CSE 273/L-18/06.04.2025/

Chomsky Normal Form - CNF

⇒ either start with exactly two non-terminals or one terminal or allow empty string.

$$\left.\begin{array}{l} A \rightarrow BC \\ A \rightarrow 1 \\ A \rightarrow \epsilon \end{array}\right\}$$ three possibilities for production rules.

CFG to CNF:

(i) Reduce the CFG

- Remove useless symbols is optional.
  ⇒ if not applied the we may face redundant operation.

(ii) Convert the CFG + CNF.

CFG to CNF conversion:

Example - Page - 98

Need to convert: $S \rightarrow aAD$

$A \rightarrow aB \mid bAB$

following CNF Rule: $B \rightarrow b$

$D \rightarrow d$

Already in reduced grammar form.

$\Rightarrow$ $S \rightarrow aAD$

$\Rightarrow$ $S \rightarrow PAD$

$P \rightarrow a$

$\Rightarrow$ $S \rightarrow PQ$

$P \rightarrow a$

$Q \rightarrow AD$

} Now following CNF

$A \rightarrow aB$

$\Rightarrow$ $A \rightarrow PB$

$P \rightarrow a$

$A \rightarrow bAB$

$\Rightarrow$ $A \rightarrow BAB$

$B \rightarrow b$

$\Rightarrow$ $A \rightarrow \cancel{BA}\,BR$

$B \rightarrow b$

$R \rightarrow AB$

$\therefore$ Therefore, final CNF:

$S \rightarrow PQ$

$Q \rightarrow AD$

$P \rightarrow a$

$A \rightarrow PB$

$A \rightarrow BR$

$R \rightarrow AB$

$B \rightarrow b$

$D \rightarrow d$

(*) Another example:

$S \rightarrow aAbB$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

$D \rightarrow b$

$S \rightarrow AABB$ $\Rightarrow$ Not allowed

$\Rightarrow$ $A \rightarrow aA \mid a$

- not a single terminal production rule. its combined and mixed.
- we can reuse if the production rule contains only one terminal.

CYK - algorithm:

- only applicable to CNF
  - so, first check for CNF or not.

    then apply CYK.

⇒ fill the table ~~with~~ for different length of the string, starting from 1.

- first take one symbol at a time
- then take two symbol at a time

  ba ⇒ make sub-string

  b, a

- then take three symbol at a time

  baa ⇒ sub-string

  b, aa
  ba, a

  } move the seperable comma one by one.

  - then cross product the outcome of these.

L-19 /08.04.2025/

Push Down Automata:

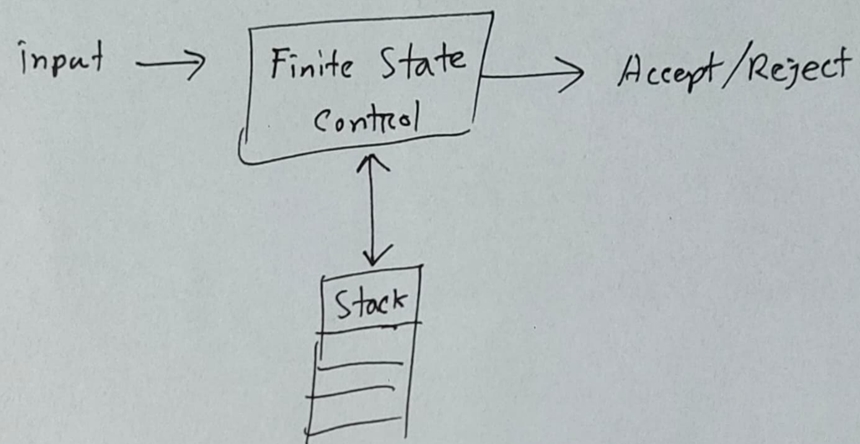
- Machine for CFG.
- use memory to store some information
  - use stack to store information.

} $0^n 1^n$, we need to store the number of '0'.

⊛ PDA ⇒ $\{Q, \Sigma, \Gamma, \delta, q_0, Z_0, F\}$

initial stack symbol, $Z_0 \in \Gamma$
- use to identify if the stack is empty or not.

Finite stack alphabet
- with full of flexibility

⇒ 

⊛ δ ⇒ transition function.

- Previous cases: FA,

$$\delta(A,0) = B$$

Decision depends on inputs only.

- But in PDA,
  decision depends on input ~~and~~ & stack both

$$\delta(q, a, X) = (p, \gamma)$$

new state

stack operation.

Q

Σ

top of the stack

* Stack operation: 4 types:

One transition function can choose any one from these.

1. (i) Push ⇒ add new item on top
2. (ii) Pop ⇒ Delete the top
3. (iii) Unchanged ⇒ No change
4. (iv) Replace ⇒ replace the top

* PDA, do not have any top operation. to see the top. we need to pop the element, and then see and push again to keep unchanged

* If ~~we push and~~

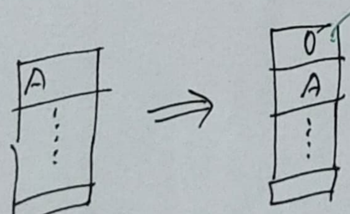
* If we pop and push same element, then no change. and if we push another element, then it is replace.

* Example:

$$\delta(q_1, 0, A) = (q_7, 0A)$$

Push operation.

add new item.



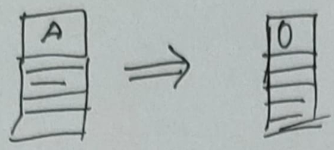
* $$\delta(q_7, 1, 0) = (q_2, \epsilon)$$

No input, which already poped is vanished.

⇒ Pop operation only.

7

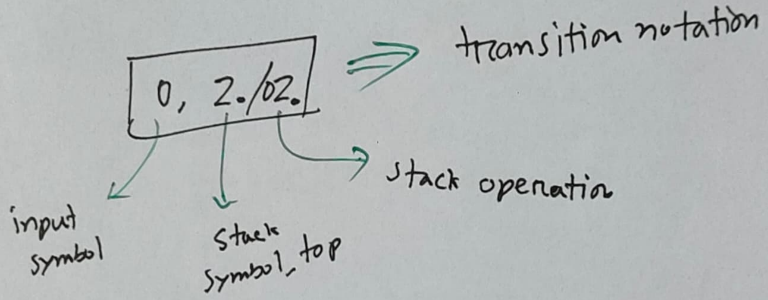
$\delta(q_1, 0, A) = (q_1, 0)$ ⇒ Replace operation



⊛ Unchanged Operation:

$$\delta(q_8, 0, A) = (q_5, A)$$

⊛ Some Notation of the machine graph:



⇒ if input symbol is 0, top of the stack is Z_0, then move to next state, and stack operation will be $0Z$.

⊛ From the example of $\boxed{\text{page - 104}}$

we can write,

no need to include 'ε'

$$P = (\{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

⊛ Number of transition from one state to another state depends on number of possible input and number of stack symbol. (Multiplication).

⊛ In the given example:

q_0 = push operation only

q_1 = pop operation only

For PDA,

$\epsilon \Rightarrow$ input is empty, no more input

$\Rightarrow$ we can ignore the next input.

⊛ $L = \{ww^R \mid w \text{ is in } (0+1)^*\}$

$\Rightarrow$ mirror string or palindrome

$\Rightarrow$

0110 0110

0111 1110

in the example:

assume that this is the mid point.

$q_0 \Rightarrow$ push all input until mid point

$q_1 \Rightarrow$ ~~RF~~ pop all and match with next input.

L-20/13.04.2025/

Quiz-2

⊛ Derivation of PDA:

$\boxed{\text{Slide - 106}}$

- Parse all state in each steps.
  - like brute force algorithm. if accepted break it.
- worst case, it can run upto the length of the string.

Deterministic PDA (DPDA):

- the set $\delta(q, s, \Gamma)$ has at most one element
- if $\delta(q, \epsilon, \Gamma)$ is not empty, then $\delta(q, s, \gamma) = \phi$

$\boxed{\text{Slide - 107}}$

- PDA acceptance
  - Empty Stack to Final state
  - Final state to empty stack

important for final.

$\boxed{\text{Slide - 108-110}}$

- PDA construction is not important.
- PDA will be given, apply different operation.

L-21 / 15.04.2025 /

Turing Machine:

- 7 tuples:

tape alphabet

$$(Q, \Sigma, \Gamma, q_0, B, F)$$

initial symbol of Γ / blank symbol.

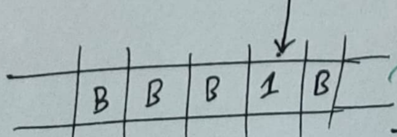
$\Rightarrow$ $\Sigma \in \Gamma$

$B \in \Gamma$, $B \notin \Sigma$

- tape:

head of the pointer

track of the tape



- we can move left & right
- we can change the pointer head position

$\delta \Rightarrow$ transition function:

$\delta(q, x)$

state

pointed symbol on tape

Example:

$\delta(q_0, 1) = (q_2, 0, R)$

replace the symbol 1 with 0.

current pointed symbol

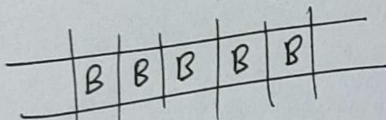
move the head position towards right direction.

R → Right

L → Left

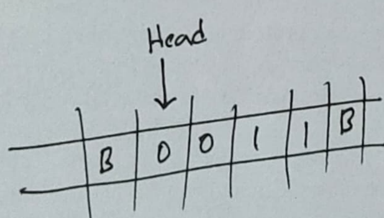
⇒ Next move direction on tape.

Tape:



- by default, all are B symbol, means blank.

⇒ first place the string on the tape, then start the machine.



From the given example, on page - 119

- if first zero found, replace with X and move to Right to find first one.
- if first one found, replace with Y and move to the Left to find X and first zero after X.

- Repeat until all are X, Y.

- When all are X, Y and B symbol found, ~~there~~ then accepted.

⇒ Here, state depends on logic.

q_0 ⇒ find the 0 and move to Right

q_1 ⇒ find the 1 and move to Left

q_2 ⇒ Go back to initial position

q_3 ⇒ check, is there any abnormality or other symbol exist, and send to final state.

q_4 ⇒ accept the string.

⊛ Construction of Turing Machine is not important for exam.

⊛ Graph or Table will be given, find out the string acceptance.

Online Class
16.04.2025
02:00 pm

L-22/16.04.2025/

⊛ Turing machine:

- models the ~~behavior~~ behaviour of a general-purpose computer.
- study the limits and capabilities of computation system.

* problem that can not be solved by a Turing machine is known as undecidable problem.

- halting problem.
  ⇒ if the problem runs infinite time or stuck on a loop, then it is known as the halting problem.

* Undecidable problem can be solved by approximation algorithms and heuristics.

  ↳ cheat code
  - use some cheat code, or extra information to solve the problem.

* Nondeterministic Polynomial (NP)

- Whatever the problem can be solved or not, ~~ett~~ varified in polynomial time.

* P vs NP (undecidable problem vs Non-deterministic Polynomial problem)

⇒

- undecidable problem fall outside the jurisdiction of NP.

- ~~Regardles~~ whather efficient verification algorithm exists or not, halting problem can not be solved.

* Halting State:

- Halting state and the final stat are not the same.
- where computation stops.

- no further transition occur,

⇒ - accepting state ⇒ can be final state
- rejecting state ⇒ trap state.

⊛ Turing machine will be given,

- process string
- derive the string

⊛ PDA will be given,

- process a string
- derivation
- acceptance conversion
- and some theory.

Final Exam

24.04.2025