Simplifications of Context-Free Grammars

A Substitution Rule

$A \to a$

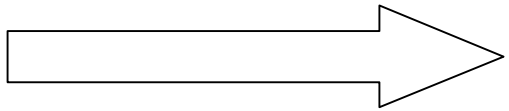
$A \to aaA$

$A \to abBc$

$B \to abbA$

$B \to b$

substitute B



equivalent grammar

$A \to a$

$A \to aaA$

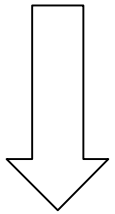
$A \to ababbAc$

$A \to abbc$

In general:

$$A \to xBz$$

$$B \to y_1 \mid y_2 \mid \cdots \mid y_3$$



substitute B

$$A \to xy_1z \mid xy_2z \mid \cdots \mid xy_nz$$

equivalent grammar

Useless Productions

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

$$S \rightarrow A$$

$$A \rightarrow aA$$

$$S \to aSb$$

$$S \to \lambda$$

$$S \to A$$

$$A \to aA$$

never terminates...

$$S \Rightarrow A \Rightarrow aA \Rightarrow aaA \Rightarrow \ldots \Rightarrow aa\ldots aA \Rightarrow \ldots$$

$S \rightarrow aSb$

$S \rightarrow \lambda$

$S \rightarrow A$

$A \rightarrow aA$ ← Useless Production

Another grammar:

$$S \to A$$

$$A \to aA$$

$$A \to \lambda$$

$$B \to bA$$

$S \to A$

$A \to aA$

$A \to \lambda$

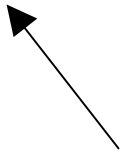
$B \to bA$

Not reachable from S

Useless Production

In general:

If $S \Rightarrow \ldots \Rightarrow xAy \Rightarrow \ldots \Rightarrow w$



$w \in L(G)$

Then A is useful

Otherwise A is useless

Removing Useless Productions

Example Grammar

$$S \to aS \mid A \mid C$$

$$A \to a$$

$$B \to aa$$

$$C \to aCb$$

First: find all variables that produce strings with only terminals

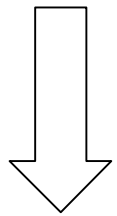
$S \to aS \mid A \mid C$

$A \to a$

$B \to aa$

$C \to aCb$

Round 1: $\{A, B\}$



Round 2: $\{A, B, S\}$

Keep only the variables
that produce terminal symbols

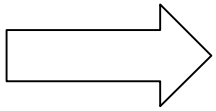
$\{A, B, S\}$

$S \to aS \mid A \mid C$

$A \to a$

$B \to aa$

$C \to aCb$



$S \to aS \mid A$

$A \to a$

$B \to aa$

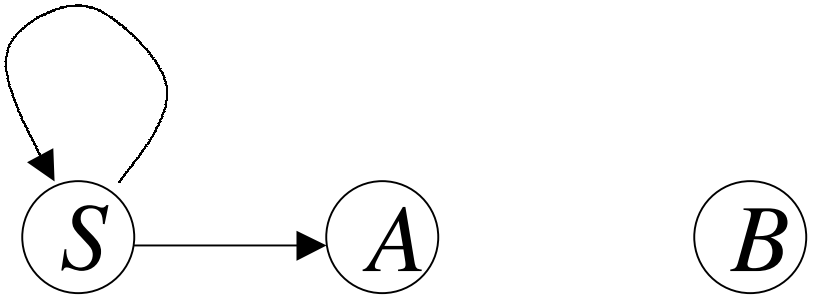
Second: Find all variables reachable from S

$S \rightarrow aS \mid A$

$A \rightarrow a$

$B \rightarrow aa$

Dependency Graph



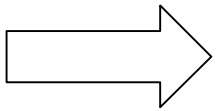
not reachable

Keep only the variables reachable from S

$S \to aS \mid A$

$A \to a$

~~$B \to aa$~~



Final Grammar

$S \to aS \mid A$

$A \to a$

Nullable Variables

λ – production : $A \to \lambda$

Nullable Variable: $A \Rightarrow \ldots \Rightarrow \lambda$

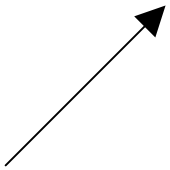
Removing Nullable Variables

Example Grammar:

$$S \to aMb$$

$$M \to aMb$$

$$M \to \lambda$$



Nullable variable

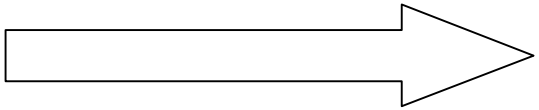
Substitute

$S \to aMb$

$M \to aMb$

~~$M \to \lambda$~~

$M \to \lambda$



$S \to aMb$

$S \to ab$

$M \to aMb$

$M \to ab$

Final Grammar

Unit-Productions

Unit Production: $A \to B$

Removing Unit Productions

Observation:

$$A \to A$$

Is removed immediately

Example Grammar:

$$S \to aA$$

$$A \to a$$

$$A \to B$$

$$B \to A$$

$$B \to bb$$

$S \to aA$

$A \to a$

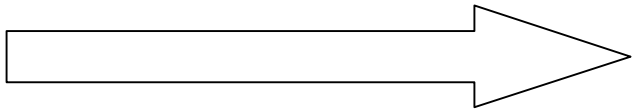
~~$A \to B$~~

$B \to A$

$B \to bb$

substitute

$A \to B$



$S \to aA \mid aB$

$A \to a$

$B \to A \mid B$

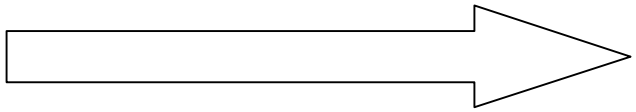
$B \to bb$

$$S \to aA \mid aB$$

$$A \to a$$

$$B \to A \mid \cancel{B}$$

$$B \to bb$$



$$S \to aA \mid aB$$

$$A \to a$$

$$B \to A$$

$$B \to bb$$

$S \to aA \mid aB$

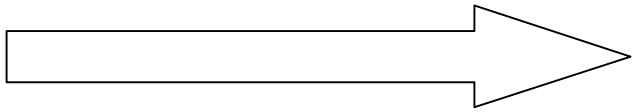
$A \to a$

~~$B \to A$~~

$B \to bb$

substitute

$B \to A$



$S \to aA \mid aB \mid aA$

$A \to a$

$B \to bb$

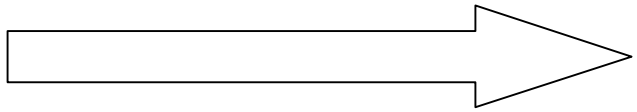
Remove repeated productions

Final grammar

$S \to aA \mid aB \mid \cancel{aA}$

$A \to a$

$B \to bb$



$S \to aA \mid aB$

$A \to a$

$B \to bb$

Removing All

1. Remove Nullable Variables

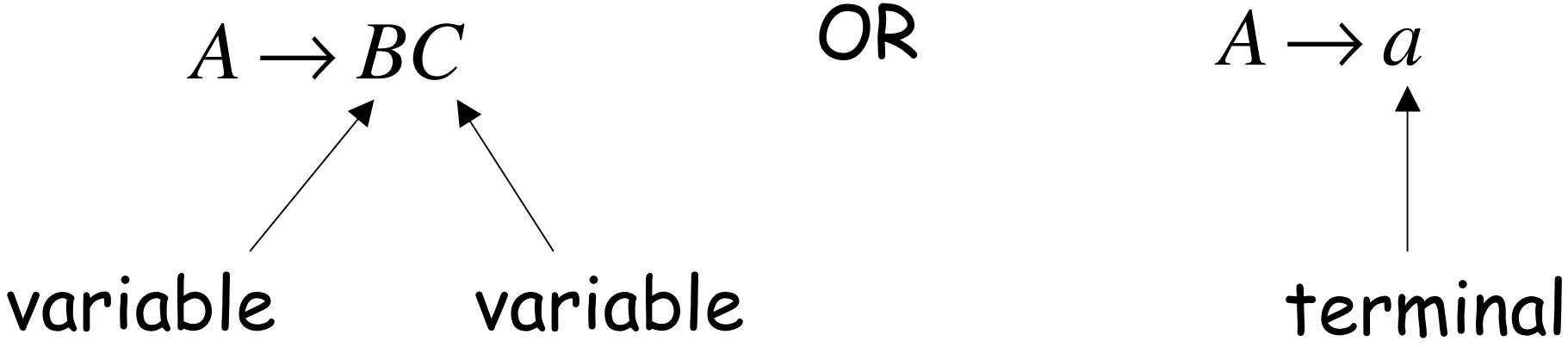
2. Remove Unit-Productions

3. Remove Useless Variables

Normal Forms
for
Context-free Grammars

Chomsky Normal Form

All productions have form:



Examples:

$$S \to AS$$
$$S \to a$$
$$A \to SA$$
$$A \to b$$

Chomsky Normal Form

$$S \to AS$$
$$S \to AAS$$
$$A \to SA$$
$$A \to aa$$

NOT Chomsky Normal Form

Convertion to Chomsky Normal Form

Example:

$$S \to ABa$$

$$A \to aab$$

$$B \to Ac$$

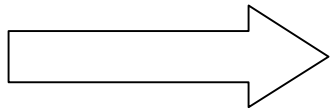
NOT

Chomsky

Normal Form

Introduce variables for terminals: T_a, T_b, T_c

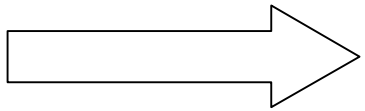
$$S \to ABa$$
$$A \to aab$$
$$B \to Ac$$



$$S \to ABT_a$$
$$A \to T_aT_aT_b$$
$$B \to AT_c$$
$$T_a \to a$$
$$T_b \to b$$
$$T_c \to c$$

Introduce intermediate variables: $V_1,\ V_2$

$$S \to ABT_a$$
$$A \to T_aT_aT_b$$
$$B \to AT_c$$
$$T_a \to a$$
$$T_b \to b$$
$$T_c \to c$$



$$S \to AV_1$$
$$V_1 \to BT_a$$
$$A \to T_aV_2$$
$$V_2 \to T_aT_b$$
$$B \to AT_c$$
$$T_a \to a$$
$$T_b \to b$$
$$T_c \to c$$

Initial grammar

$$S \to ABa$$

$$A \to aab$$

$$B \to Ac$$

Final grammar in Chomsky Normal Form:

$$S \to AV_1$$

$$V_1 \to BT_a$$

$$A \to T_aV_2$$

$$V_2 \to T_aT_b$$

$$B \to AT_c$$

$$T_a \to a$$

$$T_b \to b$$

$$T_c \to c$$

In general:

From any context-free grammar

Not in Chomsky Normal Form

We can obtain:

An equivalent grammar

in Chomsky Normal Form

The Procedure:

Remove

nullable variables

unit productions

For every symbol a :

Add new variable T_a

Add production $T_a \to a$

In productions, replace a with T_a

Replace any production: $A \to C_1 C_2 \cdots C_n$

with:

$$A \to C_1 V_1$$
$$V_1 \to C_2 V_2$$
$$\cdots$$
$$V_{n-2} \to C_{n-1} C_n$$

New variables: $V_1, V_2, \ldots, V_{n-2}$

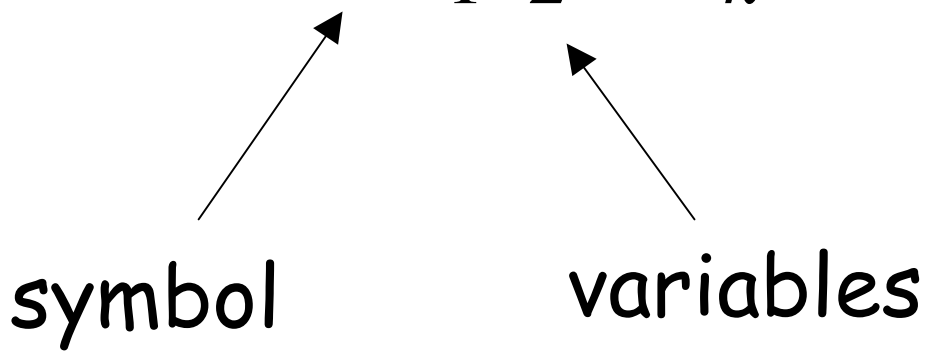
Theorem:

For any context-free grammar
there is
an equivalent grammar
in Chomsky Normal Form

Greinbach Normal Form

All rules have form:

$$A \to a\, V_1 V_2 \cdots V_k \qquad k \geq 0$$



Examples:

$$S \to cAB$$
$$A \to aA \,|\, bB \,|\, b$$
$$B \to b$$

Greinbach
Normal Form

$$S \to abSb$$
$$S \to aa$$

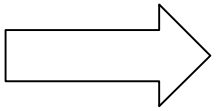
NOT

Greinbach
Normal Form

Conversion to Greinbach Normal Form

$$S \to abSb$$

$$S \to aa$$



$$S \to aT_bST_b$$

$$S \to aT_a$$

$$T_a \to a$$

$$T_b \to b$$

Greinbach Normal Form

Theorem:

For any context-free grammar

there is
an equivalent grammar
in Greinbach Normal Form

An Application of Chomsky Normal Forms

The CYK Membership Algorithm

Inputs:

- Grammar G in Chomsky Normal Form
- String w

Output: find if $w \in L(G)$

Algorithm: Input Examples

Grammar G:

$$S \to AB$$

$$A \to BB$$

$$A \to a$$

$$B \to AB$$

$$B \to b$$

String w: $aabbb$

aabbb

a	a	b	b	b
aa	ab	bb	bb	
aab	abb	bbb		
aabb	abbb			
aabbb				

$S \rightarrow AB$

$A \rightarrow BB$

$A \rightarrow a$

$B \rightarrow AB$

$B \rightarrow b$

a	a	b	b	b
A	A	B	B	B
aa	ab	bb	bb	
aab	abb	bbb		
aabb	abbb			
aabbb				

$S \rightarrow AB$

$A \rightarrow BB$

$A \rightarrow a$

$B \rightarrow AB$

$B \rightarrow b$

a	a	b	b	b
A	A	B	B	B
aa	ab	bb	bb	
	S,B	A	A	
aab	abb	bbb		
aabb	abbb			
aabbb				

$S \to AB$

$A \to BB$

$A \to a$

$B \to AB$

$B \to b$

a	a	b	b	b
A	A	B	B	B
aa	ab	bb	bb	
	S,B	A	A	
aab	abb	bbb		
S,B	A	S,B		
aabb	abbb			
A	S,B			
aabbb				
S,B				

Therefore: $aabbb \in L(G)$

Time Complexity: $|w|^3$

It can be easily converted to a parser