Theory of Computation

letter

alphabet $\Sigma = \{a, b, c\}$ finite size

$|\Sigma| = 3$

word or string on alphabet Σ $w = abacab$

$w' = aacccbbaa$

Length of word $|w| = 6$ $|w'| = 9$

ϵ as string of length 0

$w\epsilon = w = \epsilon w$

w_1 w_2

Concatenation of w_1 and w_2

$w_1 w_2$

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

a palindrom

ϵ palindrom

aba palindrom

abba "

ababa ".

Prefix of a string w

$w = abcaab$

Prefix of w: ϵ, a, ab, abc ... abcaab

Proper Prefix remove ϵ, abcaab

Suffix of w: ϵ, b, ab, aab, --- abcaab

Proper Suffix of w: remove ϵ, abcaab

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

$\Sigma^1 = \{a, b\}$ $\quad \Sigma^2 = \{aa, ab, ba, bb\}$

$\Sigma^i = \{$ all possible string of length i which can be formed from the letters $a, b \}$

$\Sigma^* = \bigcup_{i \geq 1} \Sigma^i$ con ϵ.

$\Sigma^0 = \epsilon$

Countable Set

we get a one to one mapping from $|\mathbb{N}|$.

Language is a set of strings of letters from some alphabet

Example: $\Sigma = \{a, b\}$ $\qquad \Sigma = \{a, b, c\}$

$L \subseteq \Sigma^*$ $\qquad L \subseteq \Sigma^*$

$S = \{$ infinitely many infinite length strings on some alphabet $\Sigma \}$

S countable ?

$\boxed{S \text{ uncountable}}$

$S \subseteq \Sigma^*$ countable

$f : S \longrightarrow \Sigma^*$ one lon

diagonalization S uncountable

0 1 1 0 0 — — — ∞

1 0 0 1 1 . ——

1 —

1

. . .

0	1
1	2
00	3
01	4
10	⋮
11	
000	
001	
⋮	

$S \longrightarrow$ $|S| = i \longrightarrow j$

Proof Strategies:

Induction,
Direct
Contraposition
Existence — Constructive / Non constructive
⋮
Well ordering or Extremal principal
Contradiction
⋮

Relation:

R on a set S.

$R \subseteq \underbrace{S \times S}_{\text{Cartesian Product}}$

reflexive if aRa $\forall a \in S$

transitive $aRb,\ bRc \Rightarrow aRc$

symmetric $aRb \Rightarrow bRa$

Asymmetric $aRb \Rightarrow bRa$ is false

Equivalence Relation:

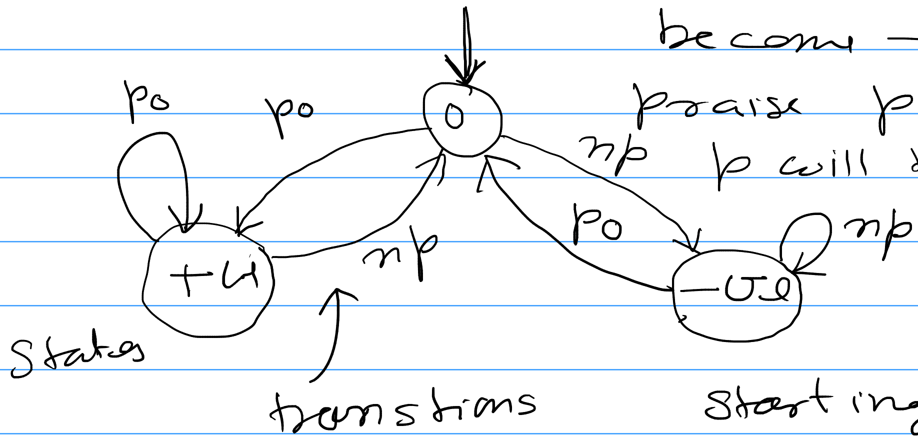
R is an equivalence relation if it is reflexive, transitive, and symmetric.

Thm R partition S in to disjoint non empty equivalence classes.

Finite Automata

states
input symbols
transition from one state to another state

p —— p_1 talks against p then mood of p will become -ve
praise p then p will become +ve



Starting state (one)
Multiple final states

input alphabet — initial state

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

Set of States — transition function — final states

$F \subseteq Q$

$$\delta : Q \times \Sigma \longrightarrow Q$$

$q', q \in Q$
$a \in \Sigma$

$$\delta(q, a) = q'$$

$$\delta(+ve, np) = 0$$

p0	np	p0	p0

0 +ve 0 +ve +ve

input string

We say that a string w accepted by the given automata when we reach final state after scanning the full string.

Design a finite automata which accept string on $\{0, 1\}$ and each of the string contains even number of 0's.

Language $L = \{ x \in \{0,1\}^* \mid x$ has even number of zero's $\}$

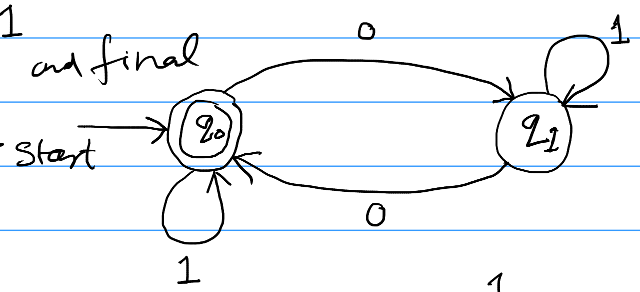
Number of 0's present in x mod 2

ϵ 0

ϵ 1

0001010 and final

1



M

rejected

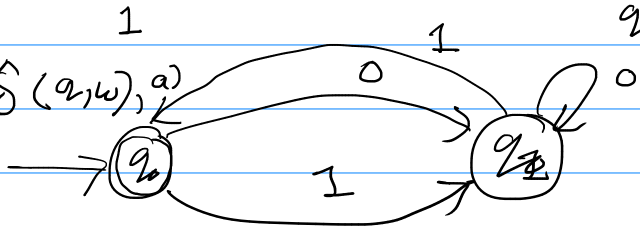
0001

$q_0\ q_1\ q_0\ q_1\ q_1$ Not final State

$\omega = 00010$

$a = 0$

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



State

$$\boxed{\begin{array}{ll} & \hat{\delta}(q, \epsilon) = q \\ \forall \text{ string } \omega \text{ and input letter } a & \hat{\delta}(q, \omega a) = \delta(\underbrace{\hat{\delta}(q,\omega)}_{p}, a) \end{array}}$$

$$\hat{\delta}(q_0, 000100) = \delta(\underline{\hat{\delta}(q_0, 00010)}, 0)$$
$$= \delta(q_0, 0)$$
$$= q_1$$

ω is accepted by FA, M if $\hat{\delta}(q_0, \omega) \in F$ (final state)

if not so then reject ω.

$L(M)$: Set of $\boxed{\text{all}}$ strings accepted by FA M.

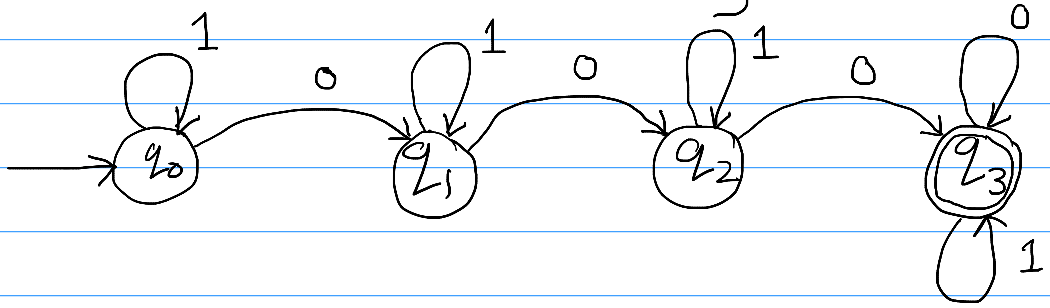
$L'(M) = \{00, 010\} = L(M)$ Not

$$\boxed{L(M) = L \quad (L(M) \subseteq L, \ L \subseteq L(M))}$$

L is accepted by FA M

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

$L = \{ x \in \{0,1\}^* \mid x$ contains at least 3 zero's $\}$

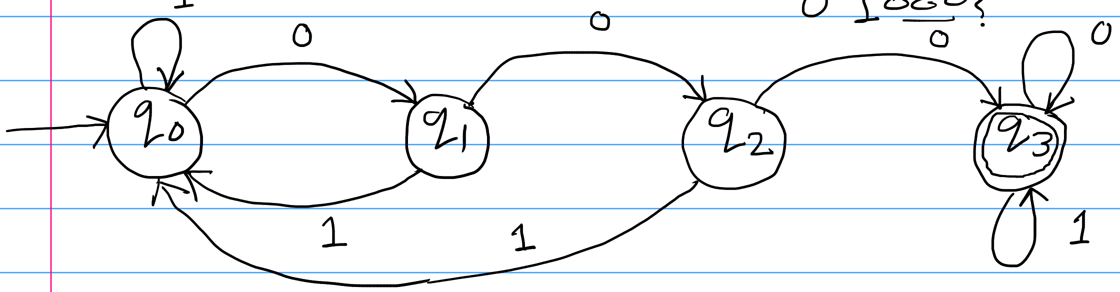


$L = \{ x \in \{0,1\}^* \mid x$ contains 000 as a factor $\}$

w contains factor y

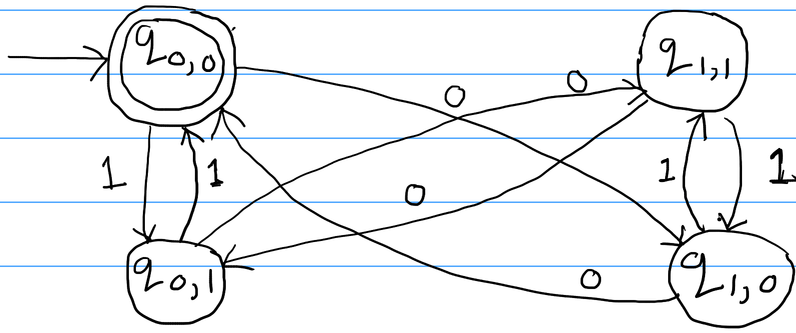
$w = u y v$ u, v might be empty

0 1 000?



$L = \{ x \in \{0,1\}^* \mid x$ contains even number of zero's and even number of 1's $\}$

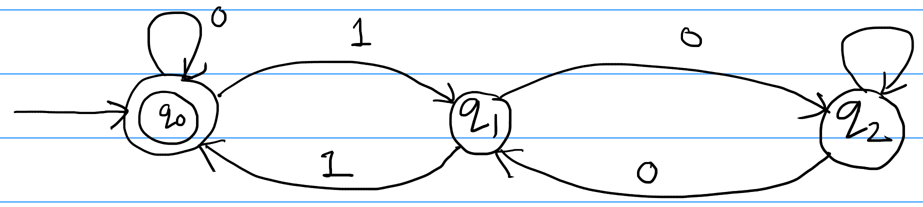
4 possibilities of parity of number of 0's and 1's



$d(010) = 2 \quad d(11) = 3 \quad d(0) = 0$

$$L = \{ x \in \{0,1\}^* \mid d(x) \bmod 3 = 0 \}$$

M



Prove that $L(M) = L$

$d(x) \bmod 3 = 0$

$x0 \quad d(x0) \bmod 3 = 0$

$d(x0) = 2d(x)$

$x1 \quad d(x1) = 2d(x) + 1$

$d(x1) \bmod 3 = 1$

$d(x) \bmod 3 = 1$

$d(x0) = 2d(x)$

$= 3 \times 2K + 2$

$d(x0) \bmod 3 = 2$

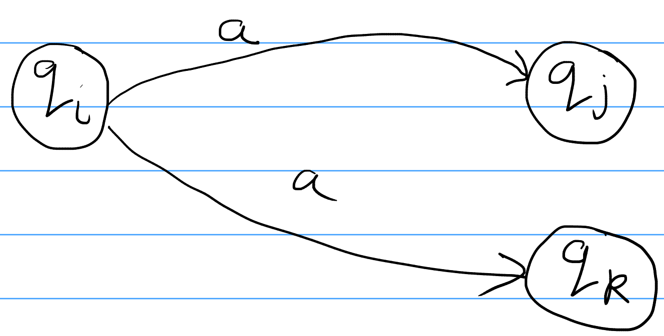
$d(x1) \bmod 3 = 0 \qquad q_0$

$$\boxed{\hat{\delta}(q_0, x) = q_{d(x) \bmod 3}}$$

Deterministic finite automata

All the languages which are accepted by DFA, we call <u>regular languages</u>

Non deterministic finite automata (NFA)



<u>allowed</u>

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

$$\delta : Q \times \Sigma \longrightarrow 2^Q$$ Power set of Q

$0\,1\,1\,0\,0\,0\,1$

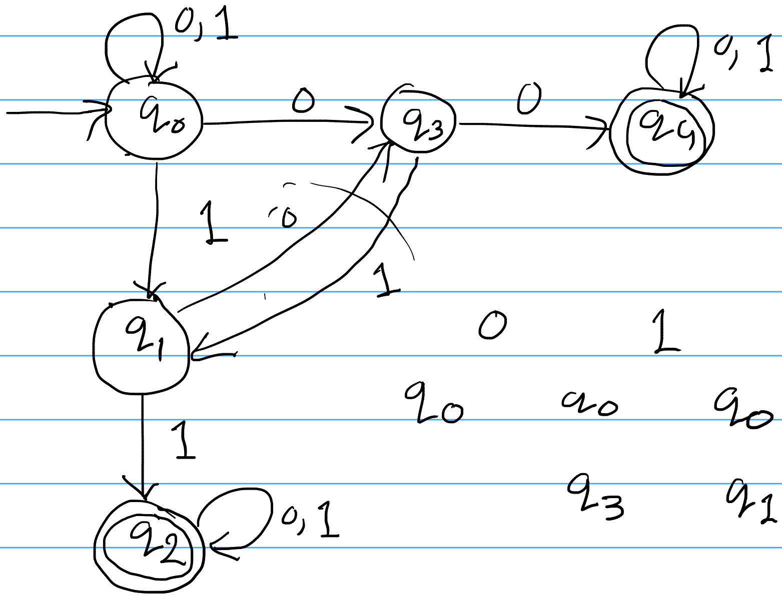
$\uparrow$ - - - - $\{q_{i_1}, q_{i_2}, \dots, q_{i_k}\}$

Accept : if $\exists$ a state which is final the last set of states.

Reject : $\forall$ state which are present in the final last set of state are non final.

$L = \{ x \in \{0,1\}^* \mid x$ has either two consecutive 0 or " " 1 $\}$

1



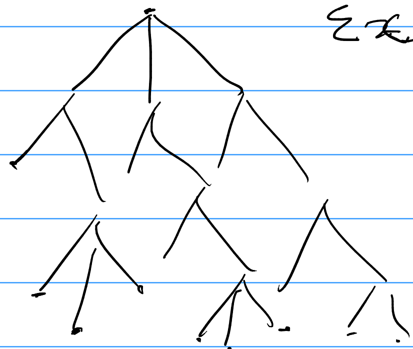
accepted

0	1	0	0	1		last state
q_0	q_0	q_0	q_0	q_0	q_0	
	q_3	q_1	q_3	q_3	q_1	
				q_4	q_4	

rejected

0	1	0	1		
q_0	q_0	q_0	q_0	q_0	No final State
	q_3	q_1	q_3	q_1	

Execution of DFA



Execution of NFA

final state

Defn: $\exists$ path in above tree which ends at final state

Accept:

Reject $\forall$ leaf nodes are non final

Defn 2: Guess and Verify

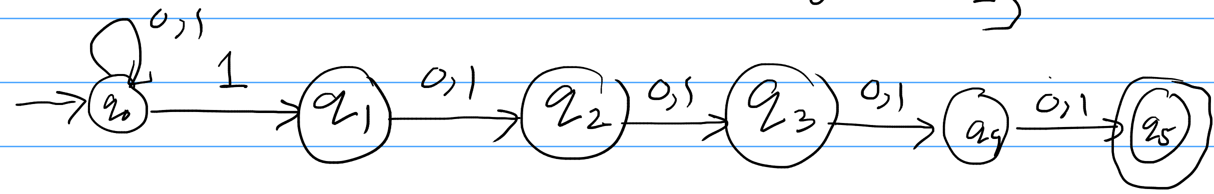
$\exists$ i/p guess

i/p : string, path

o/p : Y/N. verify the guessed path.

$$L = \{ x \in \{0,1\}^* : \text{the fifth symbol from right is } 1 \}$$

NFA for L



Proof

$$\hat{\delta}(q_0, x) = q_0 \iff d(x) \bmod 3 = 0$$
$$\hat{\delta}(q_0, x) = q_1 \iff d(x) \bmod 3 = 1$$
$$\hat{\delta}(q_0, x) = q_2 \iff d(x) \bmod 3 = 2$$

Base case $x = \epsilon$ $\quad d(x) \bmod 3 = 0$

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

The above expressions are true up to length n

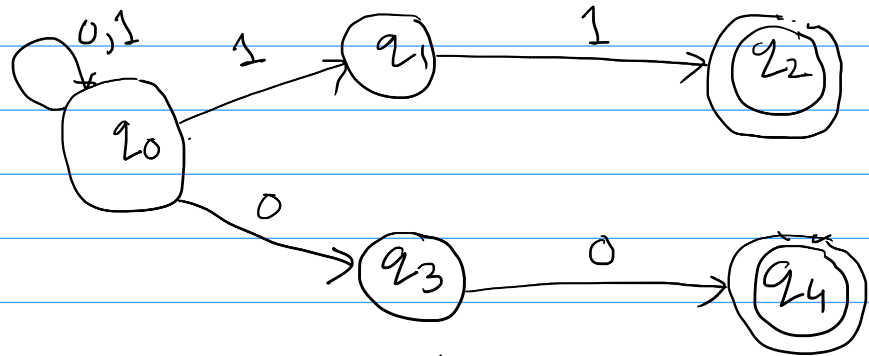
xa

$$\hat{\delta}(q_0, xa) = \delta(\hat{\delta}(q_0, x), a)$$
$$= \delta(q_{d(x) \bmod 3}, a)$$
$$= \boxed{q_{(2\, d(x) \bmod 3 + a) \bmod 3}}$$
$$= q_{d(xa) \bmod 3}$$

NFA (hopeless)

Thm:- Every NFA can be converted into a DFA.

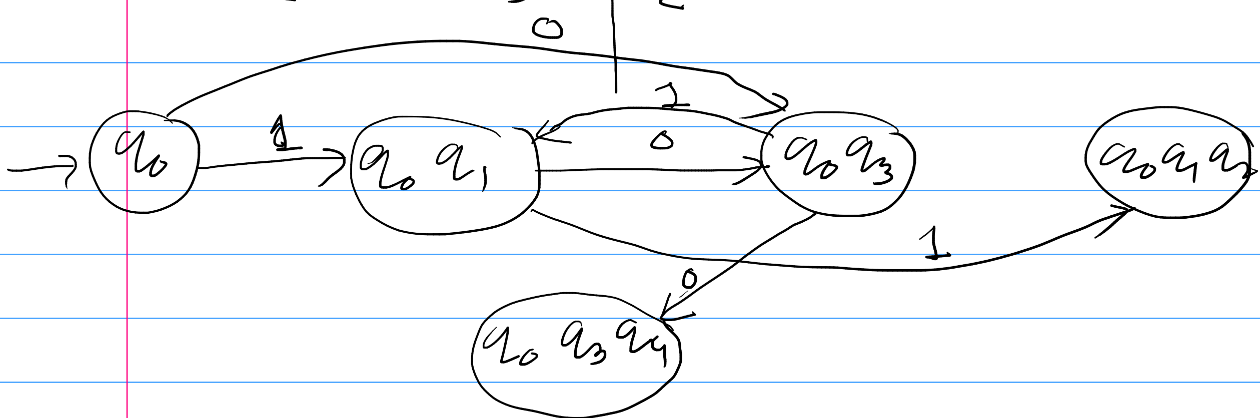
M

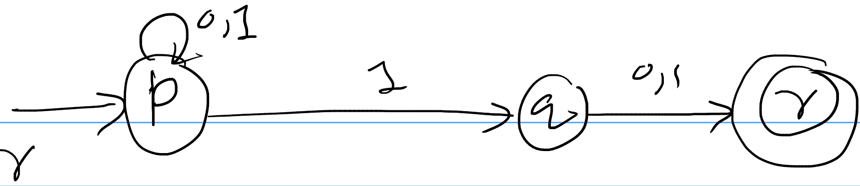


DFA $M' = (Q', \Sigma', \delta', q_0, F')$

$$\Sigma' = \Sigma$$

	0	1
q_0	$\{q_0, q_3\}$	$\{q_0, q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_3\}$	$\{q_0, q_1, q_2\}$
$\{q_0, q_3\}$	$\{q_0, q_3, q_4\}$	$\{q_0, q_1\}$
$\{q_0, q_1, q_2\}$	$\{q_0, q_3, q_2\}$	$\{q_0, q_1, q_2\}$
$\{q_0, q_3, q_4\}$	$\{q_0, q_3, q_4\}$	$\{q_0, q_1, q_4\}$



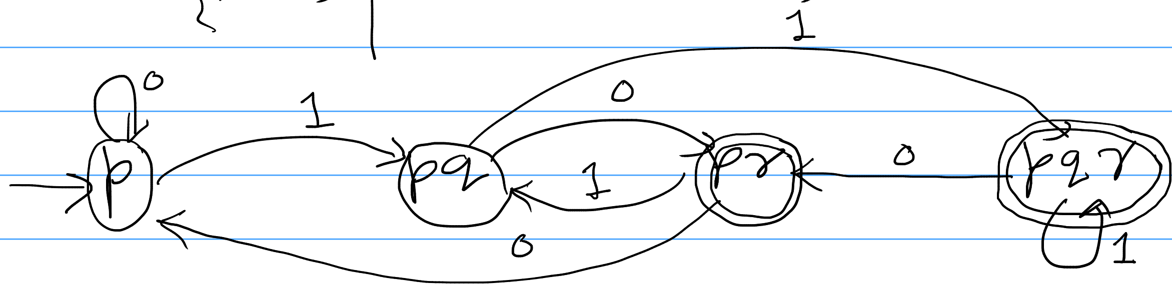


p   q   r

pq   pr   qr

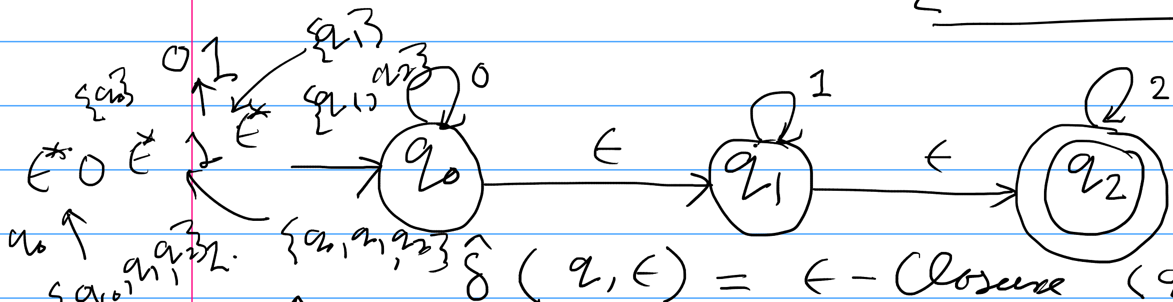
pqr

	0	1
p	p	$\{p, q\}$
$\{p, q\}$	$\{p, r\}$	$\{p, q, r\}$
$\{p, r\}$	$\{p\}$	$\{p, q\}$
$\{p, q, r\}$	$\{p, r\}$	$\{p, q, r\}$



Subset Construction

$L = \{ \underline{0^i 1^j 2^k} \mid i, j, k \geq 0 \}$



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

2. $\hat{\delta}(q, wa) = \epsilon\text{-Closure}(P)$

$w \in \Sigma^*$ $\qquad P = \{ p : \exists r \in \hat{\delta}(q_0, w),$

$a \in \Sigma$ $\qquad\qquad p \in \delta(r, a) \}$

$\epsilon\text{-Closure}(q) = \{ p : \exists$ a path from q to p labelled $\epsilon \}$

$\epsilon\text{-Closure}(q_0) = \{ q_0, q_1, q_2 \}$

$$\hat{\delta}(q_0,0) = \epsilon\text{-Closure}(\delta(\hat{\delta}(q_0,\epsilon),0))$$

$$= \epsilon\text{-closure}(\delta(\{q_0,q_1,q_2\},0))$$

$$= \epsilon\text{-closure}(\{q_0\} \cup \phi \cup \phi)$$

$$= \{q_0, q_1, q_2\}$$

$\delta(q,0)$: do not have to calculate ϵ-closure

$\hat{\delta}$

$$\hat{\delta}(q_0, 01) = \epsilon\text{-Closure}(\delta(\hat{\delta}(q_0,0), 1))$$

$$= \epsilon\text{-Closure}(\delta(\{q_0,q_1,q_2\},1))$$

$$= \epsilon\text{-Closure}(\phi \cup \{q_1\} \cup \phi)$$

$$= \{q_1, q_2\}$$

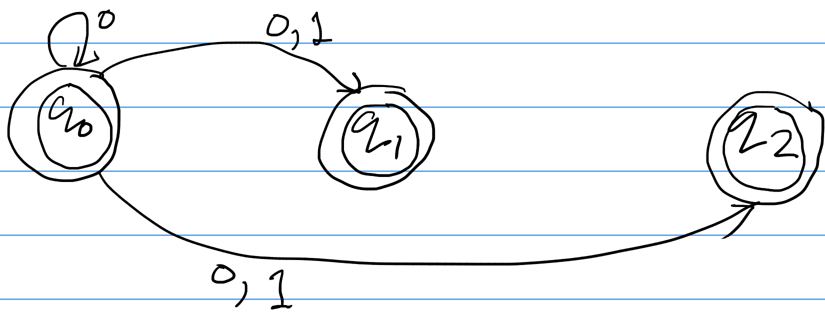
ϵ-NFA $\longleftrightarrow$ NFA

$\forall q_{\in Q} \forall a \in \Sigma$ Calculate $\hat{\delta}(q,a)$

in New NFA add transitions accordingly

$$\hat{\delta}(q_0,0) = \{q_0, q_1, q_2\}$$

$$\hat{\delta}(q_0,1) = \{q_1,q_2\}$$



NFA

Final state:- $\forall q$ $\epsilon\text{-Closure}(q) \cap F \neq \phi$

ϵ — NFA $\Leftrightarrow$ NFA $\Leftrightarrow$ DFA

Regular Expression :-

$L(a^*) =$ regular expression a^* $\{\epsilon, a, aa, aaa, \ldots\}$

$L(a^+) = a^+ = aa^* = \{a, aa, aaa, \ldots\}$

$L(\{00, 11\}^*) = \{00, 11\}^* = \{\epsilon, 00, 11, 000011, 001111,$
$1100, 110000, 111100,$
$001100, 0011110011111, \ldots\}$

$\cancel{\epsilon}$

00111

$L(\{0,1\}^*) = \{0,1\}^* = \{\epsilon, 0, 1, 001, \ldots\}$

$L_1, \quad L_2$

$L_1 L_2 = \{xy \mid x \in L_1, y \in L_2\}$

$L^i = L_1 L^{i-1}$ concatenation

$L_1^* = \bigcup_{i=0}^{\infty} L^i$

$(0+1)$ = Either 0 or 1

$(0+1)^* = \{0,1\}^*$

ϕ r.e denotes empty set

ϵ r.e $\{\epsilon\}$

a r.e $\{a\}$

$a \in \Sigma$

$*$, finite concatenation, $+$

r, s r.e. represent $L(r), L(s)$

$r+s$ " $L(r) \cup L(s)$

rs r.e. $L(r)L(s)$

r^* r.e $(L(r))^*$

r.e. $\longleftrightarrow$ DFA

r.e on $\{0,1\}$ such no two zeros are consecutive and string starts with 1

or

$11^*((01)^*1^*)^*$

$\{1, 10\}^+$ 101

r.e. $\longrightarrow$ ϵ-NFA

$*, >$ concatenation, $> +$ precedence relation b/w operators

r.e.: R

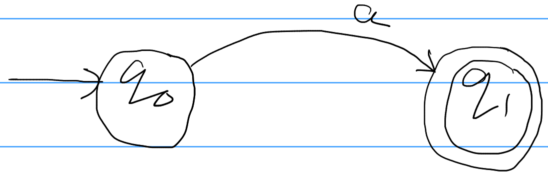
$$0^*1 + 0(00)^* + 1(01)^*$$

Base Case: (no operator)

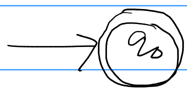
ϕ ϵ a

a

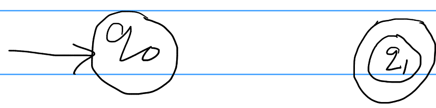
$\phi = \{ \}$



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



ϵ



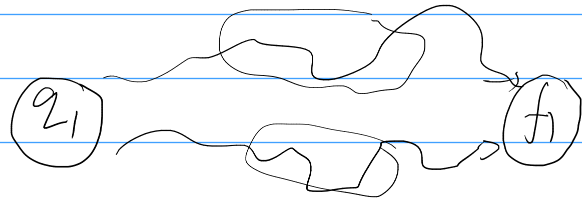
Assume that the theorem is true for r.e. with fewer than i numbers of operators.

r.e. $r = r_1 + r_2$

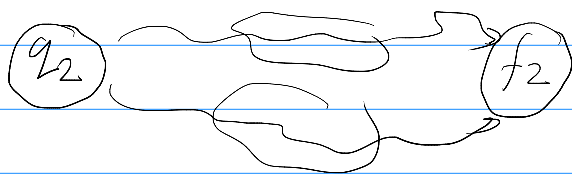
$r_1 r_2$

$(r_1)^*$

r_1

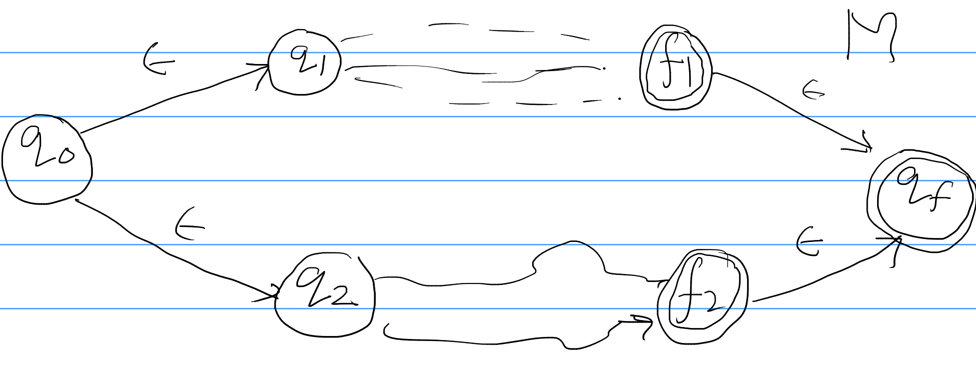


r_2



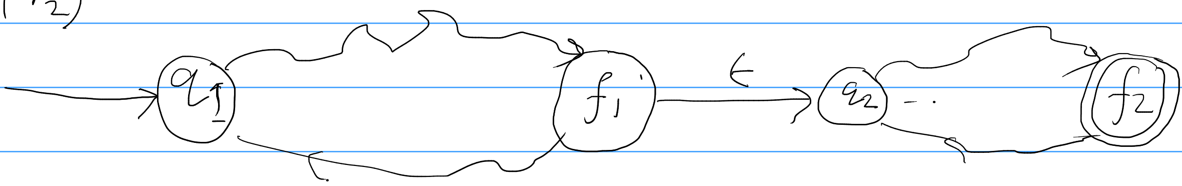
$L(r_1 + r_2)$

$x \in L(r_1 + r_2)$



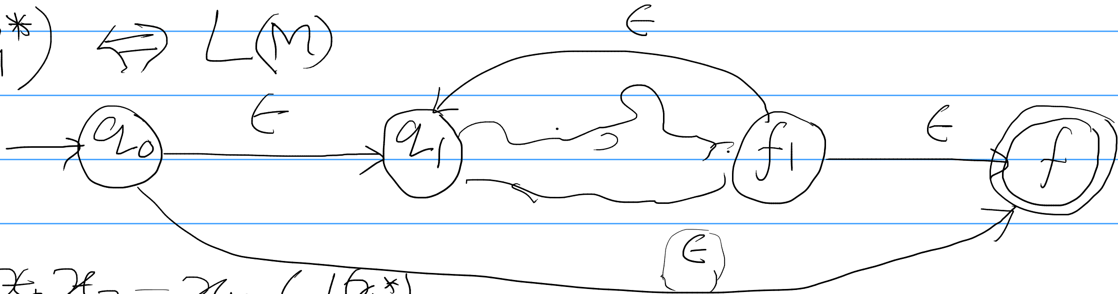
$x \in L(M)$

$L(r_1 r_2)$



$L(r_1^*) \Leftrightarrow L(M)$

r_1^*

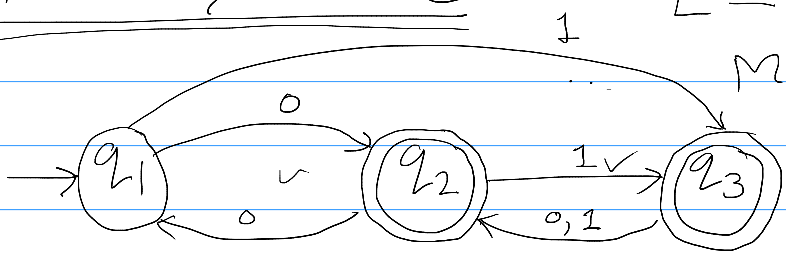


$x_1 x_2 x_3 - x_k \in L(r_1^*)$

$x_i \in L(r_1)$ $\quad 1 \le i \le k$

DFA $\longrightarrow$ r.e.

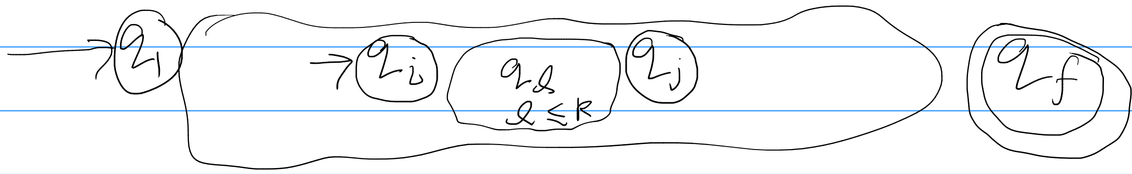
$L = L(M)$

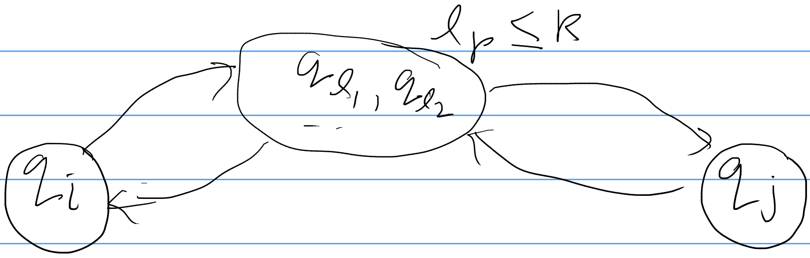


$0(00)^*$

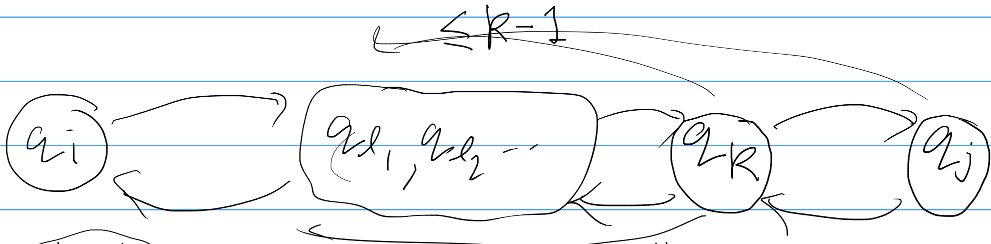
Recursively :

R_{ij}^{k} : denotes all string that takes q_i to q_j without visiting states q_l such that $l > k$.





$R_{ij}^{k} =$



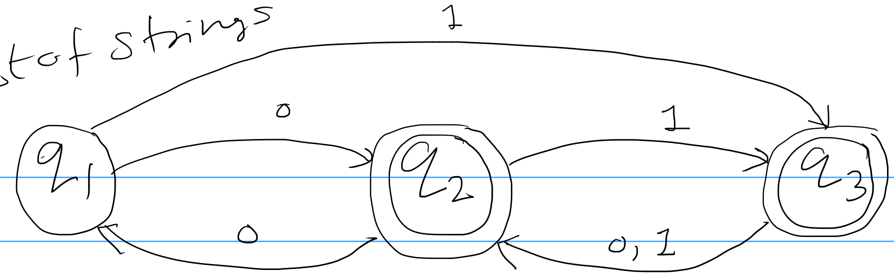
$$R_{ij}^{k} = \underline{R_{ij}^{k-1}} \cup \underbrace{R_{ik}^{k-1}\left(R_{kk}^{k-1}\right)^{*} R_{kj}^{k-1}}_{\text{using } q_k}$$

Base Case

$$R_{ij}^{0} = \{a : \delta(q_i, a) = q_j\} \quad i \neq j$$

$$\{a \cup \{\epsilon\} \ \delta(q_i, a) = q_j\} \quad i = j$$

R - Set of strings



$$r_{12}^3 + r_{13}^3 = ?$$

$$r_{12}^3 = r_{13}^2 (r_{33}^2)^* r_{32}^2 + r_{12}^2$$

$$r_{13}^2, \quad r_{33}^2, \quad r_{32}^2, \quad r_{12}^2$$

$$r_{13}^2 = r_{12}^1 (r_{22}^1)^* r_{23}^1 + r_{13}^1$$

$$r_{11}^2 = (00)^*$$

	$k=0$	$k=1$	$k=2$
r_{11}^k	ϵ	ϵ	$(00)^*$
r_{12}^k	0		
r_{13}^k	1		
r_{21}^k	0		
r_{22}^k	ϵ		
r_{23}^k	1		

$$r_{11}^1 = \epsilon$$

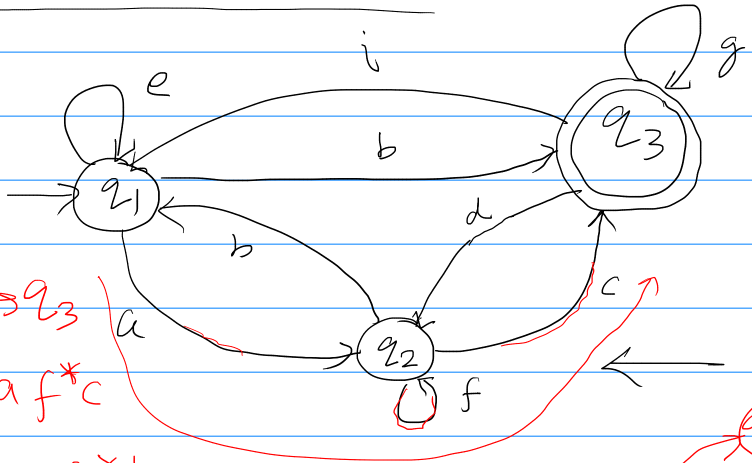
$$r_{22}^1 = r_{21}^0 (r_{11}^0)^* r_{12} + r_{22}^0$$

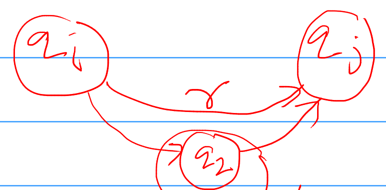
$=$

$$r_{11}^2 = r_{12}^1 (r_{22}^1)^* r_{21} + r_{11}^1$$

$=$

Alternate Method





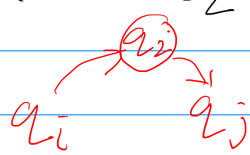
delete

$q_1 \to q_2 \to q_3$

remove q_2

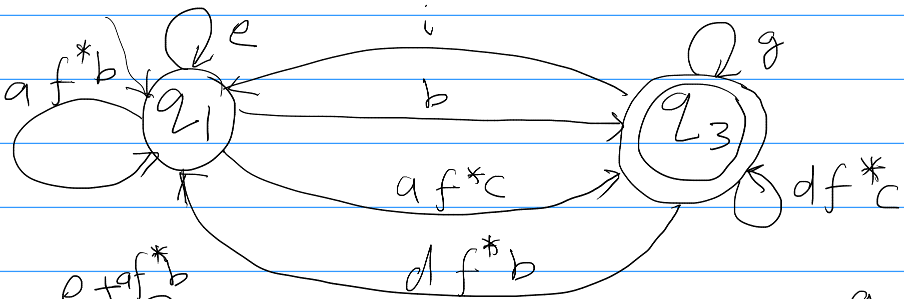
(q_1, q_3) af^*c

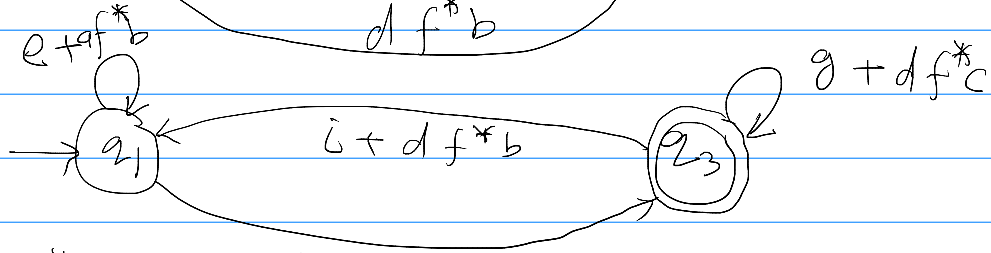
(q_1, q_1) af^*b.



(q_3, q_1) df^*b

(q_3, q_3) df^*c





$$(e+af^*b)^*(b+af^*c)(g+df^*c)^* = X$$

$b+af^*c$

$$(X(i+df^*b)X)^* + X$$

FA with o/p

Mealy M/C — o/p will be associate with transition

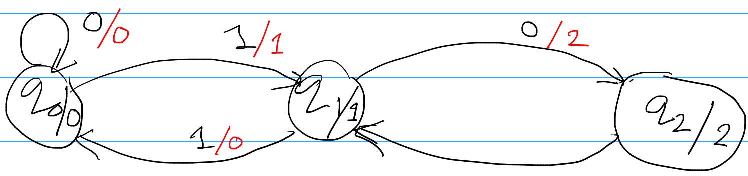
Moore M/C — o/p will be associated with states

Moore M/C

$d(x) \bmod 3 = 0$, $x \in \{0,1\}^*$

i/p: 0110

o/p: 00100



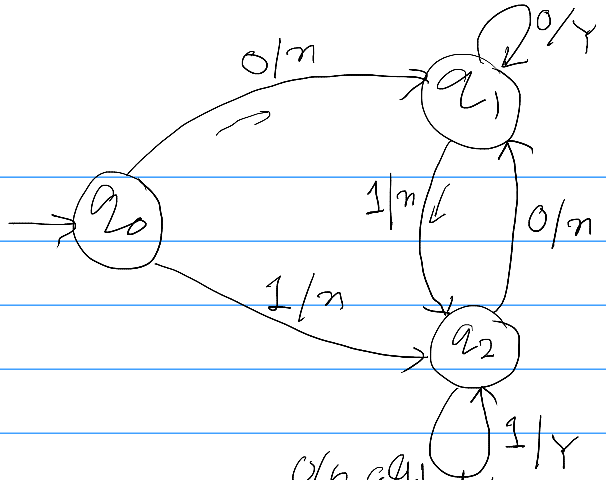
0 1 0 2 — — 0

Mealy M/C

$x \in \{0,1\}^*$ ends with 00 or 11

i/p: 0 1 1 0

o/p: n n Y n



o/p alphabet   o/p function

$$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$$

$$\begin{cases} \lambda : Q \longrightarrow \Delta & \text{Moore} \\ \lambda : Q \times \Sigma \longrightarrow \Delta & \text{Mealy.} \end{cases}$$

<u>New M/C</u>

$Q \times \Delta$

$M = (Q, \Delta, \Sigma, \delta, \lambda, -)$ <u>Mealy</u>

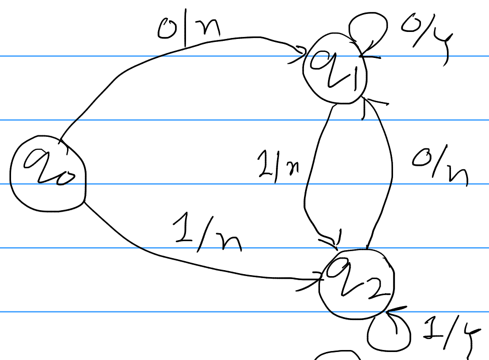
$\Delta = \{y, n\}$

$[q, Y]$ $[q, n]$ $\forall q \in Q$

$$\delta([q, b], \underset{}{\overset{\in\Sigma}{a}}) = [\delta'(q,a), \lambda(q,a)]$$

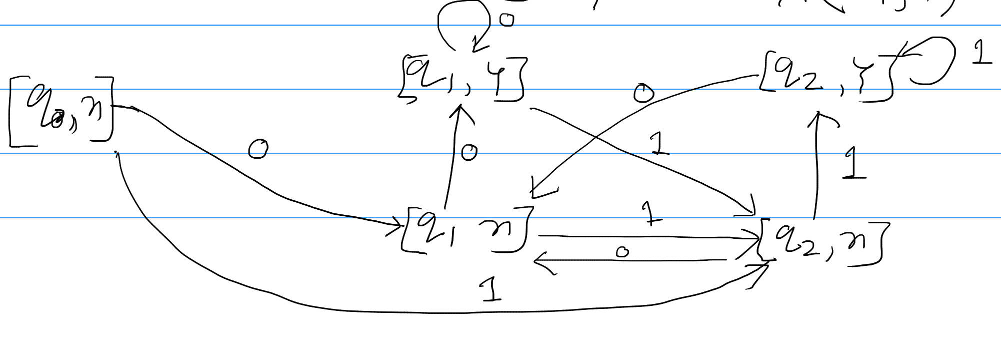
transition function for new M/C   old M/C   old

<u>Moore</u>



$\delta'(q_1, 1) = q_2$

$\lambda(q_1, 1) = n$



Pumping Lemma

DFA M finite number of states. n

$x \in \Sigma^*$ $|x| > n$

i/p $x = u\,v\,w$

$$\overbrace{q_0 \quad q_1 \quad q_2 \cdots}^{u} \underbrace{\overbrace{q_j \quad q_{j+1} \quad q_{j+2} \cdots q_k}^{v}}_{} \overbrace{q_{k+1} \cdots q_{n-1}}^{w}$$

$|uv| \leq n$ Looping.

$q_0 \quad q_1 \quad q_2 \qquad q_j \quad q_{k+1} \cdots q_{n-1}$

uw accepted if x has got accepted.

$v^i = vv \cdots$ i times

uv^iw will also be accepted.

Let L be a regular Language. Then the following property holds for L.

$\exists\, k \geq 0$ such that for any $z \in L$ and $|z| \geq k$ $\exists\, (u, v, w)$ such that

$z = uvw$ $|v| \geq 1$, $|uv| \leq n$ and for all $i \geq 0$ the string $uv^iw \in L$.

necessary condition not sufficient

$p \to q$ Contrapositive $\bar{q} \to \bar{p}$

quantifiers $\forall, \exists$ are alternating.

$$\left\{\begin{array}{l} \underline{\forall} \, k \geq 0 \quad \underline{\exists} \, z \in L \quad |z| \geq k \quad \text{and} \\ \underline{\forall} \, (u, v, w) \text{ with } z = uvw \text{ and } |v| \geq 1, \; |uv| \leq n \\ \underline{\exists} \, i \text{ such that } u v^i w \notin L \end{array}\right\}$$

$\Longrightarrow \{L \text{ is not regular}\}$

Combinatorial Game

$\forall$ player 1

$\exists$ player 2

Player 1 ($\forall$) | player 2 ($\exists$)

Pick $k \geq 0$

$|z| \geq k, \; z \in L$

$z = uvw \quad |v| \geq 1$
$|uv| \leq n$

$u v^i w \quad i \geq 0$

if $u v^i w \notin L$
player 2 will win
otherwise player 1 will win.

$L = \{ a^{k^2} \mid k \geq 1 \}$ Not regular.

$\subseteq a^*$ is regular

choose $n > 0$

$$z \in L \quad |z| = n^2$$

$$z = a^{n^2}$$

$$z = uvw = \underline{a}^{n^2}$$

$1 \le |v|$

$|uv| \le n$

$$u = a^i \quad v = a^j$$

$$w = a^{n^2-i-j}$$

$j \ge 1$

$i + j \le n$

$$uv^2w = a^i a^j a^j a^{n^2-i-j}$$

$$= a^{\boxed{n^2+j}} \in L \quad \text{Not}$$

$n^2 + j = m^2$ possible for all j

Hence L is not regular.

integer number corresponding to $1x$

$$L = \{ 1x \mid x \in \{0,1\}^* \quad d(1x) \text{ is prime} \}$$

Not regular.

{ infinitely many primes.
Fermats Little theorem

$a^p \equiv a \bmod p$ when p is prime

finitely many primes

$$p_1, p_2, - - p_n$$

$$M = p_1 p_2 \cdots p_n + 1$$

M is not divisible by any of the above prime numbers.
means we have more prime numbers

Fermats' little theorem

Lemma: $\binom{p}{r}$ is divisible by p prime

$$\binom{p}{r} = \frac{p(p-1)\cdots(p-r+1)}{r(r-1)\cdots 1}$$ when p is a prime $r<p$

$r(r-1)\cdots 1$ can not divide p

Since $\binom{p}{r}$ is an integer $p \mid \binom{p}{r}$

Thm $a^p - a$ is divisible by p. (prime)

Induction: $a = 1$ true.

$a^p - a$ is divisible by p.

$(a+1)^p - (a+1)$ is divisible by p

$a-b$ is divisible by $p \Rightarrow$

$a^k - b^k$ " " " p

$k=1$ $a-b$ is divisible by p

$k=m$ $a^m - b^m$ " " " p

Prove for

$k = m+1$ $a^{m+1} - b^{m+1}$ divisible by p.

$$a^m - b^m = pt$$

$$a^{m+1} - ab^m = apt$$

$$a-b = pl$$
$$a = pl + b$$

$$a^{m+1} - p\ell b^m - b^{m+1} = apt$$

$$a^{m+1} - b^{m+1} = (at + \ell b^m)\, p$$

Hence proved

$a - b$ divisible by p

$a \equiv b \bmod p$ change of notation

Congruence

$\Downarrow$

$a^k \equiv b^k \bmod p$

binary numbers

x $\quad$ $d(x)$

yx $\quad$ $d(yx) = 2^{|x|} d(y) + d(x)$

yyx $\quad$ $d(yyx) = 2^{|yx|} d(y) + d(yx)$

$$= 2^{|y|+|x|} d(y) + \overbrace{2^{|x|} d(y) + d(x)}$$

$$= d(y)\, 2^{|x|} (2^{|y|} + 1) + d(x)$$

$$d(yyyx) = d(y)\, 2^{|x|} (2^{2|y|} + 2^{|y|} + 1) + d(x)$$

$$d(zyyyx) = d(z)\, 2^{|x| + 3|y|} + d(y)\, 2^{|x|} (2^{2|y|} + 2^{|y|} + 1) + d(x)$$

$$L = \{ 1x \mid x \in \{0,1\}^* \; d(1x) \text{ is prime} \}$$

Suppose L is regular.

Pick $n \geq 0$, $z \in L$ $|z| \geq n$

$z = uvw$ $\quad$ $d(z)$ as prime

uv^iw $\forall i \geq 0$ $\quad$ $d(uv^iw)$ is prime

$\Rightarrow uv^iw \in L$

Not the case for all i

$$z = u v w = 1x \in L \quad x \in \{0,1\}^*$$

$$d(z) = d(uvw)$$

$$p = 2^{|v|+|w|} d(u) + 2^{|w|} d(v) + d(w) \text{ is prime}$$

$$i = p$$

claim $d(uv^p w)$ is not prime

$$d(uv^p w) = \{ d(u) 2^{|w|+p|v|} + 2^{|w|} d(v) \left(1 + 2^{|v|} + 2^{2|v|} + \cdots 2^{(p-1)|v|}\right) + d(w) \}$$

$p \neq 2$

$$2^p \equiv 2 \mod p \qquad \frac{2(2^{p-1}-1)}{p}$$

$$2^{(p-1)|v|} \equiv 1 \mod p \Leftarrow 2^{(p-1)} \equiv 1 \mod p \qquad p \neq 2$$

$$S = 1 + 2^{|v|} + 2^{2|v|} + \cdots 2^{(p-1)|v|} = \frac{2^{p|v|}-1}{2^{|v|}-1}$$

$$(2^{|v|}-1) S = 2^{p|v|} - 1 \leftarrow$$

$$2^{(p-1)|v|} \equiv 1 \mod p$$

$$a \equiv b \mod p \quad \Downarrow$$

$$\Downarrow$$

$$ac \equiv bc \mod p \quad 2^{(p-1)|v|} \times 2^{|v|} \equiv 1 \times 2^{|v|} \mod p$$

$$2^{p|v|} \equiv 2^{|v|} \mod p$$

$$2^{p|v|} - 1 \equiv 2^{|v|} - 1 \mod p$$

multiply with S

$$S(2^{p|v|}-1) \equiv S(2^{|v|}-1) \mod p$$

$$S(2^{p|v|}-1) \equiv (2^{|v|p}-1) \mod p$$

$$(S-1)(2^{p|v|}-1) \equiv 0 \mod p$$

Claim! $\boxed{2^{p|v|}-1 \bmod p \not\equiv 0}$

$$2^{p|v|} \equiv 2^{|v|} \bmod p$$

$d(uvw) = p$

$$2^{p|v|}-1 \equiv 2^{|v|}-1 \bmod p$$

$|v| < |z| = |uvw|$

$\Rightarrow$ $2^{p|v|}-1$ not divisible by p

$\forall v$ $\because d(v) < d(uvw) = p$

$2^{|v|}-1 < d(uvw) = p$

Hence $S-1$ is divisible by p.

$$\Rightarrow S \equiv 1 \bmod p$$

$$\Rightarrow d(uv^pw) = d(u)\, 2^{|w|+p|v|} + d(v)\, 2^{|w|} S + d(w)$$

$$\equiv d(u)\, 2^{|w|+p|v|} + d(v)\, 2^{|w|} + d(w) \bmod p$$

$$2^{p|v|} \equiv 2^{|v|} \bmod p$$

$$\equiv d(u)\, 2^{|w|+|v|} + \underbrace{d(v)}_{\overset{\shortparallel}{p}} 2^{|w|} + d(w) \bmod p$$

$$\equiv 0 \bmod p$$

$$S \equiv 1 \bmod p$$

$$d(u) 2^{|w|+p|v|} + d(w) + d(v) 2^{|w|} S \equiv d(v) 2^{|w|} + d(w) + d(u) 2^{|w|+p|v|} \bmod p$$

$$\underline{d(uv^pw)} \equiv \underline{d(v) 2^{|w|} + d(w) + d(u) 2^{|w|+p|v|}} \bmod p$$

$$2^{|v|p} \equiv 2^{|v|} \bmod p$$

$$\equiv d(v) 2^{|w|} + d(w) + d(u) 2^{|w|+|v|} \pmod p$$

$$\equiv \underline{d(uvw)} \bmod p$$

$$\equiv p \bmod p$$

$$d(uv^pw) \equiv 0 \bmod p$$

Hence L is not regular.

$2^p \equiv 2 \mod p$ **FLT**

$2^{p|v|} \equiv 2^{|v|} \mod p$ (hold)

$2^{p|v|} - 1 \equiv 2^{|v|} - 1 \mod p$ (hold)

$d(uvw) = p$ prime

$2^{|v|} < p$

Closure Properties of regular languages

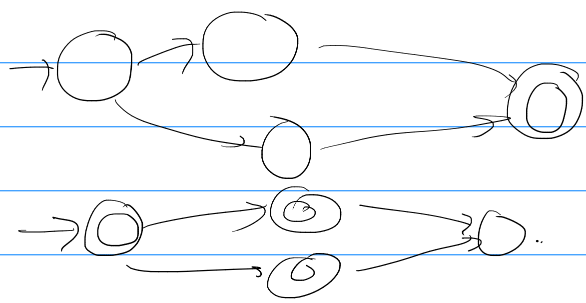
L_1, L_2 are regular $\Rightarrow$

$L_1 \cup L_2,\ L_1 L_2,\ L_1^*,\ L_1 \cap L_2,\ \overline{L_1}$

$L_1^R = \{w^R \mid w \in L_1\}$ are regular.

$\overline{L_1}$: Let M be a DFA which accepts L_1.

$\left\{\begin{array}{l}\text{all non final state make it final} \\ \text{all final " " " non final}\end{array}\right.$



L

$\overline{L}$

$L_1 \cap L_2$: $\quad L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ regular

Non Construction Proof

$L_1 = L(M_1) \qquad M_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$

$L_2 = L(M_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$

$$M = (Q_1 \times Q_2, \ \Sigma, \ \delta, \ q_{01} \times q_{02}, \ F)$$

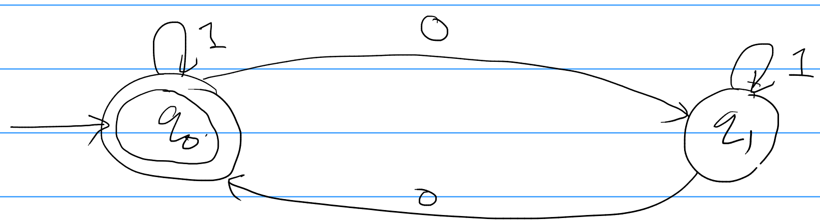
$F = F_1 \times F_2 \qquad (\overset{\in Q}{q_i}, \overset{\in F_1}{q_f}) \notin F$

$$\delta((q_i, q_j), a) = (\delta_1(q_i, a), \delta_2(q_j, a))$$

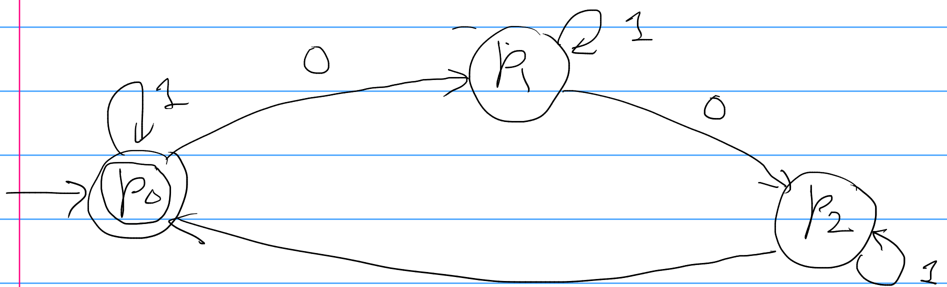
$L_1 = \{ x \in \{0,1\}^* \mid$ x contains even number of zeros $\}$

$L_2 = \{ x \in \{0,1\}^* \mid$ number of zeros in x mod 3 is 0 $\}$

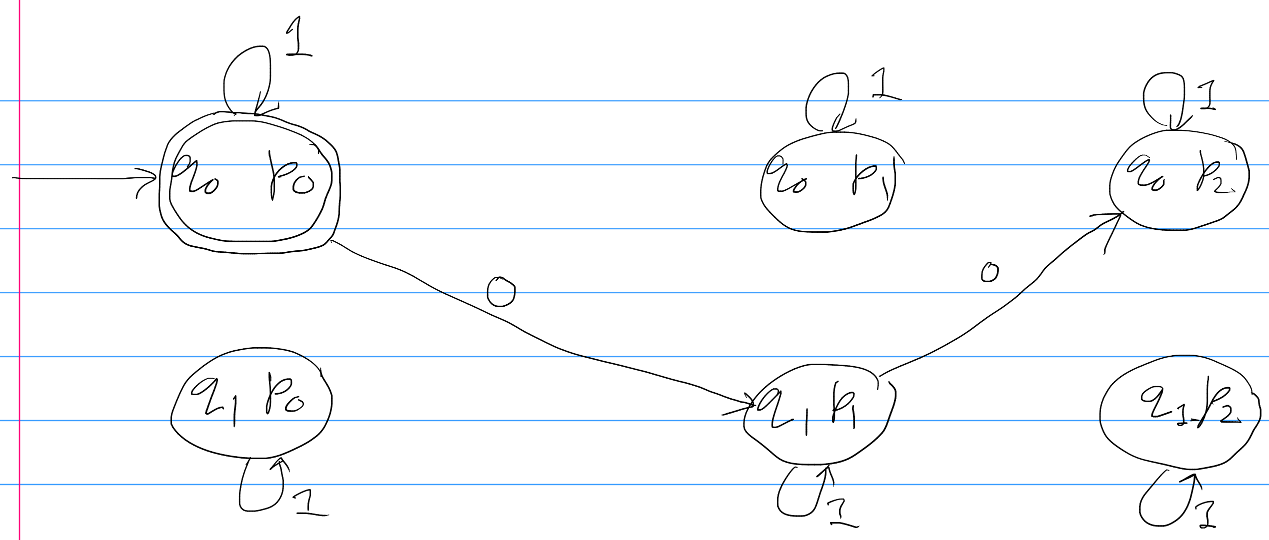
$L_1 \cap L_2 = \{ x \in \{0,1\}^* \mid$ number of zeros in x is divisible by 2 and 3 $\}$



L_1



L_2



Let L be a regular language.

$L^R = \{ \omega^R \mid \omega \in L \}$

$\omega = a_1 a_2 \cdots a_n \in L$

$\omega^R = a_n a_{n-1} \cdots a_2 a_1 \in L^R$

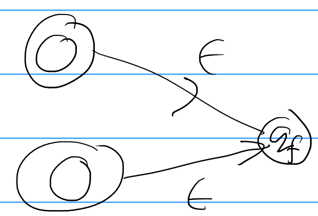
Claim: L^R is regular.

DFA M for L

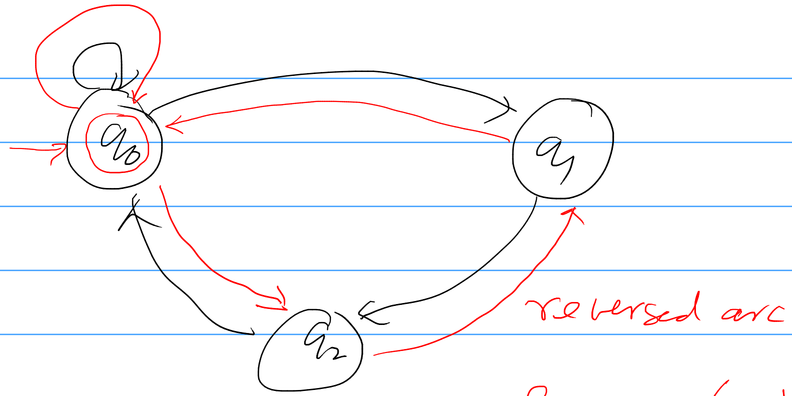
M' : 1. if M contains more than one final states then

(a) add a new final state q_f

(b) add ϵ transition from the all the previous final states to q_f



2: Reverse all the arc present in the above obtained DFA.



$w \in L(M) \iff w^R \in L(M')$

Homomorphism

$0 \to abc$

$1 \to ac$

011

$abcacac$

Σ, Γ alphabets

$$h : \Sigma \longrightarrow \Gamma^*$$

$$w = a_1 a_2 \cdots a_n \in \Sigma^*$$

$$h(w) = h(a_1)\, h(a_2) \cdots h(a_n) \in \Gamma^*$$

$h(0) = abc$ $\quad \Sigma = \{0,1\}$

$h(1) = ac$ $\quad \Gamma = \{a,b,c\}$

$w = 011$ $\quad h(w) = abcacac$

$h(L) = \{ h(w) \mid w \in L \}$ $\quad \Gamma = \{a, b, c,$

$\Gamma^* = \{ab, cac, \dots\}$

$h(0) = cbc$

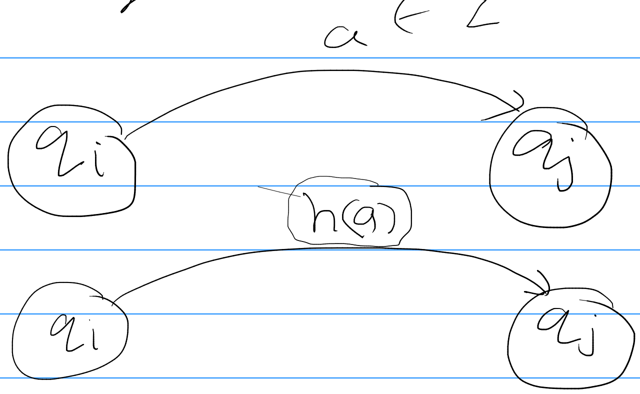
$h(1) = ab$

$w = 01001$ $\quad$ accepted

$h(w) = cbc\, ab\, cbc\, cbc\, ab$

$h(w) \to w$

if L is regular then h(L) is also regular.



transition graph

generalized transition graph

$\forall\, i, j$ and a in the original M/C.

$$x \in L(M) = L \iff h(x) \in L(M')$$

Let L_1 and L_2 be languages on same alphabet

$$h(x) = \omega$$

$$\omega \in L(M') \Rightarrow \exists\, x \in L(M)$$

$$\forall\, x \in L(M) \Rightarrow h(x) \in L(M')$$

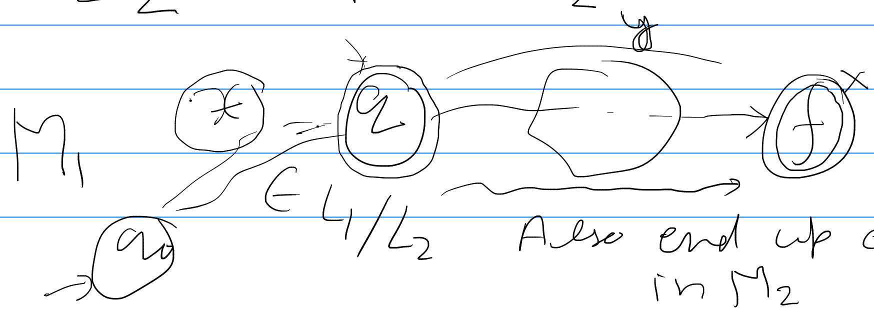
L_1, L_2 regular

right quotient

Claim: $L_1/L_2 = \{x : xy \in L_1 \text{ for some } y \in L_2\}$ regular.

L_1 DFA M_1

L_2 DFA M_2



Also end up at final state in M_2

M_1 $\quad q_0$

$$\left\{\begin{array}{l} M_i' = (Q, \delta, \Sigma, \{q_i\}, F) \\ L(M_i') \cap L(M_2) \text{ is not empty} \\ \text{then make } \underline{q_i} \text{ as final state} \end{array}\right.$$

$L_1 = \{ \underline{a^n b^m} : n \geq 1, m \geq 0 \} \cup \underline{\{ba\}}$ regular

$L_2 = \{ \underline{b}^m : m \geq 1 \}$ $\quad m \geq 0$ $\quad$ regular

$L_2 = \{ b, b^2, b^3, \dots \}$

$L_1/L_2 = \{ a^n b^m : n \geq 1, m \geq 0 \}$ $\quad \epsilon \in L_2$

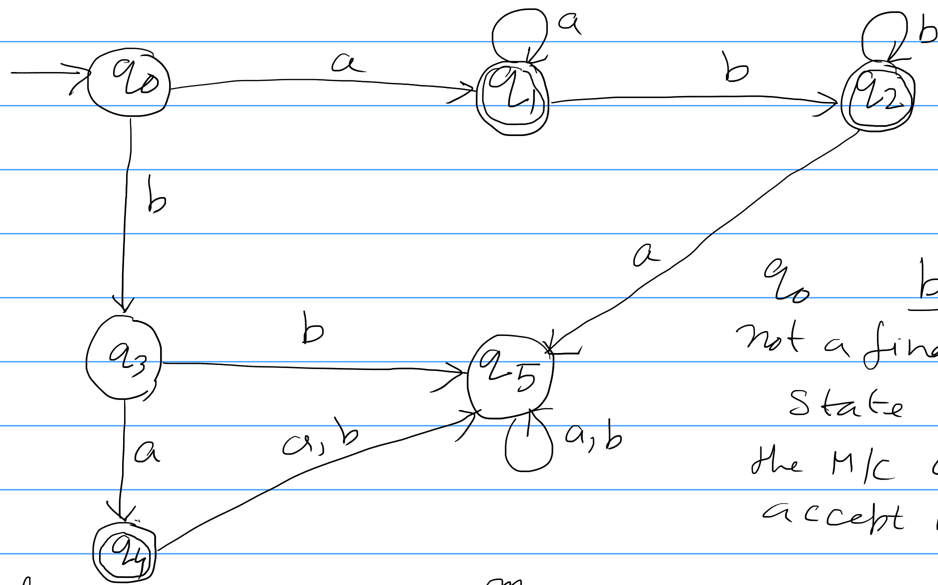
$L_1/L_2 = \{ x \mid \exists y \in L_2 \text{ such that } xy \in L_1 \}$

$ba \in L_1$ but there exist no string in L_2 which ends with a.

$x = a^n \in L_1/L_2$ $\quad i \geq 1$

$y = b^i \in L_2$ $\quad xy = a^n b^i \in L_1$

$a^n b \in L_1/L_2$



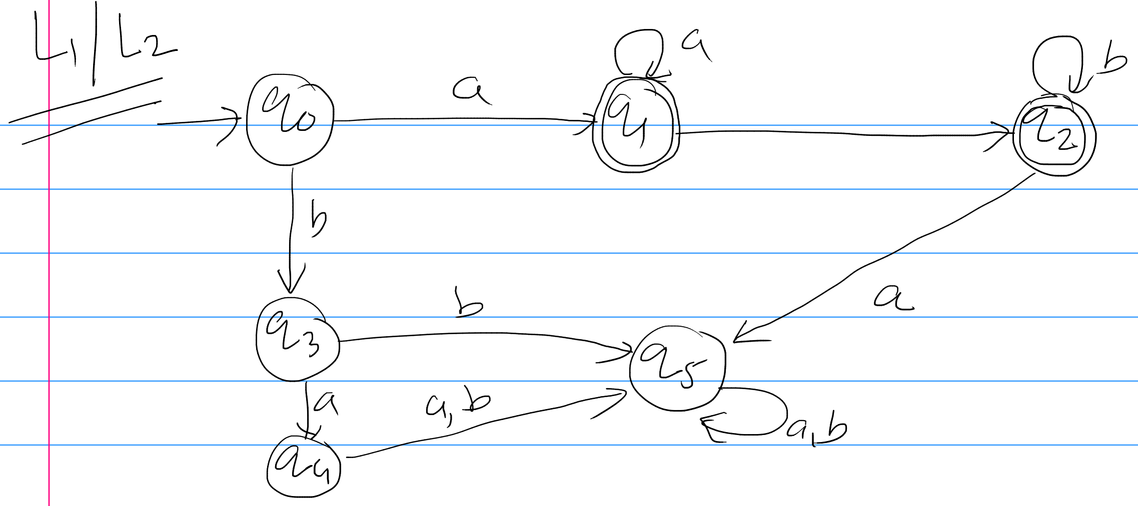
M_1

Check from each state whether we can reach final state by scanning $\underline{b}^m$. $m \geq 1$

q_0 $\quad \underline{b b}^m$ $m \geq 0$

not a final State in the M/C which accept L_1/L_2

L_1 / L_2



Equivalence Relation

S

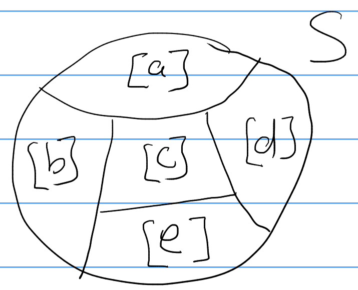
Relation $R \subseteq S \times S$ is an equivalence relation if the following conditions hold

(a) $a R a \quad \forall a \in S$

(b) $a R b \Rightarrow b R a \quad \forall a, b \in R$

(c) $a R b \wedge b R c \Rightarrow a R c \quad \forall a, b, c \in R.$

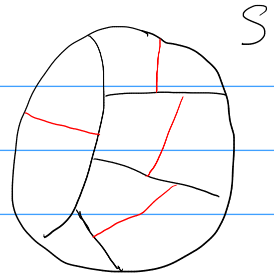
If R is an equivalence relation then it partition the set S in to equivalence classes.



$$[a] = \{ x \mid a R x \}$$

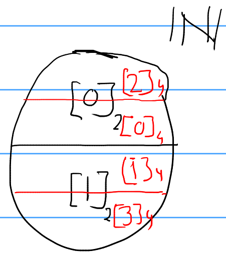
Refinement of equivalence relation R.

An equivalent relation R' is a refinement of R if every equivalent class in R' is completely contained in a equivalent class in R



R' partition

R partition.



R mod 2

R' mod 4

refinement of R.

$$x R' y \Rightarrow x R y$$

$\Rightarrow$ R' is refinement of R.

An equivalent relation R has finite index if number of equivalence classes in R is finite.

Language $L \subseteq \Sigma^*$

Equivalence relation

$x R_L y$ if and only if $\forall z \in \Sigma^*$

either xz and yz both belong to L

or neither xz nor yz belong to L.

x, y are not related in R_L

$x \not R_L y \Leftrightarrow \exists z$ such that $xz \in L$ and $yz \notin L$

or $xz \notin L$ and $yz \in L$.

it also implies that x, and y are in two different equivalent class

DFA $M = (Q, \Sigma, \delta, q_0, F)$

$x, y \in \Sigma^*$ $\quad x R_M y \iff \delta(q_0, x) = \delta(q_0, y) = q \in Q$

Claim: Equivalence relation –

$[x]$ — state $\quad$ $\boxed{L = \text{Union some equivalent classes.}}$

$y \in [x] \quad x R_M y \iff \delta(q_0, x) = \delta(q_0, y)$

$\boxed{\text{right invariant} \quad R \subseteq \Sigma^* \times \Sigma^* \quad \text{if } x R y \text{ then } xz R yz \ \forall z \in \Sigma^*}$

R_M is right invariant:–

$$x R_M y \iff \delta(q_0, x) = \delta(q_0, y) = q$$

$$\forall z \in \Sigma^* \quad xz R_M yz \iff \delta(q_0, xz) = \delta(q_0, yz)$$

$$\delta(\delta(q_0, x), z) = \delta(\delta(q_0, y), z)$$

$$\delta(q, z) = \delta(q, z)$$

Myhill Nerode Theorem:– The following 3 statements are equivalent.

1. $L \subseteq \Sigma^*$ is accepted by some FA
2. L is the union of some of the equivalence classes of a right invariant equivalence relation of finite index.

3. Equivalence relation R_L be defined by: $x R_L y \iff \forall z \in \Sigma^*$ $xz \in L$ exactly when $yz \in L$. Then R_L is of finite index.

Proof: $(1) \Rightarrow (2)$

L $\quad M = (Q, \Sigma, \delta, q_0, F) \leftarrow$

$$x R_M y \iff \delta(q_0, x) = \delta(q_0, y)$$

R_M is right invariant.

R_M has finite index.

L = union of equivalence classes which are associated with final states in M.

$(2) \Rightarrow (3)$

R' for statement 2

We will prove that R' is a refinement of R_L.

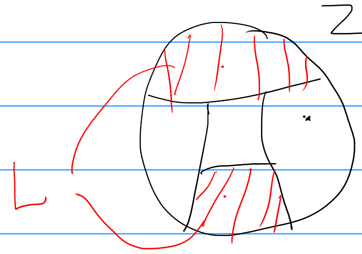
$$x R' y \Rightarrow \forall z \in \Sigma^* \; xz R' yz$$

because R' is right invariant.

$$xz \in L \iff yz \in L$$

$\Uparrow$ definition of R_L

Σ^*



$x R_L y$

hence R' Refines R_L

Since R' has finite index.

R_L finite index.

(3) $\Rightarrow$ (1)

$x R_L y \Leftrightarrow \forall z \in \Sigma^* \quad xz \in L$
exactly when $yz \in L$.

R_L has finite index.

<u>Claim:</u> R_L is right invariant.

$\underline{x R_L y} \Rightarrow \forall z' \in \Sigma^* \quad \underline{xz' R_L yz'}$

$\forall z' \in \Sigma^* \; xz' R_L yz'$

$x R_L y \Rightarrow \forall z \in \Sigma^* (xz \in L \Leftrightarrow yz \in L)$

$\Rightarrow \forall z'z'' \in \Sigma^* \; (xz'z'' \in L \Leftrightarrow yz'z'' \in L)$

$\Rightarrow \forall z'' \in \Sigma^* \; \underbrace{((xz')z'' \in L \Leftrightarrow (yz')z'' \in L)}$

$\Rightarrow xz' R_L yz'$

R_L is right invariant.

Construct <u>DFA</u> M which accepts L.

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

R_L has finitely many equivalence classes.

Each equivalence class will work as state for M.

$[x] = \{y : x R_L y\}$

$\underline{\delta([x], a) = [xa]}$ Well defined or <u>Consistent</u>

not well defined function

$\nexists \; \begin{cases} f(m/n) = m+n \\ f(3/2) = 5 \\ f(6/4) = 10 \end{cases}$

$y \in [x]$ $\quad \hat{\delta}(q_0, y) = \hat{\delta}(q_0, x)$

may arise in the previos definition

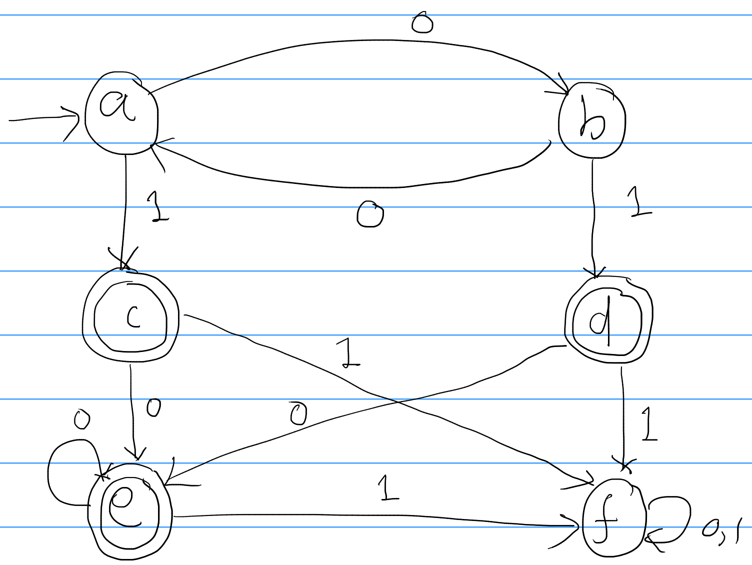
$y \in [x]$ $\quad \hat{\delta}(q_0, xa) \neq \hat{\delta}(q_0, ya)$

Since R_L is right invariant

$[xa] = [ya]$

$q_0 = [\epsilon]$ $\quad F = \{ [x] \mid x \in L \}$

$L = L(0^* 1 0^*)$



R_M

$[a] = (00)^*$

$[b] = (00)^* 0$

$[c] = (00)^* 1$

$[d] = (00)^* 01$

$[e] = (00)^* 100^* + (00)^* 0100^* = X = 0^* 100^*$

$[f] = (00)^* 011(0+1)^* + (00)^* 11(0+1)^* + X1(0+1)^* = 0^* 10^* 1(0+1)^*$

$L = [c] \cup [d] \cup [e]$

R_L

$\begin{cases} 1 : & \text{No one's in string} \\ 2 : & \text{Exactly one 1 in string} \\ 3 : & \geq 2, \text{ 1's in string.} \end{cases}$

Union of above 3 sets is $\{0,1\}^*$

(1) (2) R_L

$00 \in$ first set

$01 \in$ Second Set

$z = 1$

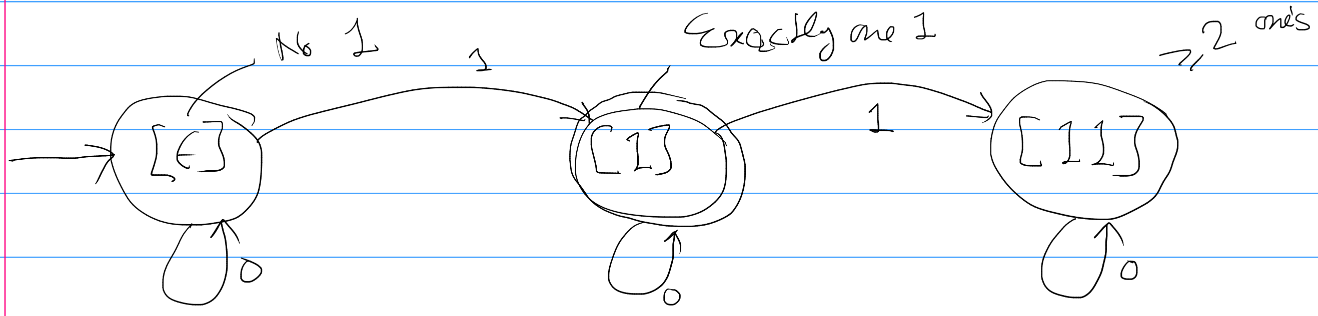
$001 \in L$

$011 \notin L$

$00 \in 1^{st}$ Set

$000 \in 1^{st}$ Set

$\forall z \in \{0,1\}^*$ $\underline{00z \; R_L \; 000z}$



Minimization of states in DFA.

States p, q

p is indistinguishable to q

xR_Ly $\forall z \in \Sigma^*$ $xz, yz \in L$ or $xz, yz \notin L$

if $\exists\ x \in \Sigma^*$ such that $\hat{\delta}(p, x) \in F$ and $\hat{\delta}(q, x) \notin F$ or vice versa.

Context Free Grammars

$G = (V, T, P, S)$

Set Production

Start symbol.

Set of terminals

Set of variables

$V = \{E\}$ $\quad T = \{+, *, (,), id\}$

$$P: \{ S \to E$$
$$E \to E+E$$
$$E \to E*E$$
$$E \to (E)$$
$$E \to id \}$$

G is context free if LHS of P contains exactly one variable and RHS contains some string from alphabet $(V \cup T)^*$

$$E \to E+E \mid E*E \mid (E) \mid id$$

Derivations $V = \{\}$, S.
$T = \{a, b\}$

$P: \quad S \to aSb \qquad S \to ab$

$$S \to aSb \to aaSbb \to aaabbb$$

aaabbb is a string generated from G.

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

$A \to \beta \in (V \cup T)^*$ Production

$\alpha A \gamma \to \alpha \beta \gamma$ one step derivation

$\alpha, \gamma \in (V \cup T)^*$

$\alpha A \gamma \xrightarrow{*} \alpha \beta \gamma$ in zero for more steps

$aSb \xrightarrow{*} a^4 S b^4$

$G, \quad L(G) = \{x \mid x \in T^*, S \xrightarrow{*} x\}$

Language generated by the grammar G

$L, \quad G$

L is generated by G if $L(G) = L$.

G_1 and G_2 are equivalent if $L(G_1) = L(G_2)$

G: $T = \{a, b\}$, S, $V = \{\ \}$

$P = \{S \rightarrow aSb \mid ab\}$

$\underline{L(G) = \{a^n b^n \mid n \geq 1\}}$

Proof: $\underline{a^{n-1} S b^{n-1}}$

$\underline{S = ab}$ $\underline{a^n b^n}$

1: $\boxed{S \rightarrow aSb}$ $\underline{\alpha \in (V \cup T)^*}$

2: $\boxed{S \rightarrow ab}$ $\underline{|x| + 2}$

Apply 1 production same number of times and then use 2^{nd} production if you want to stop $\underline{a^n b^n}$

G:

$V = \{S, A, B\}$ $T = \{a, b\}$

$P = \{S \rightarrow aB \mid bA,$

$A \rightarrow aS \mid bAA \mid a,$

$B \rightarrow bS \mid aBB \mid b\}$

Claim: $L(G) = \{x \in \{a,b\}^*$: number of a in x is same as number of b in x$\}$

right most

abbbAA

↓

abbbAa

↓ abbbAA → abbbaA

abbbaa left most

$S \rightarrow \underset{\smile}{aB} \rightarrow \underset{\smile}{abS} \rightarrow \underset{\smile}{abbA}$

$\rightarrow \underset{\smile}{abbbAA} \rightarrow \underset{\smile}{abbbaa}$

↓

abbbaa sentential form $\in (V \cup T)^*$

A derivation a production is applied to the left most variable of any sentential form then the derivation is said to be left most.

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

To Prove $S \xrightarrow{*} w \in \{a,b\}^*$ $\iff$ w has equal numbers of a's and b's.

use induction.

induction parameter Length of w : $|w|$

$A \xrightarrow{*} w$ $\iff$ w has exactly one more a than b

$B \xrightarrow{*} w$ $\iff$ w has exactly one more b than a.

$A \to bAA \to baa$

$A \to aS \to abA \xrightarrow{*} abbaa$

Base Case: $|w| = 1$

$A \to a$ $\quad$ $B \to b$

a $\quad$ b $\quad$ holds

Induction hypothesis: Assume the above statements are true $|w| \leq k-1$

Need to prove for $|w| = k$

$S \xrightarrow{*} w$

$S \to aB$ $\quad$ $S \to bA$

$S \to aB$ $\quad$ $w = a w_1$

$|w_1| = k-1$

By Induction hypothesis $B \xrightarrow{*} w_1$ number of b's in w_1 is exactly one more than number of a's in w_1

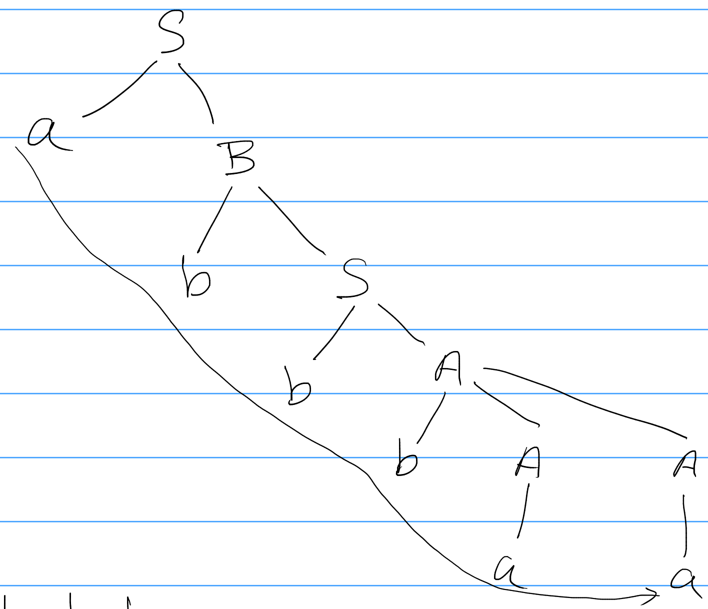
if $S \stackrel{*}{\Rightarrow} w$, then w contains equal number of a's and b's

if w contains equal number of a's and b's then
$$S \stackrel{*}{\Rightarrow} w.$$

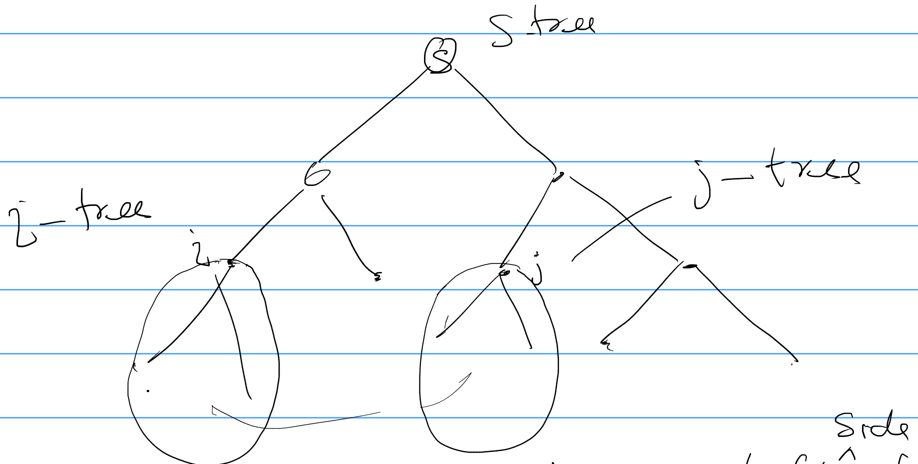
w.l.g. $w = a w_1$

$$B \stackrel{*}{\Rightarrow} w_1$$
$$S \stackrel{*}{\Rightarrow} aB.$$

<u>Derivation tree</u>



yield of above derivation tree: a b b b a a



→ yield of i tree will be on left side of yield of j tree in the yield of S tree

$G = (V, T, P, S)$

$w \in L(G)$ derivation tree for w.

$S \xrightarrow{*} \alpha \iff \exists$ a derivation tree in G with yield α.

Sentential form

$G = (V, T, P, S)$ CFG

Leaf node

Vertex has a label from $V \cup T \cup \{\epsilon\}$

root has label S.

if A is vertex which is interior then $A \in V$

if a vertex has label A and $X_1, X_2, \ldots X_k$ are sons of A in derivation tree then,

$$A \rightarrow X_1 X_2 \ldots X_k \in P$$

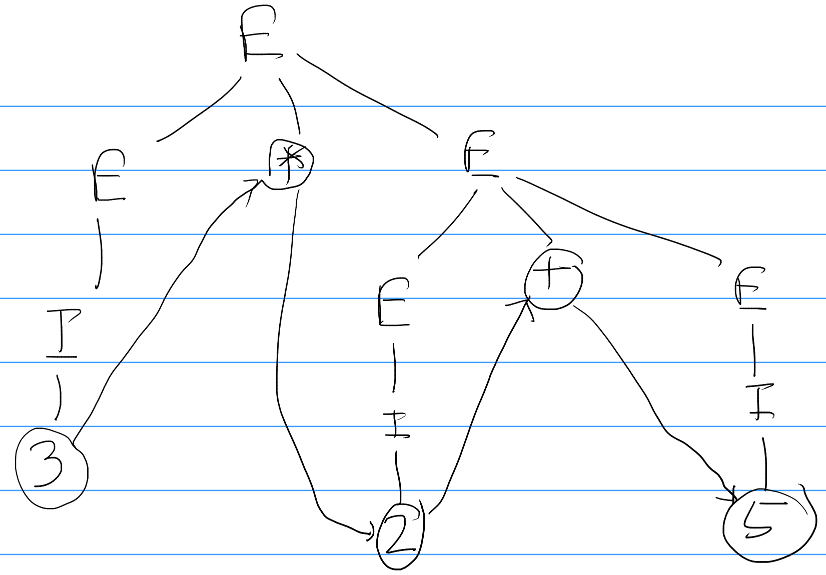
$$P: \begin{array}{ll} E \rightarrow I \mid E+E \mid E*E & V = \{E, I\} \\ I \rightarrow \epsilon \mid 0 \mid 1 \mid 2 \ldots \mid 9 & T = \{0, 1, 2, \ldots 9\} \end{array}$$

$3*2+5$

Derivation:

$$E \Rightarrow E*E \Rightarrow E*E+E$$
$$\Rightarrow I*E+E \Rightarrow I*I+E$$
$$\Rightarrow I*I+I \xRightarrow{*} 3\times 2+5$$

3*2+5



3*2+5

Left most derivation

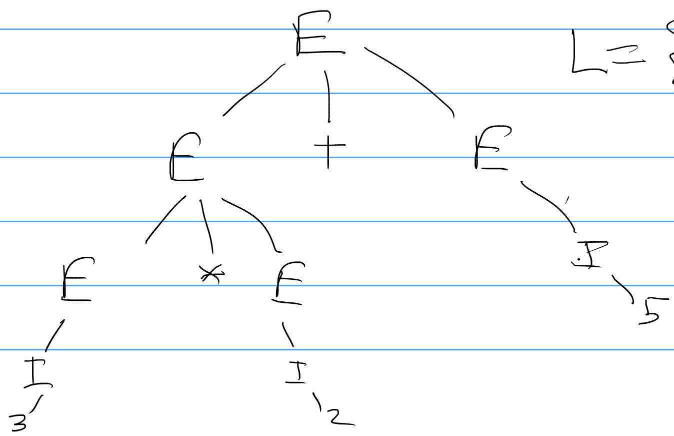
$$E \Rightarrow E*E \Rightarrow I*E \Rightarrow 3*E$$
$$\Rightarrow 3*E+E \overset{*}{\Rightarrow} 3*2+E \Rightarrow 3*2+5$$

Ambiguous Grammars

A grammar G is ambiguous if we have more than one parse tree for a string in L(G).

" " " more than one left most
or " " " right most
derivation.

$\boxed{3*2+5}$ = 11
= 21



$L = \{a^n b^m c^m d^n \mid n \geq 1, m \geq 1\}$
$\cup \{a^n b^m c^m d^n \mid n \geq 1, m \geq 1\}$

Simplification of CFG

1. Useless symbols.
2. Remove ϵ Productions
3. " Unit Productions

Useless Symbols:-

Usefull symbols: V'

$V \setminus V'$ set of useless symbols.

$T \setminus T'$

$$G = (V, T, P, S)$$

$A \in V$ $\quad \boxed{A} \xrightarrow{*} w \in T^*$ usefull.

$$P: \begin{cases} S \to AB \mid a \\ A \to a \end{cases}$$

$\boxed{B \xrightarrow{*} w \in L \subseteq \{a\}^*}$ use

$A \xrightarrow{*} a \in L \subseteq \{a\}^*$ usefu

$A \in V$ does not generate string the we call it none generating symbols.

$X \in V \cup T$

$S \xrightarrow{*} \alpha X \beta \qquad \alpha, \beta \in (V \cup T)^*$

Start symbol.

$\hookrightarrow$ use full.

Otherwise it is use less.

$A \to a\beta$

Use less symbol we name it non reacha-ble symbol.

$G = (V, T, P, S)$

$V' = \{A \mid A \to \omega \in T^*\}$

$V \ni A' \to X_1 X_2 \text{---} X_k$

$\underline{\forall i \quad X_i \xrightarrow{*} \omega \in T^*}$

$\Downarrow$

if $\forall i$ $X_i \in V'$ then $A' \in V'$

$V \setminus V'$ Set of useless symbols

non reachable symbols

reachable symbols $V' = \{S\}$

if $A \in V'$

$T' =$ and $A \to \alpha_1 \mid \alpha_2 \cdots \mid \alpha_n$

$\alpha_i \in (V \cup T)^*)$

then add all variable from α_i in V'

" " " terminals from α_i in T'.

$V \setminus V'$, $T \setminus T'$ set of non reachable symbol.

$S \to AB \mid a$

$A \to a$ $\quad L(G) = \{\omega : S \xrightarrow{*} \omega\}$

1. non generating symbols. $\quad X$ will be remo
2. non reachable "

$S \not\to \alpha X \beta$ $\quad S \overset{*}{\not\to} \alpha X \beta$

$X \xrightarrow{*} \omega$ $\quad \boxed{X \xrightarrow{*} \omega}$

usefull $\{S, A\}$
useless $\{B\}$

$$S \rightarrow a$$
$$A \rightarrow a$$
$$S \xrightarrow{*} A \quad \text{No}$$

$\{A\}$ useless : non reachable from S.

$S \rightarrow a$ Simplified grammar

$$S \rightarrow AB \mid a$$
$$A \rightarrow a$$

$S \rightarrow AB$

$$S \rightarrow a$$
$$\underline{A} \rightarrow a$$

$\}$ Not OK

Removal of ϵ Production

$$S \rightarrow ABaC$$
$$A \rightarrow BC$$
$$B \rightarrow b \mid \epsilon$$
$$C \rightarrow D \mid \epsilon$$
$$D \rightarrow d$$

$V = \{S, A, B, C, D\}$

$T = \{a, b, d\}$

Collect all variables X such that

$$X \xrightarrow{*} \epsilon$$

$\{A, B, C\}$ nullable variables

$B \rightarrow \epsilon$, $C \rightarrow \epsilon$

$A \rightarrow BC \rightarrow \epsilon$

$A \to \underline{X_1 X_2 \cdots X_n}$

$X_{i_1}, X_{i_2} \cdots X_{i_m}$, $m \le n$

$\{i_1, i_2 \cdots i_m\} \subseteq \{1, 2, \cdots n\}$ are nullable variables.

add productions

$A \to X_{j_1} X_{j_2} \cdots X_{j_k}$

$\{j_1, j_2 \cdots j_k\} \subseteq \{1, 2, \cdots n\} \setminus N$

$N \subseteq \{i_1, i_2, \cdots i_m\}$

$$\underline{S \to A B a C}$$

$A \xrightarrow{*} \epsilon$, $B \xrightarrow{*} \epsilon$ $C \xrightarrow{*} \epsilon$

$\{A, B, C\}$

$\underline{\{A\}}$, $\{B\}$, $\{C\}$

$\{A, B\}$, $\{B, C\}$, $\{A, C\}$

$\{A, B, C\}$

$S \to BaC$	$\{A\}$
$S \to AaC$	$\{B\}$
$S \to ABa$	$\{C\}$
$S \to aC$	$\{A, B\}$
$S \to Ba$	$\{A, C\}$
$S \to Aa$	$\{B, C\}$
$S \to a$	$\{A, B, C\}$

$$\left\{ \begin{array}{l} S \to ABaC \mid BaC \mid ABa \mid AaC \mid aC \mid Ba \\ \qquad\quad Aa \mid a \end{array} \right.$$

$S \to ABC$ if A, B, C are all nullable.

then do not include $S \to \epsilon$

* $A \to BC$

$A \to BC \mid B \mid C$

$B \to b$

$C \to D$

$D \to d$

Removal Unit Production

$G = (V, T, P, S)$

$A, B \in V$

$A \to B$ Unit Production

Collect all variables $X \in V$ such that

$X \xrightarrow{*} Y \in V$

$S \to Aa \mid B$

$B \to A \mid bb$

$A \to a \mid bc \mid B$

$\{S, B, A\}$

$S \xrightarrow{*} B$

$B \xrightarrow{*} A$

$A \xrightarrow{*} B$

$S \xrightarrow{*} A$ because $S \xrightarrow{*} B \xrightarrow{*} A$

non unit production

$S \rightarrow Aa$

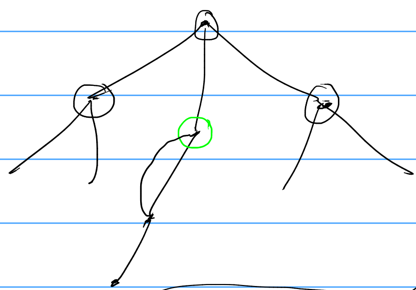
$B \rightarrow bb$

$A \rightarrow a / bc$

add — because $A \xrightarrow{*} B$

$A \rightarrow a / bc / bb$

$B \rightarrow bb / a / bc$ — bcz $B \xrightarrow{*} A$



because $S \xrightarrow{*} A$, $S \xrightarrow{*} B$

$S \rightarrow a / bc / bb \mid Aa$

Safe Order

1. Remove ϵ Production
2. " unit "
3. " useless "