Theory of Automata
Context Free Grammars

Week-9-Lecture-02

Hafiz Tayyeb Javed

Contents

- Examples of Non regular Grammar
- Mapping of CFG into GTG
- Mapping of CFG into FA
- Mapping of GTG into CFG
- Mapping of FA into CFG

Example

$S \rightarrow aSa \mid aBa$

$B \rightarrow bB \mid b$

- First production builds equal number of a's on both sides and recursion is terminated by $S \rightarrow aBa$
- Recursion of $B \rightarrow bB$ may add any number of b's and terminates with $B \rightarrow b$

- $L(G) = \{a^n b^m a^n \; n>0, m>0\}$

example

$L(G) = \{a^n b^m c^m d^{2n} \mid n>0, m>0\}$

- Consider relationship between leading a’s and trailing d’s.

$S \rightarrow aSdd$

In the middle equal number of b’s and c’s

- $S \rightarrow A$
- $A \rightarrow bAc$
- This middle recursion terminates by $A \rightarrow bc$.

- Grammar will be

$S \rightarrow aSdd \mid aAdd$

$A \rightarrow bAc \mid bc$

Example

Consider another CFG

$S \rightarrow aSb \mid aSbb \mid \Lambda$

- Language defined is

$L(G) = \{a^n b^m \mid 0 \leq n \leq m \leq 2n\}$

Example

- A grammar that generates the language consisting of even-length string over {a, b}
  S → aO | bO | Λ
  O → aS | bS

- S and O work as counters i.e. when an S is in a sentential form that marks even number of terminals have been generated

- Presence of O in a sentential form indicates that an odd number of terminals have been generated.

- The strategy can be generalized, say for string of length exactly divisible by 3 we need three counters to mark 0, 1, 2

  S → aP | bP | Λ
  P → aQ | bQ
  Q → aS | bS

Even-Even

- $\Sigma = \{a,b\}$

Productions:

- $S \rightarrow SS$
- $S \rightarrow XS$
- $S \rightarrow \Lambda$
- $S \rightarrow YSY$
- $X \rightarrow aa$
- $X \rightarrow bb$
- $Y \rightarrow ab$
- $Y \rightarrow ba$

Devise a grammar that generates strings with even number of a’s and even number of b’s

Remarks

- We have seen that some regular languages can be generated by CFGs, and some non-regular languages can also be generated by CFGs.

- In Chapter 13, we will show that ALL regular languages can be generated by CFGs.

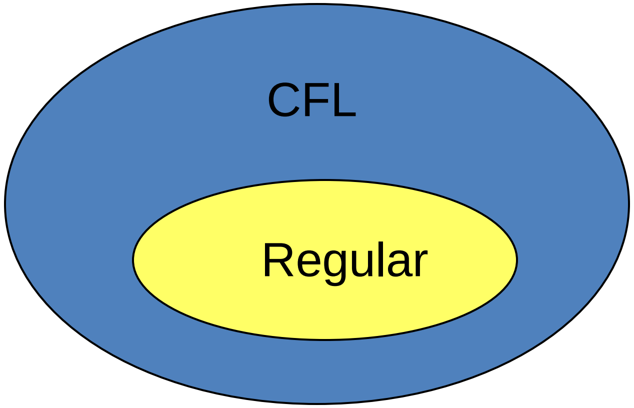
- In Chapter 16, we will see that there is some non-regular language that cannot be generated by any CFG.

- Thus, the set of languages generated by CFGs is properly **larger** than the set of regular languages, but properly **smaller** than the set of all possible languages.

Regular Grammar

Given an FA, there is a CFG that generates exactly the language accepted by the FA.

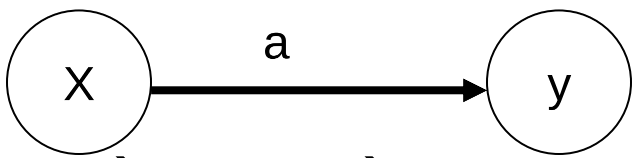
- In other words, all regular languages are CFLs

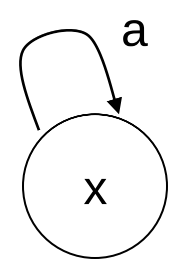


Creating a CFG from an FA

Step-1 The Non-terminals in CFG will be all names of the states in the FA with the start state renamed S.

Step-2 For every edge





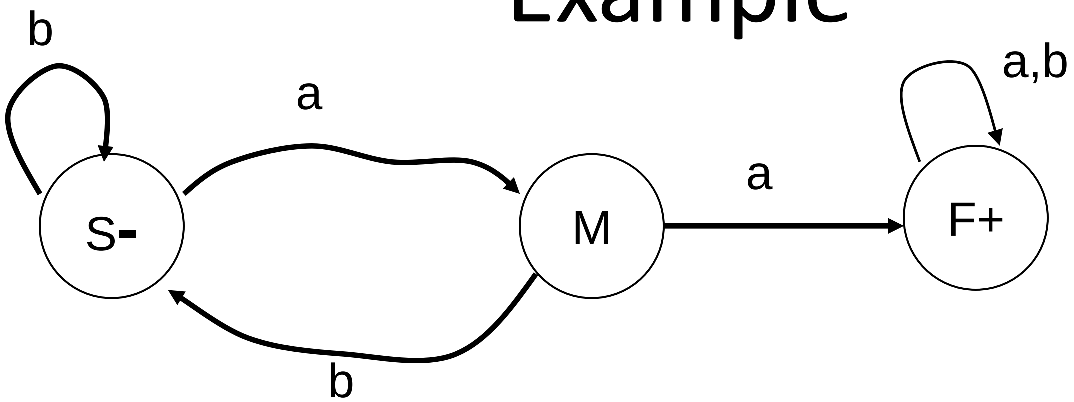
Create productions $X \rightarrow aY$ or $X \rightarrow aX$

Do the same for b-edges

Step-3 For every final-state X, create the production

$X \rightarrow \Lambda$

Example



$S \rightarrow aM$

$S \rightarrow bS$

$M \rightarrow aF$

$M \rightarrow bS$

$F \rightarrow aF$

$F \rightarrow bF$

$F \rightarrow \Lambda$

Note: It is not necessary that each CFG has a corresponding FA. But each FA has an equivalent CFG.

Regular Grammar

Theorem 22:

If all the productions in a given CFG fit one of the two forms: Non-terminal → semiword

or Non-terminal → word

(Where the word may be a Λ or string of terminal), then the language generated by the CFG is Regular.

Proof:

For a CFG to be regular is by constructing a TG from the given CFG.

Proof contd.

- Let us consider a general CFG in this form

$N_1 \rightarrow w_1N_2$ $\quad$ $N_7 \rightarrow w_{10}$

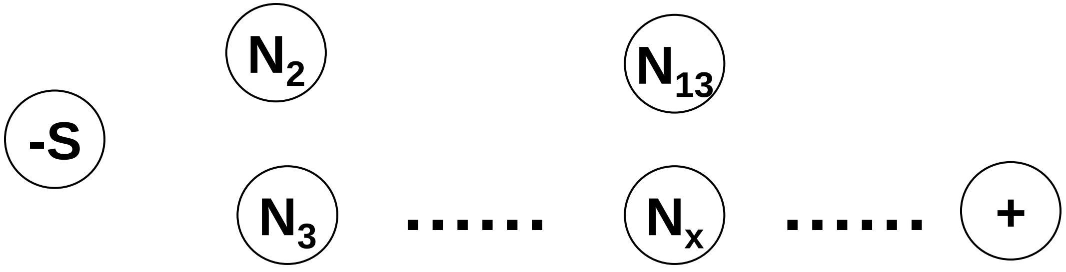
$N_1 \rightarrow w_2N_3$ $\quad$ $N_{18} \rightarrow w_{23}$

$N_2 \rightarrow w_3N_4$ $\quad$ ---------------

--------------- $\quad$ ---------------

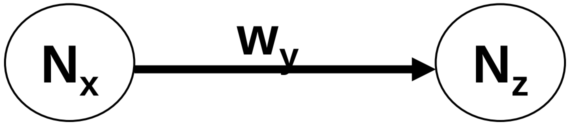
Where N's are non-terminal and w's are the string of terminal and part w_yN_z are semiwords.

Let N_1=S. Draw a small circle for each N and one extra circle labelled +, the circle for S we label (-)

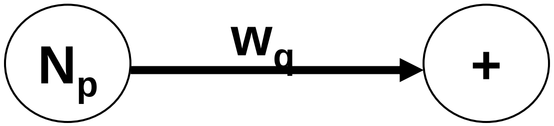


Proof contd.

- For each production of the form $N_x \rightarrow w_y N_z$, draw a directed edge from state N_x to N_z with label w_y.



- If Nx = Nz, the path is a loop
- For every production of the form $N_p \rightarrow w_q$, draw a directed edge from Np to + and label it with w_q even if $w_q = \Lambda$.



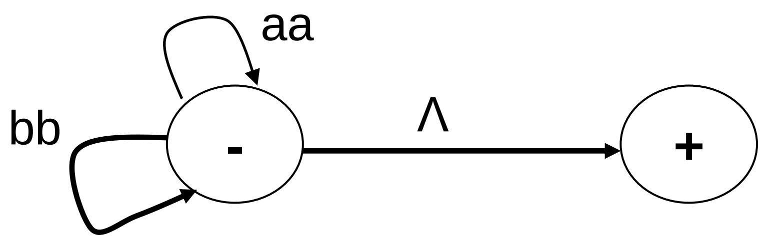
- Any path in TG form – to + corresponds to a word in the language of TG (by concatenating symbols) and simultaneously corresponds to sequence of productions on the CFG generating words.
- Conversely every production of the word in the CFG:

S ➔ wN ➔ wwN ➔ wwwN ➔ ….. ➔ wwwww

Corresponds to a path in this TG.

Example

- Consider the CFG S → aaS | bbS | Λ



- The regular expression is given by (aa + bb)*.
- Consider the CFG

S→aaS | bbS | abX | baX | Λ

X→ aaX | bbX | abS | baS

- Language accepted?
- EVEN-EVEN