CS-205

Books : Introduction to Automata Theory, Languages & computation ✓
- John E Hopcroft, Jeffrey D Ullman (Narosa, 1989)
- ——, Rajeev Motwani ——

Endsem - 50% | midsem - 20% | Quizzes - 2-4 - 20% | Relative

Elements of the theory of computation - Henry R Lewis, Christos H Papadimitriou - (Pearson Edition)

- Peter Linz
- Dexter C Kozen
- Thomas A Sudkamp

Tower of Hanoi

START →

Albhabets - Strings - Language

★ CLASSES OF LANG

1. Regular lang — finite automata
2. context free lang - push down automata
3. recursively enumerable lang - turing machine

★ ALPHABET

finite nonempty set - Alphabet (Σ) or (Γ)

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

symbols (a, b, c, d ... / 0, 1, 2, ... starting) convention

$\Sigma_1 = \{0, 1\}$

$\Sigma_3 = \{a\}$

$\Sigma_4 = \{0, 1, 2\}$

$\Sigma = \{\Delta, \square, \uparrow\}$ ✓ (But not conventional)

★ STRING

A string over an alphabet Σ is a finite sequence of symbols from Σ

CS-206

Books: Introduction to Automata Theory, Languages & Computation ✓
- John E Hopcroft, Jeffrey D Ullman (Narosa, 1989)
- ——, Rajeev Motwani ——

Endsem - 50% | Midsem - 30% | Quizes - 2-4 - 20% | Relative

Elements of the Theory of computation - Henry R Lewis, Christos H Papadimitrous - (Pearson Edition).

- Peter Linz
- Dexter C Kozen
- Thomas A Sudkamp

Tower of Hanoi

START ⇒

Alphabets - Strings - Language

★ CLASSES OF LANG

1. Regular lang — finite automata
2. context free lang - push down automata
3. recursively enumerable lang - turing machine

★ ALPHABET

finite nonempty set - Alphabet (Σ) or (Γ)

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

symbols: a, b, c, d, ... / 0, 1, 2, ... (starting) — convention

$\Sigma_1 = \{0, 1\}$

$\Sigma_3 = \{a\}$

$\Sigma_4 = \{0, 1, 2\}$

$\Sigma = \{\Delta, \square, \uparrow\}$ ✓ (But not conventional)

★ STRING

A string over an alphabet Σ is a finite sequence of symbols from Σ

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

- ab
- aab
- b
- ϵ - empty string (or λ)

length of string $|abaab| = 5$

$|\epsilon| = 0$

String & denoted by $\{ \ldots u, v, w, x, y, z \}$ constant

ending

$x = abaa$

$|x| = 4$

★ **LANGUAGE**

$\Sigma = \{0, 1\}$

set of all strings over Σ

$\Sigma^* = \{ \epsilon, 0, 1, 00, 01, 10, 11, 000, 001, \ldots \}$

$L \subset \Sigma^*$

L is said to be Language over alphabet Σ

CS-205

- Concatenation of strings

x, y

$x \cdot y \quad xy$

$x = a_1 a_2 \dots a_m$

$y = b_1 b_2 \dots b_n$

$xy = a_1 a_2 \dots a_m b_1 b_2 \dots b_n$

$|xy| = |x| + |y|$

$\varepsilon x = x \varepsilon = x$

- A_n = set of all strings of length n over Σ

$\Sigma^* = \bigcup_{n \geq 0} A_n$

① Substring : x is a substring of y if we can write $y = uxv$

② prefix : $y = xv \quad v = \varepsilon$

③ Suffix : $y = ux \quad v = \varepsilon$

- $|x|_a$ no. of occ. of a in x (variance alphabet)
- $|x|_y$ " " " substring y in x

$\Sigma = \{0, 1\}$ $\qquad \Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000 \dots\}$

- $L = \phi$ $\gg$ different
- $L = \{\varepsilon\}$
- L = any set of all strings that begin with 0
  2) End with 10
  3) has 010 as substring
  4) have even length

1) $L = \{0x \mid x \in \{0,1\}^*\}$

3) $L = \{x010y \mid x, y \in \{0,1\}^*\}$

4) $L = \{x \in \{0,1\}^* \mid |x| = 2n ; n \in \mathbb{Z}\}$

2) $L = \{x10 \mid x \in \{0,1\}^*\}$

Concatenation of languages

- L_1, L_2
  $L_1L_2 = \{xy \mid x \in L_1 \text{ and } y \in L_2\}$

$L_1 = \{0, 10\}$
$L_2 = \{11\}$
$L_1L_2 = \{011, 1011\}$

$\{\varepsilon\}L_1 = L_1\{\varepsilon\} = L_1$

$L_1L_2 \neq L_2L_1$ in general.

$LL = L^2$

$L^n = L^{n-1}L$

- Klean closure/star

$$L^* = \bigcup_{n\geq 0} L^n$$

$L^0 = \{\varepsilon\}$

① $L = \{01\}$

$L^* = L^0 \cup L^1 \cup L^2 \cup \ldots$

$= \{\varepsilon, 01, 0101, 010101, \ldots\}$

$L^* = \{(01)^n \mid n \geq 0\}$

② $\Sigma = \{0,1\}$ $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000 \ldots$

$L = \{0,1\}$ $L^* = L^0 \cup L^1 \cup L^2 \ldots = \{\varepsilon\} \cup \{0,1\} \cup \{00, 01, 10, 11\} \ldots$

$L^* = \{\varepsilon, 0, 1, 00, 01, 10, 11 \ldots$

$L^* = \Sigma^*$

- Positive closure

$$L^+ = \bigcup_{n\geq 1} L^n$$

(If ε is in L then only L^+ have ε otherwise no ε in L^+)

$\boxed{L^* = L^+ \cup \{\varepsilon\}}$

$L_1(L_2L_3) = (L_1L_2)L_3$

$L_1L_2 \neq L_2L_1$ in general

$L\{\epsilon\} = \{\epsilon\}L = L$

$L\phi = \phi L = \phi$

✓ $\phi^* = \{\epsilon\}$ $\times\phi$ $\boxed{\phi^0 = \{\epsilon\}}$?

$\{\epsilon\}^* = \{\epsilon\}$

★ $(L_1 \cup L_2)L_3 = (L_1L_3 \cup L_2L_3)$

Assume

$x \in (L_1 \cup L_2)L_3$

$\Leftrightarrow$ $x = x_1 x_2$

★ $\boxed{L^* L = L L^* = L^+}$

if $\epsilon \in L$, $L^* = L^+$

---

① $(L^*)^* = L^*$

② $(L_1L_2)^* L_1 = L_1(L_2L_1)^*$

③ $(L_1 \cup L_2)^* = (L_1^* L_2^*)^*$

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

$L = \{aba\}$ $\underbrace{\phi, \{\epsilon\}, \{a\}, \{b\}}_{\text{basis elements}}$

$\{a\}$ $\{b\}$

$\{ab\}$ $\{a\}$ $\{a\}^*$

$\{aba\}$

REGULAR EXPRESSION

★ Regular expression over Σ, recursively as follows.

1- ϕ, ϵ and a, for each a in Σ are regular expression representing the language ϕ, $\{\epsilon\}$ and $\{a\}$, respectively

2- If r_1 and r_2 are regular expressions representing languages R_1 and R_2 respectively then so are

a) $(r_1 + r_2)$ representing the language $R_1 \cup R_2$

b) $(r_1 r_2)$ representing the language $R_1 R_2$

c) (r_1^*) representing the language R_1^*

$\Sigma = \{0, 1\}$.

ϕ - ϕ

ϵ - $\{\epsilon\}$

a - $\{a\}$

$\Sigma^* = (0+1)^*$

L_1 - starting with 0 - $0(0+1)^*$

L_2 - such that 01 is a substring $(0+1)^* 01 (0+1)^*$

L_3 - Even length string. $((0+1)(0+1))^*$

L_4 - no. of zeroes is even -

odd -

exactly one occurance of 2 consequtive 0s

atmost

altleast

Every lang. doesnot have regular expression

Mental health and ...

Mental health:

WHO definition:

A state of comp...

being and not m...

informity

mental health: M...

A state of wel...

his/her abilit...

life, can wor...

make a contri...

Anxiety

Depress...

5-5-2015

17 Jan, Tuesday

No class Make up will be taken later

Regular language - representable by regex

ϕ . ~~ϵ~~ $\{a\}$

$\phi - \phi$

$\epsilon - \{\epsilon\}$

$a - \{a\}$

r - regex - $L(r)$ - language

$\Sigma = \{0,1\}$

doesnot begin with 10 :

$$(11+00+01)(0+1)^* + \epsilon + 0 + 1$$

$$r_1 \approx r_2 \quad \text{if} \quad L(r_1) = L(r_2)$$

$$r_1(r_2+r_3) \approx r_1r_2 + r_1r_3$$

$$r_1r_2 \not\approx r_2r_1 \text{ in general}$$

$$r\epsilon \approx \epsilon r \approx r$$

$$r_1(r_2r_3) \approx (r_1r_2)r_3$$

$$r\phi \approx \phi r \approx \phi$$

$$b^+(a^*b^* + \epsilon)b = b^+a^*b^+$$

$$b^+a^*b^*b + b^+b$$

$$b^+a^*b^+ + b^+b$$

$$b^+a^*b^+$$

$\{a^nb^m \mid m,n \geq 0\}$ a^*b^*

$\{a^nb^n \mid n \geq 0\}$ no regular exp. ← not regular

proof?

$\{a^p \mid p \text{ is a prime}\}$ ✗ regular

is n a prime no ?
if $a^n \in \{a^p \mid p$ is a prime $\}$

~~context~~ free grammar

Sentence $\Rightarrow$ <Sub><word><obj>
$\Rightarrow$ NP V Obj
$\Rightarrow$ (Article N) V Obj
$\Rightarrow$ ~~the~~ The boy eats NP
$\Rightarrow$ The boy eats N(icecream)

Formally, a context free grammar (CFG) is a quadruple

$G = (N, \Sigma, P, S)$

where N is a finite set of nonterminal
Σ is a finite set of terminals
$S \in N$ is the start symbol
P is a finite set of production rules / production

P: (A, α)
~~string over the set $N \cup \Sigma$~~
~~$\alpha \in (N \cup \Sigma$~~ $\alpha \in (N \cup \Sigma)^*$
$A \in N$

$(A, \alpha) = A \to \alpha$

$(N \cup \Sigma)^*$
$\left(\underset{\text{derives}}{\Rightarrow}\right)$ binary relation

$\alpha \overset{G}{\Rightarrow} \beta$
$\alpha = \alpha_1 A \alpha_2$
$\beta = \alpha_1 \gamma \alpha_2$
$A \to \gamma$
$\alpha_1 A \alpha_2 \underset{G}{\Rightarrow} \alpha_1 \gamma \alpha_2$

$\alpha \overset{*}{\Rightarrow} \beta$
reflexive
transitive closure

Length of derivation
$\alpha \overset{*}{\Rightarrow} \beta$

$G = (N, \Sigma, P, S)$

$\alpha \in (N \cup \Sigma)^*$
Sentential form

$S \overset{*}{\Rightarrow} \alpha$

α is a sentence (word) (string) if α is ~~a~~ string of terminals

$(N \cup \Sigma)^*$ → lowercase greek letters

$S \overset{*}{\Rightarrow} x$
(string from alphabet) → lower case letter
S for starting symbols - (Non terminals capital letters)

Language generated by grammar
$L(G) = \{ x \in \Sigma^* \mid S \overset{*}{\Rightarrow} x \}$

$\Sigma = \{a\}$

$\left(\begin{array}{l} S \to \varepsilon \\ S \to aS \end{array} \right) \leftarrow \Sigma^*$

$S \to aS \to a$
$(aS\ S \to \varepsilon)$

$N = \{S\}$
$P = \{ S \to \varepsilon,\ S \to aS \}$

$= \{ S \to \varepsilon \mid aS \}$
~~SO~~ S is the start symbol.

$\Sigma = \{a, b\}$
every string contains at least 2 a's
at most
exactly
set of all string having even, odd no. of a's, even a's, odd b's

CS-205

$G = (N, \Sigma, P, S)$

$L(G) = \{ x \in \Sigma^* \mid S \overset{*}{\Rightarrow} x \}$

$L(G) = \{ \}$

G_1, $S \rightarrow \varepsilon \mid aS$

$L(G) = \{ a^n \mid n \geq 0 \}$

a^*

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

$S \rightarrow aS \mid bS \mid \varepsilon$

$(a+b)^*$

$S \rightarrow abS \mid \varepsilon$

$(ab)^*$

$S \rightarrow aSb \mid \varepsilon$

$L(G) = \{ a^n b^n \mid n \geq 0 \}$

~~$S \rightarrow aSb \mid bSa \mid \varepsilon$~~

$S \rightarrow aSa \mid bSb \mid \varepsilon$

palindromes of even length

$S \rightarrow aSb \mid bSa \mid \varepsilon$

All strings with Equal no. of a and b

$S \rightarrow aSa \mid bSb \mid a \mid b$ ← palindromes of odd length

$S \rightarrow aSa \mid bSb \mid a \mid b \mid \varepsilon$ ← all palindromes

~~Set~~ Having exactly two a

$b^* a b^* a b^*$

G1 $S \rightarrow BaBaB$

$B \rightarrow bB \mid \varepsilon$

Q. $S \to bS \mid aA$
$A \to bA \mid aB$
$B \to bB \mid \varepsilon$

$b^* a b^* a b^*$

$L(G_1) = L(G_2)$

at least two a's
at most two a's

even no. of a's

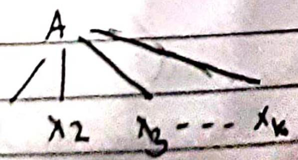
$\Sigma = \{a, b, c\}$
Substring abc doesnot occur in any string

$S \to bS \mid cS \mid aB \mid \varepsilon$
$B \to aB \mid cS \mid bC \mid \varepsilon$
$C \to aB \mid bS \mid \varepsilon$

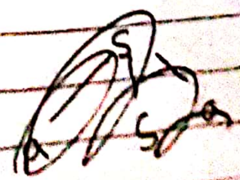
Parse tree or Derivation tree

$S \overset{*}{\Rightarrow} \alpha$ $\qquad$ $S \overset{*}{\Rightarrow} x$ $\quad x \in \Sigma^*$
$\alpha \in (N \cup \Sigma)^*$

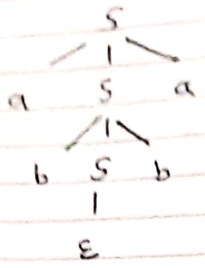
$A \longrightarrow X_1 X_2 \ldots X_k$
$X_i \in (N \cup \Sigma)$



yield = all symbols of leaves from left to right



$S \to aSb \mid bSb \mid \varepsilon$
$S \Rightarrow abba$: $S \Rightarrow aSa \Rightarrow abSba \Rightarrow abba$



$S \rightarrow S+S \mid S*S \mid (S) \mid a \mid b$

$a+b*a$

① $S \Rightarrow S+S \Rightarrow S+S*S \Rightarrow a+S*S \Rightarrow a+b*S$

$\Downarrow$

$a+b*a$

lmc #deriv ② $S \Rightarrow S*S \Rightarrow S+S*S$ ....

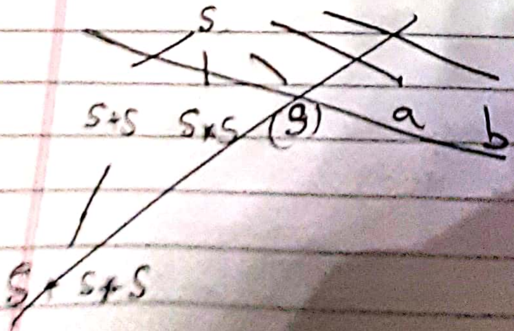
③ $S \Rightarrow S*S$

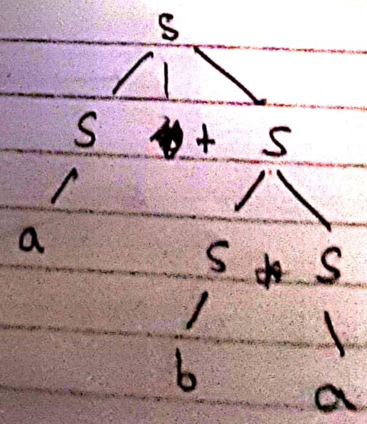
$\Rightarrow S*a$

$\Rightarrow S+S*a$

$\vdots$

$a+b*a$





Leftmost derivation:

Ambiguous gramm...

we have 2

In $a+b*a$

Leftmost derivation:

$S \underset{lm}{\Rightarrow} x_1 x_2 \ldots x_1 \ldots x_k$

first non terminal send " $x_1 \rightarrow \beta$

$\Rightarrow x_1 x_2 \ldots \beta \ldots x_k$

first non terminal $x_j \rightarrow \beta$ and repeat

Ambiguous grammar: $L(G)$ if for any string in grammar we have 2 leftmost derivations.

In $a + b * a$ we have 2 leftmost derivations.

5-20

Context-free grammar (CFG)

$G = (N, \Sigma, P, S)$

$A \to \alpha \quad A \in N, \ \alpha \in (N \cup \Sigma)^*$

context free language (CFL)

if $A \to \alpha \in P \Rightarrow \alpha = xBy$ or $\alpha = x$

→ linear grammar

right linear grammar if

$A \to xB \quad x \in \Sigma^*$

$A \to x \quad B \in N$

Left linear

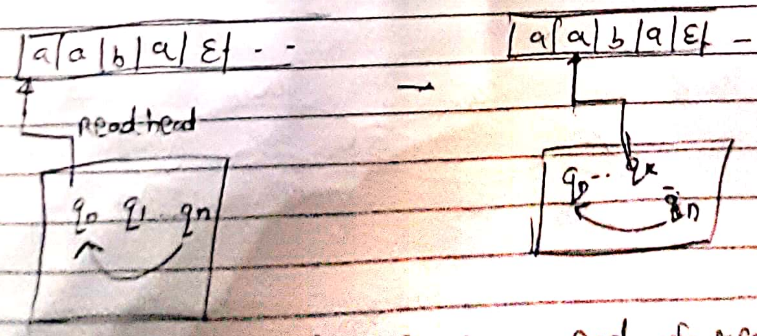
$A \to Bx \quad x \in \Sigma^*$

$A \to x \quad B \in N$

A lang. generated by right linear is regular.

Deterministic finite Automata (DFA)

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



if it enters final state on end of reading it is ~~excepted~~ accepted.

Formally, a DFA A is a quintuple

$A = (Q, \Sigma, \delta, q_0, F)$

Q · finite set of state

Σ - input alphabet

$q_0 \in Q$ is the start/initial state

$F \subseteq Q$ is the set of final or accept states

δ - state transition function

$Q \times \Sigma \rightarrow Q$ is the (state) transition f

$\delta(q, a) = p$

Total - for every symbol it goes to some state

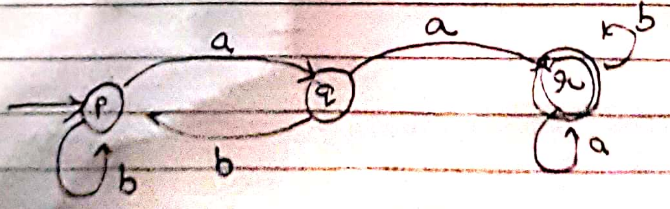
$A = (Q, \Sigma, \delta, p, F)$

$Q = \{p, q, r\}$

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

$F = \{r\}$

δ	a	b
$\rightarrow$ p	q	p
q	r	p
r	r	r



Any string which contains aa as substring.

Extended transition f

$\hat{\delta}(q, x)$

1. $\hat{\delta}(q, \epsilon) =$

2. $\hat{\delta}(q,$

$\hat{\delta}(p, aba) =$

$\hat{\delta}(p, ab)$

$\hat{\delta}(p,$

$\hat{\delta}(q$

Lang

Extended transition function

$\delta(q, a) = p$

$\hat{\delta}(q, x)$

1. $\hat{\delta}(q, \epsilon) = q$
2. $\hat{\delta}(q, xa) = \delta(\hat{\delta}(q, x), a)$

$\hat{\delta}(p, aba) = \delta(\hat{\delta}(p, ab), a)$

$\hat{\delta}(p, ab) = \delta(\hat{\delta}(p, a), b)$

$\hat{\delta}(p, a) = \delta(\hat{\delta}(p, \epsilon), a)$
$= \delta(p, a)$

$= \delta(\delta(\delta(p, a), b), a)$

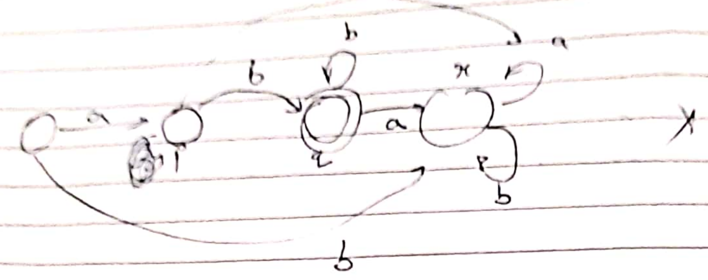
$\hat{\delta}(q_0, x) \in F \Rightarrow$ accepted

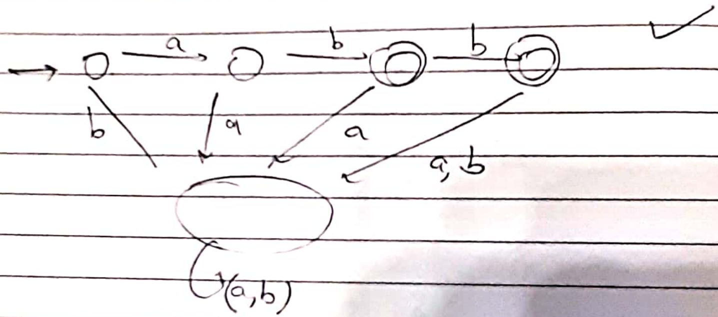
$L(A) = \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in F\}$
Language of a DFA over Σ

$L(A) = \{x \in \Sigma^* \mid$ aa is a substring of $x\}$
$|x|_{aa} \geq 1$

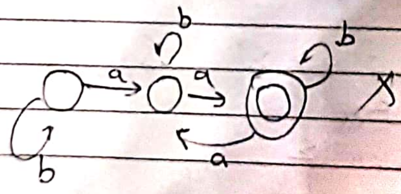
$\Sigma = \{a, b\}$

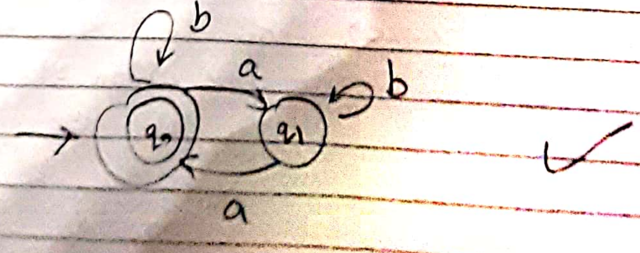
$L = \{ab, abb\}$





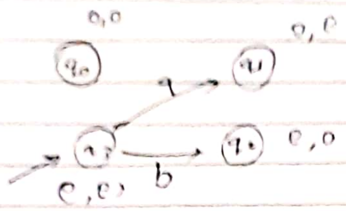
even no. of a's





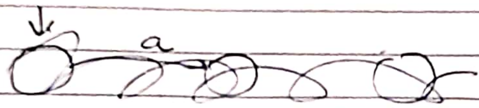
having odd no of a's

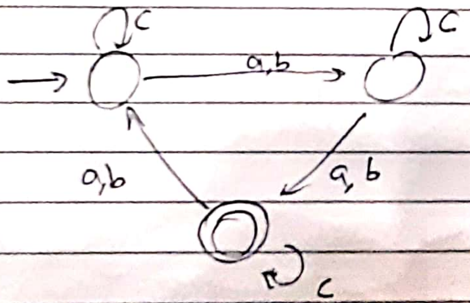
having a is even b is no odd



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

$L = \{x \in \Sigma^* \mid |x|_a + |x|_b \equiv 2 \bmod 3$





club ab as a single unit

$\Sigma = \{0, 1\}$

s.t binary digit should be a multiple of 3

11 ✓

101 ✗

110 ✓

$L = \{x \in \{a, b\}^* \mid |x| = 2 + 3n\}$

$L = \{a^m b^n \mid m \geq 1, n \geq 0\}$

$L = \{$ $\{a, b\}$ begin and end with the same ~~same~~ symbol

$L = \{a^n b^n \mid n \geq 0\}$ ✗ DFA

Length prime ✗

A lang accepted by DFA is regular

CS-205

Configurations or

$Q \times \Sigma^*$

x

(q_0, x)

$c = (q, x$

$c' = (p, y$

(q, x)

c

$c =$

$L(A) = \{$

Non determ

CS-203

Configurations or Instantaneous Description (ID)

$Q \times \Sigma^*$

x

$(q_0, x) \longrightarrow (p, \varepsilon)$

$C = (q, x)$

$C' = (p, y)$

$(q, x) \vdash (p, y)$ provided $x = ay$
$\delta(q, a) = p$

$C \vdash^* C'$ $C_0, C_1, C_2 \ldots, C_n$

$C = C_0 \vdash C_1 \vdash C_2 \vdash \ldots \vdash C_n = C'$ ← computation of DFA

$L(A) = \{ x \in \Sigma^* \mid (q_0, x) \vdash^* (p, \varepsilon)$ for some $p \in F \}$
language of the DFA

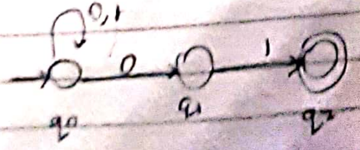
Non deterministic finite automata (NFA)

$N = (Q, \Sigma, \delta, q_0, F)$

$\delta : Q \times \Sigma \rightarrow \cancel{P(Q)}\ 2^Q$ $\quad P(Q)$

$\hat{\delta} : Q \times \Sigma^* \longrightarrow 2^Q$

All strings of form x01



$\delta(q_1, 0) = \phi$
$\delta(q_2, 0) = \phi$
$\delta(q_2, 1) = \phi$

δ	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	ϕ	$\{q_2\}$
(q_2)	ϕ	ϕ

0001 ✓

A string is accepted if there is a way to go from start to final state.

~~$L(N) = \{x \in \Sigma^* \mid$~~

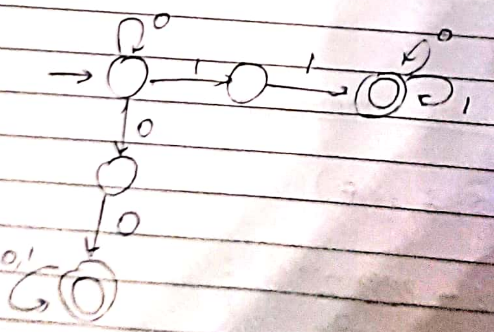
$$\hat{\delta}(q, \epsilon) = \{q\}$$

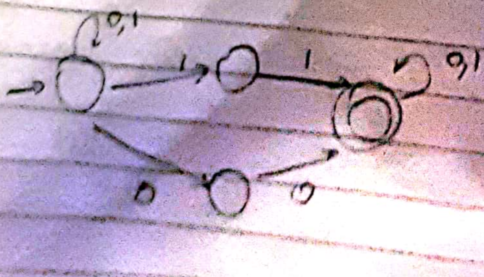
$$\cancel{\hat{\delta}(q, xa) = \delta(\hat{\delta}(q,x), a)}$$

$$\hat{\delta}(q, xa) = \bigcup_{p \in \hat{\delta}(q,x)} \delta(p, a)$$

$$L(N) = \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \cap F \neq \phi\}$$

Set of all string s.t 00 or 01 is a substring.





$N = (Q_N, \Sigma, \delta_N, q_0, F_N$ ...

$D = (Q_D, \Sigma, \delta_D, \{q_0\}$ ...

Q_D = set of subsets of ...

F_D = Set of subsets(S) ...

$\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p$ ...

Theorem If $D = (Q_D, \Sigma, \delta_D, F$ ...

constructed from NFA

$N = (Q$ ...

constru...

Proof: Induction on length ...

$|w| = 0$

$\hat{\delta}_D(\{q_0\}, w) =$ ...

✓ Base $|w| = 0$

$w = \epsilon$

$\hat{\delta}_D(\{q_0\}, \epsilon$ ...

$\hat{\delta}_N(q_0, \epsilon)$ ...

IS — $|w| = n+1$

$w = xa$

$\hat{\delta}_D(\{q$ ...

$\hat{\delta}_D(\{q_0\}, xa$ ...

$N = (Q_N, \Sigma, \delta_N, q_0, F_N)$
$D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$

Q_D - set of subsets of Q_N

F_D = set of subsets S of Q_N s.t. $S \cap F_N \neq \phi$

$\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a) \quad \forall\ S \subseteq Q_N$ or $S \in Q_D$

Theorem If $D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ is the DFA constructed from NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ by the subset construction, then $L(D) = L(N)$

Proof: Induction on length of string

$|w| = 0$

$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ ← To prove

Base $|w| = 0$

$w = \epsilon$

$\hat{\delta}_D(\{q_0\}, \epsilon) = \{q_0\}$

$\hat{\delta}_N(q_0, \epsilon) = \{q_0\}$

IS — $|w| = n+1$

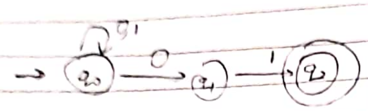
$w = xa$

$\hat{\delta}_D(\{q_0\}, x) = \hat{\delta}_N(q_0, x)$ by IH

$\hat{\delta}_D(\{q_0\}, xa) = \delta_D(\hat{\delta}_D(\{q_0\}, x), a)$
$= \delta_D(\hat{\delta}_N(q_0, x), a)$?
$= \hat{\delta}_N(q_0, xa)$

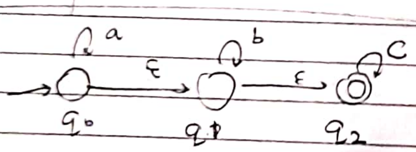
$\delta_D(\{p, q\}) = \bigcup$

ε-NFA



DFA

δ_D	0	1
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0 q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$



$a^* b^* c^*$

CS-207

$S = \{a, b, \ldots, z\}$

$|S| = n$

T

code

$\gamma : S \rightarrow N$

$\alpha \rightarrow 1010$

$\gamma(\alpha) = 1010$

CODEWORD

① Fixed length code

Average bits for symbol

$= \sum_{x \in S} f_x |\gamma(x)|$

$f_a = 0.32$, $f_b = 0.25$

a 11
b 01
c 001
d 10
e 000

Space efficient code

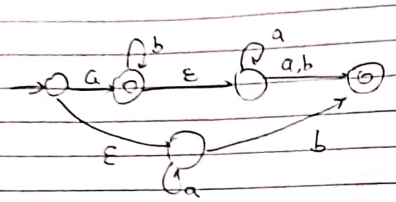
a 0
b 1
c 0

CS-205

NFA with ε-transitions (ε-moves)
(ε-NFA)

$N = (Q, \Sigma, \delta, q_0, F)$

$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$



$ab^* + a^*b + ab^*a^*(a+b)$

$L(N) = \{ x \in \Sigma^* \mid \hat{\delta}(q_0, x) \cap F \neq \phi \}$

ε-closure of a state
$q \in E(q)$

If $p \in E(q)$
& $r \in \delta(p, \varepsilon)$
$\Downarrow$
$r \in E(q)$

$E(S) = \bigcup_{p \in S} E(p)$

$\hat{\delta} : Q \times \Sigma^* \rightarrow 2^Q$

1. $\hat{\delta}(q, \varepsilon) = E(q)$
2. $\hat{\delta}(q, xa) = E\left(\bigcup_{p \in \hat{\delta}(q,x)} \delta(p, a) \right)$

Theorem For every NFA with ε transition ... transition

$N = (Q, \Sigma, \delta, q_0, F)$ NFA ...

$N' = (Q, \Sigma, \delta', q_0, F')$

To prove: $L(N) = L(N')$

$\delta'(q, a) = \hat{\delta}(q, a)$

$F' = F \cup \{ q \in Q \mid E(q) ...$

Induction on length of x

$\hat{\delta}'(q_0, x) = \hat{\delta}(q_0, x)$

$x = \varepsilon$

$\hat{\delta}'(q_0, \varepsilon) = \{q_0\}$
$\hat{\delta}(q_0, \varepsilon) = E(q_0)$

$\varepsilon \in L(N) \Leftrightarrow \hat{\delta}(q_0 ...$
$\Leftrightarrow$ ...
$\Leftrightarrow$ ...
$\Leftrightarrow \varepsilon \in$ ...

Base case length 1 ...

Theorem: For every NFA with ϵ transition $\exists$ a NFA without ϵ transition.

$N = (Q, \Sigma, \delta, q_0, F)$ NFA with ϵ-moves

$N' = (Q, \Sigma, \delta', q_0, F')$

To prove: $L(N) = L(N')$

$\delta'(q, a) = \hat{\delta}(q, a)$

$F' = F \cup \{q \in Q \mid E(q) \cap F \neq \phi\}$

Induction on length of x

$\hat{\delta}'(q_0, x) = \hat{\delta}(q_0, x)$

$x = \epsilon$ ✗

$\hat{\delta}'(q_0, \epsilon) = \{q_0\}$
$\hat{\delta}(q_0, \epsilon) = E(q_0)$ ✗

$\epsilon \in L(N) \Leftrightarrow \hat{\delta}(q_0, \epsilon) \cap F \neq \phi$
$\Leftrightarrow E(q_0) \cap F \neq \phi$
$\Leftrightarrow q_0 \in F'$
$\Leftrightarrow \epsilon \in L(N')$

Base case length 1

$x = wa$

$\hat{\delta}'(q_0, wa) = \delta'(\hat{\delta}'(q_0, w), a)$ by IH

$= \delta'(\hat{\delta}(q_0, w), a)$

$= \hat{\delta}(\hat{\delta}(q_0, w), a)$

$= \hat{\delta}(q_0, wa)$

$\hat{\delta}'(q_0, x) = \hat{\delta}(q_0, x)$

$x \in L(N')$

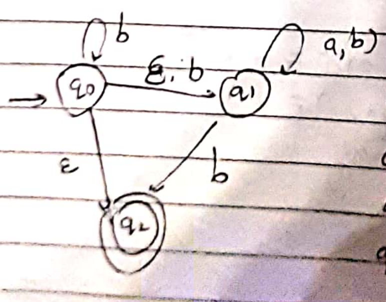
$\hat{\delta}'(q_0, x) \cap F' \neq \phi$

IF $\hat{\delta}'(q_0, x) \cap F \neq \phi$

$q \in \hat{\delta}(q_0, x)$ s.t $E(q) \cap F \neq \phi$

$\Rightarrow q \in \hat{\delta}(q_0, x)$ and $E(q) \cap F \neq \phi$

$q \in F'$



$\delta' = \hat{\delta}$	a	b	ε
q_0	ϕ	$\{q_0, q_1\}$	$\{q_1, q_2\}$
q_1	$\{q_1\}$	$\{q_1, q_2\}$	ϕ
q_2	ϕ	ϕ	ϕ

δ	a	b
$\rightarrow q_0$	$\{q_1\}$	$\{q_0, q_1, q_2\}$
q_1	$\{q_1\}$	$\{q_1, q_2\}$
q_2	ϕ	ϕ

$$\delta'(q_0, a) = \hat{\delta}(q_0, a)$$

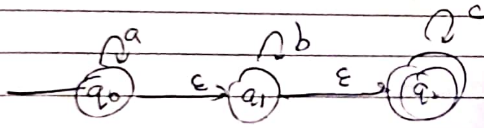
$$= E(\delta(\hat{\delta}(q_0, \varepsilon), a))$$

$$= E(\delta(\{q_0, q_1, q_2\}, a))$$

$$= E(\{q_1\})$$

$$= \{q_1\}$$

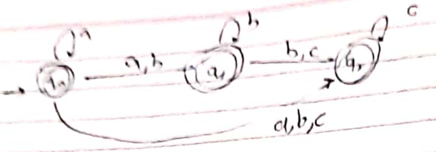
Construct DFA



$\delta' = \hat{\delta}$	a	b	c	ε
$\rightarrow q_0$	$\{q_0\}$	ϕ	ϕ	~~{q0, q1}~~ $\{q_0, q_1, q_2\}$
q_1	ϕ	$\{q_1\}$	ϕ	$\{q_1, q_2\}$
q_2	ϕ	ϕ	$\{q_2\}$	$\{q_2\}$

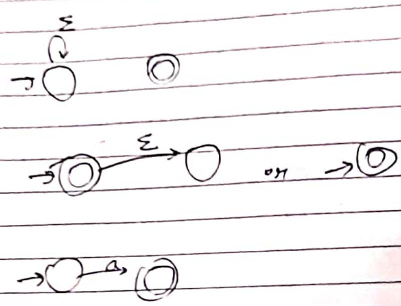
δ	a	b	c
$\rightarrow q_0$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	ϕ	$\{q_1, q_2\}$	$\{q_2\}$
q_2	ϕ	ϕ	$\{q_2\}$

Every finite automata has equivalent RegEx and vice versa.

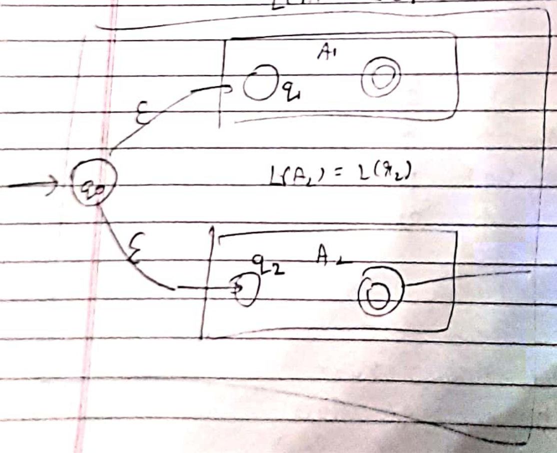


regex → finite automata

$L(\phi)$



$L(A_1) = L(r_1)$



can make final state equal

$A_1 = (Q_1, \Sigma, q_1, \delta_1, F_1)$

$A_2 = (Q_2, \Sigma, q_2, \delta_2, F_2)$

$A = (Q, \Sigma, q_0, \delta, F)$

$Q = Q_1 \cup Q_2 \cup \{q_0\}$

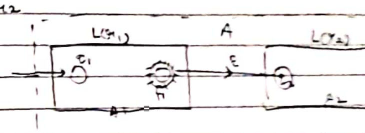
$F = F_1 \cup F_2$

$\delta(q,a) = \begin{cases} \delta_1(q,a) \\ \delta_2(q,a) \\ \{q_1, q_2\} \end{cases}$

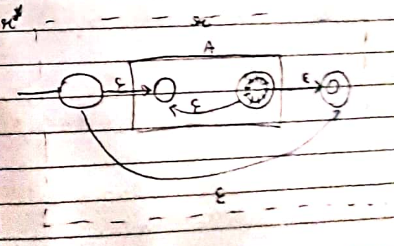
For every regular expression there exists finite automata

$L(r) = L(A)$

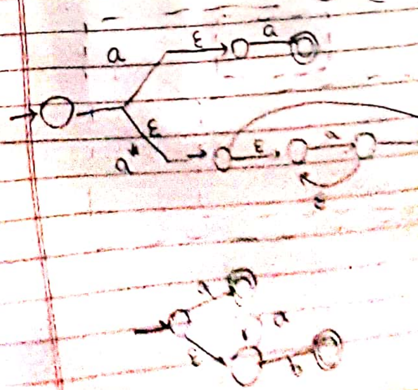
$r_1 r_2$



r^*



$a^* b + a$

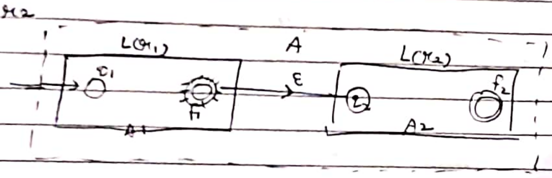


$$\delta(q,a) = \begin{cases} \delta_1(q,a) & q \in Q_1 \\ \delta_2(q,a) & q \in Q_2 \\ \{q_1, q_2\} & q = q_0 \ \& \ a = \varepsilon \end{cases}$$

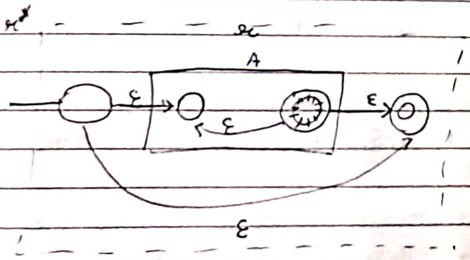
For every regular expression There exist an equivalent finite automata

$L(r) = L(A)$

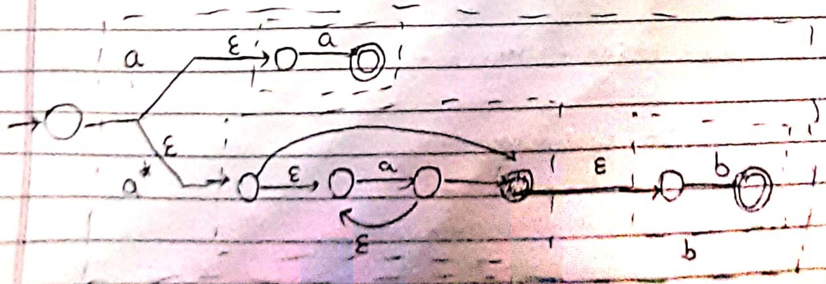
$r_1 r_2$

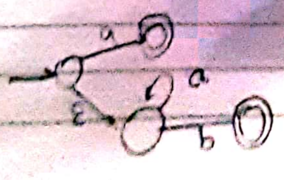


r^*



$a^* b + a$





For every DFA $\exists$ an equivalent regex.

$A = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1 \_\_\_ q_n\}$

$F = \{q_{f_1}, q_{f_2} \_\_\_ q_{f_k}\}$ $\quad k \le n$

$R_i = \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) = q_i\}$

assume R_i has regex r_i

$L(A) = R_{f_1} \cup R_{f_2} \_\_\_ R_{f_k}$

$r = r_{f_1} + r_{f_2} + \_\_\_ r_{f_k}$

$q_i \xrightarrow{a_1, a_2 \ldots a_k} q_j$

$S_{ij} = a_1 + a_2 + \_\_\_ a_k$ (regex for i to j)

$R_j = R_0 S_{0j} \cup R_1 S_{1j} \_\_\_ R_n S_{nj}$

$r_j = r_0 S_{0j} + r_1 S_{1j} + r_2 S_{2j} \_\_\_ r_n S_{nj}$

★ $r_0 = r_0 S_{00} + r_1 S_{10} + r_i S_{i0} \; r_n S_{n0} + \varepsilon$

$r_1 = r_0 S_{01} + r_1 S_{11} + r_i S_{i1} \; r_n S_{n1}$

$\vdots$

$r_j = r_0 S_{0j} + r_1 S_{1j} + \ldots r_i S_{ij} + \_\_\_ r_n S_{nj}$

$\vdots$

$r_n = r_0 S_{0n} + r_1 S_{1n} + \_\_\_ r_i S_{in} + \_\_\_ r_n S_{nn}$

s, t known regex

$r = t + rs$

Arden's principle

$r = ts^*$

$r_0 = r_0 b + r_1 a$

$r_1 = r_0 a + r_1 b$

$r_0 = (\varepsilon + r_1 a) b^*$

$r_1 = (\varepsilon + r_1 a) b^* a + \ldots$

$= r_1 (ab^* a) + \ldots$

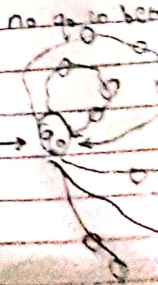
$\boxed{r_1 = b^* a (b + \ldots}$

Thm - ~~If L = L(A)~~, If A is a ...
by a regular express...

Proof - $A = (Q, \Sigma, \delta, q_0, F)$
Induction on $|Q|$
Consider a DFA w...
1 - $F = \phi$ or $F = \ldots$
$r = \phi$ $\quad r = \ldots$

IH - True for $< n$ sta...

IS - $L(A) = L_1^* L_2$ ...
set of string...
from q_0 to ...
no q_0 in bet...



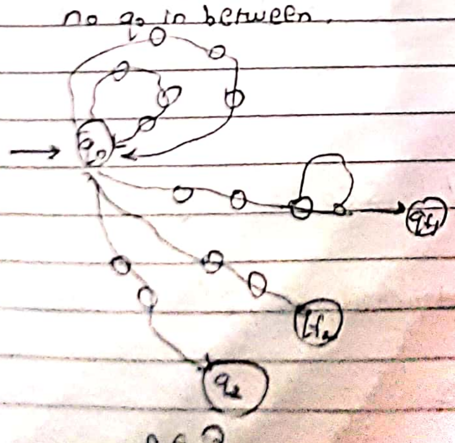
$L(A) = r_1$

$r_0 = r_0 b + r_1 a + \epsilon$
$r_1 = r_0 a + r_1 b$
$r_0 = (\epsilon + r_1 a) b^*$ ← from Arden's principle
$r_1 = (\epsilon + r_1 a) b^* a + r_1 b$
$= r_1 (ab^* a + b) + b^* a$
$\boxed{r_1 = b^* a (b + ab^* a)^*}$

Thm - ~~If L = L(A)~~, If A is a DFA, Then L(A) can be represented by a regular expression

Proof - $A = (Q, \Sigma, \delta, q_0, F)$
Induction on $|Q| = n$
Consider a DFA with only one state
1 - $F = \phi$ or $F = Q$
$r = \phi$ $r = \Sigma^*$

IH - True for $< n$ states
IS - $L(A) = L_1^* L_2$ ⟶ set of strings $q_0 \to$ one of final state
L_1: set of strings from q_0 to q_0 no q_0 in between.



$q \in Q$
$x \in \Sigma^*$

$P(q,x)$ set of all strings that appears on the path while processing x after q.

$P(q,x) = \bigcup_{i=1}^{n} \hat{\delta}(q, a_1 a_2 \ldots a_i)$

$L_1 = \{ x \in \Sigma^* \mid \hat{\delta}(q_0, x) = q_0 \}$

~~$L_2 = \{ x \in \Sigma^* \mid \hat{\delta}$~~

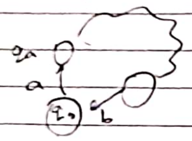
$L_2 = \begin{cases} L_3 & \text{if } q_0 \notin F \\ L_3 \cup \{\epsilon\} & \text{if } q_0 \in F \end{cases}$

$L_3 = \{ x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in F, q_0 \notin P_{(q_0, x)} \}$

$L(A) = L_1^* L_2$

Claim L_1 is regular

$A = \left\{ (a,b) \in \Sigma \times \Sigma \;\middle|\; \begin{array}{l} \hat{\delta}(q, axb) = q_0 \text{ for some } x \\ \delta(q_0, a) \neq q_0 \\ q_0 \notin P(q_a, x) \text{ where} \\ \quad q_a = \delta(q_0, a) \end{array} \right.$



$L_{(a,b)} = \{ x \in \Sigma^* \mid \hat{\delta}(q_0, axb) = q_0$ and $q_0 \notin P(q_a, x)$ where $q_a = \delta(q_0, a) \}$

$A_{(a,b)} = (Q', \Sigma, \delta', q_a, F')$

$Q' = Q - \{q_0\}$

$F' = \{ q \in Q' \mid \delta(q, b) = q_0 \}$

$\delta' = \delta$ restricted to Q'

$\delta' = \delta |_{Q' \times \Sigma}$

Claim $A_{(a,b)}$ accepts the lang $L_{(a,b)}$
(DFA)

CS-205

$x \in L(A_{(a,b)})$

$\hat{\delta}(q_0, axb) = \hat{\delta}(\delta(q_0,a), xb)$

$= \hat{\delta}(q_a, xb)$

$= \delta(\hat{\delta}(q_a, x), b)$

$= \delta(p, b) \quad p \in F'$

$= q_0$

$x \in L_{(a,b)}$

HW: Converse of this

$$L_1 = B \cup \bigcup_{(a,b) \in A} a\, L_{(a,b)}\, b$$

set of self loops from $q_0 \to q_0$

set of all symbols

$B = \{a \in \Sigma \mid \delta(q_0, a) = q_0\}$

Claim2 L_3 is regular

$C = \{a \in \Sigma \mid \delta(q_0, a) \notin q_0\}$

$a \in C$

$L_a = \{x \in \Sigma^* \mid \hat{\delta}(q_0, ax) \in F,\ q_0 \notin P_{(q_0, x)}\}$

$A_a = (Q', \Sigma, \delta', q_a, F'')$

$Q' = Q - \{q_0\}$

$\delta' = \delta|_{Q' \times \Sigma}$

$F'' = F - \{q_0\}$

HW- $L(A_a) = L_a$

$x \in L(A_a) \Leftrightarrow \hat{\delta}'(q_a, x) \in F''$

$\Leftrightarrow \hat{\delta}'(q_a, x) \in F$

$\Leftrightarrow \hat{\delta}(q_a, x) \in F$

$\Leftrightarrow \hat{\delta}(\delta(q_0, a), x) \in F$

$L_3 = \bigcup_{a \in C} a L_a$

DFA $\equiv$ reg. expre...

Right linear

Regular grammar

$G = (N, \Sigma, P, S)$

$A \to xB$

$A \to x$

To show

Regular gram...

Theorem- If A is a DF... regular grammar

Proof $A = (Q, \Sigma, \delta, ...$

$G = (N, \Sigma, P, S)$

$N = Q$

$S = q_0$

$P = \{ A \to ...$

in addition

$q_0 \in F$, then

$x \in L(A)$

$\Leftrightarrow \hat{\delta}(q_0, ax) \in F$

$\Leftrightarrow x \in L_a$

$L = \bigcup_{a \in C} a L_a$

DFA $\equiv$ Reg expression

Right linear
Regular grammar

$G = (N, \Sigma, P, S)$

$A \longrightarrow xB$

$A \rightarrow x$

To show

Regular grammar $\equiv$ FA

Theorem: If ~~L~~ A is a DFA, then $L(A)$ can be generated by a regular grammar

Proof

$A = (Q, \Sigma, \delta, q_0, F)$

$G = (N, \Sigma, P, S)$

$N = Q$

$S = q_0$

$P = \{ A \rightarrow aB \mid \delta(A, a) = B \} \cup \{ A \rightarrow a \mid \delta(A, a) \in F \}$

in addition if the initial state ~~is a final state~~ $q_0 \in F$, then include $S \rightarrow \epsilon$ in P

$x \in L(A)$

$x = a_1 a_2 \ldots a_n \in L(A)$

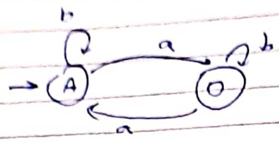
$\hat{\delta}(q_0, a_1 a_2 \ldots a_n) \in F$

$\delta(q_0, a_1) = q_1 \quad - \quad q_0 \rightarrow a_1 q_1$

$\delta(q_1, a_2) = q_2 \quad - \quad q_1 \rightarrow a_2 q_2$

$\delta(q_{n-1}, a_n) = q_n \in F$ --- $q_{n-1} \to a_n$

For converse follow reverse process



$A \to aO \mid bA$

$O \to bO \mid aA \mid \varepsilon$

Regular grammar $\longrightarrow$ DFA

Generalised Finite automata

$p \xrightarrow{x} q$ allow transitions on string $x = a_1a_2 \ldots a_k$

$p \xrightarrow{a_1} p_1 \xrightarrow{a_2} p_2 \ldots \xrightarrow{a_k} q$

GFA = $(Q, \Sigma, X, \delta, q_0, F)$

X is a finite subset of Σ^* on which we define transitions.

$Q = N \cup \{f\}$

$q_0 = S$

$F = \{f\}$

$B \in \delta(A, x)$ iff $A \to xB \in P$

$f \in \delta(A, x)$ iff $A \to x \in P$

HW - Construction

Epsilon prod. $A \to \epsilon$

Unit prod. $A \to B$

CS-205

PUSHDOWN AUTOMATA

PDA ≡ CFG

A PDA is deterministic if $\forall q \in Q$ and $z \in \Gamma$

1. whenever $\delta(q, \varepsilon, z)$ is nonempty $\delta(q, a, z)$ is empty for all $a \in \Sigma$

2. $\delta(q, a, z)$ has only one choice $\forall q \in Q$, $z \in \Gamma$ & $a \in (\Sigma \cup \{\varepsilon\})$

Nondeterministic PDA ← $\{ww^R \mid w \in \{a,b\}^*\}$ ← non deterministic

$\{wcw^R \mid w \in \{a,b\}^*\}$ ← deterministic.

DPDA – class of language accepted is DCFL

DPDA ≠ PDA

L(M) – PDA that accepts by final state

N(M) – PDA " " " emptying the stack.

★ If $L = L(M)$ for some pda M. Then $L = N(M')$ for some pda M'

Proof. $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$

$M' = (Q \cup \{q_e, q_0'\}, \Sigma, \Gamma \cup \{x_0\}, \delta', q_0', x_0, \phi)$

δ': 1. $\delta'(q_0', \varepsilon, x_0) = (q_0, z_0 x_0)$

2. $\delta'(q, a, z) = \delta(q, a, z)$ $\forall q \in Q$, $z \in \Gamma$, $a \in \Sigma \cup \{\varepsilon\}$.

3. $\forall q \in F$, $z \in \Gamma \cup \{x_0\}$,

$\delta'(q, \varepsilon, z)$ contain (q_e, ε)

4. $\delta'(q_e, \varepsilon, z) = (q_e, \varepsilon)$

$\forall z \in \Gamma \cup \{x_0\}$

if $x \in L(N)$

$\delta(q_0, x, z_0) \vdash^*_{M} (p, \epsilon, \alpha)$ $p \in F$, $\alpha \in \Gamma^*$

$\delta'(q_0', x, x_0) \vdash'_{M'} (q_0, x, z_0 x_0)$

$\vdash^*$

$(p, \epsilon, z\alpha)$

$\vdash^* 1$

(q_e, ϵ, α)

$\vdash^*$

$(q_e, \epsilon, \epsilon)$

Then $x \in N(M')$

Therefore these two are equivalent

Reverse process

If $L = N(M)$ then $L = L(M')$

① ✓ same

② ✓ same

$M^* = (Q \cup \{q_f, q_0'\}, \Sigma, \Gamma \cup \{x_0\}, \delta', q_0', x_0, \{q_f\})$

③ $p, \epsilon, x_0 \vdash q_f, \epsilon, \epsilon$

In DPDA it is not the case.

★ If $L = N(P)$, for some DPDA P. Then L has the "prefix property"

$x \in L$ then no proper prefix of x will be in L

(a^* is not accepted by the DPDA)

(Proof by contradiction)

x xy

$(q_0, x, z_0) \vdash^* (p, \epsilon, \epsilon)$

$(q_0, xy, z_0) \vdash^* (p, y, \epsilon)$ stack is empty

cannot move

So xy is not accepted by DPDA

- If L is accepted by some DPDA by empty stack. Then there exists that empty by final state $L = L(P')$

★ Converse doesnot work

If L has the prefix property DPDA P, then there exists $L' = N(P')$

A language $L = N(P)$ for the prefix property DPDA P'

PDA ≡ CFG

- Any grammar G cons

Thm Proof - If L is a CFL

$G = (N, \Sigma, P, S)$

$L = L(G)$

Assume that

$A \rightarrow$

$M = (\{q\},$

$\delta(q, a, A)$ co

whenever

- If L is accepted by some DPDA P that accept by empty stack. Then there exists a DPDA P' that empty by final state i.e

$L = L(P')$

★ Converse doesnot work

If L has the prefix property and $L = L(P)$ for some DPDA P, then there exists a DPDA P' s.t

$L' = N(P')$

A language $L = N(P)$ for some DPDA P iff L has the prefix property and $L = L(P')$ for some DPDA P'

PDA ≡ CFG

- Any grammar G construct PDA

Thm If L is a CFL, then there exists a PDA M st $L = N(M)$

Proof- $G = (N, \Sigma, P, S)$

$L = L(G)$

Assume that G is in GNF

$$A \rightarrow aB_1B_2 \ldots B_k, \; k \geq 0$$

$$M = (\{q\}, \Sigma, \Gamma, \delta, q, S, \phi) \qquad \Gamma = N$$

$\delta(q, a, A)$ contains (q, α) whenever $A \rightarrow a\alpha$ is in P.

Thm: If L is a CFL, then there is a pda M s.t. $L = N(M)$.

Proof.

$G = (N, \Sigma, P, S)$

$M = (\{q\}, \Sigma, N, \delta, q, S, \phi)$

$\delta(q, a, A)$ contain (q, γ)

whenever $A \to a\gamma$ is in P

$S \underset{G}{\overset{*}{\Rightarrow}} x\alpha$ iff $(q, x, S) \overset{*}{\vdash} (q, \varepsilon, \alpha)$

$N(M) = L(G)$

(If) Assume $(q, x, S) \overset{i}{\vdash} (q, \varepsilon, \alpha)$

Base $i = 0$

$x = \varepsilon$

$S = \alpha$

$i \geq 1$ $x = ya$

$(q, ya, S) \overset{i-1}{\vdash} (q, a, \beta) \vdash (q, \varepsilon, \alpha)$

$(q, y, S) \overset{i-1}{\vdash} (q, \varepsilon, \beta)$

$S \underset{G}{\overset{*}{\Rightarrow}} y\beta$

$\beta = A\gamma$

$\delta(q, a, A) = (q, \eta)$

$\eta\gamma = \alpha$

$(q, a, \beta) = (q, a, A\gamma) \vdash (q, \varepsilon, \eta\gamma) = (q, \varepsilon, \alpha)$

$A \to a\eta$

(only if) Suppose

Base $i = 0$

IH $i \geq 1$

S

(q, y, S)

(q, ya, S)

Eg- $S \to aAB$

$A \to a \mid aB$

$B \to a \mid b$

Thm- Let

Proof- $M =$

$G =$

$N =$

P

1

2

$S \Rightarrow \omega\beta \rightarrow ya\eta\gamma \rightarrow \underbrace{y}_{x} q\alpha$

(Contd.) Suppose $S \overset{i}{\underset{q}{\Rightarrow}} x\alpha$

Base $i = 0$ $\quad S \Rightarrow S$

IH $-$ $i \geq 1$

$A \rightarrow a\eta$

$S \overset{i-1}{\Rightarrow} yA\gamma \overset{1}{\Rightarrow} \underbrace{ya}_{x}\underbrace{\eta\gamma}_{\alpha}$

$(q, y, S) \vdash^{*} (q, \varepsilon, A\gamma)$

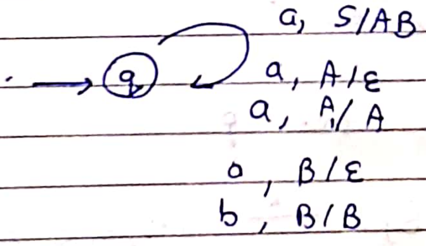
$(q, ya, S) \vdash^{*} (q, a, A\gamma) \vdash (q, \varepsilon, \alpha)$

Eg $-$ $S \rightarrow aAB$

$A \rightarrow a \mid aA$

$B \rightarrow a \mid bB$

$\delta(q, a, S) = (q, AB)$



Thm. Let $L = N(M)$ for some PDA M, then L is CFL

Proof. $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \phi)$

$G = (N, \Sigma, P, S)$

$N = \{ \langle qAp \rangle \}$. $\quad q, p \in Q \quad A \in \Gamma$

P
1. $S \rightarrow \langle q_0 Z_0 q \rangle$ for each $q \in Q$
2. $\langle qAq_{m+1} \rangle \rightarrow a \langle q_1 B_1 q_2 \rangle \langle q_2 B_2 q_3 \rangle \ldots \langle q_m B_m q_{m+1} \rangle$

s.t $\delta(q, a, A)$ contains $(q_1, B_1 B_2 \ldots B_m)$

for each $q_1, q_2, \ldots q_{m+1}$ in Q

Scanned by CamScanner

$\langle qAp \rangle \xrightarrow{*} x$ if A is erased from q to p.

$\langle qAp \rangle \xRightarrow{i} x$ iff $(q, x, A) \vdash^{*} (p, \varepsilon, \varepsilon)$

if $(q, x, A) \vdash^{i} (p, \varepsilon, \varepsilon)$

if $i = 1$

$\delta(q, x, A)$ contain (p, ε)

$x = a$ or ε

$\langle qAp \rangle \xRightarrow{1} a$

$i > 1$

$x = ay$

$(q, ay, A) \vdash^{1} (q_1, y, B_1 B_2 \ldots B_n) \vdash^{i-1} (p, \varepsilon, \varepsilon)$

$y = y_1 y_2 \ldots y_n$

y_1 pops B_1

$q_2, q_3, \ldots q_{m+1} = p$

$(q_j, y_j, B_j) \vdash^{*} (q_{j+1}, \varepsilon, \varepsilon) \leftarrow$ IH

this will take less than i moves.

$\langle q_j B_j q_{j+1} \rangle \xRightarrow{*} y_j$

$\langle qAp \rangle \rightarrow a \langle q_1 B_1 q_2 \rangle \ldots \langle q_m B_m q_{m+1} \rangle$

$\xRightarrow{*} a y_1 \ldots$

$\Rightarrow a y_1 y_2 \ldots y_n$

$\Rightarrow ay = x$

$M = (\{q_0, q_1\}, \{0,1\}, \{x, z_0\}, \delta, q_0, z_0, \phi)$

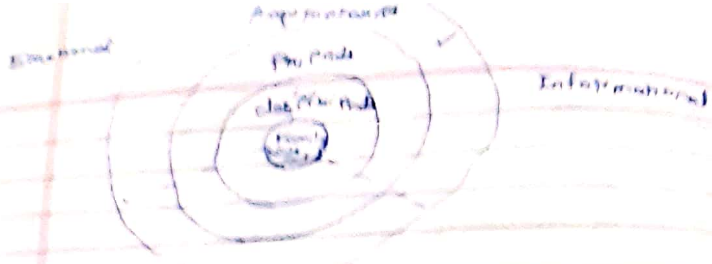
$\delta(q_0, 0, z_0) = \{(q_0, xz_0)\}$
$\delta(q_0, 0, x) = \{(q_0, xx)\}$
$\delta(q_0, 1, x) = \{(q_1, \varepsilon)\}$
$\delta(q_1, 1, x) = \{(q_1, \varepsilon)\}$
$\delta(q_1, \varepsilon, x) = \{(q_1, \varepsilon)\}$
$\delta(q_1, \varepsilon, z_0) = \{(q_1, \varepsilon)\}$

$N = \{ S, \langle q_0 x q_0 \rangle, \ldots \}$

1. $S \rightarrow \langle q_0 z_0 q_0 \rangle \mid \langle q_0 z_0 q_1 \rangle$
2. $\langle q_0 z_0 q_0 \rangle \rightarrow 0 \langle q_0 x q_0 \rangle \langle q_0 z_0 q_0 \rangle \mid 0 \langle q_0 x q_1 \rangle \langle q_1 z_0 q_0 \rangle$

$\langle q_0 z_0 q_1 \rangle \rightarrow 0 \langle q_0 x q_0 \rangle \langle q_0 z_0 q_1 \rangle \mid 0 \langle q_0 x q_1 \rangle \langle q_1 z_0 q_1 \rangle$

3 $\langle q_0 x q_1 \rangle \rightarrow 1$



Tangible

altruistic, volunteering community work to form connections

CS-205

DPDA accepts DFA

$\delta_P(q, a, z_0) = (\delta_D(q, a), z_0)$

PDA – closed under compl.
DPDA – not closed under compl.

$L_1 \cdot G_1 = (N_1, \Sigma_1, P_1, S_1)$
$L_2 \; G_2 = (N_2, \Sigma_2, P_2, S_2)$

$L = L_1 \cup L_2 \quad G = (N, \Sigma, P, S)$

N_1 and N_2 are disjoint
S doesnot belong to N_1, N_2
$N = N_1 \cup N_2 \cup \{S\}$
$\Sigma = \Sigma_1 \cup \Sigma_2$
$P = P_1 \cup P_2 \cup \{S \rightarrow S_1 | S_2\}.$
$x \in L(G)$
$S \overset{*}{\underset{G}{\Rightarrow}} x$

$S \overset{1}{\underset{G}{\Rightarrow}} S_1 \overset{*}{\underset{G_1}{\Rightarrow}} x$

$L = L_1 L_2$

$P = P_1 \cup P_2 \cup \{S \rightarrow S_1 S_2\}.$

CFL's are closed
CFL's are not
$L \cap R$
CFL
CFL are

$L \quad G = (N, \Sigma, P, S)$
$L^* \quad G' = (N, \Sigma, P', S)$

$P' = P \cup \{ S \to SS | \epsilon \}$

HW — CFL's are closed under reversal

CFL's are not closed under complement.

$L_1 = \{a^i b^i c^j \mid i, j \geq 0\}$
$L_2 = \{a^i b^j c^j \mid i, j \geq 0\}$

$L_1 \cap L_2 \cap L_3 = \{a^i b^i c^i \mid i \geq 0\}$

regular

$L \cap R$ can construct PDA.

CFL

R — $A = (Q_A, \Sigma_A, \delta_A, q_A, F_A)$

L — $M = (Q_M, \Sigma, \Gamma, \delta_M, q_M, Z_M, F_M)$

$L \cap R$ — $M' = (Q, \Sigma, \Gamma, \delta, (q_M, q_A), Z_M, F_M \times F_A)$

$Q = Q_M \times Q_A$

$\delta((p,q), a, X) = ((p', q'), \alpha)$ prove by induction.

CFL are closed under substitution.

$L \subseteq \Sigma$

$s(L)$

consider L $G = (N, \Sigma, P, S)$

$\forall a \in \Sigma$ L_a is a CFL $G_a = (N_a, \Sigma_a, P_a, S_a)$

N, N_a - disjoint

$$G' = (N', \Sigma', P', S')$$

$$N' = N \cup \bigcup_{a_i \in \Sigma} N_{a_i}$$

$$\Sigma' = \bigcup_{a_i \in \Sigma} \Sigma_{a_i}$$

P' consists of

1. $\bigcup_{a_i \in \Sigma} P_{a_i}$

2. The production of P but with each terminal a in the RHS of a production replaced by S_a everywhere.

$w \in L(G')$ iff. $w \in S(L)$

(if) $w \in S(L)$

$x = a_1 \ldots a_n \in L$

$x_i \in S(a_i)$

$w = x_1 x_2 \ldots x_n \; (\in S(a_1) S(a_2) \ldots S(a_n))$

$S \overset{*}{\Rightarrow} S_{a_1} S_{a_2} \ldots S_{a_n}$

$S \overset{*}{\Rightarrow} w$

$w \in L(G')$

similarly converse.

CFL's are closed under homomorphism as well.

Pumping lemma

$z = uvwxy$

vx is not empty i.e. $|vx| > 0$

$|vwx| \le n$

$\forall i = 0 \quad uv^iwx^iy \in L$

Assume CFG is in CNF

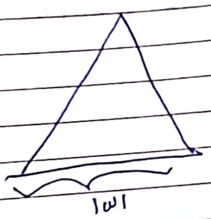
$A \to BC$

$A \to a$

Consider a string

n – length of longest path from root to leaf in parse tree

S



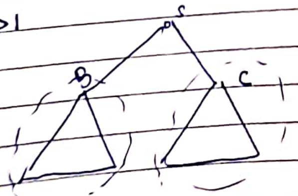
$|w|$

$|w| \le 2^{n-1}$

Base $n=1$

$n>1$

S



B C

IH IH

Suppose $|N| = k$. L $G = (N, \Sigma, P, S)$

$n = 2^k$

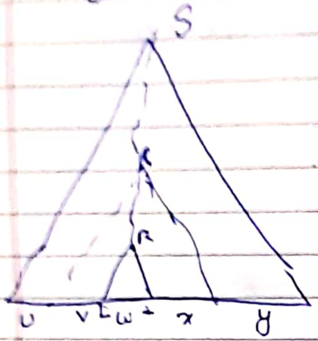
$z \in L(G)$

$|z| \ge n = 2^k$

$k+1$ length path

$k+2$ nodes.

k terminals but k+1 nodes in the path
atleast one repetition



$|vx| > 0$

$S \overset{*}{\Rightarrow} \alpha R \beta$

$S \overset{*}{\Rightarrow} uRy \Rightarrow uvRxy \Rightarrow uv\,vRx\,xy$

$\{a^i b^i c^i \mid i \geq 0\}$.

$z = a^k b^k c^k$

* ww
* a^p

Acts of kindness
Act of kindness is a
towards someone or so
feeling - " I can

What goes around co

✓ Compassion is the bas
✓ True happiness consi
✓ without kindness, there can be

Social implications

Multiple Scle
Negativity →

- Being kind leads
'positively'
- It impacts self perce
sense of meaning fu
discovers hidden
- Kindness can jump