Greibach Theorem

Every CFL L where $\epsilon \notin L$ can be generated by a CFG in Greibach normal form.

Proof idea: Let $G = (V, \Sigma, R, S)$ be a CFG generating L. Assume that G is in Chomsky normal form

- Let $V = \{A_1, A_2, \ldots, A_m\}$ be an ordering of nonterminals.
- Construct the Greibach normal form from Chomsky normal form

Greibach Theorem

Every CFL L where $\epsilon \notin L$ can be generated by a CFG in Greibach normal form.

Proof idea: Let $G = (V, \Sigma, R, S)$ be a CFG generating L. Assume that G is in Chomsky normal form

- Let $V = \{A_1, A_2, \ldots, A_m\}$ be an ordering of nonterminals.
- Construct the Greibach normal form from Chomsky normal form

$N = \{A_1, \ldots, A_m\}$ ordering of N

Classification of Rules

- **Forward recursive** if

$$A_i \to A_j \gamma \quad \text{with } j > i,$$

or

$$A_i \to b\gamma$$

- **Backward recursive** if

$$A_i \to A_j \gamma \quad \text{with } j < i$$

- **Left recursive** if

$$A_i \to A_i \gamma$$

- $A_1 \rightarrow A_2 A_3$
- $A_2 \rightarrow A_3 A_1 \mid b$
- $A_3 \rightarrow A_1 A_2 \mid a$
- $A_3 \rightarrow A_3 A_1 A_3 A_2 \mid b A_3 A_2 \mid a$

Lemma Let $B \to \alpha_1 | \cdots | \alpha_k$ be all B rules in G.
Let $A \to \gamma B \delta$ be a rule.

Let G1 obtained from G by

- removing $A \to \gamma B \delta$
- Adding $A \to \gamma \alpha_1 \delta \mid \gamma \alpha_2 \delta \mid \cdots \mid \gamma \alpha_n \delta$

Then $L(G) = L(G1)$.

Proof: Exercise.

$A_1 \to A_2 A_3 \quad A_2 \to A_3 A_1 \mid b \quad A_3 \to b A_3 A_2 \mid a$ } Forward rules only.

We can transform this to GNF

$A_3 \to b A_3 A_2 \mid a$

$A_2 \to b A_3 A_2 A_1 \mid a A_1 \mid b$

$A_1 \to$

Greibach Theorem

Every CFL L where $\epsilon \notin L$ can be generated by a CFG in Greibach normal form.

Proof idea: Let $G = (V, \Sigma, R, S)$ be a CFG generating L. Assume that G is in Chomsky normal form

- Let $V = \{A_1, A_2, \ldots, A_m\}$ be an ordering of nonterminals.
- Construct the Greibach normal form from Chomsky normal form

Construction

Aim: Make all rules forward recursive.

1. Modify the rules in R so that if $A_i \to A_j\gamma \in R$ then $j > i$
2. Starting with A_1 and proceeding to A_m this is done as follows:
   (a) Assume that productions have been modified so that for $1 \leq i \cancel{\leq} k$, $A_i \to A_j\gamma \in R$ only if $j > i$
   (b) If $A_k \to A_j\gamma$ is a production with $j < k$, generate a new set of productions substituting for the A_j the rhs of each A_j production
   (c) Repeating (b) at most $k - 1$ times we obtain rules of the form $A_k \to A_p\gamma$, $p \geq k$
   (d) Replace rules $A_k \to A_k\gamma$ by removing left-recursive rules

Construction

1. Modify the rules in R so that if $A_i \to A_j\gamma \in R$ then $j > i$
2. Starting with A_1 and proceeding to A_m this is done as follows:
   (a) Assume that productions have been modified so that for $1 \le i \le k$, $A_i \to A_j\gamma \in R$ only if $j > i$
   (b) If $A_k \to A_j\gamma$ is a production with $j < k$, generate a new set of productions substituting for the A_j the rhs of each A_j production
   (c) Repeating (b) at most $k - 1$ times we obtain rules of the form $A_k \to A_p\gamma, p \ge k$
   (d) Replace rules $A_k \to A_k\gamma$ by removing left-recursive rules

Construction

1. Modify the rules in R so that if $A_i \rightarrow A_j\gamma \in R$ then $j > i$
2. Starting with A_1 and proceeding to A_m this is done as follows:
   (a) Assume that productions have been modified so that for $1 \leq i \leq k$, $A_i \rightarrow A_j\gamma \in R$ only if $j > i$
   (b) If $A_k \rightarrow A_j\gamma$ is a production with $j < k$, generate a new set of productions substituting for the A_j the rhs of each A_j production
   (c) Repeating (b) at most $k - 1$ times we obtain rules of the form $A_k \rightarrow A_p\gamma$, $p \geq k$
   (d) Replace rules $A_k \rightarrow A_k\gamma$ by removing left-recursive rules

Removing left-recursion

Left-recursion can be eliminated by the following scheme:

- If $A \to A\alpha_1|A\alpha_2 \ldots |A\alpha_r$ are all A left recursive rules, and $A \to \beta_1|\beta_2| \ldots |\beta_s$ are all remaining A-rules then chose a new nonterminal, say B
- Add the new B-rules $B \to \alpha_i|\alpha_i B$, $1 \leq i \leq r$
- Replace the A-rules by $A \to \beta_i|\beta_i B$, $1 \leq i \leq s$

This construction preserve the language L.

Removing left-recursion

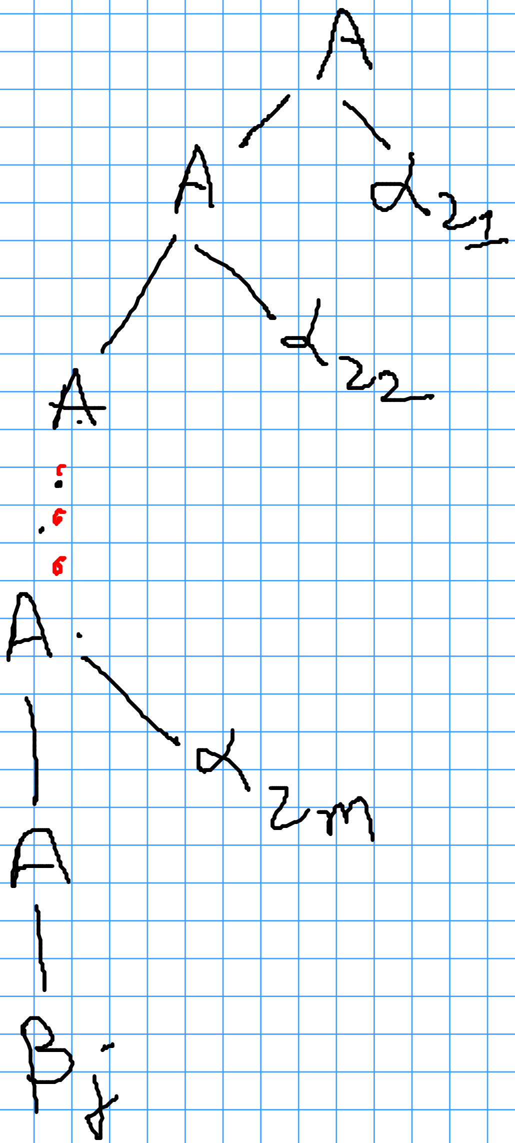
$A \to A\alpha_1 \to A\alpha_2\alpha_1 \xrightarrow{*} A\alpha_m\alpha_{m-1} \cdots \alpha_1 \to \beta_j\alpha_m \cdots \alpha_1$

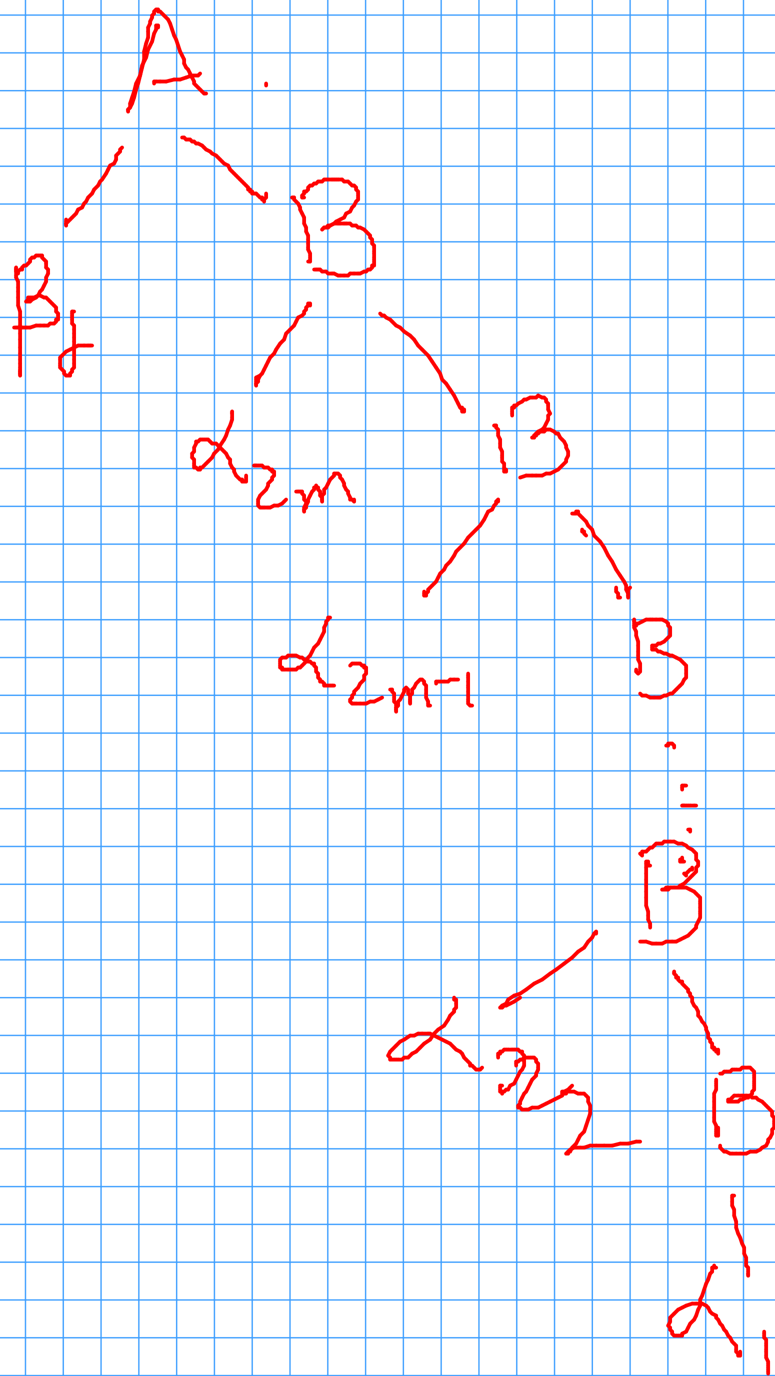
Left-recursion can be eliminated by the following scheme:

- If $A \to A\alpha_1|A\alpha_2 \ldots |A\alpha_r$ are all A left recursive rules, and $A \to \beta_1|\beta_2| \ldots |\beta_s$ are all remaining A-rules then chose a new nonterminal, say B } G
- Add the new B-rules $B \to \alpha_i|\alpha_i B$, $1 \leq i \leq r$ } G1
- Replace the A-rules by $A \to \beta_i|\beta_i B$, $1 \leq i \leq s$

This construction preserve the language L.

Theorem $L(G) = L(G1)$





Grammar G

Removing left-recursion

Left-recursion can be eliminated by the following scheme:

- If $A \to A\alpha_1 | A\alpha_2 \ldots | A\alpha_r$ are all A left recursive rules, and $A \to \beta_1 | \beta_2 | \ldots | \beta_s$ are all remaining A-rules then chose a new nonterminal, say B
- Add the new B-rules $B \to \alpha_i | \alpha_i B$, $1 \leq i \leq r$
- Replace the A-rules by $A \to \beta_i | \beta_i B$, $1 \leq i \leq s$

This construction preserve the language L.

$A \to A\alpha_{z_1} \to A\alpha_{z_2}\alpha_{z_1} \Rightarrow A\alpha_{z_m} \cdots \alpha_{z_1} \to \beta_j \alpha_{z_m} \cdots \alpha_{z_1}$

$A \to \beta_j B \to \beta_j \alpha_{z_m} B \to \beta_j \alpha_{z_m} \alpha_{z_{m-1}} B \overset{*}{\Rightarrow} \beta_j \alpha_{z_m} \alpha_{z_2} B$

- All $A_k \to \beta_i \mid \beta_i B$ rules are forward
- Hence, by substitution, we can turn A_k rules in Griebach Form
- All $B_k \to \alpha_i \mid \alpha_i B_k$ have

  $$\alpha_i \in \left(N \cup \Sigma \cup \{B_1, \dots, B_{k-1}\}\right)^*$$

- By substitution we can turn B_k rules in Griebach Form.

```
        begin
1)          for k := 1 to m do
                begin
2)                  for j := 1 to k - 1 do
3)                      for each production of the form A_k → A_j α do
                            begin
4)                              for all productions A_j → β do
5)                                  add production A_k → βα;
6)                              remove production A_k → A_j α
                            end;
7)                  for each production of the form A_k → A_k α do
                        begin
8)                          add productions B_k → α and B_k → αB_k;
9)                          remove production A_k → A_k α
                        end;
10)                 for each production A_k → β, where β does not
                        begin with A_k do
11)                         add production A_k → βB_k
                end
        end
```

More on Greibach NF

See Introduction to Automata Theory, Languages, and Computation, J.E, Hopcroft and J.D Ullman, Addison-Wesley 1979, p. 94–96

Example

Convert the CFG

$G = (\{A_1, A_2, A_3\}, \{a, b\}, R, A_1)$

where

$R = \{A_1 \to A_2A_3, A_2 \to A_3A_1|b, A_3 \to A_1A_2|a\}$

into Greibach normal form.

Example

Convert the CFG

$G = (\{A_1, A_2, A_3\}, \{a, b\}, R, A_1)$

where

$R = \{A_1 \rightarrow A_2A_3, A_2 \rightarrow A_3A_1|b, A_3 \rightarrow A_1A_2|a\}$

into Greibach normal form.

Example

Convert the CFG

$G = (\{A_1, A_2, A_3\}, \{a, b\}, R, A_1)$

where

$R = \{A_1 \rightarrow A_2A_3, A_2 \rightarrow A_3A_1|b, A_3 \rightarrow A_1A_2|a\}$

into Greibach normal form.

- $A_3 \rightarrow A_2A_3A_2 | a$
- $A_3 \rightarrow A_3A_1A_3A_2 | bA_3A_2 | a$

Solution

1. *Step 1:* ordering the rules: (Only A_3 rules violate ordering conditions, hence only A_3 rules need to be changed).
   Following the procedure we replace A_3 rules by:
   $A_3 \rightarrow A_3A_1A_3A_2|bA_3A_2|a$
2. Eliminating left-recursion we get: $A_3 \rightarrow bA_3A_2B_3|aB_3|bA_3A_2|a$,
   $B_3 \rightarrow A_1A_3A_2|A_1A_3A_2B_3$
3. All A_3 rules start with a terminal. We use them to replace $A_1 \rightarrow A_2A_3$. This introduces the rules $B_3 \rightarrow A_1A_3A_2|A_1A_3A_2B_3$
4. Use A_1 production to make them start with a terminal

Solution

1. *Step 1:* ordering the rules: (Only A_3 rules violate ordering conditions, hence only A_3 rules need to be changed).
   Following the procedure we replace A_3 rules by:
   $A_3 \rightarrow A_3A_1A_3A_2|bA_3A_2|a$
2. Eliminating left-recursion we get: $A_3 \rightarrow bA_3A_2B_3|aB_3|bA_3A_2|a$,
   $B_3 \rightarrow A_1A_3A_2|A_1A_3A_2B_3$
3. All A_3 rules start with a terminal. We use them to replace $A_1 \rightarrow A_2A_3$. This introduces the rules $B_3 \rightarrow A_1A_3A_2|A_1A_3A_2B_3$
4. Use A_1 production to make them start with a terminal

Solution

1. *Step 1:* ordering the rules: (Only A_3 rules violate ordering conditions, hence only A_3 rules need to be changed).
   Following the procedure we replace A_3 rules by:
   $A_3 \to A_3A_1A_3A_2|bA_3A_2|a$
2. Eliminating left-recursion we get: $A_3 \to bA_3A_2B_3|aB_3|bA_3A_2|a$,
   $B_3 \to A_1A_3A_2|A_1A_3A_2B_3$
3. All A_3 rules start with a terminal. We use them to replace $A_1 \to A_2A_3$. This introduces the rules $B_3 \to A_1A_3A_2|A_1A_3A_2B_3$
4. Use A_1 production to make them start with a terminal