Formal languages

01/03/96

— set of strings

ch. 1 Alphabets, Languages

1) Alphabet :- finite non-empty set of symbols

$$\Sigma_1 = \{a, b\}$$

$$\Sigma_2 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

$$\Sigma_3 = \{a, b, c, \ldots\ldots, z\}$$

$$\Sigma_4 = \{\Delta, \square\}$$

2) String/Word over alphabet Σ : finite sequence of symbols from Σ

3) Language over alphabet Σ is a set of strings over Σ

$$L_1 = \{ab, aab, aaab, \ldots\ldots\}$$

$= \{w / w$ is a string over Σ_1 and w has 1 or more a's followed by one

4) Star Closure of Σ

Σ^* = set of all possible strings over Σ

= Universal Language

Operations on strings

1) **Length of String** = # of symbols in string

eg: $|abaa| = 4$

$|\epsilon| = 0$

2) **Concatenation of two strings**

Appending of one string to another is Concatenatio.
Appending z to w is denoted by $w.z$ or wz

$w\epsilon = \epsilon w = w$

$|wz| = |w| + |z|$

3) **Exponentiation**

$$w^n = \begin{cases} \epsilon, & n = 0 \\ ww^{n-1}, & n > 0 \end{cases}$$

$w^3 = www$

$w^0 = \epsilon$

$w = aab$

$w^3 = aabaabaab$

4) **Suffix, prefix**

x is a Suffix of w if there exists y such that $w=$

x is a Prefix of w if there exists y such that $w=$

y could be ϵ

if $y \neq \epsilon$ then proper Suffix/Prefix

5) Substring/Subword

x is a substring of w if there exists y, z such that $w = y\,x\,z$

6) Reverse

$$w^R = \begin{cases} w & \text{if } w = \epsilon \\ y^R a & \text{if } w = ay \;;\; a \in \Sigma,\ y \in \Sigma^* \end{cases}$$

$$(xy)^R = y^R x^R$$

$$(x^R)^R = x$$

Operations on languages 01/06/9

Let L_1, L_2 be two languages

$$L_1 \cdot L_2 = \{ w_1 \cdot w_2 \mid w_1 \in L_1 \text{ and } w_2 \in L_2 \}$$

ex: $L_1 = \{a, bb\}$

notation $L = L^1$

$L_2 = \{aa, b\}$

$L \cdot L = L^2$

$L \cdot L \cdot L = L^3$ ...

$L_1 \cdot L_2 = \{aaa, ab, bbaa, bbb\}$

Set operators

$L_1 \cup L_2 = \{ w \mid w \in L_1 \text{ or } w \in L_2 \}$ (union)

$L_1 \cap L_2 = \{ w \mid w \in L_1 \text{ and } w \in L_2 \}$ (intersectio

$L_1 - L_2 = \{ w \mid w \in L_1 \text{ and } w \notin L_2 \}$ (difference)

$\overline{L_1} = \Sigma^* - L_1$ (complement

Star-Closure

$$L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup L^3 \cup L^4 \ldots\ldots$$

$\Sigma = \{a, b\}$; $L = \{a, bb\}$

$$L^* = \underbrace{\{\epsilon\}}_{L^0} \cup \underbrace{\{a, bb\}}_{L^1} \cup \underbrace{\{aa, abb, bba, bbbb\}}_{L^2} \cup$$
$$\underbrace{\{aaa, aabb, abba, abbbb, bbaa, bbabb, bbbba, bbbbbb\}}_{L^3}$$
$$\ldots\ldots\ldots\ldots \underbrace{\{\}}_{L^{10}}$$

$$L^+ = \bigcup_{i=1}^{\infty} L^i$$

Summary of operations on languages

1) Union
2) Concatenation
3) star closure
4) intersection
5) difference
6) Complement
7) Plus closure
8) exponentiation
9) reverse

1) $A^+ = A \cdot A^* = A^* \cdot A$

2) $(A^*)^* = A^*$

3) $(A^+)^+ = A^+$

Thm. $(A \cdot B)^R = B^R \cdot A^R$

$A = \{ab, b\}$ $A^R = \{ba, b\}$

$B = \{aab, ba\}$ $B^R = \{baa, ab\}$

$A \cdot B = \{ab\,aab, bba, \ldots\}$

$A \cdot B^R = \{ba\,baa, bab, \ldots\}$

Ch. 2

Σ = alphabet = finite non-empty set of symbols

Σ^* = set of all strings over Σ

Theorem: Σ^* is Countably infinite

Proof sketch: $\Sigma = \{a, b, c\}$

$\Sigma^* = \{\epsilon, a, b, c, aa, ab, ac, ba, bb, bc,$
$ca, cb, cc, aaa, aab, aac, \ldots$

Theorem: $\mathcal{L}$ = set of all languages over Σ is uncountab

Proof: Diagonalization argument

Regular languages

Def: It is defined as follows:

1) $\{a\}$ is regular for each $a \in \Sigma$
2) $\{\epsilon\}$ is regular
3) $\{\ \}$ is regular
4) If L_1 and L_2 are regular then
   - $L_1 L_2$ is regular ($L_2 L_1$ is also regul
   - $L_1 \cup L_2$ is regular
   - $L_1{}^*$ is regular ($L_2{}^*$ is regular)
5) Nothing else is regular

01/08/97

2) $\mathcal{L}_2$ = Set of words over $\{a, b, c\}$ with no substring 'ac'

3) Set of words over $\{a, b, c\}$ with atleast one 'ac' as a substring

4) Set of words with even # a's = {bbb, baabbbaba, ...

$$(b^* a b^* a b^*)^* b^*$$

5) Set of words with odd # a's

$$b^* a b^* (b^* a b^* a b^*)^*$$

6) words which begin with 'a' or end with 'b'

$$a(a \cup b)^* \cup (a \cup b)^* b$$

7) $\mathcal{L}_2$ = words with even # of a's and even # of b's

$$\mathcal{L}_1 = L[\,aa \cup bb \cup (ab \cup ba)(aa \cup bb)^*(ab \cup ba)^*$$

<u>Th. 2.2.2</u>

1) $r \cup s = s \cup r$
2) $r \cup \phi = r = \phi \cup r$

11) $r(sr)^* = (rs)^* r$
12) $(r^* s)^* = \varepsilon \cup (r \cup s)^* s$

Finite Automata (FA)

Deterministic (DFA) — Non-deterministic (NFA)

- states
- transitions

graphed notation

$\bigcirc$: states

$\rightarrow\bigcirc$: start state (exactly one)

$\circledcirc$: final state (zero or more)

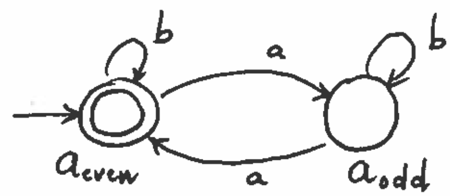
01/10/97

DFA

states, transitions Goto X

ex:

1) set of words with even # a's



→ *

$$b^*(b^*ab^*ab^*)^*$$

→ * $\omega_1 = abaab$

$$(q_{even}, abaab) \vdash_m (q_{odd}, baab)$$
$$\vdash_m (q_{odd}, aab)$$
$$\vdash_m (q_{even}, ab)$$
$$\vdash_m (q_{odd}, b)$$
$$\vdash_m (q_{odd}, \epsilon)$$

ω_1 is not recognised.

<u>def.</u> $(q_1, \omega_1) \vdash^*_m (q_2, \omega_2)$

if $\delta(\delta(\delta(q_1, \sigma_1), \sigma_2) \ldots, \sigma_k) = q_2$

$$\omega_1 = \sigma_1 \sigma_2 \sigma_3 \cdots\cdots \sigma_k \omega_2$$

<u>def.</u> ω_1 is recognised/accepted by m if

$$(s, \omega) \vdash^*_M (f, \epsilon) \text{ and } f \in F$$

$$M = (Q, \Sigma, s, F, \delta)$$

<u>def.</u> language accepted by m

Notation

X **def.** A DFA is $(\phi, \Sigma, s, F, \delta)$

ϕ – finite set of states

Σ – input alphabet

$s \in \phi$ – start state

$F \leq \phi$ – final state

$$\delta : \underbrace{\phi}_{\text{present state}} \times \underbrace{\Sigma}_{\text{present i/p symbol}} \longrightarrow \underbrace{\phi}_{\text{next state}}$$

def. configuration : (q, w) q : Current state
w : String (remaining i/p to be scanned)

$(s, \underbrace{w}_{\text{entire string}})$: initial configuration

def. $(q_1, w_1) \vdash_M (q_2, w_2)$

$\vdash_M$: apply one transition

if $\delta(q_1, \sigma) = q_2$ and $w_1 = \sigma w_2$

$$q_1 \xrightarrow{\sigma} q_2$$

$\omega_2 = bbabaaba$

$(q_{even}, bbabaaba) \vdash_m$

$\vdots$

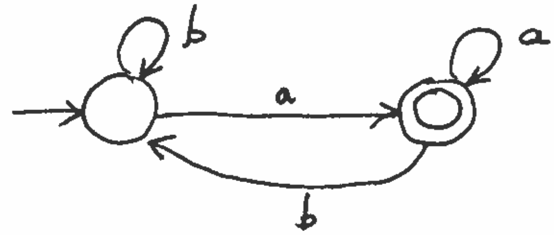
$\vdash_m (q_{even}, \epsilon)$

ω_2 is accepted

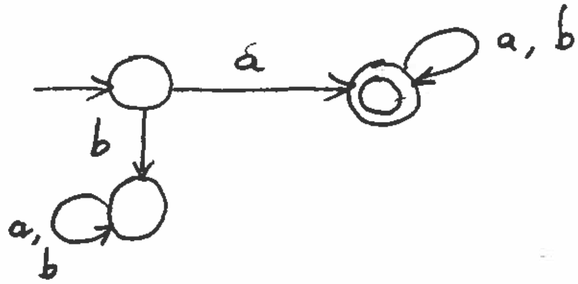
---

2) words that end with 'a'

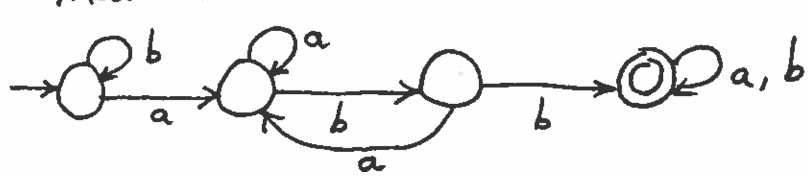
$(a \cup b)^* a$



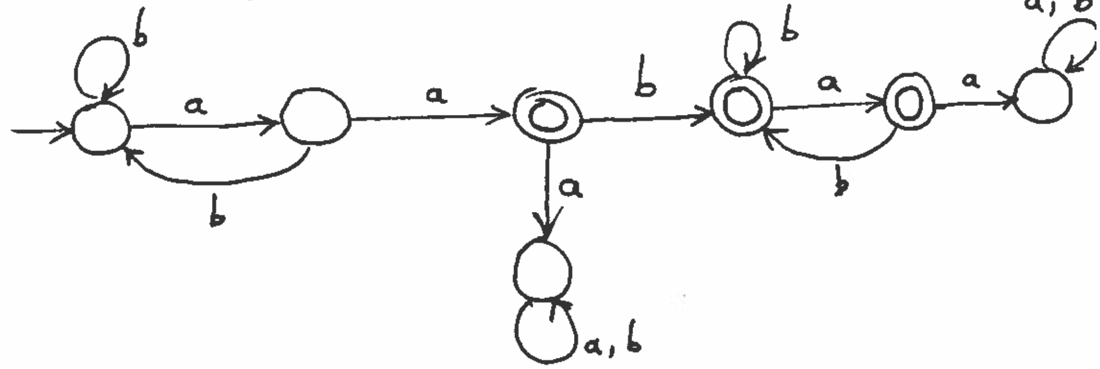
3) words that begin with 'a'



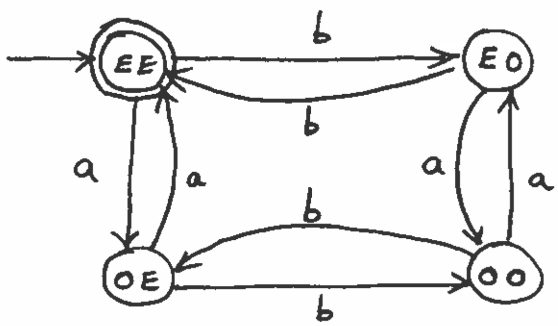
4) words that have 'abb' as a substring



5) words with exactly one occurence of 'aa' as a substring

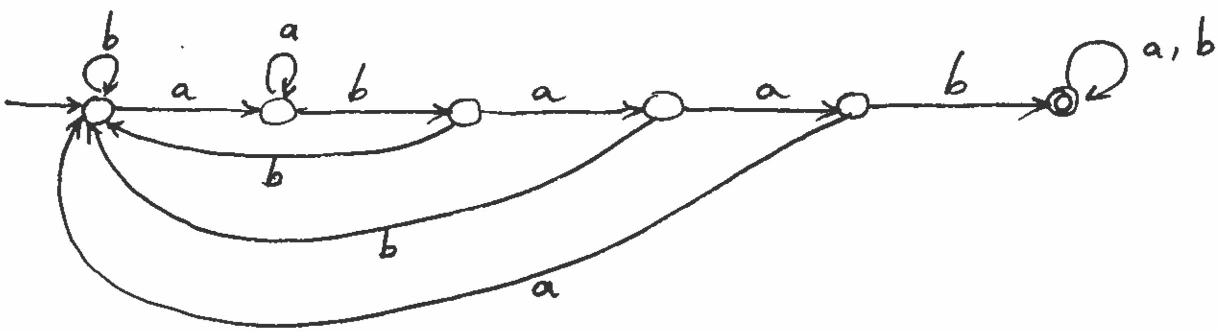


6) EVEN – EVEN — words with even # a's and even # b's

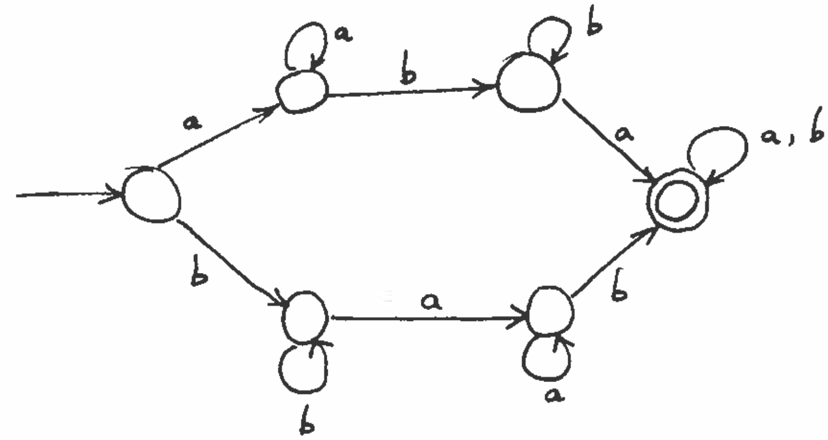


$$[aa \cup bb \cup (ab \cup ba)(aa \cup bb)^*(ab \cup ba)]^*$$

* 7) words with 'abaab' as a substring (H/W)



Ex : $L = \{ w \mid w$ has both 'ab' and 'ba' as a Substring $\}$



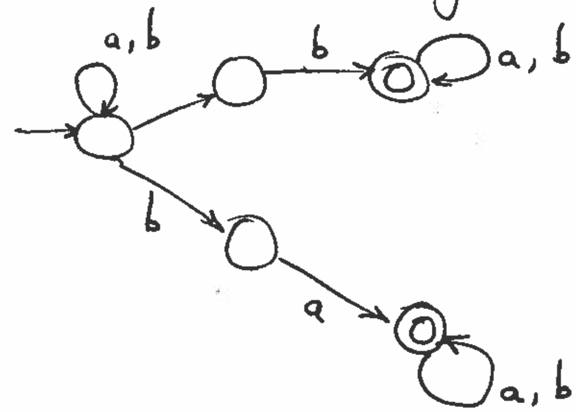
NFA

we allow one, zero or more transitions from a given state on the same input symbol.

In an NFA w is accepted if we reach a final state after scanning all symbols of w by making the right choice

- NFA's are easier to Construct
- Given an NFA, easier to determine the language

ex: words with substring 'ab'

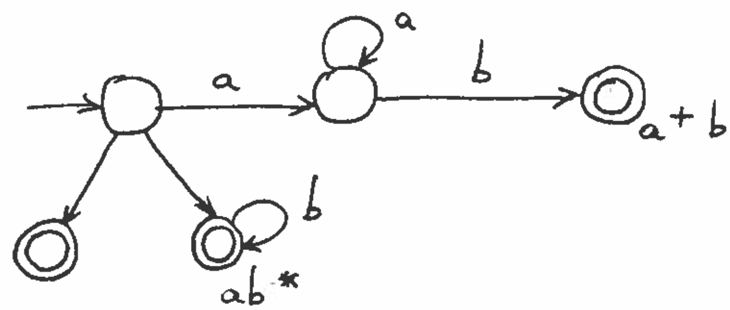


Ex: $a^*b \cup ab^*$

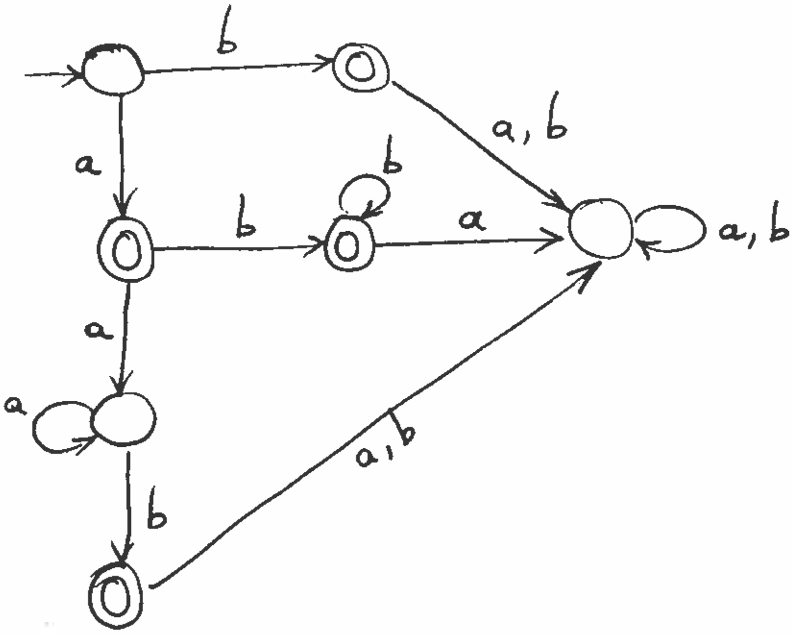
b, ab, aab, aaab, aaaab

a, ab, abb, abbb, abbbb .....

NFA

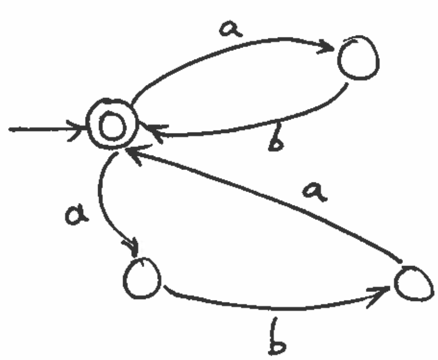


DFA



ex: $(ab \cup aba)^*$

NFA



ω = ab aaa

- $(q_0, abaaa)$
  - $(q_1, baaa)$
    - (q_0, aaa)
      - (q_1, aa) FAIL
      - (q_2, aa) FAIL
  - $(q_2, baaa)$
    - (q_3, aaa)
      - (q_0, aa)
        - (q_1, a) FAIL
        - (q_2, a) FAIL

ω is not in language

def: NFA $M = (Q, \Sigma, s, F, \Delta)$

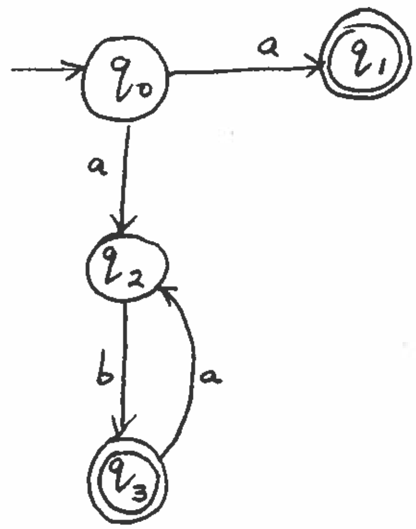
Q = finite set of states
Σ = input alphabet
$s \in Q$ = start state
$F \subseteq Q$ = set of final states
$\Delta \subseteq Q \times \Sigma \times Q$

NFA $\equiv$ DFA

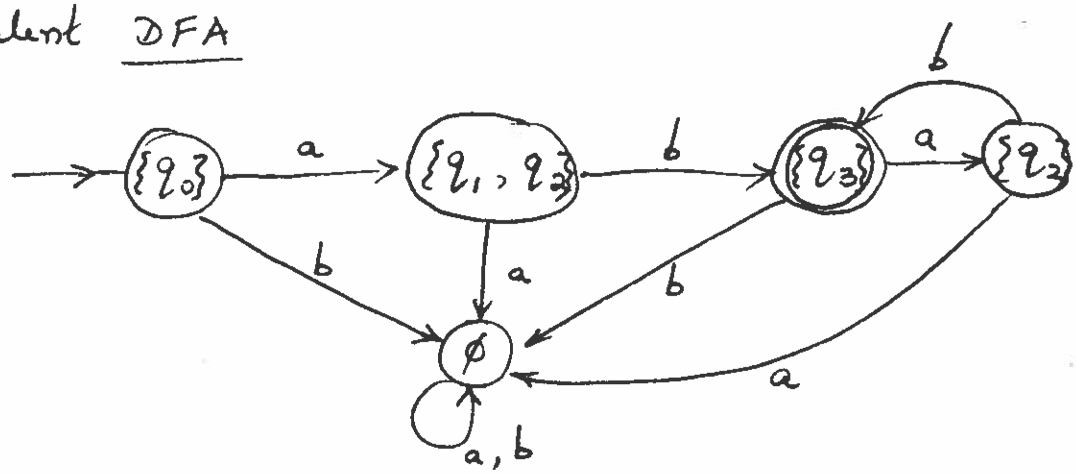
1) Every DFA is also an NFA

2) Given an NFA, $M = (Q, \Sigma, s, F, \Delta)$
we can construct an equivalent DFA
$M' = (Q', \Sigma, s', F', \delta)$ $\rightarrow$ $L(M) = L(M')$

ex: NFA



equivalent DFA



$Q' = 2^{Q}$ = set of all subsets of Q

$s' = \{s\}$

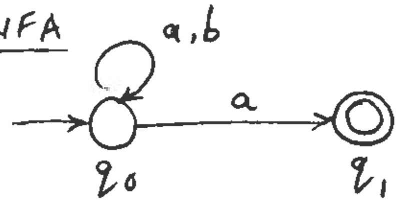
$F' = \{A \subseteq Q \mid A \cap F \neq \phi\}$

$\delta = \delta(\{q_1 \ldots\ldots q_n\}, \sigma) = \{p_1 \ldots\ldots p_n\}$

01/15/97

NFA to DFA Conversion

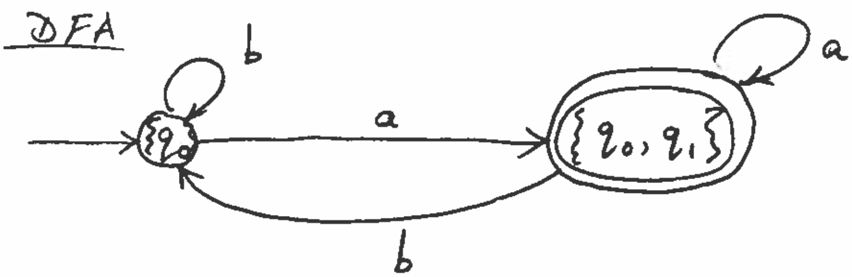
ex: NFA



$(a \cup b)^* a$

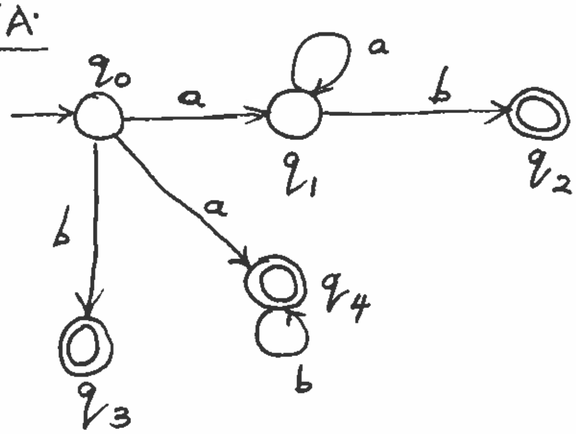
words ending with 'a'

DFA



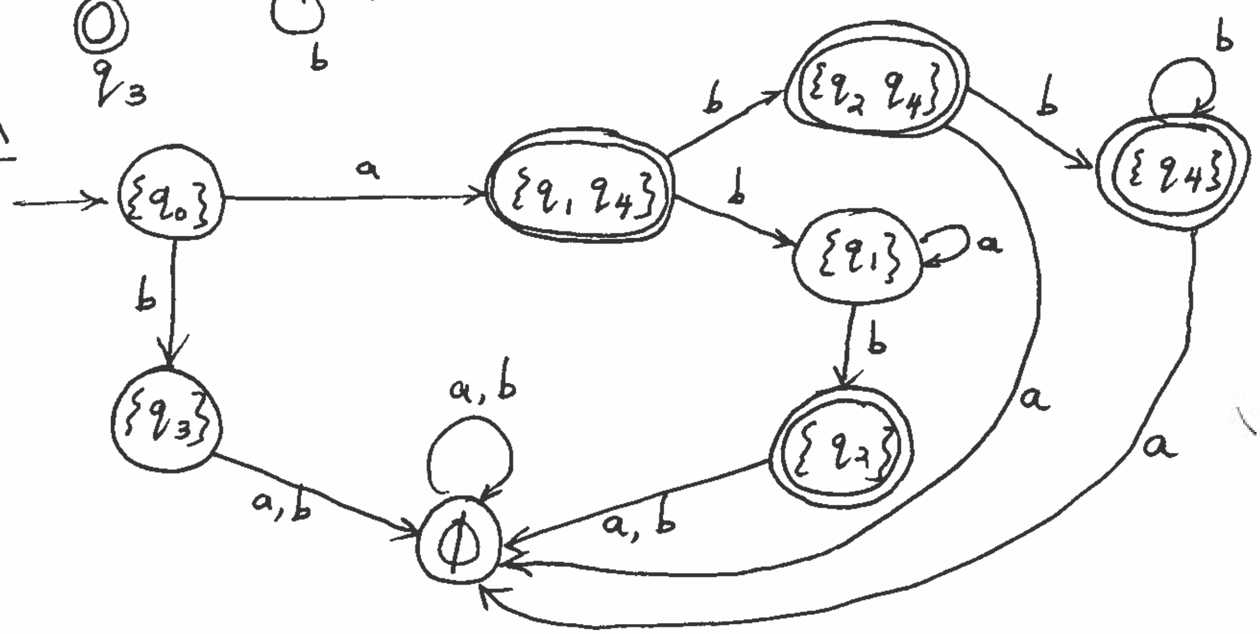
DFA/NFA in prolog

NFA:



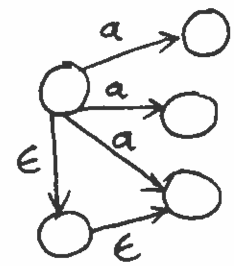
$a^* b \cup ab^*$

DFA



* (pg 75 # 2.2)

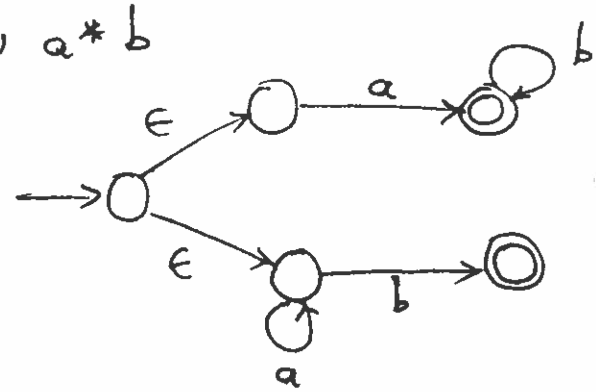
2.7 ϵ-transitions in NFA's



NFA w/out ϵ-transitions

NFA with ϵ-transitions

$ab^* \cup a^*b$



Alg to convert NFA with ϵ-transitions into NFA without ϵ-transitions

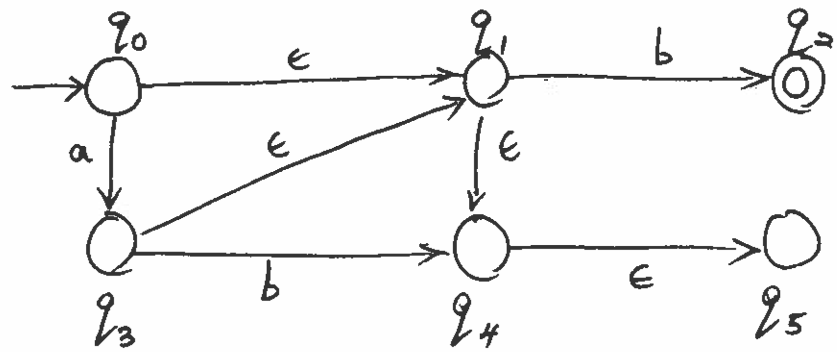
def ϵ-closure(q) = $\{p \mid p$ can be reached from q without reading any input ie by following only ϵ-transitions$\}$

state

def d-transition(q, σ) = $\{p \mid$ there is a dire transition from q to on $\sigma\}$

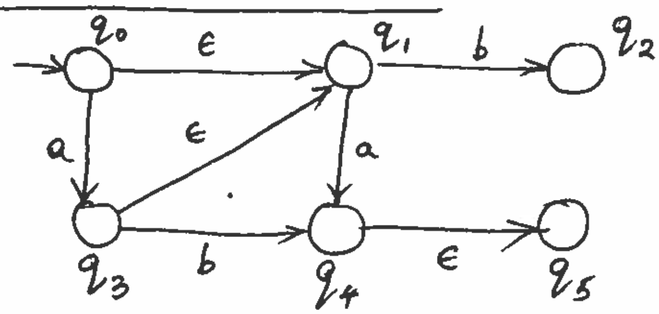
→

ex:



q	ϵ - closure(q)
q_0	q_0, q_1, q_4, q_5
q_1	q_1, q_4, q_5
q_2	q_2
q_3	q_3, q_1, q_4, q_5
q_4	q_4, q_5
q_5	q_5

ϵ - transitions in NFA's

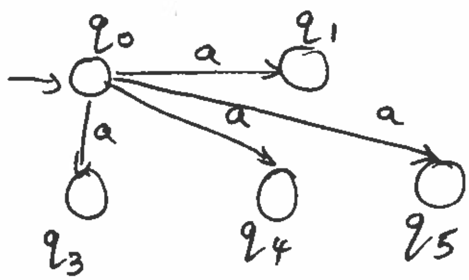


$$A'(q, \sigma) = \epsilon\text{-}cl(d(\epsilon\text{-}cl(q), \sigma))$$

$\epsilon - cl\,(q_0, a) = \{q_0, q_1\}$

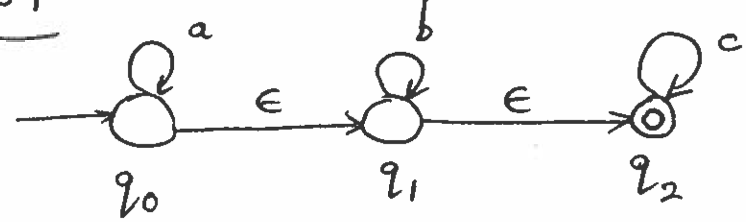
$d = \{q_3, q_4\}$

$\epsilon - cl = \{q_3, q_1, q_4, q_5\}$



pg 59

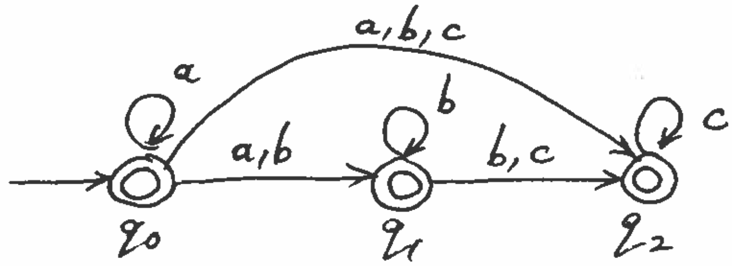
2.7.5



$q_0 \xrightarrow{\epsilon\text{-closure}} q_0, q_1, \underline{q_2} \xrightarrow{d}$ $\xrightarrow{a} q_0 \xrightarrow{\epsilon} q_0 q_1$; $\xrightarrow{b} q_1 \longrightarrow q_1 q_2$; $\xrightarrow{c} q_2 \longrightarrow q_2$

$q_1 \xrightarrow{\epsilon\text{-closure}} q_1, \underline{q_2} \xrightarrow{d}$ $\xrightarrow{a} \phi \longrightarrow \phi$; $\xrightarrow{b} q_1 \longrightarrow q_1 q_2$; $\xrightarrow{c} q_2 \longrightarrow q_2$

$q_2 \xrightarrow{\epsilon\text{-closure}} \underline{q_2} \xrightarrow{d}$ $\xrightarrow{a} \phi \longrightarrow \phi$; $\xrightarrow{b} \phi \longrightarrow \phi$; $\xrightarrow{c} q_2 \longrightarrow q_2$



2.8 Finite Automata & Regular Expressions 01/22/

FA $\Longleftrightarrow$ RE

NFA with ϵ-trans.

$\big\downarrow$ rmet

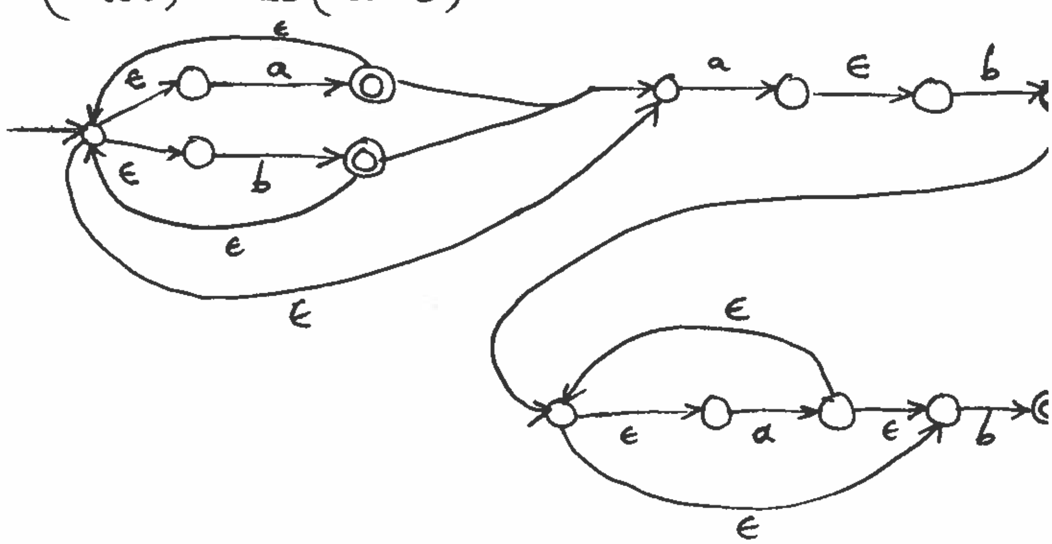
NFA without ϵ-trans.

$\big\downarrow$ ntd

DFA

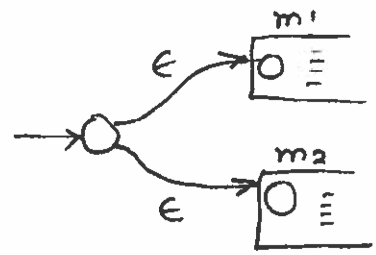
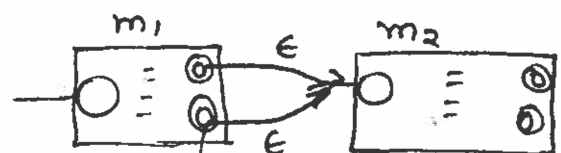
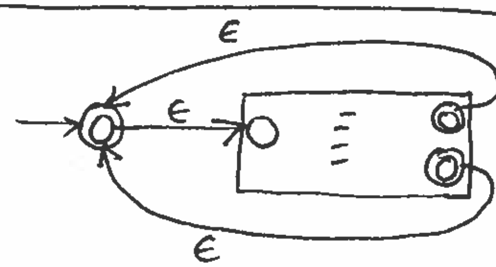
reg. exp.	NFA with ϵ-trans.
ϕ	
a	

ex: $(a \cup b)^* \, ab(a^* b)$ RE $\rightarrow$ FA



4

→ *

reg. exp.	NFA with ε-trans.
ϕ	
a	
$r \cup s$ m_1 m_2	
$r \cdot s$ m_1 m_2	 change final states to non-final states
r^*	 change o (don't cha ~~check~~ fin states (doesn't matt

→

FA $\longrightarrow$ RE

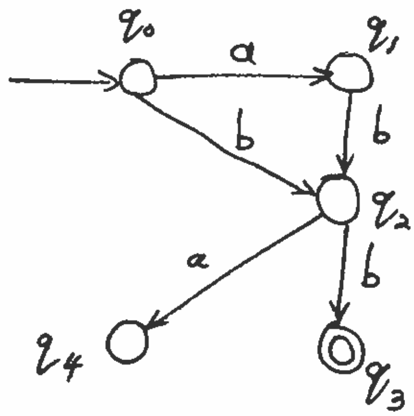
Def: NFA $M = \{Q, \Sigma, \delta, F, \Delta)$

$A_i = \{ \omega \mid \Delta(q_i, \omega) \cap F \neq \phi \}$

= set of words accepted by M if q_i is the start state

$A_0 = \mathcal{L}(M)$

ex:



$$
\begin{aligned}
A_0 &= aA_1 \cup bA_2 && = abb \cup bb \\
A_1 &= bA_2 = bb \\
A_2 &= bA_3 \cup aA_4 && = b\epsilon \cup a \cdot \phi = b \\
A_3 &= \phi \cup \epsilon = \epsilon \\
A_4 &= \phi
\end{aligned}
$$

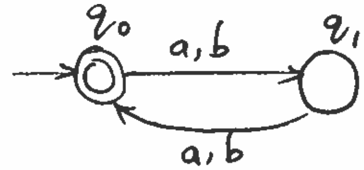
2.8.2 Arden's Lemma

$X = AX \cup B$ where ϵ is not in A has a unique solution.

$$X = A^*B$$

$\longrightarrow$

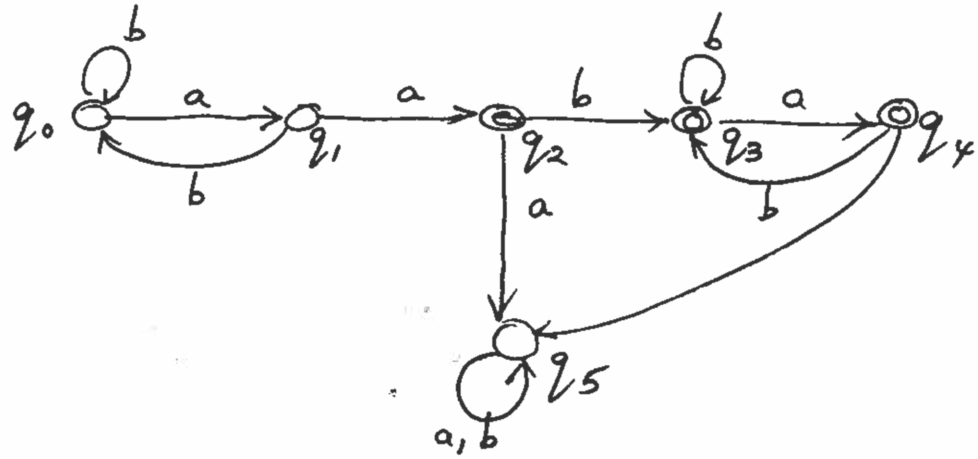
ex:



$$A_0 = aA_1 \cup bA \cup \epsilon = (a \cup b)A_1 \cup \epsilon = \underbrace{(a\cup b)(a\cup b)}_{A} A_0 \cup$$

$$A_1 = (a \cup b) A_0$$

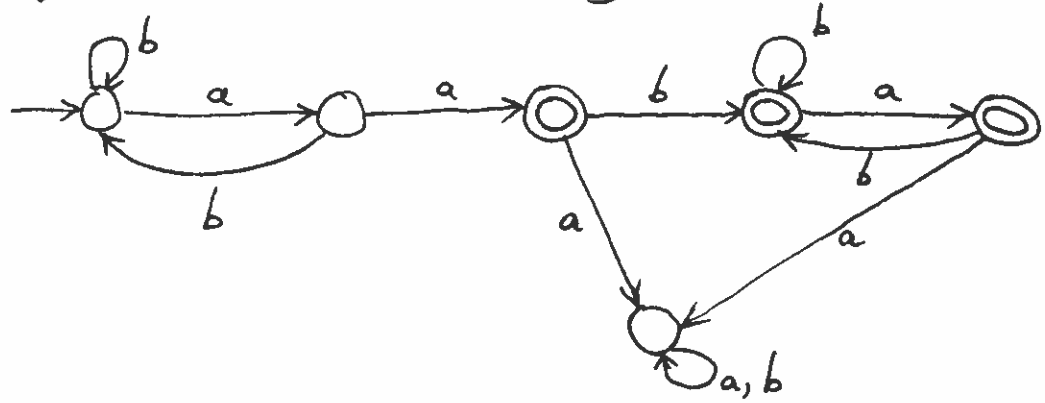
ex: words with one occurence of 'aa'



$$A_0 = bA_0 \cup aA_1$$
$$A_1 = bA_0 \cup aA_2$$
$$A_2 = bA_3 \cup aA_5 \cup \epsilon$$
$$A_3 = bA_3 \cup aA_4 \cup \epsilon$$
$$A_4 = bA_3 \cup aA_5 \cup \epsilon$$
$$A_5 = (a \cup b) A_5 = \phi$$
$$= (a \cup b) A_5 \cup \phi$$
$$A_5 = (a \cup b)^* \cdot \phi = \phi$$

01/27/96

exactly one 'aa' as Substring



$$E = (b \cup ab)^* a \left[\, b(b \cup ab)^* (a \cup \epsilon) \cup \epsilon \,\right]$$

2.9 Properties of Regular languages

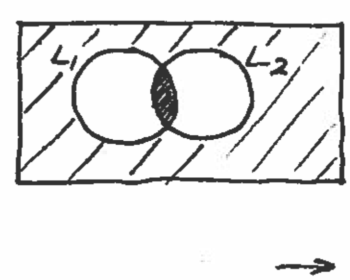
Theorem: If L_1 and L_2 are regular then
$(L_1 \cap L_2)$ is regular and
$\overline{L_1} = (\Sigma^* - L)$ is regular

($L_1 \cup L_2$, L_1^* , $L_1 L_2$ are also regular

i.e. regular languages are closed under the operation of union, concatenation, Kleene star, intersectio[n] complementation.

Integers are closed under '+'
Positive integers are not closed under '−'

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$



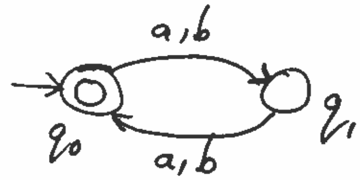
→

Alg: Given DFA M_1 for L_1
DFA M_2 for L_2

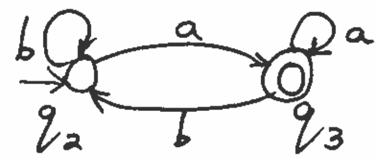
Construct DFA M for $L_1 \cap L_2$

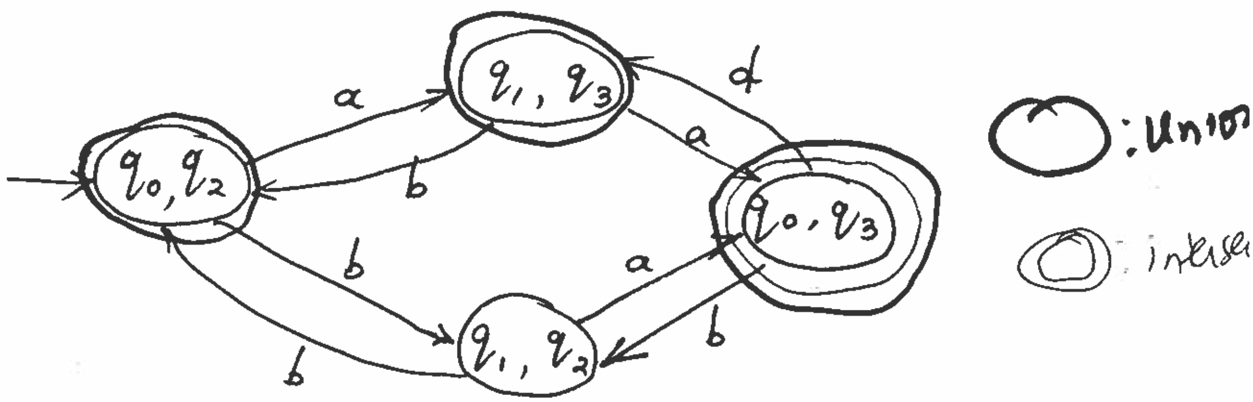
ex:

even length



ending with 'a'





Pumping Lemma for Regular Languages

L: regular, infinite $\Rightarrow$ There are words of arbitrary length $\Rightarrow$ There are words whose length is $> n$

$\Downarrow$

dfa exists

$\Downarrow$

finite # states $= n$

FACT: Every finite language is regular

→

ω = length $n+1$ = $a_1 a_2 \ldots\ldots\ldots a_n a_{n+1}$

path $\quad \to q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} q_3 \cdots\cdots q_{n-1} \xrightarrow{a_n} q_n \xrightarrow{a_{n+1}} q_{n+1}$

There are repeating states in this path

q_i, q_j are the same $\quad 0 \leq i < j \leq n+1$

Pumping Lemma (for regular languages)

Let L be regular. Then, There exists a Constant n such that

for all ω, if ω is in L and $|\omega| \geq n$ Then

There exists u, v, x such that

(1) $\omega = uvx$

and (2) $|v| \geq 1$

and (3) $|uv| \leq n$

and (4) uv^ix is in L for $i \geq 0$

ex: of non-regular language

$$L = \{a^k b^k \mid k \geq 1\}$$

$$= \{ab, aabb, aaabbb, \ldots\ldots\}$$

Proof:

1) Assume L is regular
2) Pumping Lemma guarantees Constant n
3) Pick $\omega = a^n b^n$

4) $\omega = uvx$ where $u = a^{k_1}$; $k_1 + k_2 \leq n$

$v = a^{k_2}$; $k_1 + k_2 + k_3 = n$

$x = a^{k_3} b^n$; $k_2 \geq 1$

uv^ix is in L

<u>$i=2$</u>

$uvvx = a^{k_1} a^{k_2} a^{k_2} a^{k_3} b^n$ is in L

This is not true (Contradiction)

$\therefore$ Not a regular language.

01/29/97

Pumping Lemma

Let L be an infinite regular language. Then <u>there exists</u>, Constant n such that <u>for every</u> w
if $w \in L$ <u>and</u> $|w| \geq n$ Then
There exits u, v, x such That
(1) $w = uvx$
and (2) $|v| \geq 1$
and (3) $|uv| \leq n$
and (4) $uv^ix \in L$, $i \geq 0$

<u>Claim</u> $L = \{a^k b^k \mid k \geq 1\}$ is not regular

<u>Proof:</u> Let L be regular.
Then, P.L. guarantees Constant n.
Let $w = a^n b^n$
Clearly $w \in L$ and $|w| \geq n$
$w = a^n b^n = uvx \ldots\ldots$

Since $|uv| \leq n$

$u = a^{k_1}$ $\qquad k_1 + k_2 \leq n$

$v = a^{k_2}$ $\qquad k_2 \geq 1$

$x = a^{k_3} b^n$ $\qquad k_1 + k_2 + k_3 = n$

Consider uv^2x
P.L. states that $uv^2x \in L$

ex: $L = \{a^k b^l \mid k > l\}$ is not regular

Proof: Let L be regular

Then P.L. guarantees constant n

Choose $\omega = a^{n+1} b^n$

$\omega = uvx$ $\quad |uv| \leq n$, $|v| \geq 1$, $uv^i x \in L$ for $i \geq 0$

$= \underbrace{a^{k_1}}_{u} \underbrace{a^{k_2}}_{v} \underbrace{a^{k_3} b^n}_{x}$; $k_1 + k_2 + k_3 = n+1$

$k_2 + k_1 \leq n$

pick $i = 0$ $\qquad k_2 \geq 1$

$uv^0 x = a^{k_1} a^{k_3} b^n$

$= a^{k_1 + k_3} b^n$

Since $k_2 \geq 1$

$k_1 + k_3 \leq n$

Since $k_1 + k_3 \leq n$

$a^{k_1 + k_3} b^n$ will have less a's

Hence L is not regular

$$uv^2x = a^{k_1}a^{2k_2}a^{k_3}b^n = a^{(k_1+k_2+k_3)+k_2}b^n$$
$$= a^{n+k_2}b^n$$

Since $k_2 \geq 1$ $\quad uv^2x = a^{n+k_2}b^n \notin L$

(contradiction)

Hence L is not regular

<u>Claim</u> :

$L = \{a^{k^2} \mid k \geq 1\}$ is not regular

<u>Proof</u> :

Let L be regular

Then, P.L. guarantees constant n

$\longrightarrow$ Let $\boxed{\omega = a^{n^2}}$

clearly $\omega \in L$ and $|\omega| \geq n$

$$\omega = uvx = \underbrace{a^{k_1}}_{u}\underbrace{a^{k_2}}_{v}\underbrace{a^{k_3}}_{x} \qquad k_1+k_2+k_3 = n^2$$

Consider uv^2x

$$n^2 \underset{\because k_2 \geq 1}{<} |uv^2x| = n^2+k_2 \leq n^2+n < (n+1)^2$$

$\therefore$ $|uv^2x|$ is not a perfect square

$\therefore$ uv^2x is not in L

But P.L. says it is in L (Contradiction)

<u>Claim</u> $L = \{a^k \mid k$ is prime$\}$ is not regular

<u>Proof:</u> Let L be regular
Then the P.L. guarantees a Const. n.
Let $\underline{w = a^m}$; where m is prime and $m \geq n$

P.L. says, $w = uvx$
$= a^{k_1} a^{k_2} a^{k_3}$ $k_1 + k_2 + k_3 = m$

Consider $a^{k_1}(a^{k_2})^{m+1} a^{k_3}$

$= a^{k_1} a^{k_2 m + k_2} a^{k_3}$

$= a^{(k_1 + k_2 + k_3) + k_2 m}$

<u>Claim</u> EQUAL $= \{w \mid \#a\text{'s in } w = \# b\text{'s in } w\}$

<u>Proof:</u> $L(a^* b^*) \cap \text{EQUAL} = \{a^n b^n \mid n \geq 0\}$

regular   reg. lang. are closed under $\cap$ <u>∴ not regular</u>   not regular

<u>Skip Th. 2.9.2</u>

Try Prob 2.9.3. pg 71

ch. 3 Context - free languages

01/29/97

3.3 def: Context-free Grammar

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

N = finite set of non-terminal symbols
Σ = alphabet
$S \in N$ = start symbol
P : finite set of production rules of the form

$X \longrightarrow \alpha$ where $X \in N$
α is any string in $(N \cup \Sigma)^*$

ex:
$S \longrightarrow aSa$
$S \longrightarrow bSb$
$S \longrightarrow \epsilon$

$S \overset{1}{\Rightarrow} aSa \overset{1}{\Rightarrow} aaSaa \overset{2}{\Rightarrow} aabSbaa \overset{1}{\Rightarrow} aabaSabaa$

- even
- palindrome

$\overset{3}{\Longrightarrow} aaba\,abaa$

ex:
$S \longrightarrow AB$
$A \longrightarrow aAb$
$A \longrightarrow a$
$B \longrightarrow bBa$
$B \longrightarrow b$

$S \Rightarrow AB \overset{*}{\Longrightarrow} a^{n+1} b^n b^{m+1} a^m$

~~ex:~~ $\cancel{S \longrightarrow aB}$ $L(G) = \{ a^{n+1} b^n\, b^{m+1} a^m / m \geq 0\ n \geq$

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

ex:

$$G: \quad \begin{array}{l} S \longrightarrow aB \\ S \longrightarrow bA \\ \hline A \longrightarrow aS \\ A \longrightarrow bAA \\ A \longrightarrow a \\ \hline B \longrightarrow bS \\ B \longrightarrow aBB \\ B \longrightarrow b \\ \hline \end{array}$$

$$S \Rightarrow aB \Rightarrow aaBB \Rightarrow aabbAB \Rightarrow aabbaB \Rightarrow aabbab$$

$L(G) =$ EQUAL
$=$ equal # a's & b's

$A \overset{*}{\Rightarrow} \omega$ iff ω has one more 'a' than 'b's
$B \overset{*}{\Rightarrow} \omega$ iff ω has one more 'b' than 'a's
$S \overset{*}{\Rightarrow} \omega$ iff ω has equal no. of 'a's & 'b's

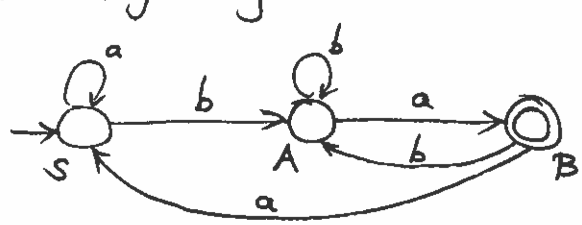
def: A regular grammar is any cfg in which the production rules are restricted to be of the form

$$X \longrightarrow \alpha Y \quad \text{or} \quad X \longrightarrow \alpha$$

where X, Y : non terminals & $\alpha \in \Sigma^*$

Alg 1 : Given a reg. lang. Construct a reg. grammar

ex:



non-terminals

$$G: \quad \begin{cases} S \longrightarrow aS \\ S \longrightarrow bA \end{cases}$$

$$\begin{cases} A \longrightarrow bA \\ A \longrightarrow aB \end{cases}$$

$$\{ B \longrightarrow aS \qquad B \longrightarrow bA \qquad B \longrightarrow \epsilon$$

~~pumping~~ eg: abba

$S \Rightarrow aS \Rightarrow abA \Rightarrow abbA \Rightarrow abbaB \Rightarrow abba$

Alg2: Given a reg. grammar, G Construct a NFA/DFA to accept L(G)

ex:

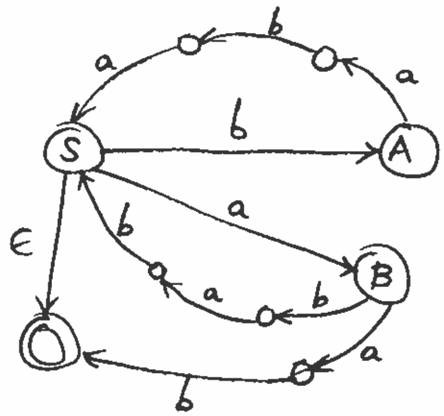
$S \longrightarrow aB$

$S \longrightarrow bA$

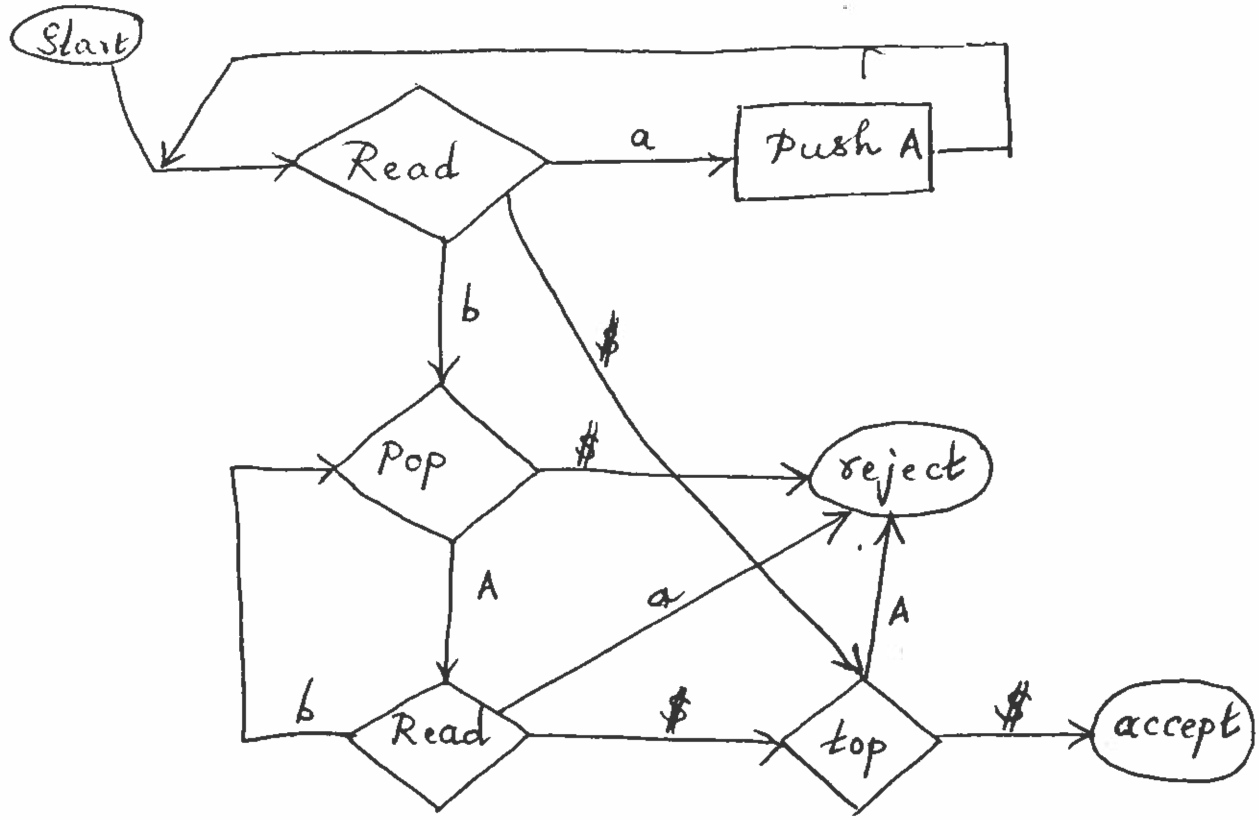
$S \longrightarrow \epsilon$

$A \longrightarrow abaS$

$B \longrightarrow babS$ $\qquad B \longrightarrow ab$



3.7 Pushdown Automata (PDA) 02/10/97



Δ	$(a, \$)$	(a, A)	$(b, \$)$	(b, A)	$(\epsilon, \$)$
q_0	$(q_0, A\$)$	(q_0, AA)		(q_1, ϵ)	$(q_2, \$)$
q_1				(q_1, ϵ)	$(q_2, \$)$

q_0 – start state

q_1 – final state

$$L(M) = \{ \omega \in \Sigma^* \mid \underbrace{(s, \omega, z)}_{\text{Configuration}} \vdash_M^* (p, \epsilon, u) \text{ and } p \in F \}$$

$$(s, \omega, z) \vdash^{\text{one step}} (\;,\;,\;) \vdash^{\text{one step}} (\;,\;,\;)$$

s: Current state
ω: what is remaining in input
z: what's on stack

$(s, \omega, \$)$
initial Configuration

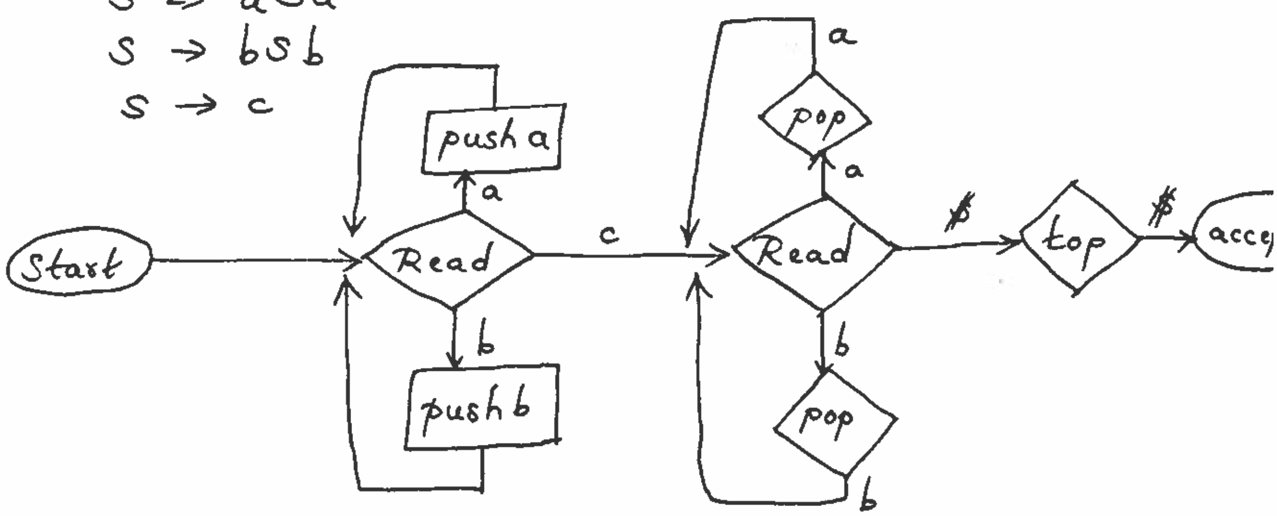
$$N(M) = \{ \omega \in \Sigma^* \mid (s, \omega, z) \vdash_M^* (p, \epsilon, z) \text{ and } p \in F \}$$

ex:

$$L = \{ \omega c \omega^R \mid \omega \text{ in } \{a, b\}^* \} \qquad \Sigma = \{a, b, c\}$$

odd palindrome with 'c' in middle

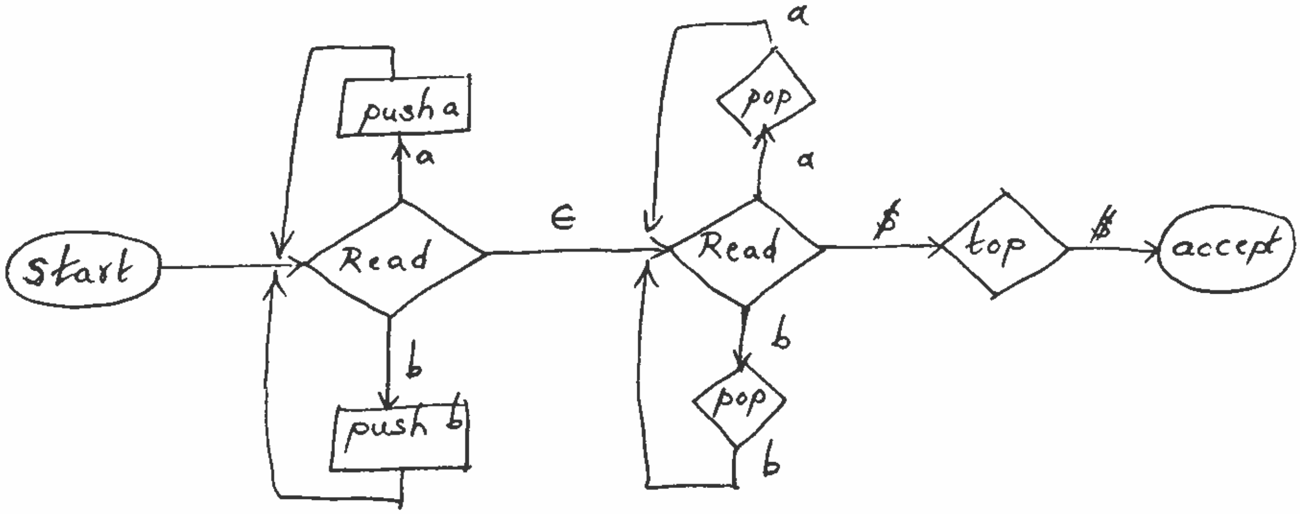
$S \Rightarrow aSa$
$S \rightarrow bSb$
$S \rightarrow c$



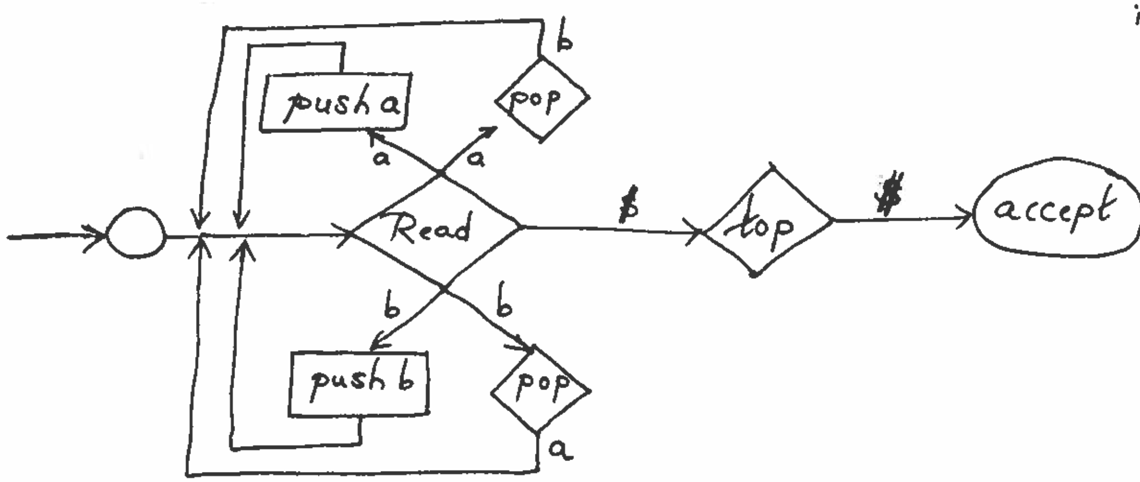
→

ex: $\mathcal{L} = \{ \omega \cdot \omega^R \mid \omega \text{ in } \{a,b\}^* \}$

even palindrome



ex: $EQUAL = \{ w \mid w \text{ in } \{a,b\}^* \text{ and } \# a\text{'s in } w = \# b\text{'s in } w \}$



$S \rightarrow aB$

$S \rightarrow bA$

$B \rightarrow bS$

$B \rightarrow aBB$

$B \Rightarrow b$

$A \rightarrow aS$

$A \rightarrow bAA$

$A \rightarrow a$

$ = excess # a's seen so far

or

excess # b's seen so far

Properties of Context-free Languages

02/13/97

CFL's are closed under union, Concatenation and Kleene star.

Let L_1 and L_2 be CFL's.

Then $G_1 = (N_1, \Sigma_1, P_1, S_1)$ for L_1 i.e. $L(G) = L_1$

$G_2 = (N_2, \Sigma_2, P_2, S_2)$ for L_2 i.e. $L(G) = L_2$

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

Such that $L(G) = L(G_1) \cup L(G_2)$

use renami
to ensure thi

$$N = N_1 \cup N_2 \cup \{S\} \qquad [N_1 \cap N_2 = \phi]$$

$$\Sigma = \Sigma_1 \cup \Sigma_2 \ (\text{normally } \Sigma_1 = \Sigma_2)$$

$$P = P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\}$$

$S \to S_1 S_2$ Concatenation

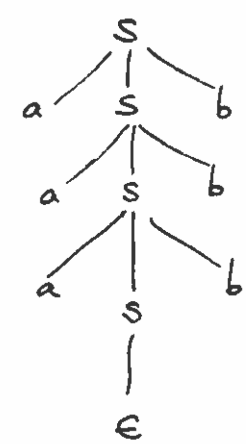
$S \to \epsilon$ star
$S \to S_1 S$

3.4 Derivation tree/ Parse tree

Ambiguity

ex: 1. $S \to aSb$
2. $S \to \epsilon$

$S \Rightarrow aSb \Rightarrow aaSbb$
$\Rightarrow aaaSbbb$
$\Rightarrow aaabbb$



$\longrightarrow$

ex:

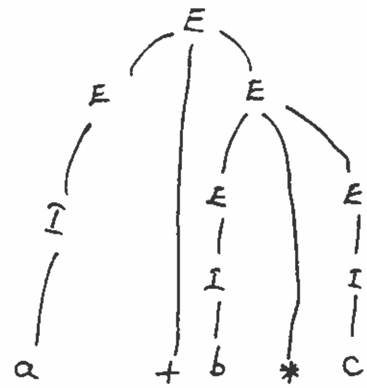
$E \longrightarrow E + E$
$E \longrightarrow E * E$
$E \longrightarrow (E)$
$E \longrightarrow I$
$I \longrightarrow a / b / c$

$\omega = a + b * c$

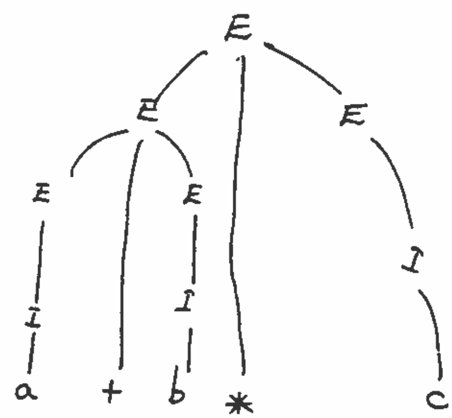
$N = \{E, I\}$

$\Sigma = \{a, b, c, +, *, (,)\}$

$$
\begin{aligned}
E &\Rightarrow E + E \\
&\Rightarrow I + E \\
&\Rightarrow a + E \\
&\Rightarrow a + E * E \\
&\Rightarrow a + I * E \\
&\Rightarrow a + b * E \\
&\Rightarrow a + b * I \\
&\Rightarrow a + b * c
\end{aligned}
$$



$$
\begin{aligned}
E &\Rightarrow E * E \\
&\Rightarrow E * I \\
&\Rightarrow E * c \\
&\Rightarrow E + E * c \\
&\Rightarrow I + E * c \\
&\Rightarrow a + E * c \\
&\Rightarrow a + I * c \\
&\Rightarrow a + b * c
\end{aligned}
$$

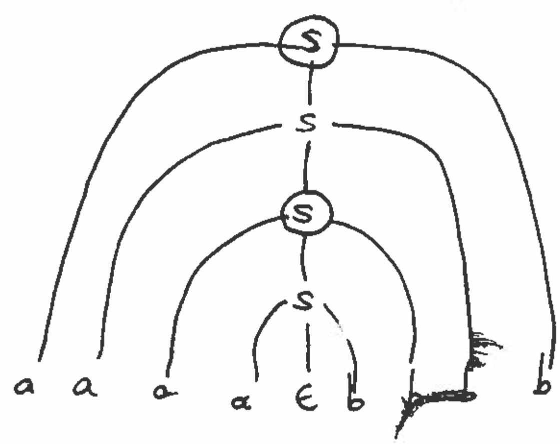


3.6 Pumping Lemma for CFL

motivation

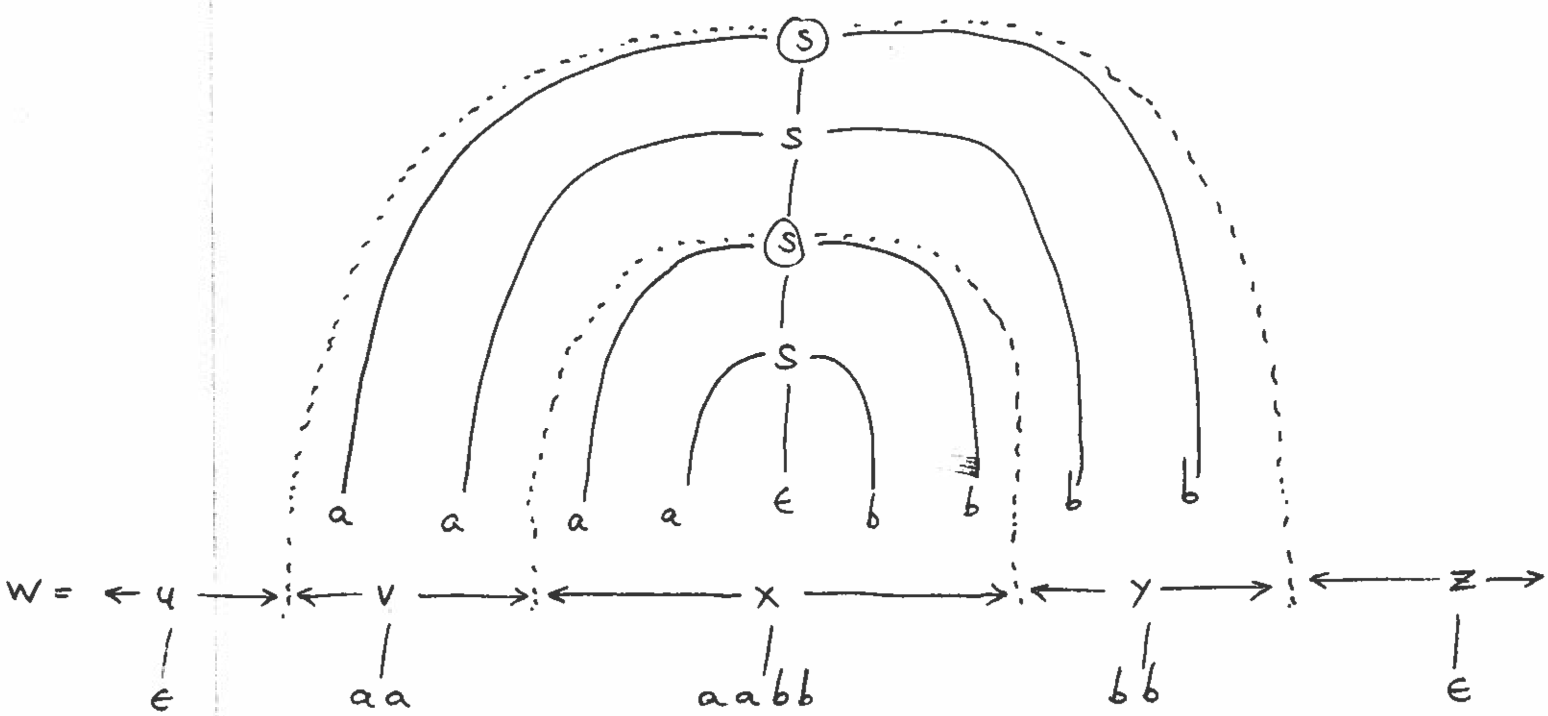
ex: $\begin{cases} S \rightarrow aSb \\ S \rightarrow \epsilon \end{cases}$

$\longrightarrow$



1) "repeating non-terminal" in some path from root to leaf in the parse tree.

instead of expanding 2^{nd} S the way it was expanded in the parse tree, try expanding it like the 1^{st} S.



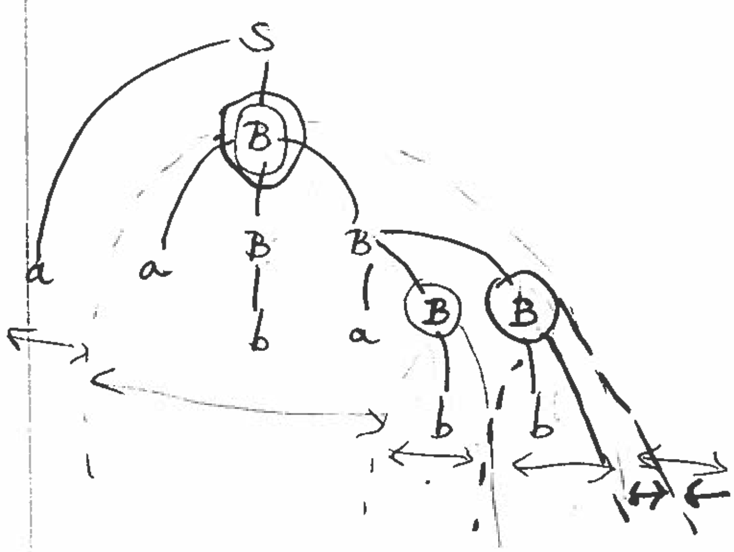
$\Rightarrow w = u v^i x y^i z \in L \; ; \; i \geq 0$

ex : EQUAL

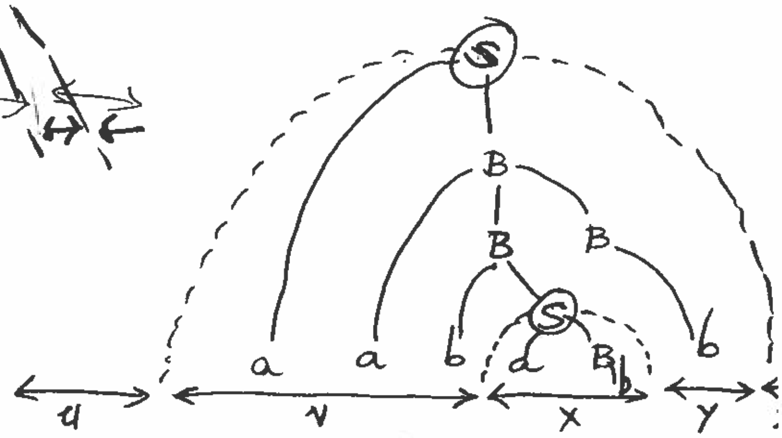
$S \rightarrow aB$
$S \rightarrow bA$
$A \rightarrow a$
$A \rightarrow bAA$
$A \rightarrow aS$
$B \rightarrow b$
$B \rightarrow aBB$
$B \rightarrow bS$

$w = aababb$

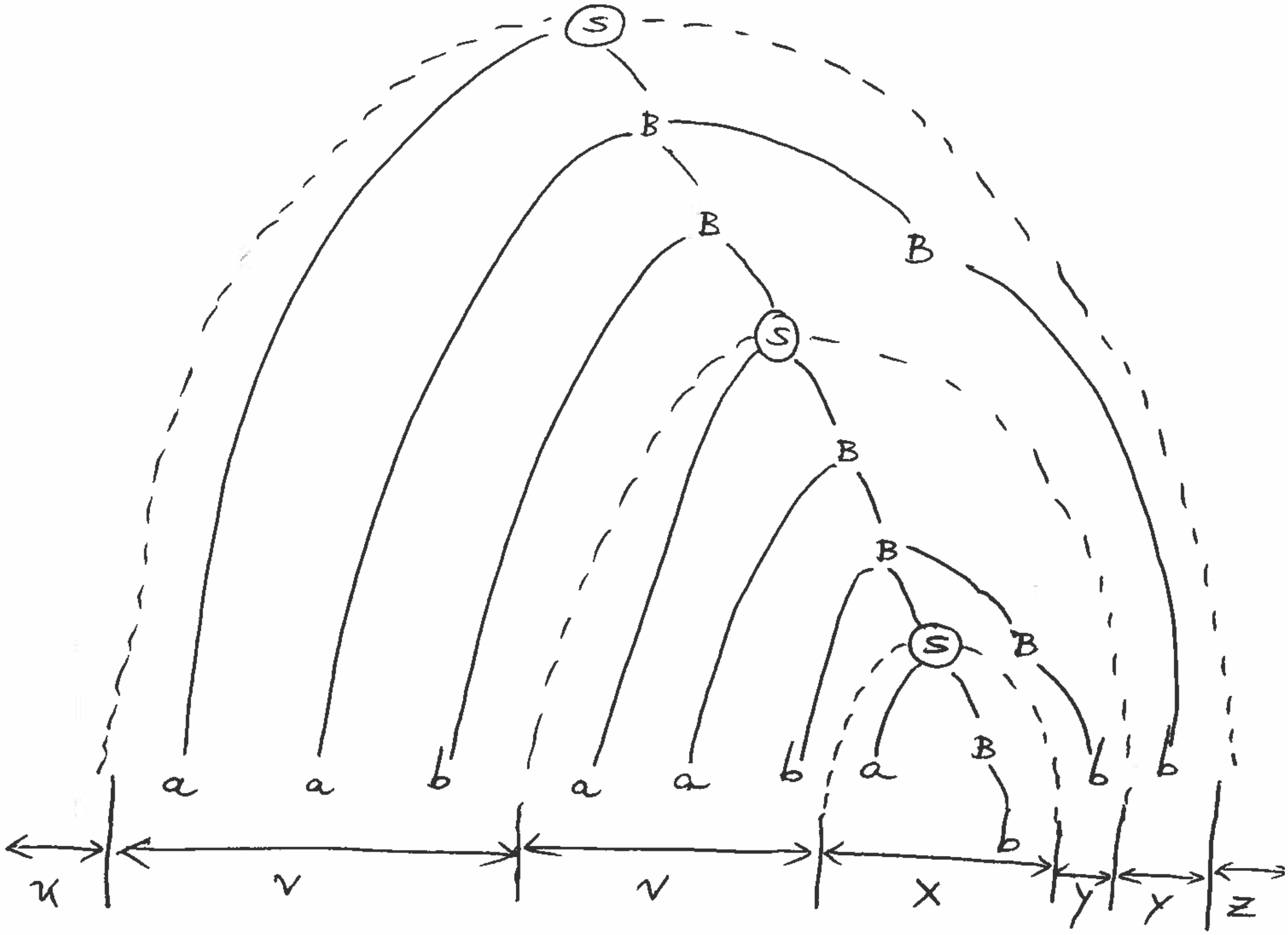
$S \Rightarrow aB$
$\Rightarrow aaBB$
$\Rightarrow aabB$
$\Rightarrow aabaBB$
$\Rightarrow aababB$
$\Rightarrow aababb$



$S \Rightarrow aB$
$\Rightarrow aaBB$
$\Rightarrow aabSB$
$\Rightarrow aabaBB$
$\Rightarrow aababB$
$\Rightarrow aababb$



$u = \epsilon$
$v = aab$
$x = ab$
$y = b$
$z = \epsilon$



$uv^ixy^iz \in L$ $|vy| \geq 1$

$|vxy| \leq n$

Pumping Lemma for CFL's

Let L be any (infinite) CFL. Then, there exists a Constant N such that for all w

if $w \in L$ and $|w| \geq N$ then

there exists u, v, x, y, z such that $w = uvxyz$

and

1) $|vy| \geq 1$

and 2) $|vxy| \leq n$

and 3) $uv^i x y^i z \in L$, $i \geq 0$

---

ex: $L = \{ a^n b^n c^n \mid n \geq 0 \}$ is not CFL

Proof: Let L be Context-free
then P.L. guarantees constant N

choose $w = a^N b^N c^N$
Consider the possibilities for v, y

Either → v or y have 'ab' or 'bc' as a substri

$v = a^{k_1}$; $y = a^{k_2}$

$v = a^{k_1}$; $y = b^{k_2}$

$v = a^{k_1}$; $y = c^{k_2}$

$v = b^{k_1}$; $y = b^{k_2}$

$v = b^{k_1}$; $y = c^{k_2}$

⇒ $i = 2$ only of the 3 grou will increas

'b' will appea before 'a'

'c' will appec before 'b'

Claim: $L = \{a^n b^{n^2} \mid n \geq 0\}$ is not Context-free

Proof: Let L be a CFL
Then P.L. guarantees N
Choose $\underline{w = a^N b^{N^2}}$

Consider all possibilities for v, y

1) v or y has 'ab' as a Substring

Either

$v = a^{k_1}$; $y = a^{k_2} \Rightarrow i = 2$ $\quad \#b's = N^2$, $\#a's = N + k_1 + k_2$, $k_1 + k_2 \geq 1$

$v = a^{k_1}$; $y = b^{k_2} \Rightarrow$ $\#a's = N + k_1$, $\#b's = N^2 + k_2$

$v = b^{k_1}$; $y = b^{k_2} \Rightarrow \#a's = N$, $\#b's = N^2 + k_1 + k_2$, $k_1 + k_2 \geq 1$

$$|vxy| \leq n$$

$$(N + k_1)^2 = N^2 + k_2$$

$$N^2 + k_1^2 + 2k_1 N = N^2 + k_2 \quad ; \quad k_1 = 0 ; k_2 \neq 0, \quad k_2 = 0 ; k_1 \neq 0$$

$$N < k_1^2 + 2k_1 N = k_2 \leq N$$

Contradiction

* H/W

Claim : $L = \{a^i \mid i \text{ is prime}\}$ is not CFL

$$m_{pq} = \{ a^p b^q c^r \mid p > q \}$$

$$m_{qp} = \{ a^p b^q c^r \mid q > p \}$$

Recall : CFL's are closed under union, concatenation, Kleene star.

$$S \to S_1 | S_2 \;;\; S \to S_1 S_2 \;;\; S \to S S_1 | \epsilon$$

(Un)fortunately, CFL's are not closed under complementation and intersection.

ex: Intersection

$$L_1 = \{ a^n b^n c^m \mid n \geq 0, m \geq 0 \}$$

$$L_2 = \{ a^m b^n c^n \mid n \geq 0, m \geq 0 \}$$

L_1	L_2
$S \to AC$	$S \to AB$
$A \to aAb \mid \epsilon$	$A \to aA \mid \epsilon$
$C \to cC \mid \epsilon$	$B \to bBc \mid \epsilon$

$L_1 \cap L_2 = a^n b^n c^n \therefore$ not Context-free

$A \cap B = \overline{\overline{A} \cup \overline{B}} \quad \therefore$ not Context-free

---

ex: $L = \{ a^n b^n c^n \mid n \geq 0 \}$

$\overline{L} = L_1 \cup L_2$

where $L_1 = \{ w \mid w$ is not of the form $a^{k_1} b^{k_2} c^{k_3} \}$

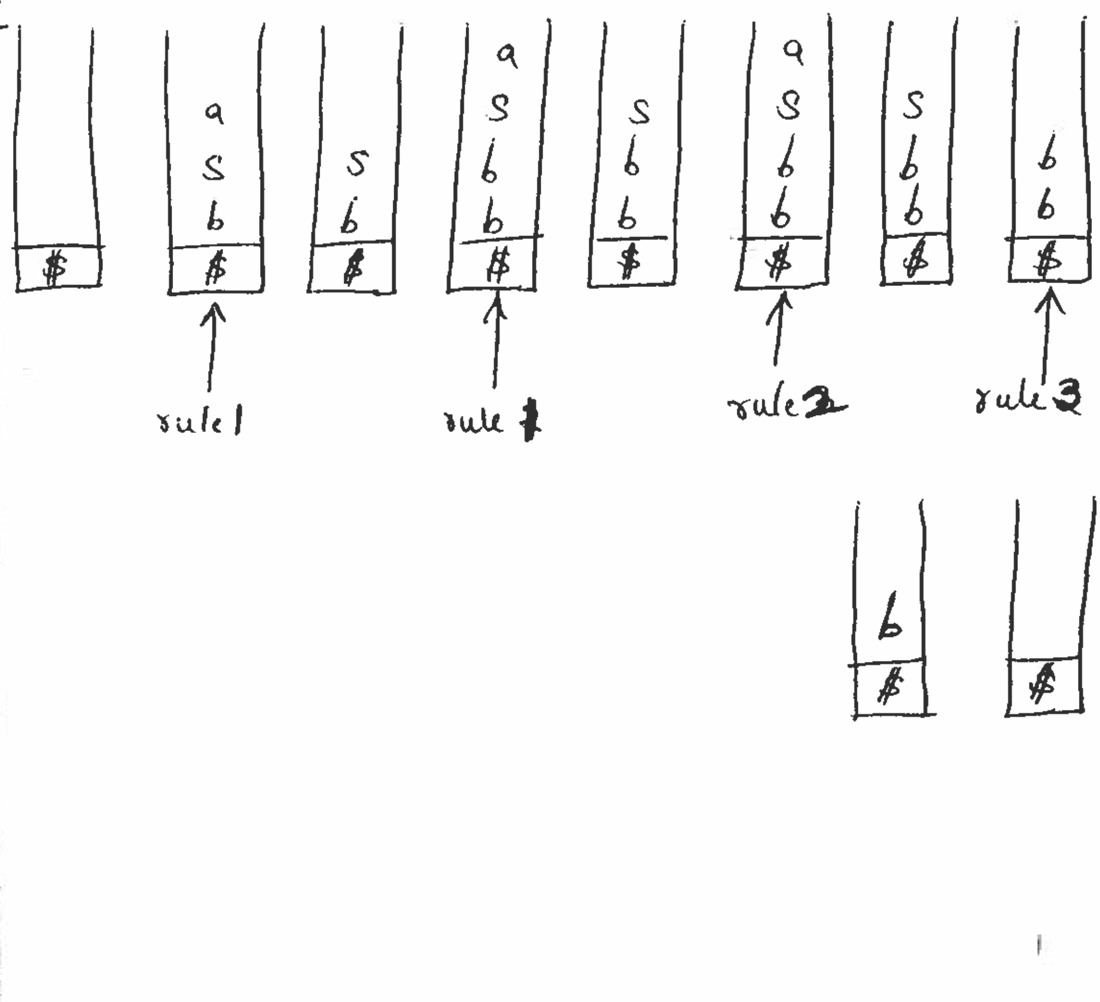
$L_2 = \{ w \mid w$ is of the form $a^{k_1} b^{k_2} c^{k_3}$ and $(k_1 \neq k_2)$ or $(k_2 \neq k_3)$ or $(k_1 \neq k_3) \}$

$L_2 = M_{pq} \cup M_{pr} \cup M_{qr} \cup M_{qp} \cup M_{rp} \cup M_{rq}$

$(\rightarrow^*)$

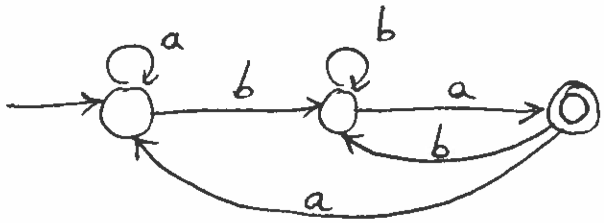
Consider $\underbrace{a}_{a}\,\underbrace{a\,a\,b}_{S}\,\underbrace{b}_{b}$

<u>Stack</u>

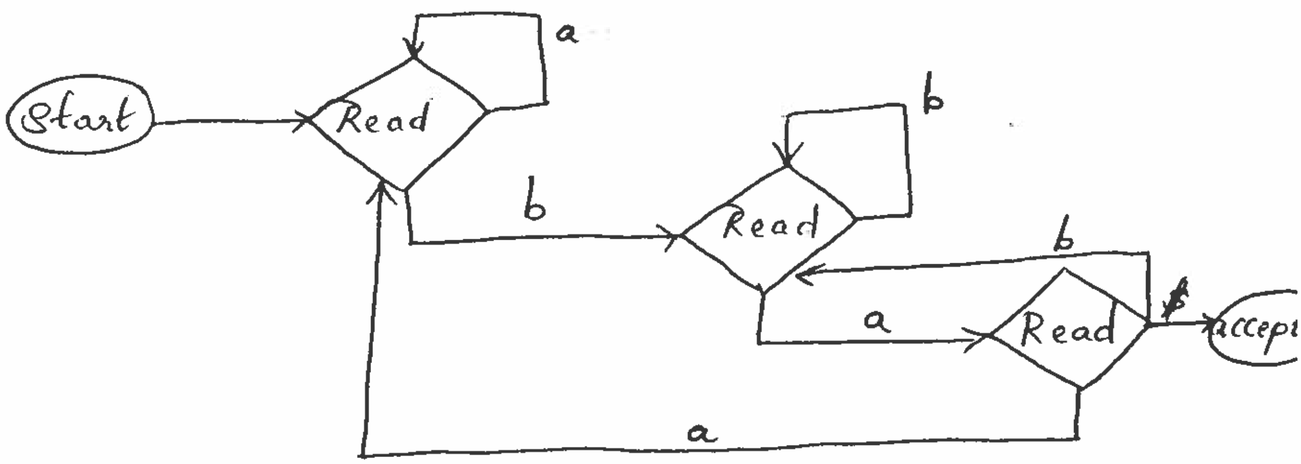


02/19/97

<u>observation</u>: Every dfa/nfa is equivalent to some nPDA (which does not operate on its stack



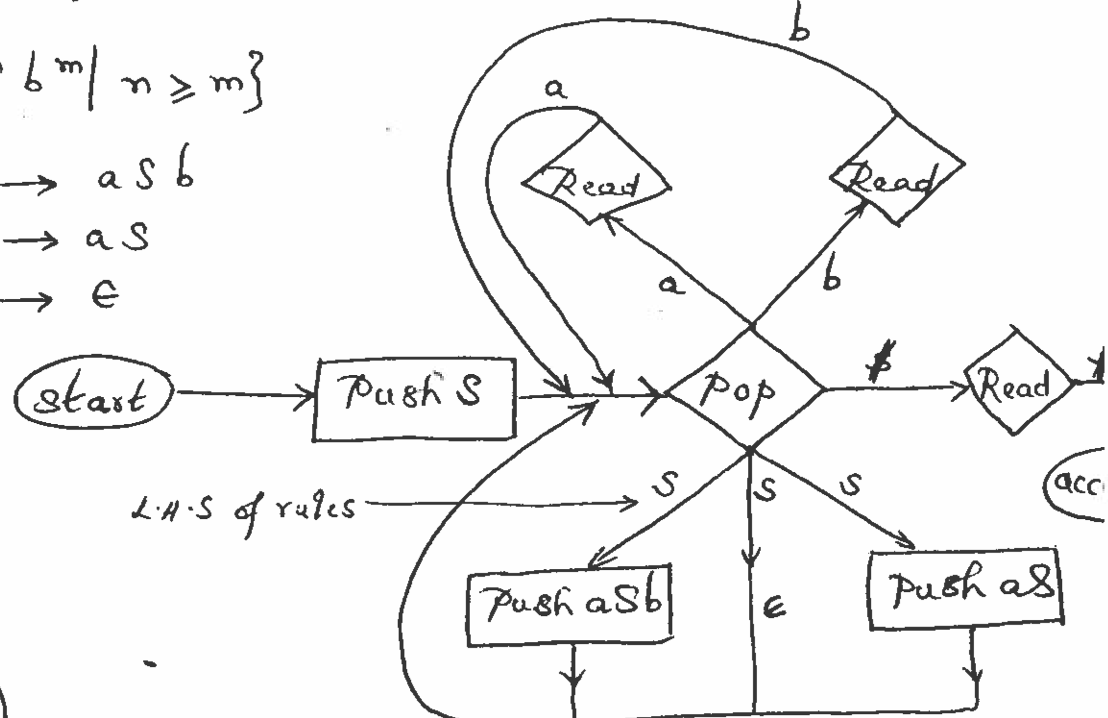
ending with 'ba'



<u>Alg</u>: CFG $\longrightarrow$ nPDA

ex: $\{a^n b^m \mid n \geq m\}$

1. $S \rightarrow aSb$
2. $S \rightarrow aS$
3. $S \rightarrow \epsilon$



($\rightarrow$ *)

$\Downarrow$

goto previous page

ch. 3	Skipped
3.1	
3.2	
3.3	
3.4	
	3.5 (Chomsky Normal form)
3.6	3.6.2 (Lemma CYK alg.)
3.7	
3.8	3.8.3 (nPDA $\rightarrow$ CFG)

ch.4 Turing Machines

02/19/97

- i/p is placed on tape
- to start with R/w head points to cell containing leftmost i/p symbol
- blank symbol, #
- R/w head can move L or R (one cell at a time)

Def: A TM $M = (Q, \Sigma, s, \delta)$

Q : finite set of states

Σ : alphabet (includes #)

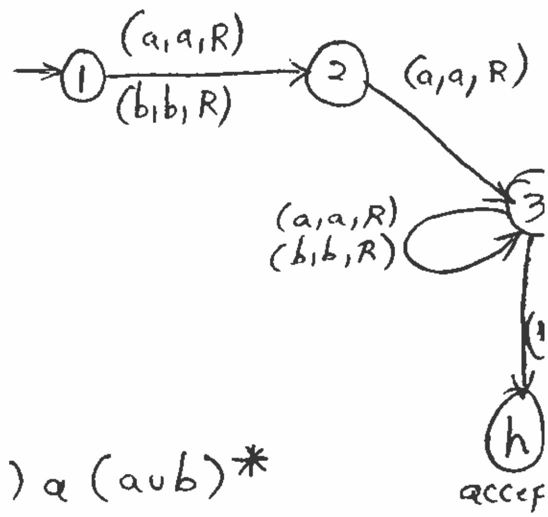
$s \in Q$: start state

h : special "state" not in Q called halt 'state'

$\delta : Q \times \Sigma \longrightarrow (Q \cup \{h\}) \times \Sigma \times \{L, R\}$

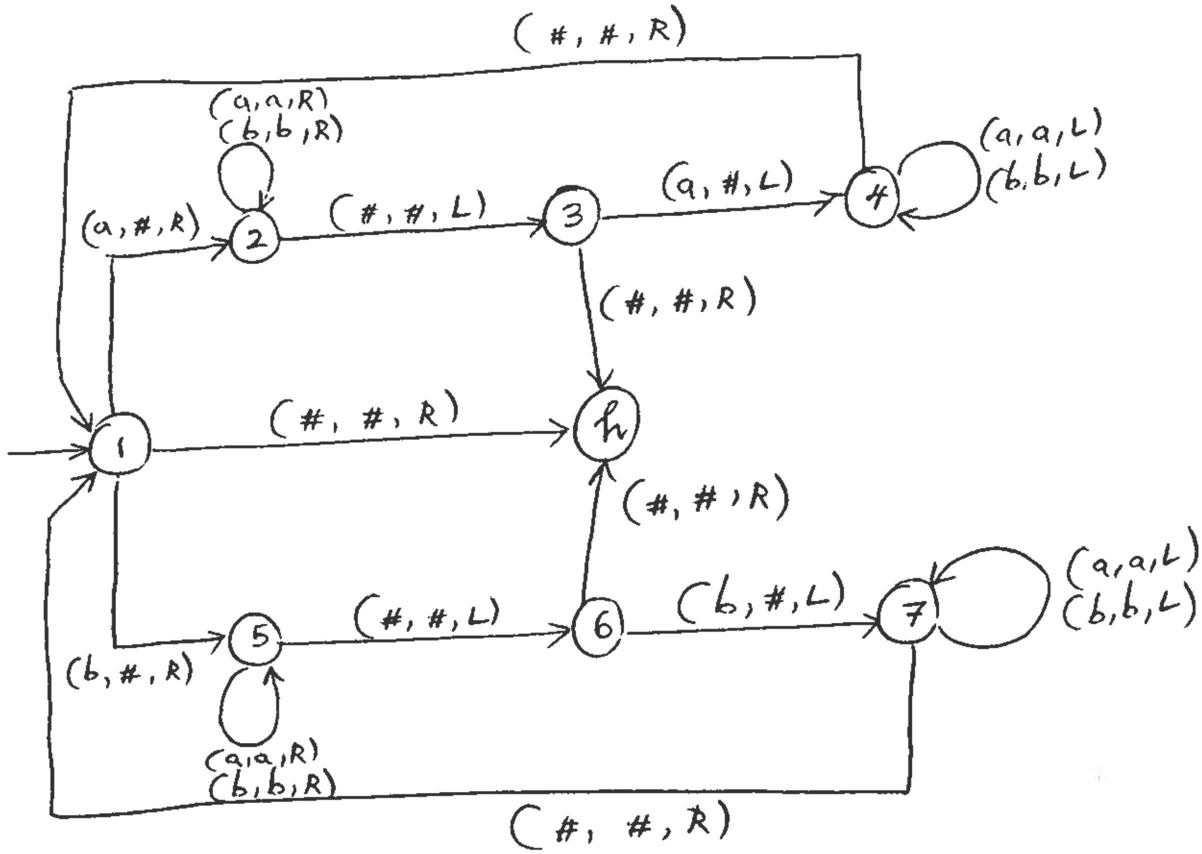
i.e. $\boxed{\delta(q, a) = (p, b, L)}$

From	To	Read	Write	Move
1	2	a	a	R
1	2	b	b	R
2	3	a	a	R
3	3	a	a	R
3	3	b	b	R
3	h	#	#	R



$\Rightarrow (a \cup b)\, a\, (a \cup b)^*$

T.M. for Palindrome



Configuration

(1, abbba) $\vdash$ (2, bbba)

$\vdash$ (2, bbba)

$\vdash$ (2, bbba)

$\vdash$ (2, bbba)

$\vdash$ (2, bbba#)

$\vdash$ (3, bbba)

$\vdash$ (4, bbb)

$\vdash$ (4, bbb)

$\vdash$ (4, bbb)

$\vdash$ (4, #bbb)

$\boxed{\vdash (1, bbb)}$

$\vdash$ (5, bb)

$\vdash$ (5, bb)

$\vdash$ (5, bb#)

$\vdash$ (6, bb)

$\vdash$ (7, b)

$\vdash$ (7, #b)

$\boxed{\vdash (1, b)}$

$\vdash$ (5, ##)

$\vdash$ (6, ##)

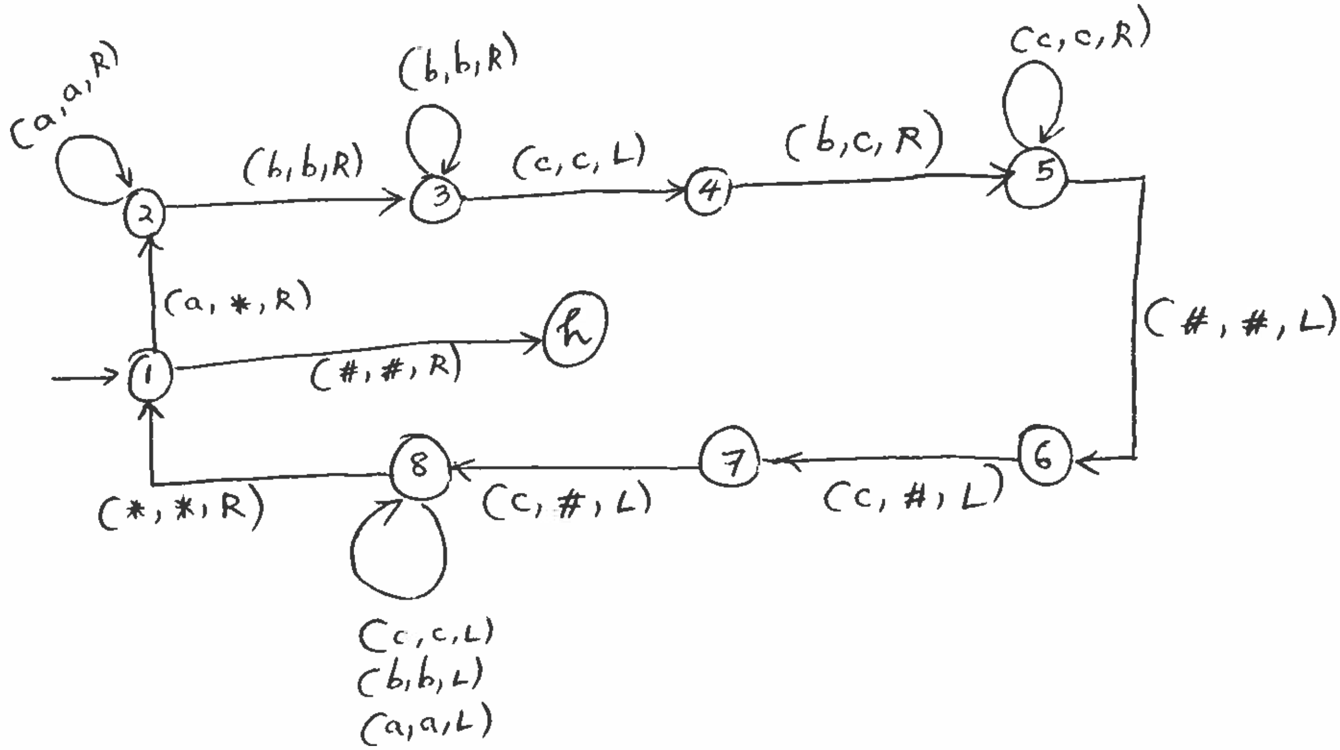
$\vdash$ (h, ##)

ex: TM for $\{a^n b^n c^n \mid n \geq 0\}$

a a a b b b c c c
* a a b b b c c c
* a a b b c c c c — delete
* a a b b c c

* * a b b c c
* * a b c c c
* * a b c

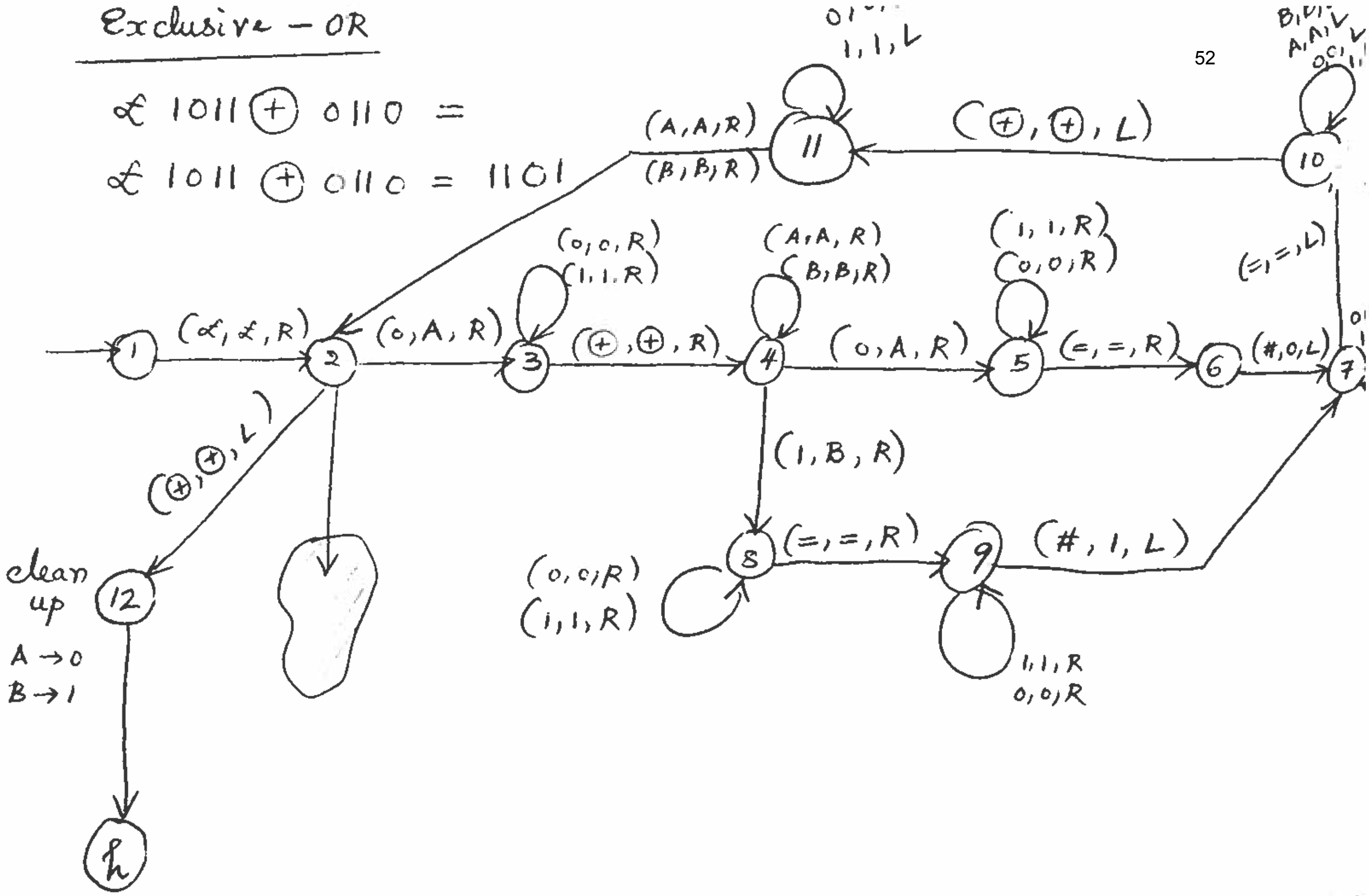
* * a b c
* * * b c
* * * c c
* * *



Exclusive – OR

$\mathcal{L}\ 1011 \oplus 0110 =$

$\mathcal{L}\ 1011 \oplus 0110 = 1101$



$$\begin{array}{r} 1011 \\ 0110 \\ \hline 1101 \end{array}$$

Unary Notation for numbers (≥ 0)

0	ϵ
1	a
2	aa
3	aaa
4	aaaa

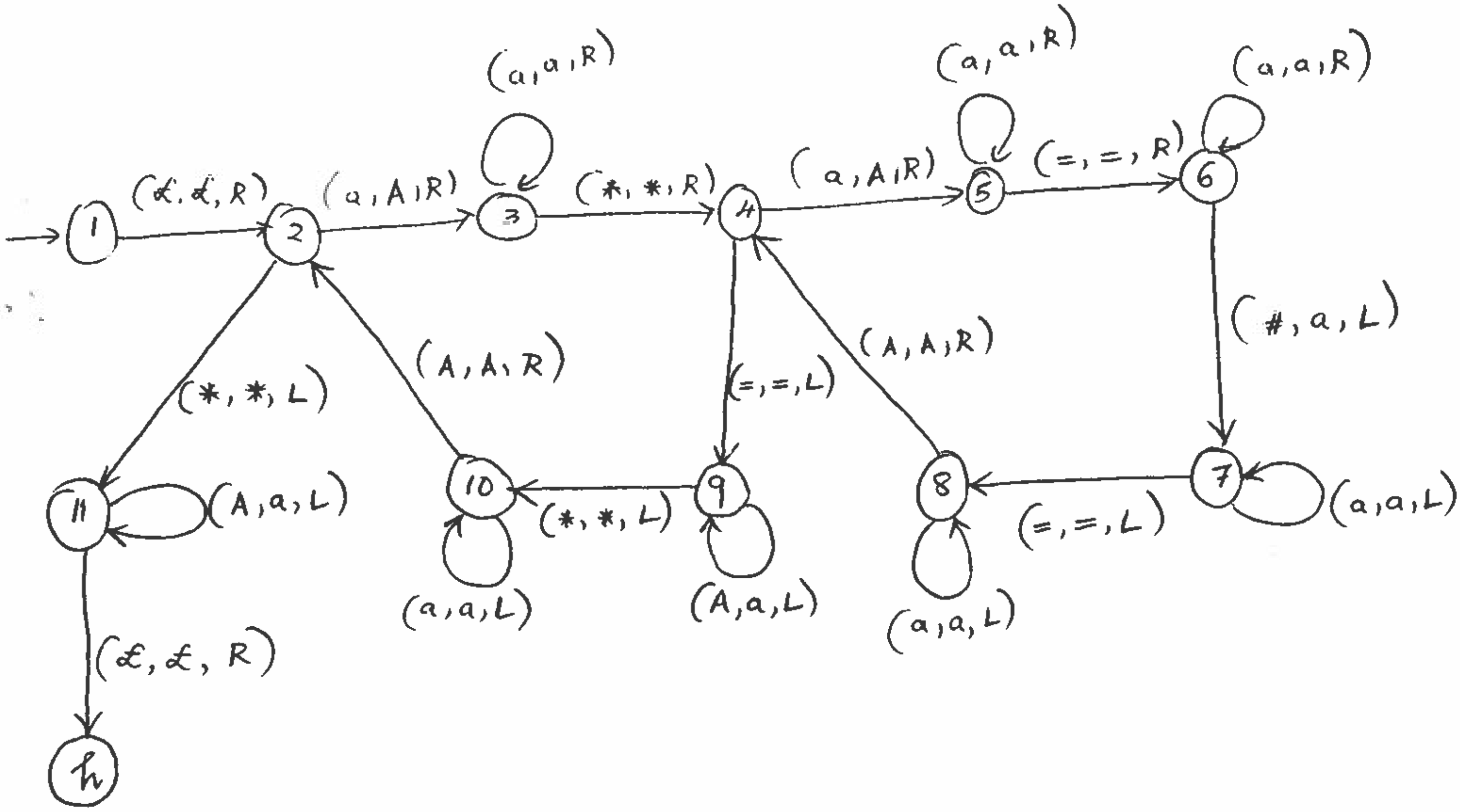
Given i/p tape as follows

£	— n_1 —	*	— n_2 —	=

↓

£	— n_1 —	*	— n_2 —	=	$n_1 * n_2$

TM to multiply 2 no's



£ aaa * aa = £ aaaaaa

02/26/97

On an input, a T.M
- halts (accepts)
- crashes (rejects)
- infinite loop (reject!)

<u>Def</u>: Let M be a T.M. (tape alphabet $= \Sigma$)
Then, Σ^* can be partitioned into 3 parts

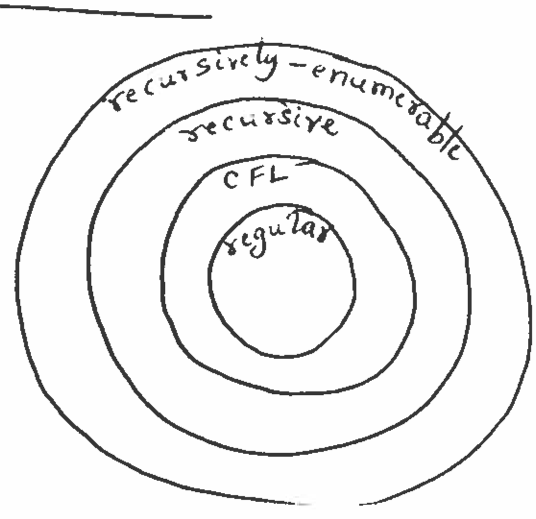
1) $\text{ACCEPT}(M) = \{ w \in \Sigma^* \mid M \text{ reaches halt state on i/p } w \}$
2) $\text{REJECT}(M) = \{ w \in \Sigma^* \mid M \text{ crashes on i/p } w \}$
3) $\text{LOOP}(M) = \{ w \in \Sigma^* \mid M \text{ runs forever on i/p } w \}$

<u>Def</u>: A language L, is <u>recursive</u> if there exists a T.M, M such that

$$\text{ACCEPT}(M) = L$$
$$\text{REJECT}(M) = \Sigma^* - L$$
$$\text{LOOP}(M) = \phi$$

<u>Def</u>: A language is <u>recursively enumerable</u> if there exist a T.M, M such that

$$\text{ACCEPT}(M) = L$$



Thm : If L is recursive then $\overline{L}$ is also recursive
(If L_1 and L_2 are recursive then so is $L_1 \cup L_2$)

Thm : If L is recursively enumerable and $\overline{L}$ is recursively enumerable then L is recursive.

Def. L is recursive if there ~~exists~~ is a T.M. M such that

$$ACCEPT(M) = L$$
$$REJECT(M) = \overline{L}$$
$$LOOP(M) = \phi$$

Def. L is recursively enumerable if there is a T.M. M such that

$$ACCEPT(M) = L$$

Encoding of T.M.

From	To	Read	Write	Move
1	2	a	a	L
1	3	b	b	R
2	1	a	a	L

States

1 for start
2 for h
3, 4, 5, 6 ..... for other states

	Code
a	aa
b	ab
#	ba
$	bb
L	a
R	b

def. $CWL = L((a^+ b a^+ b (a \cup b)^5)^*)$

Note:

(1) Every T.M. has a unique Code in CWL

(2) Not every Code in CWL Corresponds to a Valid T.M.

(Reason: may have outgoing transitions from 'halt' state = e)

def

ALAN = { w | w is in CWL <u>and</u>

(either w does not Correspond to a Valid

or M (The T.M. Corresponding to w) does no

accept w

) }

<u>def.</u> CWL = $L((a^+ b a^+ b (a \cup b)^5)^*)$

<u>def.</u> ALAN = { w | w is in CWL and
either w is an invalid Code
or { w is not accepted
by M (M is the T.M. for w)) }

<u>def.</u> MATHISON = { w | w is CWL <u>and</u>
w is accepted by M
(M is the T.M. for w) }

<u>claim</u>: ALAN is not recursively enumerable

<u>Proof</u>: by Contradiction

— Let ALAN be recursively enumerable.
Therefore, there is a T.M., M which
accepts ALAN, ie <u>ACCEPT(M) = ALAN</u>

UTM (Universal T.M.)

is in CWL and is a valid code

any string

$£\ w_1 \ \$ \ w_2 \ =$

£ row1	row2	row3	------	row$_k$ \$	W_2

£ * row1	row2	row3	------	row$_k$ \$ W_2

£ * row1	row2	row3	-------	row$_k$	\$ * v

	Code
a	aa
b	ab
#	ba
\$	bb
L	a
R	b