

# Chomsky and Greibach Normal Forms

Teodor Rus

`rus@cs.uiowa.edu`

The University of Iowa, Department of Computer Science



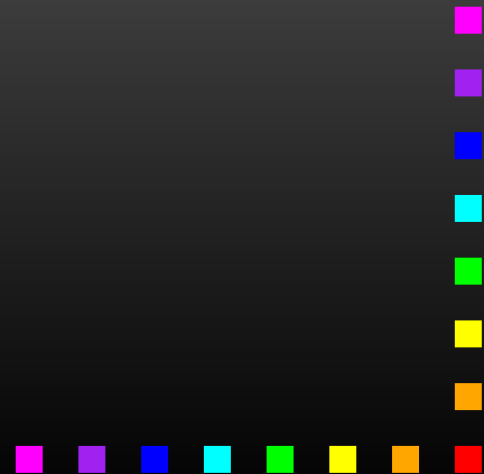
# Simplifying a CFG

- It is often convenient to simplify CFG
- One of the simplest and most useful simplified forms of CFG is called the Chomsky normal form
- Another normal form usually used in algebraic specifications is Greibach normal form



# Note

Normal forms are useful when more advanced topics in computation theory are approached, as we shall see further



# Definition

A context-free grammar  $G$  is in Chomsky normal form if every rule is of the form:

$$A \longrightarrow BC$$

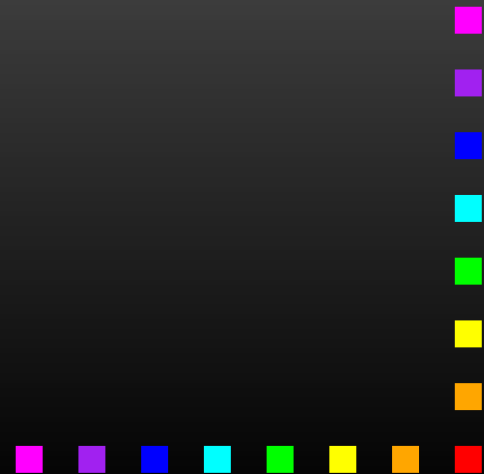
$$A \longrightarrow a$$

where  $a$  is a terminal,  $A, B, C$  are nonterminals, and  $B, C$  may not be the start variable (the axiom)



# Note

The rule  $S \longrightarrow \epsilon$ , where  $S$  is the start variable, is not excluded from a CFG in Chomsky normal form.



# Theorem 2.6

Any context-free language is generated by a context-free grammar in Chomsky normal form.

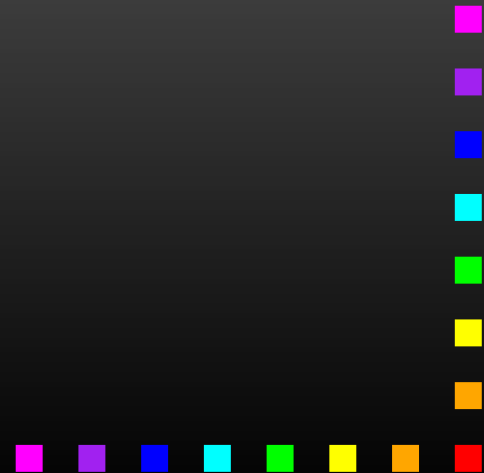
## Proof idea:

- Show that any CFG  $G$  can be converted into a CFG  $G'$  in Chomsky normal form
- Conversion procedure has several stages where the rules that violate Chomsky normal form conditions are replaced with equivalent rules that satisfy these conditions
- Order of transformations: (1) add a new start variable, (2) eliminate all  $\epsilon$ -rules, (3) eliminate unit-rules, (4) convert other rules



# Proof idea, continuation

- Check that the obtained CFG  $G'$  define the same language as the initial CFG  $G$ .



# Proof

Let  $G = (N, T, R, S)$  be the original CFG.

**Step 1:** add a new start symbol  $S_0$  to  $N$ , and the rule  $S_0 \longrightarrow S$  to  $R$

**Note:** this change guarantees that the start symbol of  $G'$  does not occur on the *rhs* of any rule





# Step 2: eliminate $\epsilon$ -rules

Repeat

1. Eliminate the  $\epsilon$  rule  $A \longrightarrow \epsilon$  from  $R$  where  $A$  is not the start symbol
2. For each occurrence of  $A$  on the *rhs* of a rule, add a new rule to  $R$  with that occurrence of  $A$  deleted

**Example:** replace  $B \longrightarrow uAv$  by  $B \longrightarrow uAv|uv$ ;

replace  $B \longrightarrow uAvAw$  by  $B \longrightarrow uAvAw|uvAw|aAvw|uvw$

3. Replace the rule  $B \longrightarrow A$ , (if it is present) by  $B \longrightarrow A|\epsilon$  unless the rule  $B \longrightarrow \epsilon$  has not been previously eliminated

until all  $\epsilon$  rules are eliminated



# Step 3: remove unit rules

Repeat

1. Remove a unit rule  $A \longrightarrow B \in R$
2. For each rule  $B \longrightarrow u \in R$ , add the rule  $A \longrightarrow u$  to  $R$ , unless  $B \rightarrow u$  was a unit rule previously removed

until all unit rules are eliminated

**Note:**  $u$  is a string of variables and terminals



# Convert all remaining rules

## Repeat

1. Replace a rule  $A \longrightarrow u_1 u_2 \dots u_k$ ,  $k \geq 3$ , where each  $u_i$ ,  $1 \leq i \leq k$ , is a variable or a terminal, by:

$$A \longrightarrow u_1 A_1, A_1 \longrightarrow u_2 A_2, \dots, A_{k-2} \longrightarrow u_{k-1} u_k$$

where  $A_1, A_2, \dots, A_{k-2}$  are new variables

2. If  $k \geq 2$  replace any terminal  $u_i$  with a new variable  $U_i$  and add the rule  $U_i \longrightarrow u_i$

until no rules of the form  $A \longrightarrow u_1 u_2 \dots u_k$  with  $k \geq 3$  remain



# Example CFG conversion

Consider the grammar  $G_6$  whose rules are:

$$S \longrightarrow ASA|aB$$

$$A \longrightarrow B|S$$

$$B \longrightarrow b|\epsilon$$

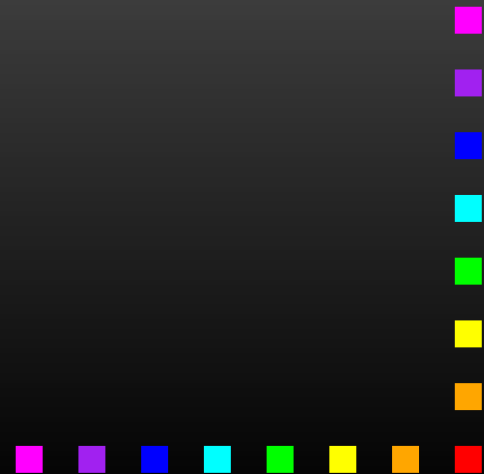
After first step of transformation we get:

$$S_0 \longrightarrow S$$

$$S \longrightarrow ASA|aB$$

$$A \longrightarrow B|S$$

$$B \longrightarrow b|\epsilon$$



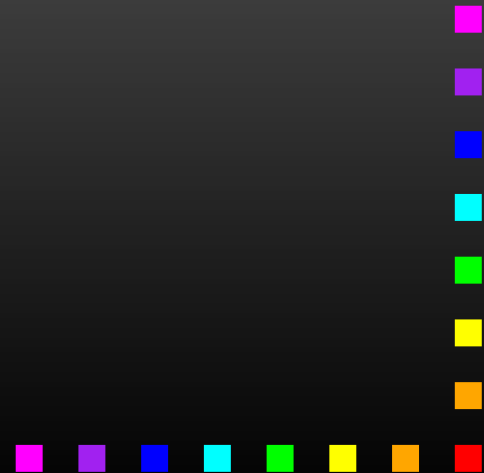
# Removing $\epsilon$ rules

Removing  $B \rightarrow \epsilon$ :

$$\begin{aligned} S_0 &\longrightarrow S \\ S &\longrightarrow ASA|aB|a \\ A &\longrightarrow B|S|\epsilon \\ B &\longrightarrow b \end{aligned}$$

Removing  $A \rightarrow \epsilon$ :

$$\begin{aligned} S_0 &\longrightarrow S \\ S &\longrightarrow ASA|aB|a|SA|AS|S \\ A &\longrightarrow B|S \\ B &\longrightarrow b \end{aligned}$$



# Removing unit rule

Removing  $S \rightarrow S$ :

$$\begin{aligned} S_0 &\longrightarrow S \\ S &\longrightarrow ASA|aB|a|SA|AS \\ A &\longrightarrow B|S \\ B &\longrightarrow b \end{aligned}$$

Removing  $S_0 \rightarrow S$ :

$$\begin{aligned} S_0 &\longrightarrow ASA|aB|a|SA|AS \\ S &\longrightarrow ASA|aB|a|SA|AS \\ A &\longrightarrow B|S \\ B &\longrightarrow b \end{aligned}$$



# More unit rules

Removing  $A \rightarrow B$ :

$$S_0 \longrightarrow ASA|aB|a|SA|AS$$

$$S \longrightarrow ASA|aB|a|SA|AS$$

$$A \longrightarrow S|b$$

$$B \longrightarrow b$$

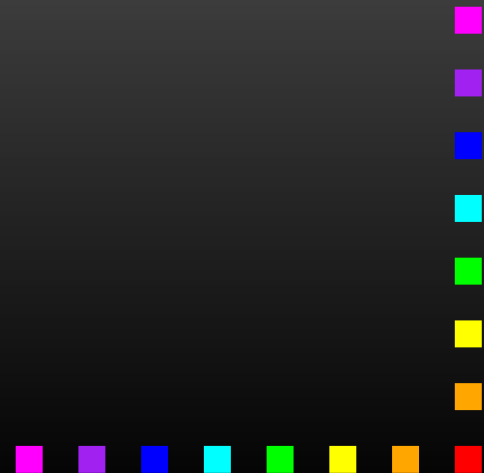
Removing  $A \rightarrow S$ :

$$S_0 \longrightarrow ASA|aB|a|SA|AS$$

$$S \longrightarrow ASA|aB|a|SA|AS$$

$$A \longrightarrow b|ASA|aB|a|SA|AS$$

$$B \longrightarrow b$$



# Converting remaining rules

$$S_0 \longrightarrow AA_1|UB|a|SA|AS$$

$$S \longrightarrow AA_1|UB|a|SA|AS$$

$$A \longrightarrow b|AA_1|UB|a|SA|AS$$

$$A_1 \longrightarrow SA$$

$$U \longrightarrow a$$

$$B \longrightarrow b$$





# Note

- The conversion procedure produces several variables  $U_i$  along with several rules  $U_i \rightarrow a$ .
- Since all these represent the same rule, we may simplify the result using a single variable  $U$  and a single rule  $U \rightarrow a$



# Greibach Normal Form

A context-free grammar  $G = (V, \Sigma, R, S)$  is in Greibach normal form if each rule  $r \in R$  has the property:  $lhs(r) \in V$ ,  $rhs(r) = a\alpha$ ,  $a \in \Sigma$  and  $\alpha \in V^*$ .

**Note:** Greibach normal form provides a justification of operator prefix-notation usually employed in algebra.



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



# 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  $i > j$
  - (b) If  $A_k \rightarrow A_j\gamma$  is a production with  $j < k$ , generate a new set of productions substituting for  $A_j$  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:

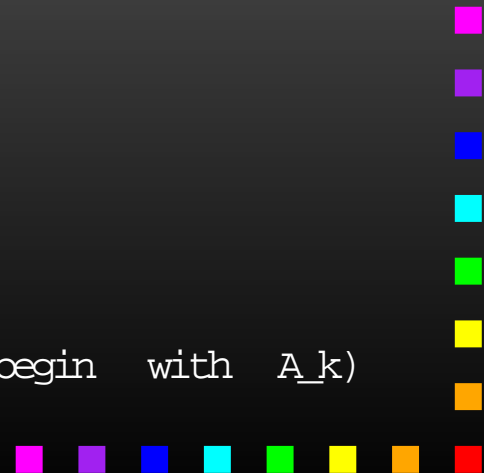
- If  $A \rightarrow A\alpha_1 | A\alpha_2 \dots | A\alpha_r$  are all  $A$  left recursive rules, and  $A \rightarrow \beta_1 | \beta_2 | \dots | \beta_s$  are all remaining  $A$ -rules then chose a new nonterminal, say  $B$
- Add the new  $B$ -rules  $B \rightarrow \alpha_i | \alpha_i B, 1 \leq i \leq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

This construction preserve the language  $L$ .



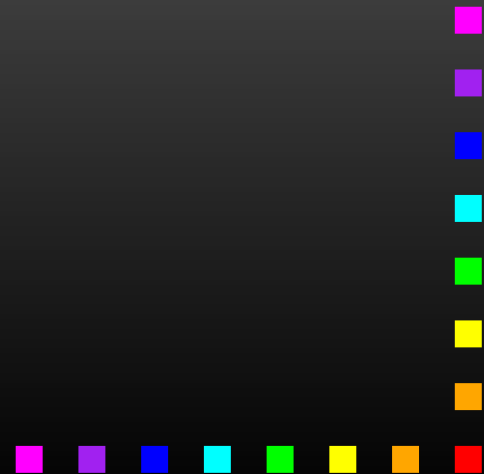
# Conversion algorithm

```
for (k=1; k<=m; k++)
  {for (j=1; j<=k-1; j++)
    {for (each A_k ---> A_j alpha)
      {
        for all rules A_j ---> beta
          add A_k ---> beta alpha
          remove A_k ---> A_j alpha
        }
      }
    for (each rule A_k ---> A_k alpha)
      {
        add rules B_k ---> alpha | alpha B_k
        remove A_k ---> A_k alpha
      }
    for (each rule A_k ---> beta, beta does not begin with A_k)
      add rule A_k ---> beta B_k
  }
```



# 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

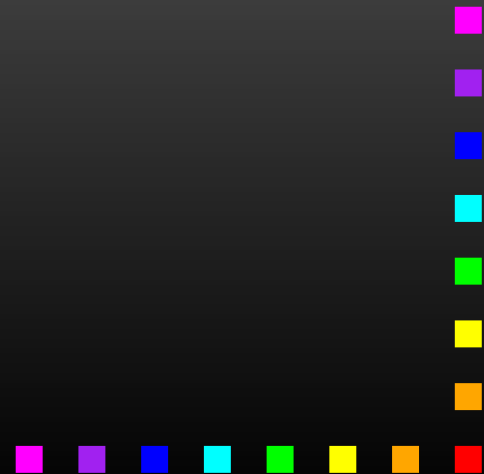
Convert the CFG

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

where

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

into Greibach normal form.





# 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 then 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

