CSE331 Automata and Computability

Lecture 17.1: Chomsky Normal Form

Presenter: Sabrina Zaman Ishita (SZI) *Scribe: Sabrina Zaman Ishita (SZI)*

To convert the Context Free Grammar to Chomsky Normal Form, following concepts are needed:

1. **Eliminate the variables that derive nothing:**
   Discover all variables that derive terminal strings. For all other variables, remove all productions in which they appear either on the left or the right.
   For example,
   Consider: **S → AB, A → aA | a, B → AB**
   Although **A** derives all strings of **a**'s, **B** derives no terminal strings.
   Thus, **S** derives nothing, and the language is empty.

2. **Eliminate unreachable variables:**
   Eliminate the variables that do not appear in any derivation from the start symbol.
   Basis: We can reach **S** (the start symbol).
   Induction: If we can reach **A**, and there is a production **A →** α, then we can reach all symbols of α.
   Remove all other symbols which are unreachable from the Start state.

3. **Eliminate all useless symbols by:**
   - Eliminate symbols that derive no terminal string.
   - Eliminate unreachable symbols.

4. **Eliminate ε-Productions:**
   We can almost avoid using productions of the form **A → ε** (called *ε-productions*).
   The problem is that ε cannot be in the language of any grammar that has no ε–productions.
   Theorem: If L is a CFL, then L-{ε} has a CFG with no ε-productions.

   Nullable Variables: To eliminate ε-productions, we first need to discover the ***nullable variables*** (eg. **A**) which are in the form **A =>* ε**.
   Basis: If there is a production **A → ε**, then A is nullable.
   Induction: If there is a production **A →** α, and all symbols of α are nullable, then A is nullable.
   Example:
   S → AB, A → aA | ε, B → bB | A
   Basis: A is nullable because of **A → ε**.
   Induction: B is nullable because of **B → A**.

Then, S is nullable because of **S → AB**.

Key idea: turn each production **A → $X_1 \ldots X_n$** into a family of productions.
For each subset of nullable X's, there is one production with those eliminated from the right side "in advance."
Except, if all X's are nullable, do not make a production with ε as the right side.
Example:
S → ABC, A → aA | ε, B → bB | ε, C → ε
A, B, C, and S are all nullable.
New grammar:
S → AB | A | B
A → aA | A
B → bB | B

5. **Eliminate Unit Productions:**
   A ***unit production*** is one whose right side consists of exactly one variable.
   These productions can be eliminated.
   Key idea: If A =>* B by a series of unit productions, and B → α is a non-unit-production, then add production A → α.
   Then, drop all unit productions.

6. **Cleaning up the Grammar:**
   Start with a CFG for L.
   Perform the following steps in order:
   - Eliminate ε-productions.
   - Eliminate unit productions.
   - Eliminate variables that derive no terminal string.
   - Eliminate variables not reached from the start symbol.

Chomsky Normal Form:
A CFG is said to be in *Chomsky Normal Form* if every production is of one of these two forms:

1. A → BC (right side is two variables).
2. A → a (right side is a single terminal).

Theorem: If L is a CFL, then L – {ε} has a CFG in CNF.

Review:

1. Eliminate ε-productions.
2. Eliminate unit productions.
3. Eliminate variables that derive no terminal string.
4. Eliminate variables not reached from the start symbol.

A CFG is said to be in ***Chomsky Normal Form*** if every production is of one of these two forms:

1. A → BC (right side is two variables).
2. A → a (right side is a single terminal).

CSE331	Automata and Computability

Lecture 17.2: Chomsky Normal Form (Examples)

Presenter: Sabrina Zaman Ishita (SZI) *Scribe: Sabrina Zaman Ishita (SZI)*

Examples on converting Context Free Grammar to Chomsky Normal Form:

1. **Example 1: ∑ = {a, b}**
   S → ASB | a
   A → aAS | a | ε
   B → SbS | A | bb

   Step 1: Add a new start state.
   S' → S
   S → ASB | a
   A → aAS | a | ε
   B → SbS | A | bb

   Step2: Remove ε-productions
   S' → S
   S → ASB | a | AS | SB
   A → aAS | a | aS
   B → SbS | A | bb

   Step 3: Remove Unit Productions
   S' → ASB | a | AS | SB
   S → ASB | a | AS | SB
   A → aAS | a | aS
   B → SbS | aAS | a | aS | bb

   Step 4: Make CNF form
   S' → XB | a | AS | SB
   S → XB | a | AS | SB
   A → YX | a | YS
   B → SZS | YX | a | YS | ZZ
   X → AS
   Y → a
   Z → b

   Step 5: Complete
   S' → XB | a | AS | SB
   S → XB | a | AS | SB
   A → YX | a | YS
   B → WS | YX | a | YS | ZZ

X → AS
Y → a
Z → b
W → SZ

2. **Example 2: ∑ = {a, b}**

S → ASA | aB

A → B | S

B → b | ε

Step 1:

S' → S

S → ASA | aB

A → B | S

B → b | ε

Step 2:

S' → S

S → ASA | aB | AS | SA | a

A → B | S

B → b

Step 3:

S' → ASA | aB | AS | SA | a

S → ASA | aB | AS | SA | a

A → b | ASA | aB | AS | SA | a

B → b

Step 4:

S' → YA | XB | AS | SA | a

S → YA | XB | AS | SA | a

A → b | YA | XB | AS | SA | a

B → b

X → a

Y → AS

3. **Example 3: $\sum = \{a, b, c\}$**

   $S \rightarrow aXbX$

   $X \rightarrow aY \mid bY \mid \varepsilon$

   $Y \rightarrow X \mid c$

   Do it yourself.

Normal form of a context free grammar makes the algorithm for membership problem (whether a string belongs to the Language of the grammar or not) to be run in Polynomial time.

CSE331 Automata and Computability

Lecture 18: CYK Algorithm

Presenter: Sabrina Zaman Ishita (SZI) *Scribe: Sabrina Zaman Ishita (SZI)*

CYK Algorithm: The algorithm to decide whether a string belongs to the language of a grammar or not.

Basics of CYK Algorithm:

- ✓ The Structure of the rules in a Chomsky Normal Form grammar
- ✓ Uses a “dynamic programming” or “table-filling approach”

Construct a triangular table:

Each row corresponds to one length of substrings

- ✓ Bottom Row – Strings of length 1
- ✓ Second from Bottom Row – Strings of length 2
- ✓ Top Row – string ‘w’

- $\mathbf{X_{i,\,i}}$ is the set of variables A such that

  $A \rightarrow w_i$ is a production of G
- Compare at most n pairs of previously computed sets:

  $(\mathbf{X_{i,\,i}}, \mathbf{X_{i+1,\,j}})$, $(\mathbf{X_{i,\,i+1}}, \mathbf{X_{i+2,\,j}})$ … $(\mathbf{X_{i,\,j-1}}, \mathbf{X_{j,\,j}})$

$X_{1,5}$				
$X_{1,4}$	$X_{2,5}$			
$X_{1,3}$	$X_{2,4}$	$X_{3,5}$		
$X_{1,2}$	$X_{2,3}$	$X_{3,4}$	$X_{4,5}$	
$X_{1,1}$	$X_{2,2}$	$X_{3,3}$	$X_{4,4}$	$X_{5,5}$
w_1	w_2	w_3	w_4	w_5

Table for string ‘**w**’ that has length 5

Example: Show the CYK Algorithm with the following example:

- CNF grammar **G**

  S → AB | BC

  A → BA | a

  B → CC | b

  C → AB | a

- **w** is baaba

Question Is **baaba** in L(G)?

{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

Calculating the Bottom ROW

- **$X_{1,2} = (X_{i,i}, X_{i+1,j}) = (X_{1,1}, X_{2,2})$**
- → {B}{A,C} = {BA, BC}
- Steps:
  - Look for production rules to generate BA or BC
  - There are two: S and A
  - **$X_{1,2}$ = {S, A}**

{S, A}				
{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

$S \rightarrow AB \mid BC$

$A \rightarrow BA \mid a$

$B \rightarrow CC \mid b$

$C \rightarrow AB \mid a$

- **$X_{2,3} = (X_{i,i}, X_{i+1,j}) = (X_{2,2}, X_{3,3})$**
- ➔ {A, C}{A,C} = {AA, AC, CA, CC} = Y
- Steps:
  - – Look for production rules to generate Y
  - – There is one: B
  - – **$X_{2,3}$ = {B}**

{S, A}	{B}			
{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

- **$X_{3,4} = (X_{i,i}, X_{i+1,j}) = (X_{3,3}, X_{4,4})$**
- ➔ {A, C}{B} = {AB, CB} = Y
- Steps:
  - – Look for production rules to generate Y
  - – There are two: S and C
  - – **$X_{3,4}$ = {S, C}**

{S, A}	{B}	{S, C}		
{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

- **$X_{4,5} = (X_{i,i}, X_{i+1,j}) = (X_{4,4}, X_{5,5})$**
- ➔ {B}{A, C} = {BA, BC} = Y
- Steps:
  - – Look for production rules to generate Y
  - – There are two: S and A
  - – **$X_{4,5}$ = {S, A}**

{S, A}	{B}	{S, C}	{S, A}	
{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

- **$X_{1,3}$ $= (X_{i,i}, X_{i+1,j})\ (X_{i,i+1}, X_{i+2,j})$**
  $= (X_{1,1}, X_{2,3}), (X_{1,2}, X_{3,3})$
- ➔ {B}{B} **U** {S, A}{A, C}= {BB, SA, SC, AA, AC} = Y
- Steps:
  - – Look for production rules to generate Y
  - – There are NONE: S and A
  - – **$X_{1,3} = \emptyset$**
  - – **no elements in this set (empty set)**

Ø				
{S, A}	{B}	{S, C}	{S, A}	
{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

$S \rightarrow AB \mid BC$
$A \rightarrow BA \mid a$
$B \rightarrow CC \mid b$
$C \rightarrow AB \mid a$

- $X_{2,4}$ $= (X_{i,i}, X_{i+1,j})\ (X_{i,i+1}, X_{i+2,j})$
  $= (X_{2,2}, X_{3,4}), (X_{2,3}, X_{4,4})$
- $\rightarrow$ {A, C}{S, C} **U** {B}{B}= {AS, AC, CS, CC,
- Steps:
  - Look for production rules to generate Y
  - There is one: B
  - $X_{2,4} = \{B\}$

Ø	{B}			
{S, A}	{B}	{S, C}	{S, A}	
{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

- $X_{3,5}$ $= (X_{i,i}, X_{i+1,j})\ (X_{i,i+1}, X_{i+2,j})$
  $= (X_{3,3}, X_{4,5}), (X_{3,4}, X_{5,5})$
- $\rightarrow$ {A,C}{S,A} **U** {S,C}{A,C}
  = {AS, AA, CS, CA, SA, SC, CA, CC} = Y
- Steps:
  - Look for production rules to generate Y
  - There is one: B
  - $X_{3,5} = \{B\}$

Ø	{B}	{B}		
{S, A}	{B}	{S, C}	{S, A}	
{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

{S, A, C}	$\leftarrow X_{1,5}$			
Ø	{S, A, C}			
Ø	{B}	{B}		
{S, A}	{B}	{S, C}	{S, A}	
{B}	{A, C}	{A, C}	{B}	{A, C}
b	a	a	b	a

- Table for string '**w**' that has length 5
- The algorithm populates the triangular table

- Is baaba in L(G)?

Yes

We can see the S in the set X_{1n} where 'n' = 5
We can see the table
the cell X_{15} = (S, A, C) then
if S $\in$ X_{15} then baaba $\in$ L(G)

- The CYK Algorithm correctly computes X_{ij} for all i and j; thus w is in L(G) if and only if S is in X_{1n}.
- The running time of the algorithm is $O(n^3)$.