2021

Theory of Computation

Kun-Ta Chuang
Department of Computer Science and Information Engineering
National Cheng Kung University

Outline

1. Methods for Transforming Grammars
2. Two Important Normal Form
3. A Membership Algorithm for CFGs*

Theorem 6.1

Let G = (V, T, S, P) be a context-free grammar. Suppose that P contains a production of the form

$$A \rightarrow x_1 B x_2$$

Assume that A and B are different variables and that

$$B \rightarrow y_1 | y_2 | \ldots | y_n$$

is the set of all productions in P which have B as the left side.

Let Ĝ=(V, T, S, Ṗ) be the grammar in which Ṗ is constructed by deleting

$$A \rightarrow x_1 B x_2$$

from P, and adding to it

$$A \rightarrow x_1 y_1 x_2 | x_1 y_2 x_2 | \ldots | x_1 y_n x_2$$

Then

$$L(\hat{G}) = L(G)$$

Example 6.1

Consider G with following productions

$$A \rightarrow a \mid aaA \mid abBc$$

$$B \rightarrow abbA \mid b$$

Using the suggested substitution for the variable B, we get the grammar Ĝ

$$A \rightarrow a \mid aaA \mid ab\boxed{abbA}c \mid ab\boxed{b}c$$

Useful Substitution Rules

- **Rule 1:** Remove Nullable Variables
- **Rule 2:** Remove Unit-Productions
- **Rule 3:** Remove Useless Variables

Nullable Variables

$\lambda - \text{production}:$ $A \rightarrow \lambda$

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

Example 6.4

$$\{a^n b^n : n \geq 1\}$$

$$S \rightarrow aS_1b$$
$$S_1 \rightarrow aS_1b \mid \lambda$$

$$\Rightarrow$$

$$S \rightarrow aS_1b \mid ab$$
$$S_1 \rightarrow aS_1b \mid ab$$

Example 6.5

Find a CFG without λ-productions equivalent to the grammar G:

Grammar G

$$
\begin{aligned}
S &\rightarrow ABaC \\
A &\rightarrow BC \\
B &\rightarrow b \mid \lambda \\
C &\rightarrow D \mid \lambda \\
D &\rightarrow d
\end{aligned}
$$

$$
\begin{aligned}
S &\rightarrow ABaC \mid BaC \mid AaC \mid ABa \mid aC \mid Aa \mid Ba \mid a \\
A &\rightarrow B \mid C \mid BC \\
B &\rightarrow b \\
C &\rightarrow D \\
D &\rightarrow d
\end{aligned}
$$

A,B, and C are nullable variables

Unit-Productions

Unit Production: $A \rightarrow B$

(a single variable in both sides)

Removing Unit Productions

Observation:

$$A \to A$$

Is removed immediately

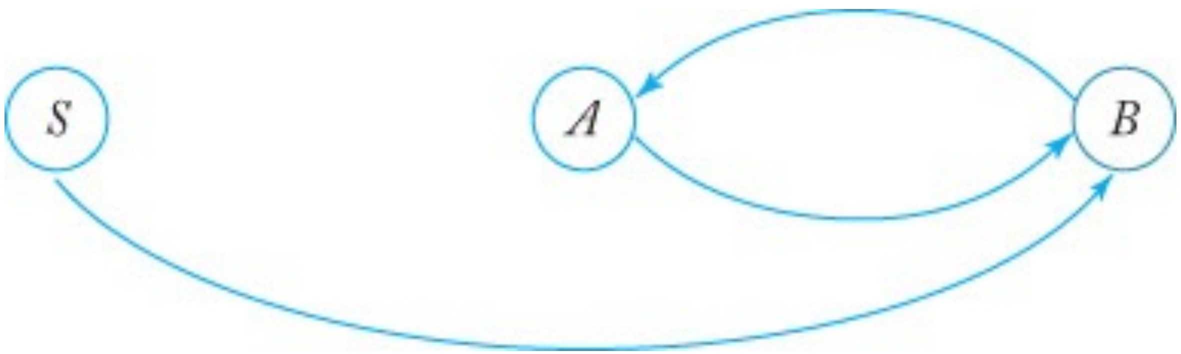
Example 6.6

Remove all unit-productions from

$$S \to Aa \mid B$$

$$B \to A \mid bb$$

$$A \to a \mid bc \mid B$$



dependency graph

$$S \to Aa \mid B$$

$$B \to A \mid bb$$

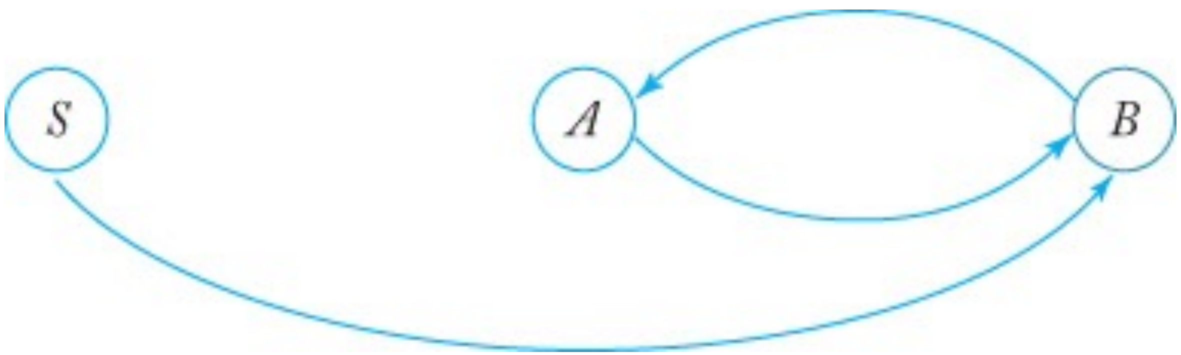
$$A \to a \mid bc \mid B$$

Example 6.6

$$\boxed{\begin{aligned} S &\to Aa \\ B &\to bb \\ A &\to a \mid bc \end{aligned}} + \boxed{\begin{aligned} S &\to a \mid bc \mid bb \\ B &\to a \mid bc \\ A &\to bb \end{aligned}} = \boxed{\begin{aligned} S &\to a \mid bc \mid bb \mid Aa \\ &\cancel{B \to a \mid bb \mid bc} \\ A &\to a \mid bb \mid bc \end{aligned}}$$

Non-unit production

New rules



dependency graph

Useless Productions

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

$$S \rightarrow A$$

$$A \rightarrow aA$$

Useless Production

Some derivations never terminate...

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

Another grammar:

$$S \to A$$

$$A \to aA$$

$$A \to \lambda$$

$$B \to bA$$

Useless Production

Not reachable from S!

In general:

contains only terminals

if $S \Rightarrow \ldots \Rightarrow xAy \Rightarrow \ldots \Rightarrow w$

$w \in L(G)$

then variable A is useful

otherwise, variable A is useless

A production $A \to x$ is useless iff any of its variables is useless

$$S \to aSb$$

$$S \to \lambda$$

Productions

Variables

$S \to A$ useless

useless $A \to aA$ useless

useless $B \to C$ useless

useless $C \to D$ useless

Removing Useless Productions

Example 6.3:

Eliminate useless symbols and productions from the grammar below:

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

$$A \to a$$

$$B \to aa$$

$$C \to aCb$$

First: find all variables that can produce strings with only terminals

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

$$A \to a$$

$$B \to aa$$

$$C \to aCb$$

$$\{A, B\}$$

$$\because S \to A$$

$$\{A, B, S\}$$

Keep only the variables
that produce terminal symbols: $\{A, B, S\}$

(the rest variables are useless)

$$S \rightarrow aS \mid A \mid \cancel{C}$$

$$A \rightarrow a$$

$$B \rightarrow aa$$

$$\cancel{C \rightarrow aCb}$$

$$\Rightarrow$$

$$S \rightarrow aS \mid A$$

$$A \rightarrow a$$

$$B \rightarrow aa$$

Remove useless productions

Second: Find all variables reachable from S

Dependency graph

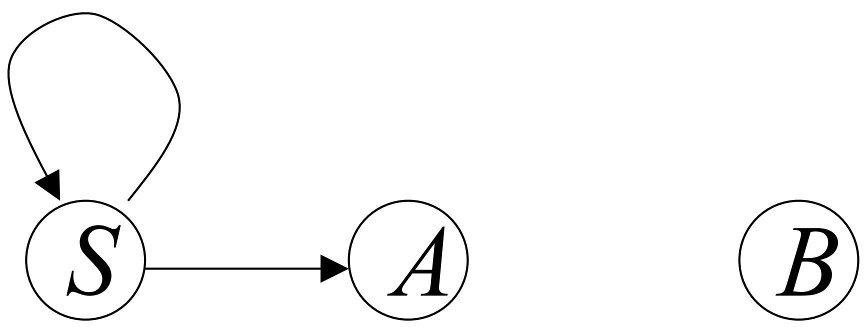
- Vertex labeled with variable
- Edge (A, B) exists iff a production form

A → xBy

$$S \to aS \mid A$$

$$A \to a$$

$$B \to aa$$



not reachable

Keep only the variables
reachable from S

(the rest variables are useless)

Final Grammar

$$S \to aS \mid A$$
$$A \to a$$
$$\cancel{B \to aa}$$

$$\Longrightarrow$$

$$S \to aS \mid A$$
$$A \to a$$

Remove useless productions

Theorem 6.5

- Let L be a CFL that does not contain λ. Then there exists a CFG that generates L and that does not have any useless-, unit-, or λ-production.

$$\boxed{S_0 \rightarrow S \mid \lambda}$$

- Which one needs to be removed first?

- Remove all undesirable productions using the following sequence of steps:
- **Step 1:** Remove λ-productions
- **Step 2:** Remove unit-productions
- **Step 3:** Remove useless-productions

Outline

1. Methods for Transforming Grammars
2. Two Important Normal Form
3. A Membership Algorithm for CFGs*

Chomsky Normal Form (CNF)

Each productions has form:

$$A \to BC \qquad \text{or} \qquad A \to a$$

variable variable terminal

Noam Chomsky

- The Grammar Guy
- 1928 –
- b. Philadelphia, PA
- PhD – UPenn (1955)
  - Linguistics
- Prof at MIT (Linguistics) (1955 - present)

Example 6.7

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

Chomsky
Normal Form

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

Not Chomsky
Normal Form

Example 6.8

- Convert the grammar with following productions to CNF:

$$S \to ABa$$

$$A \to aab$$

$$B \to Ac$$

Introduce variables for terminals: T_a, T_b, T_c

$$S \to ABa$$

$$A \to aab$$

$$B \to Ac$$

$$\Longrightarrow$$

$$S \to ABT_a$$

$$A \to T_aT_aT_b$$

$$B \to AT_c$$

$$T_a \to a$$

$$T_b \to b$$

$$T_c \to c$$

Introduce intermediate variable: V_1

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

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

Introduce intermediate variable: V_2

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

$$\Longrightarrow$$

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

Final grammar in Chomsky Normal Form:

$$S \to AV_1$$

$$V_1 \to BT_a$$

$$A \to T_aV_2$$

$$V_2 \to T_aT_b$$

$$B \to AT_c$$

$$T_a \to a$$

$$T_b \to b$$

$$T_c \to c$$

Initial grammar

$$S \to ABa$$

$$A \to aab$$

$$B \to Ac$$

Theorem 6.6

From any context-free grammar
(which doesn't produce λ)
not in Chomsky Normal Form

we can obtain:
An equivalent grammar
in Chomsky Normal Form

The Procedure

First remove:

Nullable variables

Unit productions

Then, for every symbol a :

Add production $T_a \to a$

In productions: replace a with T_a

New variable: T_a

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

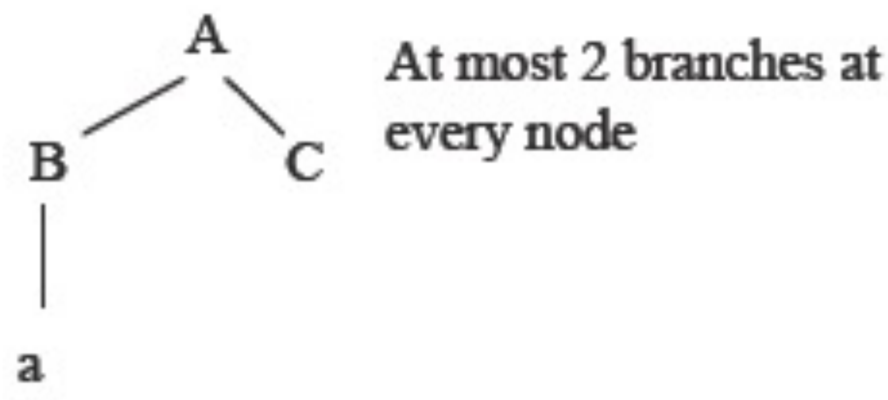
with
$$A \to C_1 V_1$$
$$V_1 \to C_2 V_2$$
$$\cdots$$
$$V_{n-2} \to C_{n-1} C_n$$

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

Theorem: For any context-free grammar (which doesn't produce λ) there is an equivalent grammar in Chomsky Normal Form

Observations

- Chomsky normal forms are good for parsing and proving theorems



- It is very easy to find the Chomsky normal form for any context-free grammar

Greibach Normal Form

All productions have form:

Sheila Greibach

PhD (1963) Harvard University

Prof. of UCLA(CS)

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

symbol

variables

Examples:

$$S \to cAB$$

$$A \to aA \mid bB \mid b$$

$$B \to b$$

Greibach
Normal Form

$$S \to abSb$$

$$S \to aa$$

Not Greibach
Normal Form

Example 6.9:

$$S \to AB$$
$$A \to aA \mid bB \mid b$$
$$B \to b$$

$\Rightarrow$

$$S \to aAB \mid bBB \mid bB$$
$$A \to aA \mid bB \mid b$$
$$B \to b$$

Example 6.10:

$$S \to abSb$$

$$S \to aa$$

$$S \to aT_bST_b$$

$$S \to aT_a$$

$$T_a \to a$$

$$T_b \to b$$

Greibach
Normal Form

Theorem 6.7:

For any context-free grammar
(which doesn't produce λ)
there is an equivalent grammar
in Greibach Normal Form

Observations

- Greibach normal forms are very good for parsing

- It is hard to find the Greibach normal form of any context-free grammar

Outline

1. Methods for Transforming Grammars
2. Two Important Normal Form
3. A Membership Algorithm for CFGs*

Membership Question:

for context-free grammar G
find if string $w \in L(G)$

Membership Algorithms: Parsers

- Exhaustive search parser: $O(P^{2|w|+1})$
- **CYK** parsing algorithm: $O(|w|^3)$

The CYK Parser

J. Cocke
D. H. Younger
T. Kasami

The CYK Membership Algorithm

Input:

- Grammar G in Chomsky Normal Form
- String w

Output:

find if $w \in L(G)$

The Algorithm

Input example:

- Grammar G :

$$S \to AB$$

$$A \to BB$$

$$A \to a$$

$$B \to AB$$

$$B \to b$$

- String w : $aabbb$

$aabbb\ (V_{15})$

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

Therefore: $aabbb \in L(G)$

Time Complexity: $|w|^3$

Observation: The CYK algorithm can be easily converted to a parser (bottom up parser)