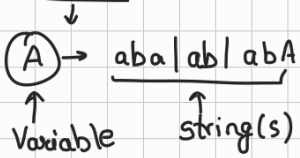
CFG

Context Fvee Grammar:
Use → Specification & compilation of Programming languages.

Terminology:

- Grammar → Collection of substitution rules {Productions}



- Rule → A line in grammar, comprising symbol and string separated by arrows!
- Variable → The symbol to the left of arrow in rule is a variable.
- Strings consits of either variables or other symbols called terminals.
- One variable is designated as the start variable.

CFG Simplification:
1. Useless symbols (Non ① generating and Non ② rechable) → dependency graph
2. ϵ-productions
3. Unit productions

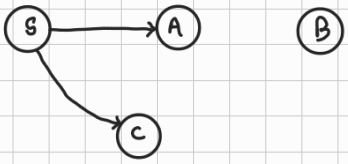
$S \to AB \mid CA$
$A \to a$ $\quad B \to BC \mid AB$
$C \to AB \mid b$

Generating = {S, A, C}

Remove non generating, i.e. B,

$S \to CA$
$A \to a$
$C \to b$

∴ dependency graph will be:



2. ϵ-productions:

e.g. $S \to AB$
$A \to aAA \mid \epsilon$
$B \to bBB \mid \epsilon$

$S \to AB \mid A \mid B$
$A \to aAA \mid aA \mid a$
$B \to bBB \mid bB \mid b$

Unit Productions:
$A \to B$, where A and B are variables.

Steps:
1.

Grammar :

- 4 tuple as $G = \{V, T, P, S\}$

  V - Variables / Non-terminal symbols
  T - Terminals
  P - Production rules.
  S - Start symbol

In CFG,

$B \rightarrow 1 | B1$ ✓

$B1 \rightarrow 11 | B11$ ✗ (can't have two.)

Regular Grammar

1) Right Linear

- Non terminals lie to right of terminals
  $A \rightarrow xB$
  $A \rightarrow x$

e.g. $S \rightarrow abS | b$

2) Left Linear

- Non terminals lie to left of terminals.
  $A \rightarrow Bx$
  $A \rightarrow x$

E.g. $S \rightarrow Sbb | b$

Derivations from Grammar

- Set of all strings that can be derived from a grammar is said to be language generated from that Grammar.

G_3 : $S \rightarrow AB$
$A \rightarrow Aa | a$
$B \rightarrow Bb | b$

$S \Rightarrow AB$
$\Rightarrow aB$
$\Rightarrow ab$

$S \Rightarrow AB$
$\Rightarrow AaB$
$\Rightarrow aaB$
$\Rightarrow aab$

Similarly,
$S \Rightarrow abb$
$S \Rightarrow aabb$

$\therefore L(G_3) = \{ ab, a^2b, ab^2, a^2b^2 \ldots \}$

$\therefore L(G_3) = \{ a^n b^m \mid n > 0 \text{ and } m > 0 \}$

Context Free languages.

Removal of Left Recursion:

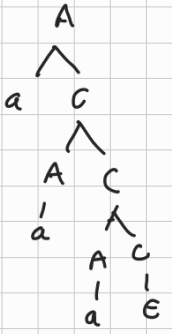
(1) if, $A \rightarrow Ax | c$
$A \rightarrow cB$ ⇒
(¥) $B \rightarrow xB | \epsilon$

e.g Remove Ambiguity in
$A \rightarrow AA | a$
→ $A \rightarrow aC$
$C \rightarrow AC | \epsilon$

(2) Left factoring:

if $A \rightarrow a\beta_1 | a\beta_2$, Confusion on what to expand A to.

$A \rightarrow aB$
$B \rightarrow \beta_1 | \beta_2$



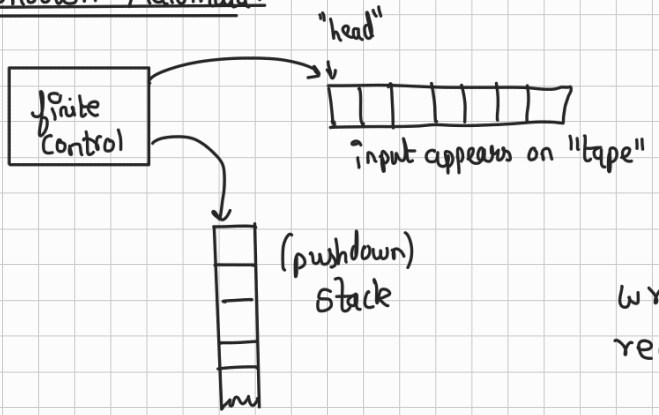
$u \Rightarrow v$ u yields v in one step

$u \overset{*}{\Rightarrow} v$ u yields v in some no. of steps

u derives v $u \Rightarrow v_1 \Rightarrow v_2 \ldots \Rightarrow v_k = v$ derivation of v from u.

It is not multicity of derivations that causes ambiguity but the existence of multiple parse trees.

Pushdown Automata:



write-add → push
read-remove → pop

(7 tuple) :→ $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

Γ → stack alphabet

$\delta : Q \times \Sigma_\varepsilon \times \Gamma_E \rightarrow P(Q \times \Gamma_E)$

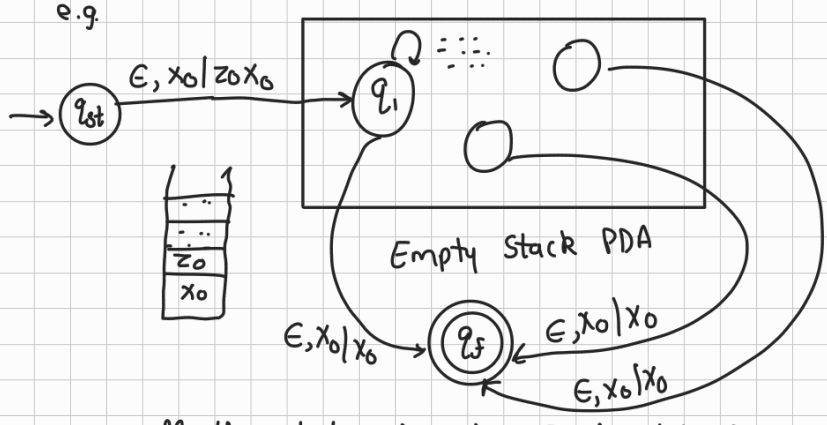
- $Q \times \Sigma_\varepsilon \times \Gamma_E$ → when we are on a particular state (Q) with a particular input symbol (Σ_E) with a particular stack top (Γ_E)
- $P(Q \times \Gamma_E)$ → when we have previous 3 we know what new state we can goto and what the new top will be

Convert Empty stack PDA to acceptance by Final state.

(EASY YYYY)

① Add a new initial state which sets the stack top to z_0 and below it another variable.

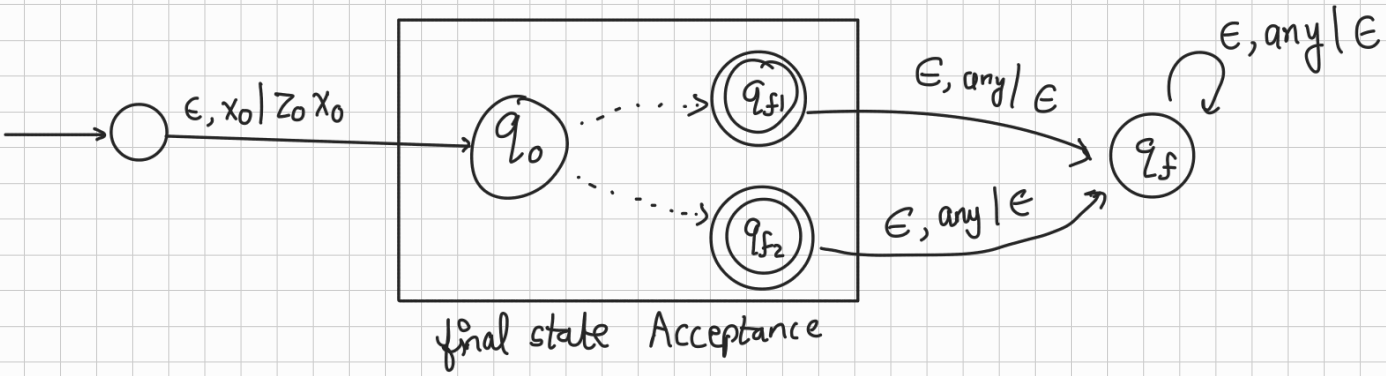
e.g.



② from all the states in the Empty state PDA, make a epsilon transition to final state if original stack top is present

// Tadaa ... done ... q_1 is the Stack Acceptance PDA.

From Final State Acceptance to Empty Stack Acceptance:

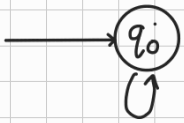


example:

$L = \{ \omega \in (0,1)^* \mid \omega$ has equal number of a's and b's $\}$

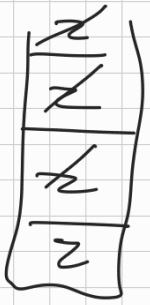
input Symbol		Stack Symbol
1	⟶	B
0	⟶	A

Acceptance by empty stack:



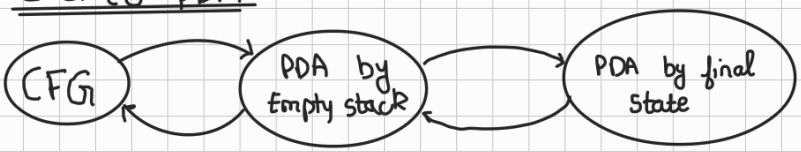
0, Z0 | A
1, Z0 | B
1, A | ϵ
1, B | B B
0, A | A A
0, B | ϵ
ϵ, Z0 | ϵ

~~iieeie~~



$G = \{$

CFG to PDA:



Steps:

1. Corresponding to each **variable A**, these in a **transition on ∈**:

$$\delta(q, \epsilon, A) = \{(q, \beta) \mid A \to \beta \text{ is a production of } G.\}$$

2. for each **terminal a**,

$$\delta(q, a, a) = \{(q, \epsilon)\}$$

e.g.

$S \longrightarrow 0S1 \mid A$
$A \longrightarrow 1A0 \mid S \mid \epsilon$

$\Sigma = \{0, 1\}$
$\Gamma = \{A, S, Z_0\}$

$P = \{\{q\}, \{0,1\}, \{0,1,A,S\}, \delta, q, S\}$

↑ Remember, you are also pushing terminals in stack here, so they as well go in stack variables.

$(q, \epsilon, S) = (q, 0S1)$
$(q, \epsilon, S) = (q, A)$
$(q, \epsilon, A) = (q, 1A0)$
$(q, \epsilon, A) = (q, S)$
$(q, \epsilon, A) = (q, \epsilon)$
$(q, 0, 0) = (q, \epsilon)$
$(q, 1, 1) = (q, \epsilon)$

$(q, \epsilon, S) = (q, 0S)$
$(q, \epsilon, S) = (q, A)$
$(q, \epsilon, A) = (q, 1A0)$
$(q, \epsilon, 0) = (q, 0)$
$(q, \epsilon, \epsilon) = (q, \epsilon)$

$(q, 0, 0) = (q, \epsilon)$
$(q, 1, 1) = (q, \epsilon)$

Accept String 010101 by empty stack PDA.

$(q, 010101, S) \vdash (q, 10101, 0S1) \vdash (q, 10101, S1) \vdash (q, 10101, A1) \vdash (q, 0101, 1A01)$
$\vdash (q, 0101, A01) \vdash (q, 0101, S01) \vdash (q, 0101, 0S101) \vdash (q, 101, S101)$
$\vdash (q, 101, A101) \vdash (q, 01, \epsilon 101) \vdash (q, 101, 101) \vdash (q, 01, 01) \vdash (q, 1, 1)$
$\vdash (q, \epsilon, \epsilon)$ ... Accepted by empty stack.

Questions at: Pg. 267 6.3.2, 6.3.3

PDA to CFG Conversions:

Given a PDA, we shall construct $G = (V, T, P, S)$, where the set of variables V consists of:

1. Special symbol S which is also the stack symbol.
2. All symbols of the form $[pXq]$, where $(p,q) \in Q$
   $X \in \Gamma$

3 operations, push, pop, skip

$Q = \{q, p\}$

push : $\delta(q, 1, z) = (q, xz)$

$[q, z, q] = 1\,[q, X, q][q, z, q]$

$[q, z, q] = 1\,[q, X\, p][p, z, q]$

$[q, z, p] = 1\,[q, X\, q][q, z, p]$

$[q, z, p] = 1\,[q, X\, p][p, z, p]$

Pop : $\delta(q, 1, x) = (q, \in)$

$[q, x, q] \rightarrow 1$

Skip : $\delta(q, 0, x) = (p, x)$ ← stack unchanged.

$[q, x, p] \rightarrow 0[p, x, p]$

$[q, x, q] \rightarrow 0[p, x, q]$

Deterministic PDA:

following conditions are met:

1. $\delta(q, a, X)$ has at most 1 member for any q in Q, a in Σ or $a = \epsilon$ and X in Γ.
2. If $\delta(q, a, X)$ is non-empty for some a in Σ, then $\delta(q, \epsilon, X)$ must be empty.

Todo: Pumping lemma for CFL and closure properties**

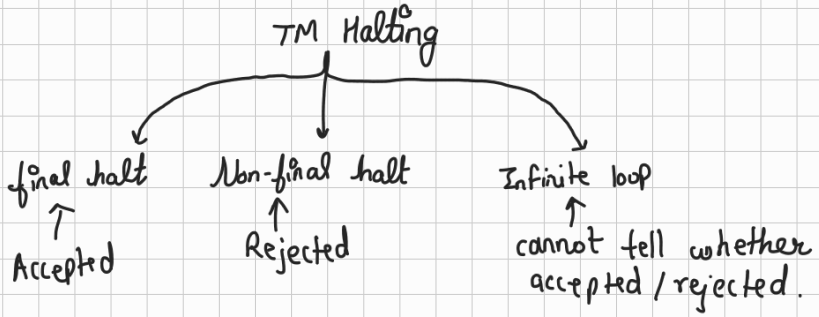
Turing Machine:

- Accepts Recursively enumerable languages
- A TM can do everything that a computer can do.
- TM is a mathematical model consisting of
  i) Infinite length tape
  ii) Read/Write head
  iii) Finite Control Unit

7-tuple $(Q, \Sigma, X, \delta, q_0, B, F)$

- X: tape alphabet
- B: Blank Symbol
- F: set of final states.

$$Q \times X \rightarrow Q \times X \times \{L, R\}$$



$L = \{ a^i b^j \mid i \leq 2j \}$

$L_1 = a^i b^j \quad i = 2j$

$L_2 = a^i b^j \quad i < 2j$

for L_1

$S_1 \rightarrow aaSb \mid \varepsilon$

for L_2:

$S_2 \rightarrow Bb \mid aBb$

$B \rightarrow Bb \mid aBb \mid aaBb \mid \varepsilon$

$S \rightarrow S_1 \mid S_2$

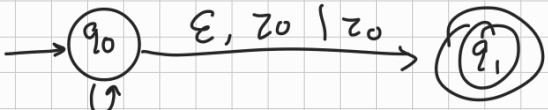
$L = \{ w \mid n_a(w) = n_b(w) + 1 \}$

a's always 1 more than b

$S \rightarrow SaSbS \mid SbSaS \mid \varepsilon$

DPDA for balanced Parantheses

$\{ \; (\; \} \;)$



$\{, z_0 \mid \{ z_0$

$(, z_0 \mid C z_0$

$(, C \mid CC$

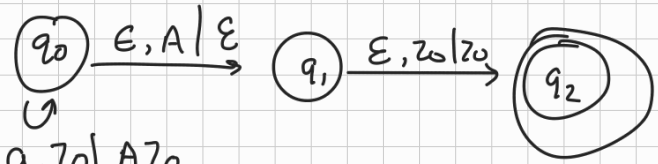
$(, \{ \mid C \{$

$\{, \{ \mid \{ \{$

$\{, C \mid \{ C$

$), C \mid \varepsilon$

$\}, \{ \mid \varepsilon$



$a, z_0 \mid A z_0$

$a, A \mid AA$

$b, A \mid \varepsilon$

$b, z_0 \mid B z_0$

$b, B \mid BB$

$A, B \mid \varepsilon$

aaab

aaabb

$[q_0, a, z_0] = [q_0, a z_0]$

$[q_0, a, A] = [q_0, AA]$

$0^n 1^m \quad n \geq m$

$S \rightarrow 0S1 \mid A$

$A \rightarrow 1A0 \mid S \mid \varepsilon$

$(q, \varepsilon, S) = (q, 0S1)$

$(q, \varepsilon, S) = (q, A)$

$(q, \varepsilon, A) = (q, 1A0)$

$(q, \varepsilon, A) = (q, S)$

$(q, \varepsilon, A) = (q, \varepsilon)$

$(q, 0, 0) = (q, \varepsilon)$

$(q, 1, 1) = (q, \varepsilon)$

$S \rightarrow xS$

$X \rightarrow aXb \mid ab$

$(q, \varepsilon, S) = (q, xS)$

$(q, \varepsilon, x) = (q, aXb)$

$(q, \varepsilon, x) = (q, ab)$

$(q, a, a) = (q, \varepsilon)$

$(q, b, b) = (q, \varepsilon)$

PDA to CFG:

$\delta(q,i,z) = (q, xz)$ $\quad \delta(q,e,x) = (q,\varepsilon)$ $\quad \delta(q,\varepsilon,z) = (q,\varepsilon)$

$S \rightarrow [qzq]$

$[q,z,q] \rightarrow i[q,x,q][q,z,q]$

$[q,x,q] \rightarrow e$

$[q,z,q] \rightarrow \varepsilon$

$(p,0,z) = (p, xz)$

$S \rightarrow pzp \mid pzq$

$[p,z,p] = 0[p\,x\,p][p\,z\,p]$

$[p,z,p] = 0[p\,x\,q][q\,z\,p]$

$[p,z,q] = 0[p\,x\,p][p\,z\,q]$

$[p,z,q] = 0[p\,x\,q][q\,z\,q]$

Pumping lemma for CFG

uv^ixy^iz

String s divided into 5 pieces,

① uv^ixy^iz is in A for every $i \geq 0$

② $|vy| > 0$

③ $|vxy| \leq p$

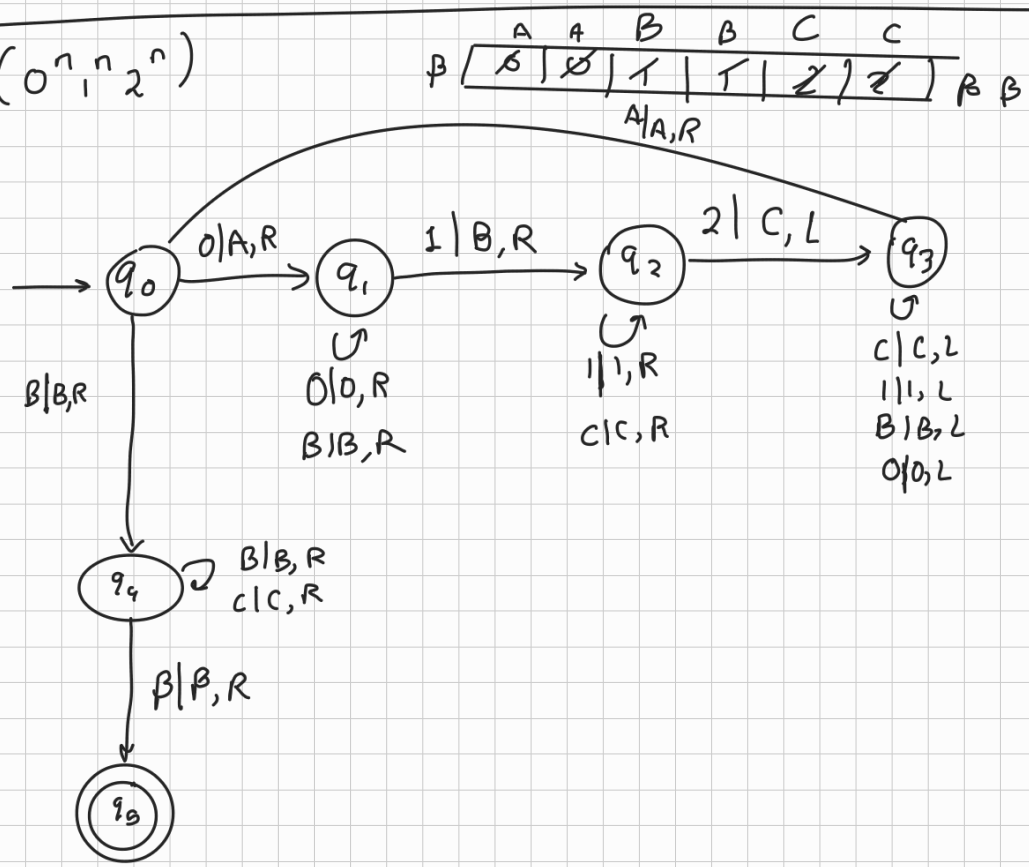
A – language

Turing Machine:

Configuration of TM (q, α_1, α_2)

- q: current state
- α_1: input read till now
- α_2: remaining input

$L = (0^n 1^n 2^n)$



TM's

Standard TM — left limited, Right Unlimited

Multi-track

Two way ∞ tape — left, Right unlimited