AUTOMATA FORMAL LANGUAGES AND LOGIC

Lecture notes
Chomsky Normal Form

Prepared by:
Kavitha K N
Assistant Professor

Department of Computer Science & Engineering
PES UNIVERSITY
(Established under Karnataka Act No.16 of 2013)
100-ft Ring Road, BSK III Stage, Bangalore – 560 085

Table of Contents:

Examples Solved:

Chomsky Normal Form

1. Introduction

A context free grammar is in chomsky normal form (CNF) if all the productions rules satisfy one of the following condition

1) A non - terminal generating terminal
   Ex: X->x
2) A non - terminal generating two non terminals
   Ex: X-> XY
3) Start symbol generating λ
   Ex: S-> λ

2. Why Chomsky Normal Form

The key advantage of CNF is, every derivation of a string of n letters have exactly 2n-1 steps Thus one can determine the string is in the language by exhaustive search of all derivations.Normal forms make many concepts much easier to handle because you can assume a simple structure for them. CNF enables a polynomial time algorithm to decide whether a string can be generated by the grammar.

If every production in the grammar is in the form X->x and X->XY then we say that the grammar is in CNF. CNF makes the parse tree a binary tree, in a binary tree , a node may have either one or two child nodes. In a parse tree obtained from the CNF grammar,

1) If a node has one child , that child is a terminal symbol
   Ex: A-> a
2) If a node has two children that , that child are non terminal symbol

3. Converting Context Free Grammar (CFG) to Chomsky Normal Form(CNF)

To convert the CFG to CNF the CFG should be clean, that is it must not have any

1) Lambda production
   Ex: A-> $\boldsymbol{\lambda}$
   The variables that have such productions are nullable variables
2) Unit Productions
   (It just replaces one variable with another variable without any progress in the derivation)
3) Useless variables:
   These are of two types
   a) Non-generating variables:

This will never lead to terminal symbols. If a non generating variable is introduced into the sentential form , it is a dead end, the sentential form can never generate any string in the language

b) Productions for unreachable variables
These variables does not appear on the right hand side of any production

3.1 Steps to Convert CFG to CNF

Follow the steps in the given order

Step 1: Eliminate lambda productions:

Step 1:Find out all nullable variables.

Step 2: In the RHS of every production where the nullable variables exists

a) Write the production rule with and without the nullable variable
b) Eliminate nullable variable production

Step2 might produce useless productions and unit productions

Note : we will ot eliminate all the λ productions for the grammar

If $\lambda \in L(G)$

In this case we set S-> λ

Step2: Eliminate Unit production

Why?

If we have

A-> B
B->C
c-> d

In order to generate d,

We should follow the derivation

Step 1:A-> B

Step 2: B->C

Step 3: C->d

Length of their sentential forms is not increasing neither the number of terminals.

Note: Every step in the derivation has to be useful either increasing the sentential form or increasing the number of terminals.

So the above step 1 ,2 and 3 are useless.

Step 3: Eliminating Useless productions and Symbols

There are two aspects to begin a useful symbol

1) Derivability

If a variable A is useful it must generate orderive string of terminals

A ⇒ w (w is a string of terminals

2) Reachability

Every useful variable must be reachable from S

S-> A ⇒w

That is,

A useful variable must be reachable from S and must deriva a string of terminals

Note:

For a given language there are many G1 , G2 ,...Gn multiple grammars, some are more elegant than others leading to better parsers and faster compilers.

-> we can convert a grammar eo Normal Form that gives us cleaner grammars with good algorithms for parsing

-> Most wel, known Normal Forms for CG's is CNF. or energy CFG there is an equivalent CNF

Here,

Each production in the grammar either generates a terminal or expands a sentential form by exactly 1

A-> a

A-> BC

-> If every production in the grammar is in this form we say G is in CNF

-> CNF makes the parse tree as binary tree

-> Algorithm for converting CFG to CNF is based on the idea of substitution

4. Example:

Convert the following CFG to CNF

1) S -> aX | Yb

X ->S | λ

Y -> bY | b

Step 1: Eliminate λ productions

S -> aX | a | Yb

X-> S

Y-> bY | b

Step 2: Eliminate Unit Productions

S -> aX | a | Yb

X -> aX | a | Yb

Y -> bY | b

Step 3: There are no useless productions

Step 4: Conversion to CNF:

S-> AX | YB | a

X -> AX | YB | a

Y -> BY | b

A -> a

B -> b

2) S → aSa | bSb | A | λ

A → a | b| λ

Solution:

Step 1: Remove λ production

S → aSa | aa | bSb | bb| A

A → a | b

Step 2: Remove unit production (S → A)

S → aSa | aa | bSb | bb| a | b

A → a | b

Step 3: Remove useless production(A)

S → aSa | aa | bSb | bb| a | b

Now the CFG is

A → a

B → b

S → ASA | BSB | AA | BB | a | b

Step 4: To CNF

A → a

B → b

C → AS

D → BS

S → CA | DB | AA | Bb | a | b

3) S → BAB
   B → bba
   A → Bc

   Solution:
   Step 1: There are no $\boldsymbol{\lambda}$ production
   Step 2 There are no unit productions
   Step 3: There are no useless production
   Step 4: To CNF
      C → a
      D → b
      E → c
      F → BA
      G → DD
      S → FB
      B → GC
      A → BE

4) S →Aa | B | Ca
   B → aB | b
   C → Db | D
   D → E | d
   E → ab
   Solution:
   Step 1: There are no $\boldsymbol{\lambda}$ production
   Step 2: Remove unit production

   S → Aa | aB | b | Ca
   B → aB | b
   C → Db | ab | d
   D → ab | d
   E → ab
   Step 3: Remove useless production
   E is useless production and Aa is useless as there is no variable A

Step 4: To CNF

X → a

Y → b

S → XB | b | CX

B → XB | b

C → DY | XY | d

D → XY | d

5) S → aAa | bBb | BB

A → C

B → S | A

C → S | λ

Solution:

Step 1: Remove λ production

S → aAa | aa | bBb | bb | BB

A → C

B → S | A

C → S

Step 2: Remove unit productions

S → aAa | aa | bBb | bb | BB

A → aAa | aa | bBb | bb | BB

B → aAa | aa | bBb | bb | BB

C → aAa | aa | bBb | bb | BB

Step 3: Remove useless production

C is useless production

S → aAa | aa | bBb | bb | BB

A → aAa | aa | bBb | bb | BB

B → aAa | aa | bBb | bb | BB

Step 4: To CNF

X → a

Y → b

P → XA

Q → YB

S → PX | XX | QY | YY | BB

A → PX | XX | QY | YY | BB

B → PX | XX | QY | YY | BB

6) E → E + T | T
T → T * F | F
F → num | id

Solution:
Step 1: There are no λ production
Step 2: Remove unit production (E → T) and(T → F)
E → E + T | T * F | F
T → T * F | num | id
F → num | id
This results in E → F , remove this
E → E + T | T * F | num | id
T → T * F | num | id
F → num | id

Step 3: There are no use less productions
Now the CFG is:
A → +
B → *
E → EAT | TBF | num | id
T → TBT | num| id
F → num | id

Step 4 : To CFG
A → +
B → *
C → EA
D → TB
E → CT | DF | num | id
T → DT | num | id
F → num | id