i = 1$ und es gibt noch keine B-Regeln).

Betrachten wir nun die innere Schleife. Hier formen wir die A_i-Regeln um. Zunächst entfernen wir führende Vorkommen von A_1, dann führende Vorkommen von A_2, ..., und schließlich führende Vorkommen von A_{i-1}. Am Ende der inneren Schleife beginnt jede A_i-Regel entweder mit einem Terminal oder mit A_j mit $j \geq i$. Im zweiten Fall ist auch das zweite Symbol der rechten Seite eine A-Variable. Nehmen wir an, das sei nicht der Fall. Dann muss es eine erste Ersetzung geben, bei der die Eigenschaft verletzt wird. Bei dieser Ersetzung haben wir eine A-Variable durch eine ihrer rechten Seiten ersetzt. Diese beginnt aber mit einem Terminal oder mit zwei A-Variablen.

Wir benutzen nun $eliminate\_left\_recursion(A_i)$ um die Regeln zu entfernen, deren rechte Seite mit A_i beginnt. Dabei führen wir mit den Bezeichnungen von oben die Regeln $A_i \to \beta_j$, $A_i \to \beta_j B_i$, $B_i \to \alpha_s$ und $B_i \to \alpha_s B_i$, $1 \leq j \leq l$ und $1 \leq s \leq t$ hinzu. Die β_j beginnen mit einem Terminal oder mit einem A_j mit $j > i$ gefolgt von einer weiteren A-Variablen und die α_s beginnen mit einer A-Variablen (wegen der zweiten Invariante). Also gilt für die neu eingeführten Regeln: die rechte Seite beginnt mit einem Terminal oder einem A_j mit $j > i$. Im letzteren Fall hat sie Länge zwei oder mehr. Für die neuen A_i-Regeln gilt sogar, dass sie entweder mit einem Terminal oder mit zwei A-Variablen beginnen.

Nach Beendigung des Programs haben wir einen azyklischen Abhängigkeitsgraphen. Wir beenden nun die Umformung wie im azyklischen Fall.