

## Derivation

The process of generating a string. Start with  $u = S$ , repeat the following until  $u$  is variable-free:

Find a variable  $A$  in  $u$ , find a rule in  $R$  of the form  $A \rightarrow w$ , replace the  $A$  by  $w$ .

Use  $A \Rightarrow u$  to denote that  $u$  is **derived** from  $A$ .

A **parse tree** (or **derivation tree**) is a tree that depicts the process of derivation.

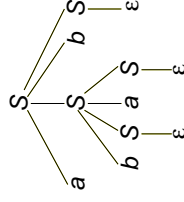
## Context-Free Languages

**Example:** The strings over  $\Sigma = \{a, b\}$  consisting of an equal number of a's and b's

$V = \{S\}$  and the derivation rules are  $S \rightarrow \epsilon \mid aSbS \mid bSaS$ .

$abab$  is derived as follows:

$$S \Rightarrow aSbS \Rightarrow abSaSbS \Rightarrow abSabbS \Rightarrow ababS \Rightarrow abab.$$



A **context-free grammar** is a 4-tuple  $G = (V, \Sigma, R, S)$ . Here

1.  $V$  is the set of **variables** (or **nonterminals**),
2.  $\Sigma$  is the set of **terminals**,
3.  $R$  is the set of **rules**, each of which is of the form

$$A \rightarrow w,$$

- where  $A \in V$  and  $w$  is a string over  $V \cup \Sigma$ ; and
4.  $S$  is the **start symbol**.

## Proof of the Theorem

**Step 1** Add new start symbol  $S_0$  with a unique production rule  $S_0 \rightarrow S$ .  
If  $S \rightarrow \epsilon \in R$  then add  $S_0 \rightarrow \epsilon$ .

**Step 2** Elimination of  $\epsilon$  rules

While there is a variable  $A \neq S_0$  such that  $A \rightarrow \epsilon \in R$

- for each rule  $r$  of the form  $B \rightarrow y$  with an  $A$  in  $y$ , replace  $r$  with the collection of all rules of the form  $B \rightarrow y'$  such that  $y'$  is constructed from  $y$  by eliminating some (possibly none) of the occurrences of  $A$ ;
- eliminate  $A \rightarrow \epsilon$ .

## Leftmost Derivation & Ambiguity

A **leftmost derivation** is the derivation in which each production rules are applied to the leftmost variable. The following derivation of  $abab$

$$S \Rightarrow a\underline{S}bS \Rightarrow ab\underline{S}aSbS \Rightarrow aba\underline{S}bS \Rightarrow abab\underline{S} \Rightarrow abab$$

is a leftmost derivation (giving the same parse tree as before) while

$$S \Rightarrow aS\underline{b}S \Rightarrow aSba\underline{S}bS \Rightarrow aSbab\underline{S} \Rightarrow aS'bab \Rightarrow abab$$

is not (and the parse tree is different as well).

A context-free grammar is **unambiguous** if it has a unique leftmost derivation for every word (sentence) it generates.

There is an **inherently ambiguous** context-free language.

## Proof of the Theorem (cont'd)

**Step 3** Elimination of Unit Rules

While there is a unit rule  $A \rightarrow B$  with  $B \in V$ ,

- eliminate the rule and
- if  $B \neq A$ , then for each rule  $B \rightarrow w$ , add  $A \rightarrow w$  (provided that  $A \rightarrow w$  wasn't previously eliminated)

## Chomsky Normal Form

A context-free grammar  $G = (V, \Sigma, R, S)$  is in **Chomsky normal form** if each rule in  $R$  is either of the form  $A \rightarrow BC$  for some  $B, C \in V - \{S\}$  or of the form  $A \rightarrow a$  with  $a \in \Sigma$ , except that  $S \rightarrow \epsilon$  is permitted.

**Theorem.** *Each context-free language  $L$  is generated by a Chomsky normal form grammar.*

### Example

$V = \{S\}, \Sigma = \{a, b\}$ , and  $R$  consists of  $S \rightarrow \epsilon \mid aSbS \mid bSaS$

**Step 1** Add  $S_0 \rightarrow S \mid \epsilon$ .

**Step 2** Eliminate  $S \rightarrow \epsilon$ . The rules are

$$\begin{aligned} S_0 &\rightarrow S \mid \epsilon, \\ S &\rightarrow ab \mid abS \mid aSbS \mid aS^2b \mid \\ &\quad ba \mid baS \mid bSaS \mid bSa^2. \end{aligned}$$

**STEP 3** Eliminate  $S_0 \rightarrow S$  and add

$$\begin{aligned} S_0 &\rightarrow ab \mid abS \mid aSbS \mid aS^2b \mid \\ &\quad ba \mid baS \mid bSaS \mid bSa^2 \end{aligned}$$

### Proof of the Theorem (cont'd)

**Step 4** Normalization

For each terminal  $d$

- add a new nonterminal  $D$ ,
- add a new rule  $D \rightarrow d$ , and
- for each rule  $A \rightarrow u, |u| > 1$ , in which  $d$  occurs, replace each occurrence of  $d$  with a  $D$ .

For each rule  $A \rightarrow w_1 \dots w_m, m \geq 3$ ,

- add a new variable  $X$  and
- replace each rule of the form  $A \rightarrow w$  with  $A \rightarrow w_1X$  and  $X \rightarrow w_2 \dots w_m$ .

### Example (concluded)

**STEP 4** The rules are

$$\begin{aligned} S_0 &\rightarrow \epsilon, \quad A \rightarrow a, \quad B \rightarrow b, \\ S_0 &\rightarrow AB \mid AX_1 \mid AX_2 \mid AX_3 \mid BA \mid BY_1 \mid BY_2 \mid BY_3, \\ S &\rightarrow AB \mid AX_1 \mid AX_2 \mid AX_3 \mid BA \mid BY_1 \mid BY_2 \mid BY_3, \\ X_1 &\rightarrow BS, \quad X_2 \rightarrow SX_4, \\ X_3 &\rightarrow SB, \quad X_4 \rightarrow BS, \\ Y_1 &\rightarrow AS, \quad Y_2 \rightarrow SY_4, \\ Y_3 &\rightarrow SA, \quad Y_4 \rightarrow AS. \end{aligned}$$

### Proof of the Theorem (concluded)

It's pretty clear the transformation "works", but strictly, we should show this.

In particular, we can argue that

- the iterations in steps 2, 3 and 4 terminate (which is not completely obvious for step 3...); and
- when we consider any parse tree based on the original grammar, we can still set up a parse tree for the same terminal string using the normal-form grammar; and conversely.