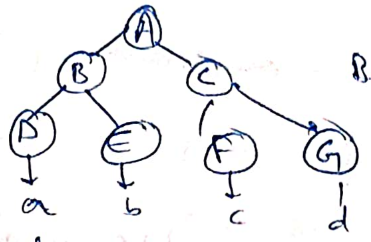
CNF $A \to BC$ 1) Binary Tree 2) for length n, min steps $(2(n)-1)$

$A \to a$ 2) length is restricted

• Membership

Var → Terminal

Var → V1 V2



Binary Tree

- $(n-1)$ steps to derive vars.
- n steps to derive terminals

Chomsky Normal Form.

\# learned CFG to CNF.

\# GNF $\boxed{A \to aV^*}$ $A \to \epsilon$ is also valid.

TOC

Greibach normal form

CFG to GNF

$S \to absb \mid aa$

$\boxed{S \to aYSY \mid aX}$ → GNF

$X \to a$

$Y \to b$

Left recursion

$A \to A\alpha \mid \beta$

$\Downarrow$

$A \to \beta A'$

$A' \to \epsilon \mid \alpha A'$

(remove)

$A \to \beta\alpha^*$

$B \to b \mid bAB \mid BBB$ Q.

(A) β_1 β_2 (A) α

$B \Rightarrow bB' \mid bABB' \mid$

$B' \to \epsilon \mid BBB'$ ← remove

direct

$B \to bB' \mid bABB' \mid b \mid bAB$

$B' \to BBB' \mid BB$

\# Push Down Automata → Deterministic PDA, → NPDA

$Q \times (\Sigma \cup \epsilon) \times \Gamma^* \to Q \times \Gamma^*$

(top symbol)

$\delta(q_0, a, z_0)$ — state, top, ↳ input

ww^R → NPDA ✓

NPDA more powerful than DPDA.

CFG to PDA convert

$\delta(q, \epsilon, \epsilon)$

$\delta(q, \text{input}, \text{bottom})$ state empty stack top

stack states

if empty stack, no final state.

PDA to CFG

longest question

3 rules

① $S \to [q_0, Z_0, P]$ for all P

② $\delta(q_0, x, A) = (P, B_1 B_2)$

write down transition

$[q_0, A, \_] \to x [P B_1 \_][\_ B_2 \_]$

③ $\delta(q, x, A) = (P, \epsilon)$ for deletion

$[q, A, P] \to x$

- Now, Remove useless production, present on RHS but not on LHS.
- remove $A \to aAA$

CFG ⇌ PDA are equivalent in power

if conversion possible

Two Stack PDA

aabbcc

PDA

(q, input, bottom stack) → $Q \times \Gamma^*$

$Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \to Q \times \Gamma^*$

$a^n b^n c^n d^n$

2 stack PDA

For conversion, be careful.

CFG → PDA

GNF — only V in stack.

No GNF — stack → V∪T may

Youtube

$S \to 0BB$

$B \to 0S \mid 1S \mid 0$

① $(q, 0, S) \to (q, BB)$

② $(q, 0, B) \to (q, S)$

③ $(q, 1, B) \to (q, S)$

④ $(q, 0, B) \to (q, \epsilon)$

⇒ (q, terminal, left) ⟶ (q, V*)

$S \to aSb$

$S \to ab$

① $\delta(q, \epsilon, S) = (q, aSb)$

② $\delta(q, \epsilon, S) = (q, ab)$

③ $\delta(q, a, a) = (q, \epsilon)$

④ $\delta(q, b, b) = (q, \epsilon)$

$\delta(q, \epsilon, \text{start}) = (q, \Gamma)$

$\delta(q, V, T) \to (q, \epsilon)$

CFL properties

Not for intersection and complementation
$\overline{L_1 \cap L_2} = \overline{L_1} \cup \overline{L_2}$

Decidibility

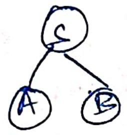
Lec 27

① Membership → string in language.

② Emptiness → First simplify CFG, check if S is useless

③ finiteness → if cycle, then infinite. Dependency Graph.

S → AB
A → a
B → b



finite.

• closed union, concatenation, kleene closure

* Turing Machine (bidirectional) can't accept ϵ $\{L = a^n b^n\}$

$\delta : (Q \times \Sigma) \rightarrow (Q \times \Sigma \times \{L, R\})$

curr state, curr. i/p alphabet (tape symbol), new state, new symbol, direction

$(q_0, a) \rightarrow (q_1, X, R)$ Transition

TM → can do multiplication

• TM is partially deterministic

• FA + Queue = TM.

• Non-Halting TM

• Halting problem of TM
(1) in ∞, but the computation
(2) is in computation, but is ∞

Turing Thesis

file 16,17

Infinite Turing Machines for each problem

• Every work that can be done by computer can also be done by TM.

NDTM → $\delta : Q \times \Sigma \rightarrow 2^{Q \times \Sigma \times \{L, R\}}$

Power of TM → languages accepted by TM. → not time & space complexity

Universal TM

A TM which can solve every problem

→ General, A TM can solve only 1 problem

transitions	input	states
T1	T2	T3

tape

TM as string of 0s & 1s

• Modifications in Turing Machine

- Language accepted by TM is Recursively Enumerable.
- ~~RL~~ is subset of ~~CFL~~.

Halting TM

final & non-final state

(Recursive)

Non Halting (RE) → $\boxed{TM}$

(Final, Non-final, ∞)

Recursively Enumerable.

- **Definitely Halt** → final state → accepted
  → non final state → not accepted

* Unrestricted Grammar

$u \to v$ {LHS ≠ ∈}

$u \in (V \cup T)^{+}$

$v \in (V \cup T)^{*}$

$\boxed{\text{Production}}$

Membership

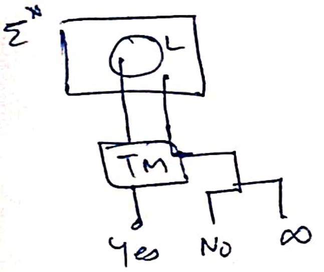
- RL, CFL, Rec. ~~…~~, CSL
- RE → no membership alg.

String, Machine → Yes / No

Halt

- Decidability (have algo)
- Reducibility

$P_1 \xrightarrow{\text{algo}} P_2$ ∀ algo
then P_1 has ~~a~~ algo



Halting Problem of TM is undecidable.
Proof

$\boxed{\text{State Entry Problem}}$

check whether ~~TM~~ entered state q.

use of Decid
Red

If a single string can be derived from 2 different paths, then grammar is **Ambiguous**

$\boxed{\text{Post Correspondence Problem.}}$

~~R~~ sequences of 2 strings. Check by arrangement, if string can be made equal.

HALTING TM → PCP → Ambiguity (CFG)

- **Modified PCP**

$w_1(w_2 \ldots w_j) = v_1(v_2 \ldots v_j)$

$\boxed{\text{Decidibility Table}}$

CYK Algo → membership algo for CFG