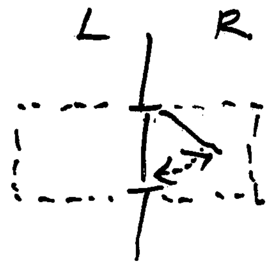
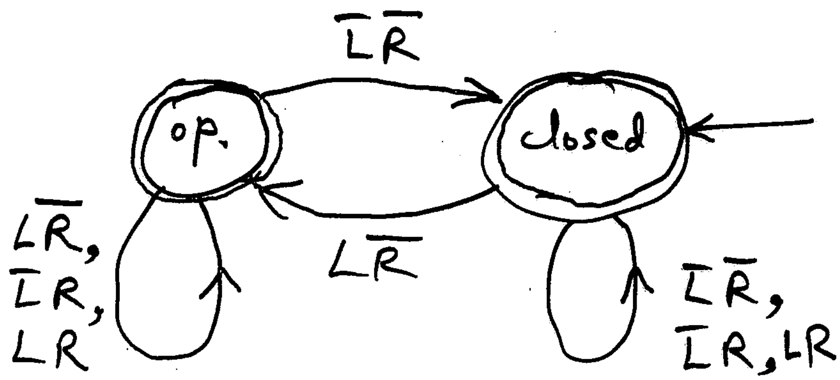


	$(\bar{L}\bar{R})$	$L\bar{R}$	$\bar{L}R$	LR
open	closed	open	open	open
closed	closed	open	~~open~~ closed	closed



- Design for sliding door (1-way) (2-way)
- Markov-chains: probablistic
- Specification ⟷ FSM ⟷ state diagram ⟷ 5-tuple ⟷ Language ⟷ Grammar
- Machine M <u>recognizes</u> language A $\equiv$ // M accepts only each $w \in A$ $L(M)$
- Machine M <u>accepts</u> a word/string $\in A$
- Computation of w on M: $\underset{(=q_0)}{r_0}\ w_1\ r_1\ w_2\ r_2 \ldots\ w_n\ \underset{(\in F)?}{r_n}$

M: $\Sigma = \{0, 1, 2, 3\}$

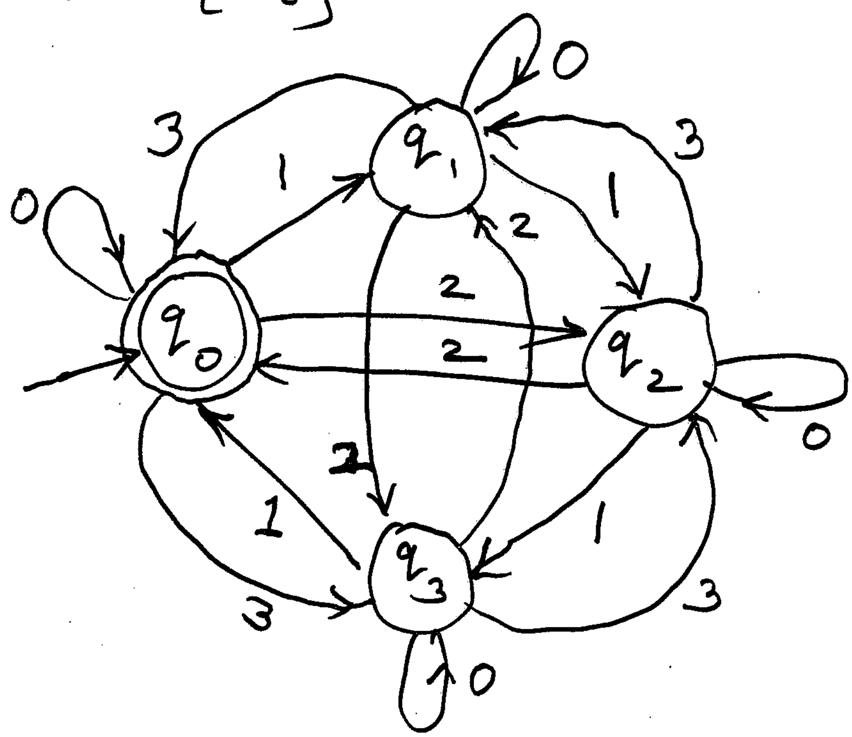
$Q = \{q_0, q_1, q_2, q_3\}$

$\delta =$

$q_0' = q_0$

$F = \{q_0\}$

$L(M) = \{ w \mid (\text{sum of symbol in } \cancel{\text{the}}\ w) \text{ mod. } 4 = 0 \}$



M: $\Sigma = \{0, 1, 2\}$

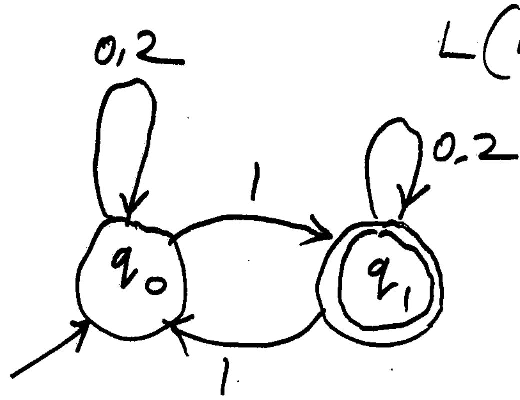
$Q = \{q_0, q_1\}$

$\delta =$

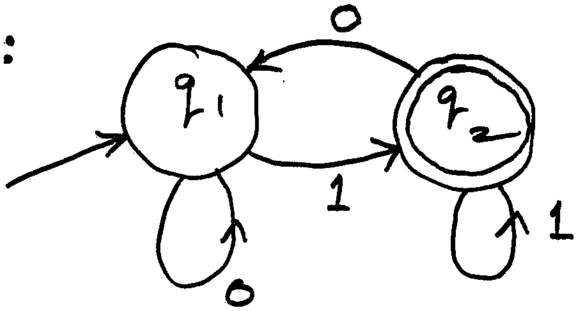
$q_0' = q_0$

$F = \{q_1\}$

$L(M) = \{ w \mid w \text{ has an odd numbe of } 1^s \}$



M :



$L(M) = \{ w \mid w \text{ ends in a } 1 \}$

$0010 \notin L(M)$

$00101 \in L(M)$

$q_1 \; q_1 \; q_1 q_2 q_1 q_2$

$Q = \{q_1, q_2\}$

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

$\delta =$

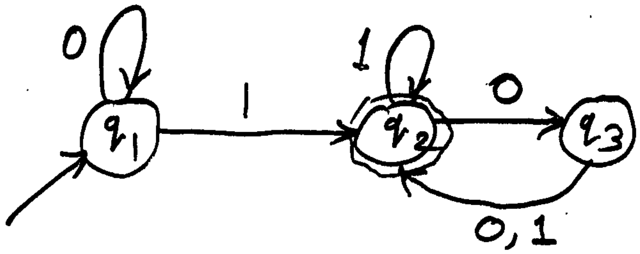
	0	1
q_1	q_1	q_2
q_2	q_1	q_2

$q_0 = q_1$

$F = \{q_2\}$

<u>F.S.A.</u> : 5-tuple $(Q, \Sigma, \delta, q_0, F)$

- Q → <u>finite</u> set of states
- Σ → <u>finite</u> alphabet set
- δ → $Q \times \Sigma \to Q$
- q_0 → $\in Q$ (<u>start</u> state)
- F → $\subseteq Q$ (set of ~~ϕ~~ accept states)



$L(M) = \{ w \mid w \text{ has an even \# of } 0^s \text{ following the last } 1 \}$

<u>Defn</u>: M accepts w $(w_1 w_2 \cdots w_n)$ if a seq of states $r_0\, r_1\, r_2 \cdots\cdots r_n$

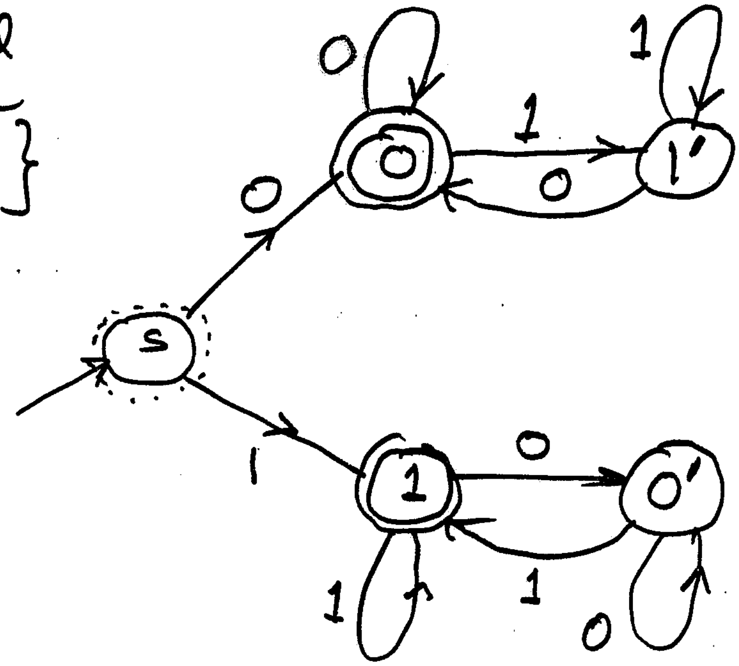
such that: ① $r_0 = q_0$, ② for $i \in [0, n-1]$ $r_{i+1} = \delta(r_i, w_{i+1})$

③ $r_n \in F$

$L(M) = \{ w \mid$ w begins and ends with the same alphabet $\}$

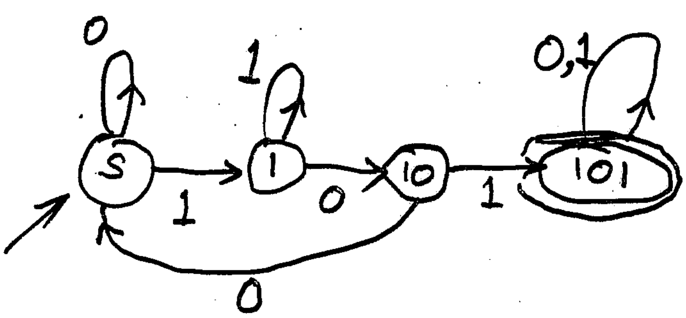
$\Sigma = \{0, 1\}$

// 'ε' for empty string

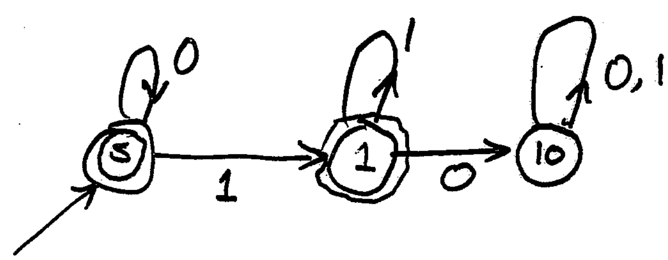


$\Sigma = \{0, 1\}$

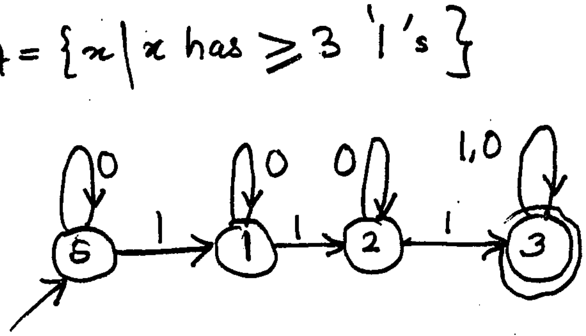
$A = \{ x \mid$ '101' is substring $\}$



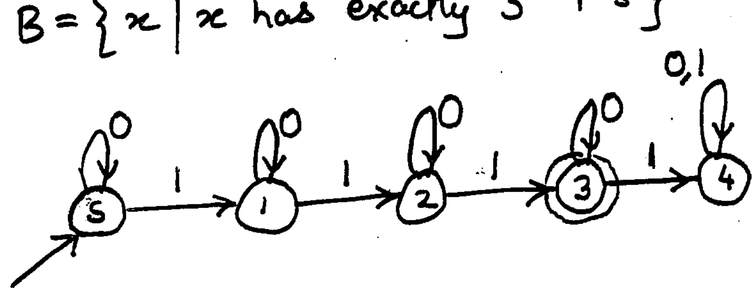
$B = \{ x \mid$ '10' is not a substring $\}$



$A = \{ x \mid x$ has ≥ 3 '1's $\}$



$B = \{ x \mid x$ has exactly 3 '1's $\}$



Concat A∘B with DFAs ?

Defn: Regular language is one that is recognized by some F.S.A.

- $A \cup B = \{ x \mid x \in A \text{ or } x \in B \}$

  eg 1: $A = \{x \mid x \text{ is a multiple of } 3\}$, $B = \{x \mid x \text{ is a multiple of } 4\}$ — $\Sigma = \{1\}$

- $A \circ B = \{ xy \mid x \in A \text{ and } y \in B \}$

  eg 1: $A = \{x \mid \text{`101' is a substring}\}$, $B = \{x \mid \text{`10' is not a substring}\}$ — $\Sigma = \{0,1\}$

- $A^* = \{ x_1 x_2 \cdots x_k \mid k \geq 0 \text{ and each } x_i \in A \}$

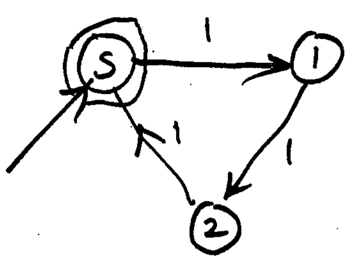
Defn: CLOSURE of a SET under an OPERATION

eg $(\mathbb{N}, +)$

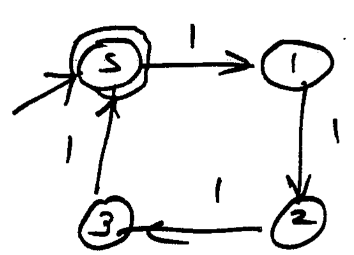
Theorem: The set of regular Langs. is closed under $\cup$

$\Sigma = \{1\}$

A:



B:



Proof: $M(A) = M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$

$M(B) = M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$

Construct $M(A \cup B) = M = (Q, \Sigma, \delta, q_0, F)$

- $Q = \{(r_1, r_2) \mid r_1 \in Q_1 \text{ and } r_2 \in Q_2\}$
- $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$

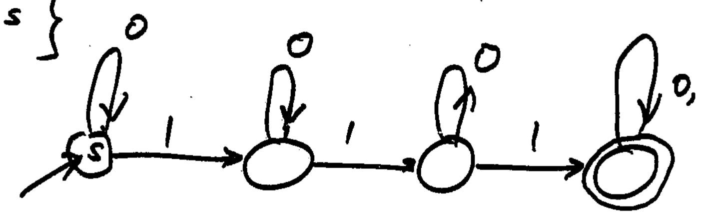
  $Q \times \Sigma \to Q$
- $q_0 = (q_1, q_2)$
- $F = \{(r_1, r_2) \mid r_1 \in F_1 \text{ or } r_2 \in F_2\}$

  $= F_1 \times Q_2 \cup Q_1 \times F_2$

Q.E.D.

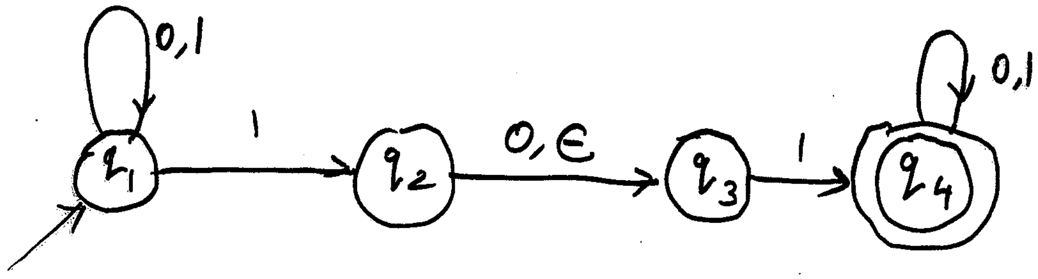
- $L(M_A) = \{x \mid x \text{ contains (at least) } 3 \text{ `1's}\}$

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



- $L(M_B) = \{x \mid 3^{rd} \text{ last symbol is `1'}\}$

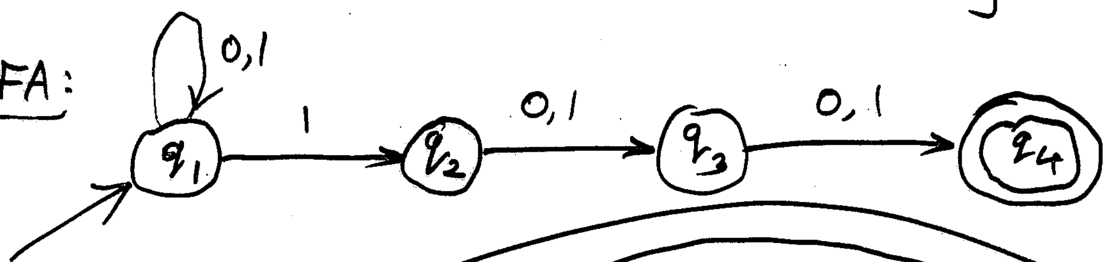
- Prove CLOSURE under $\cap$
- " " " complement



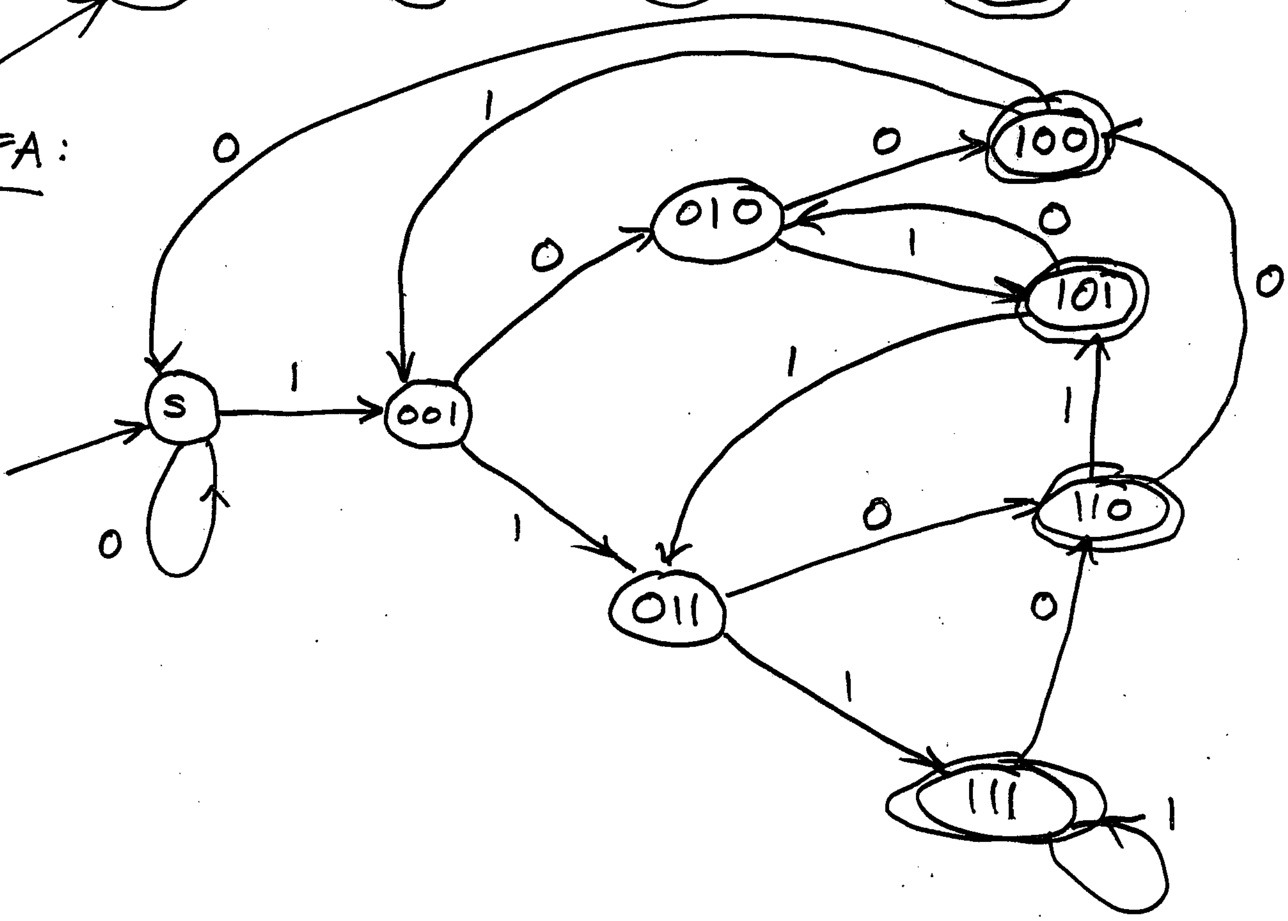
$L(M) = \{ x \mid$ '101' or '11' are substrings of $x \}$

$L(M) = \{ x \mid 3^{rd}$ last symbol is '1' $\}$

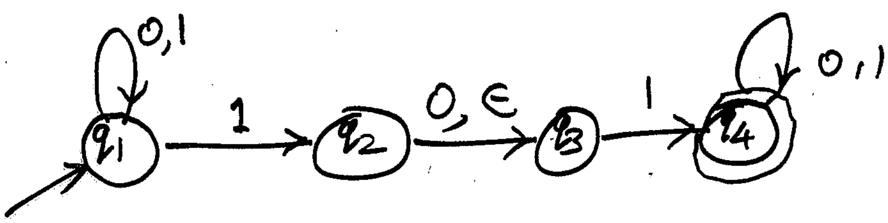
NFA:



DFA:



Nondeterministic FSMs (NFA) // or "the power of CHOICE"



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

NFA vs DFA

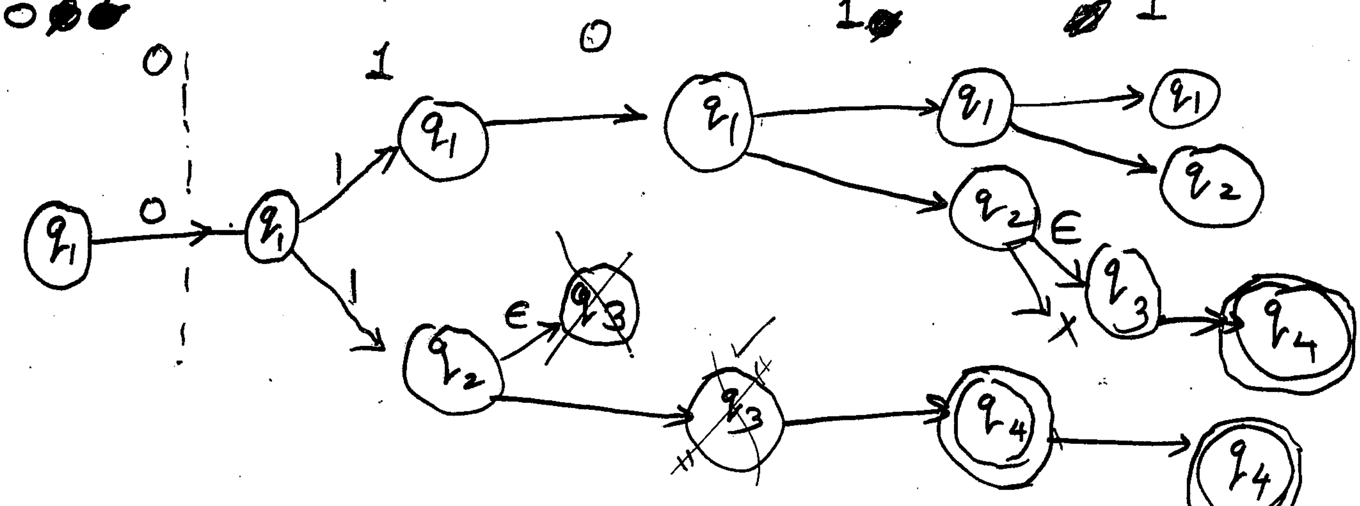
① $\forall q \in Q, \forall a \in \Sigma, \delta(q,a) = \{ \quad \}$ → set of ≥ 0 elements

eg: $\delta(3,0) = \{\}$

$\delta(1,1) = \{1,2\}$

② $\forall q \in Q$, 0 or more out-edges labelled ϵ

$w = 01011$



After any symbol, DFA-equivalent of NFA has to remember that it is in '1' state out of $2^{|Q|}$ states.

Theorem: Every NDFA has an equiv. DFA.

Given

$N = (Q_N, \Sigma, \delta_N, q_{0(N)}, F_N)$

build $D = (Q_D, \Sigma, \delta_D, q_{0(D)}, F_D)$

- $Q_D = P(Q_N)$, - $q_{0(D)} = \{q_{0(N)}\}$ // power set
- $\delta_D = ?$ For each 's' in Q_D and for 'a' in Σ

$= Q_D \times \Sigma$

$$\delta_D(s,a) = \{ q \in Q_N \mid q \in \delta_N(t_N, a), \text{ where } t_N \in S_D \}$$

(assume no ϵ transitions yet) OR $\delta_D(s,a) = \bigcup_{t \in S} \delta_N(t,a)$

Now, ϵ-Transitions:

$E(s) = \{ q \mid q$ is reachable from S by using ≥ 0 ϵ-transitions $\}$

→ $\delta_D(s,a) = \{ q \in Q_N \mid q \in E(\delta_N(t,a)), \text{ where } t_N \in S_D \}$

→ $q_{0(D)} = E(\{q_{0(N)}\})$ // $\delta_N(E(t), a$

- $F_D = \{ s \in Q_D \mid s$ contains an accept state of $N \}$
  ie $s \cap F_N \neq \phi$

Defn: 2 machines are "equivalent" if they recognize the same language

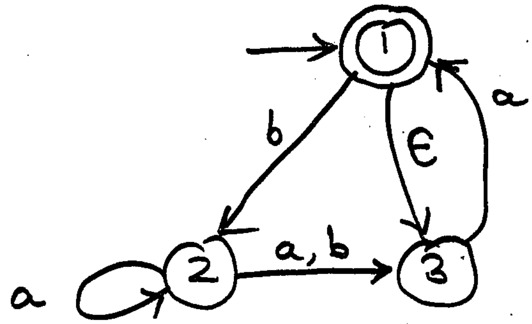
Corollary to Theorem: Lang L is regular $\Longleftrightarrow$ some NFA recognizes L

→ by defn, some DFA recognizes L

NFA :

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

$Q_N = \{1,2,3\}$



δ	a	b	ϵ
1	ϕ	$\{2\}$	$\{3\}$
2	$\{2,3\}$	$\{3\}$	ϕ
3	$\{1\}$	ϕ	ϕ

DFA : $Q_D = \{ \phi, \{1\}, \{2\}, \{3\}, \{1,2\}, \{2,3\}, \{1,3\}, \{1,2,3\} \}$

δ_D	a	b
$\{\} = \phi$	ϕ	ϕ
$\{1\}$	$\cancel{\phi}\ \cancel{\{\}}\ \phi$	$\{2\}$
$\{2\}$	$\{2,3\}$	$\{3\}$
$\{3\}$	$\{1,3\}$	ϕ
$\{1,2\}$	$\{\cancel{1},2,3\}$	$\{2,3\}$
$\{1,3\}$	$\{1,3\}$	$\{\cancel{2}\}$
$\{2,3\}$	$\{2,3,1\}$	$\{3\}$
$\{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$

$q_D = \cancel{\{1\}}\ \{1,3\}$

$F_D = \{\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}$

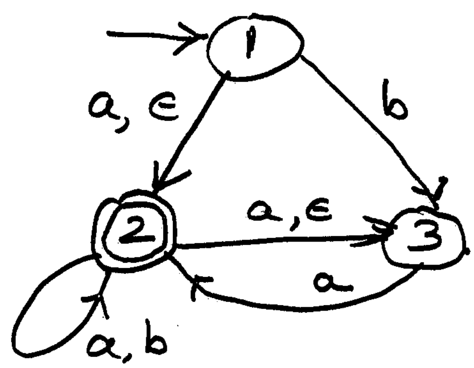
- Remove redundant states $\{1\}, \{1,2\}$ // no in-edges
- What is significance of ϕ ?

Defn: NFA $N = (Q, \Sigma, \delta, q_0, F)$, where $\delta : Q \times \Sigma_\epsilon \to P(Q)$

Defn: "Computation". N accepts w if w can be written as $y_1 y_2 y_3 \cdots y_m$ where $y_i \in \Sigma_\epsilon$ & $r_0, r_1, \ldots, r_m \in Q$ such that

1) $r_0 = q_0$ 2) $r_{i+1} \in \delta(r_i, y_{i+1})$ for $i = 0, 1, \ldots, m-1$ 3) $r_m \in F$

NFA



DFA: $Q_D = \{ \qquad \}$

δ_D	a	b
ϕ		
$\{1\}$	$\{2,3\}$	$\{3\}$
$\{2\}$	$\{2,3\}$	$\{2,3\}$
$\{3\}$	$\{2,3\}$	ϕ
$\{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{1,3\}$	$\{2,3\}$	$\{3\}$
$\{2,3\}$	$\{2,3\}$	$\{2,3\}$
$\{1,2,3\}$	$\{2,3\}$	$\{2,3\}$

$q_{0(D)} = \{1,2,3\}$

$F_D = \{ \{2\}, \{1,2\}, \{2,3\}, \{1,2,3\} \}$

NFA vs DFA

1) easier to understand 2) smaller 3) easier to design
4) introduction to "nondeterminism" in more powerful models

CLOSURE under REGULAR OPs. $\cup, \circ, *$

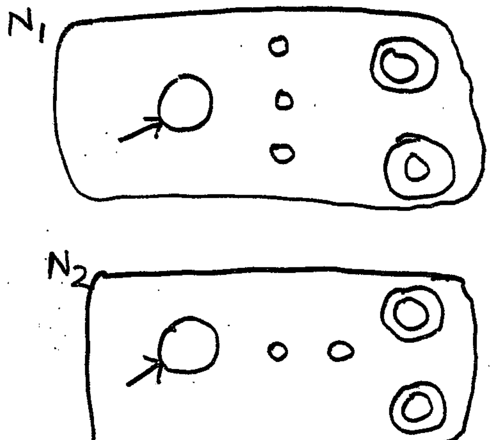
- easier to prove with NFAs

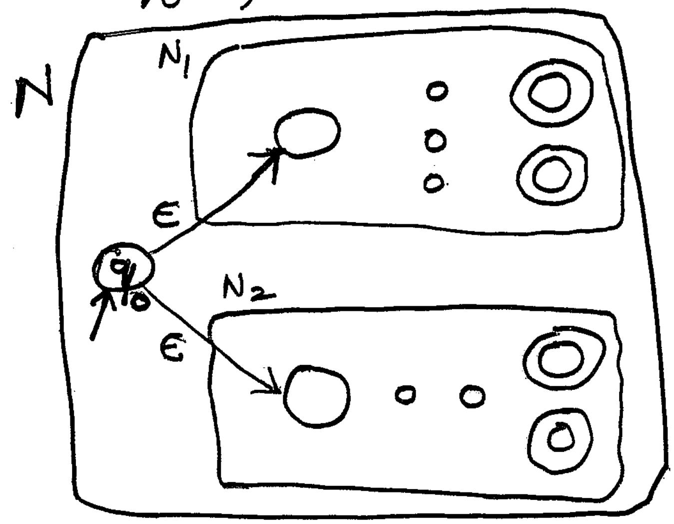
<u>Theorem</u>: set of regular languages closed under $\cup$

$L(N_1) = A_1 \qquad N_1 = (Q_1\ \Sigma\ \delta_1\ q_1\ F_1)$

$L(N_2) = A_2 \qquad N_2 = (Q_2\ \Sigma\ \delta_2\ q_2\ F_2)$

$L(N) = A_1 \cup A_2 \qquad N = (Q\ \Sigma\ \delta\ q_0\ F)$

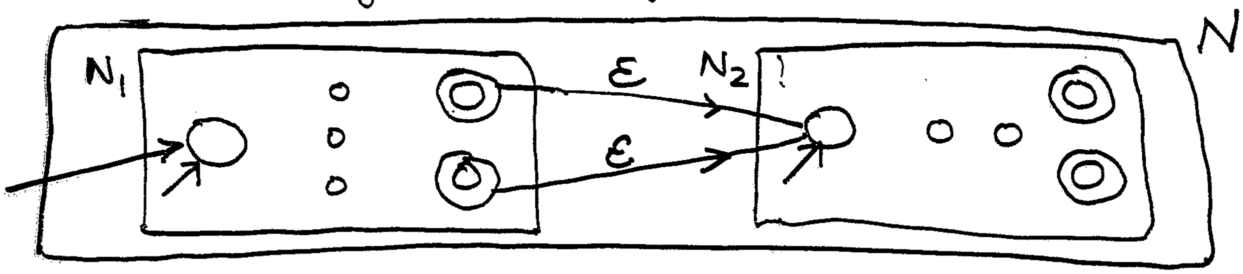




- $Q = Q_1 \cup Q_2 \cup \{q_0\}$
- $F = F_1 \cup F_2$
- $$\delta(q,a) = \begin{cases} \delta_1(q,a) & \text{if } q \in Q_1 \\ \delta_2(q,a) & \text{if } q \in Q_2 \\ \{q_1, q_2\} & \text{if } q = q_0 \text{ and } a = \epsilon \\ \emptyset & \text{if } q = q_0 \text{ and } a \neq \epsilon \end{cases}$$

eg: $\Sigma = \{1\}$; $A_1 = \{$multiple of 3 '1's$\}$ $A_2 = \{$multiple of 4 '1's$\}$

Theorem: Set of regular languages closed under concat



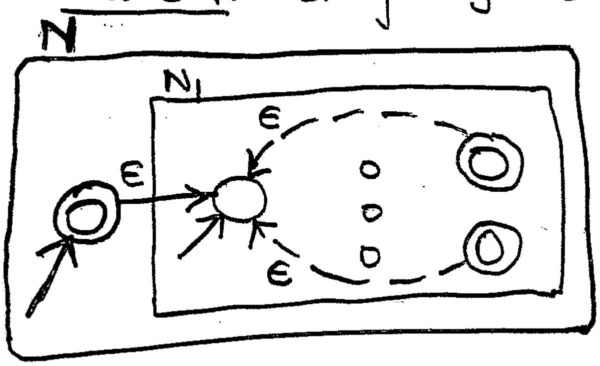
- $Q = Q_1 \cup Q_2$ — $q_0 = q_1$ — $F = F_2$

- $$\delta(q,a) = \begin{cases} \delta_1(q,a) & \text{if } q \in Q_1 \text{ and } q \notin F \\ \delta_1(q,a) & \text{if } q \in F_1 \text{ and } a \neq \varepsilon \\ \delta_1(q,a) \cup \{q_2\} & \text{if } q \in F_1 \text{ and } a = \varepsilon \\ \delta_2(q,a) & \text{if } q \in Q_2 \end{cases}$$

eg: $\Sigma = \{0,1\}$; $A_1 = \{x \mid x \text{ has at least 3 `1's}\}$

$A_2 = \{x \mid x \text{ has exactly 3 `1's}\}$ or $\{x \mid 3^{rd} \text{ last is `1'}\}$

Theorem: Set of regular languages closed under 'star'



$N_1 = (Q_1 \;\; \Sigma \;\; \delta_1 \;\; q_1 \;\; F_1)$

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

- $Q = Q_1 \cup \{q_0\}$
- $q_0 = \text{new state}$
- $F = F_1 \cup \{q_0\}$

- $$\delta(q,a) = \begin{cases} \delta_1(q,a) & \text{if } q \in Q_1 \text{ and } q \notin F_1 \\ \delta_1(q,a) & \text{if } q \in F_1 \text{ and } a \neq \varepsilon \\ \delta_1(q,a) \cup \{q_1\} & \text{if } q \in F_1 \text{ and } a = \varepsilon \\ \{q_1\} & \text{if } q = q_0 \text{ and } a = \varepsilon \\ \phi & \text{if } q = q_0 \text{ and } a \neq \varepsilon \end{cases}$$

Think: Why is q_0 needed?

REGULAR EXPRESSION : its value is a language ∴ $L(R)$

eg $(0 \cup 1)\,1^* \equiv (\underbrace{\{0\} \cup \{1\}}_{\text{union}}) \underbrace{\circ}_{\text{concat}} \underbrace{\{1\}^*}_{\text{star}}$

<u>precedence</u>

decrease ↓ $*$, $\circ$, $\cup$

- Σ : all strings of length '1' over alphabet set Σ
- Σ^*: all strings over that alphabet

<u>Defn</u>: $R(\Sigma)$ is a regular expression if R is // GRAMMAR

1) a, for any $a \in \Sigma$ // language $\{a\}$
2) ε // language $\{\varepsilon\}$
3) ϕ // language $\{\}$
4) $(R_1 \cup R_2)$, where R_1 and R_2 are regular expressions
5) $(R_1 \circ R_2)$, —— " ——
6) R_1^* , —— " ——

<u>Note</u>: $R \circ \varepsilon = R$ $\quad$ $R \cup \varepsilon$ adds ε to R ie, $L(R) \cup \{\varepsilon\}$

$R \circ \phi = \phi$ $\quad$ $R \cup \phi = R$

$\phi^* = \{\varepsilon\}$

Practice: What is $L(R(\Sigma))$ for $\Sigma = \{0,1\}$:

1) $\Sigma^* 10 \Sigma^*$
2) $\Sigma^* 10 \Sigma\Sigma$
3) $(\Sigma\Sigma)^*$
4) $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$
5) $(1 \cup \varepsilon)\, 0^*$
6) $(1 \cup \phi)\, 0^*$

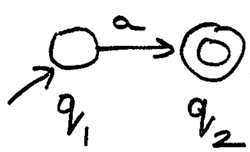
// a^+, Σ^+ = 1 or more instances

Lemma: $\underbrace{L(R)}$ is a $\underbrace{\text{regular language}}$.

(ie, lang described by (regular) expr is accepted by a FA)
generated by Rules 1–6

1) $R = a,\ a \in \Sigma$

$L(R) = \{a\}$



$N = (\{q_1, q_2\}, \Sigma, \delta, q_1, \{q_2\})$

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

$\delta(s, b) = \phi$ if $s \neq q_1$ or $b \neq a$

2) $R = \varepsilon$

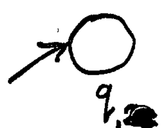
$L(R) = \{\varepsilon\}$

$N = (\{q_1\}, \Sigma, \delta, q_1, \{q_1\})$

$\delta(s, b) = \phi$, ~~for~~ $\forall s \in Q\ \forall b \in \Sigma$

3) $R = \phi$

$L(R) = \phi$



$N = (\{q_1\}, \Sigma, \delta, q_1, \phi)$

$\delta(s, b) = \phi$, $\forall s \in Q\ \forall b \in \Sigma$

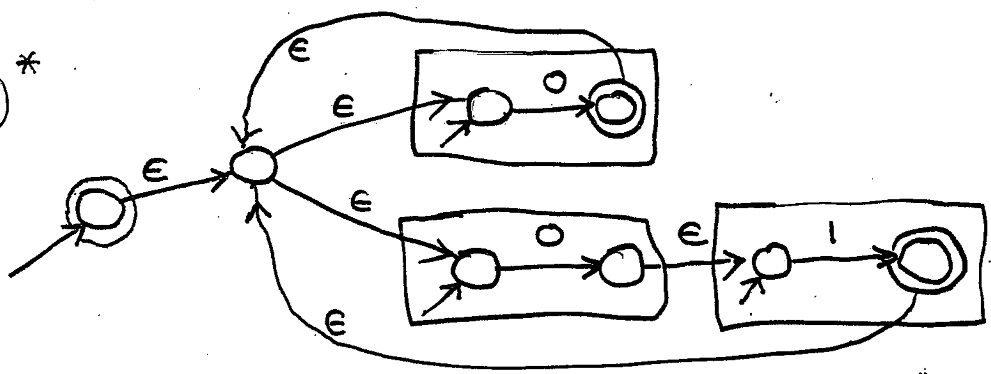
4) $R = R_1 \cup R_2$

5) $R = R_1 \circ R_2$

6) $R = R_1^*$

} already proved CLOSURE OF Regular languages under $\cup, \circ, *$

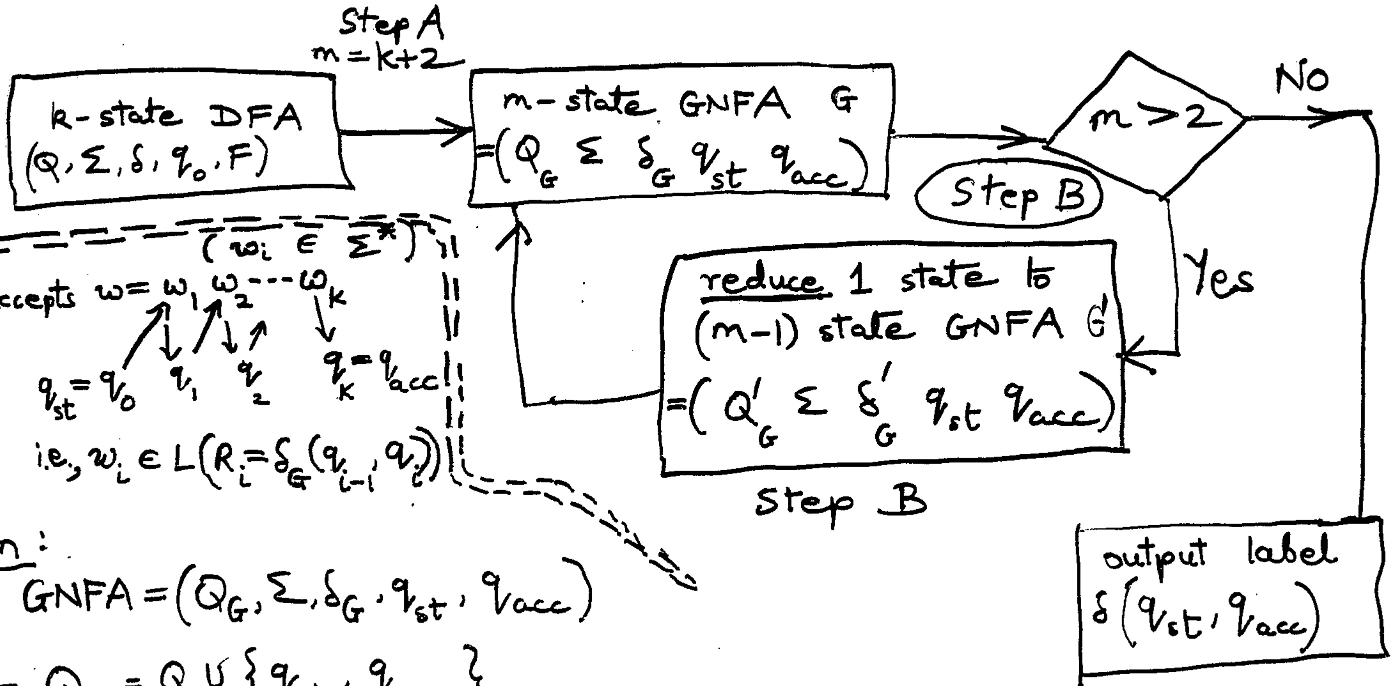
eg $(0 \cup 01)^*$



eg $(0 \cup 01)^* 10$

Now, 2-state NFA for $(0 \cup 01)^*$

Lemma: $\underbrace{\text{A regular language}}_{\text{accepted by a FA}}$ is $\underbrace{\text{generated by a regular expression}}_{L(R)}$



$(w_i \in \Sigma^*)$

G accepts $w = w_1 w_2 \cdots w_k$ if

$q_{st} = q_0 \quad q_1 \quad q_2 \quad q_k = q_{acc}$

i.e., $w_i \in L(R_i = \delta_G(q_{i-1}, q_i))$

Defn:

$GNFA = (Q_G, \Sigma, \delta_G, q_{st}, q_{acc})$

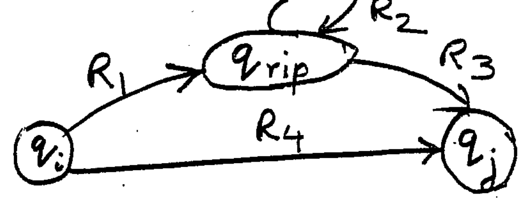
- $Q_G = Q \cup \{q_{st}, q_{acc}\}$
- $\delta_G : (Q_G - \{q_{acc}\}) \times (Q_G - \{q_{st}\}) \longrightarrow \{R \mid R \text{ is a reg. expr over } \Sigma\}$

$\xrightarrow{\text{i.e.}} L(R(\Sigma))$

Step A:

1) ε-edge (q_{st}, q_0) 2) ε-edge (q_f, q_{acc}), $\forall q_f \in F$
3) Multiple labels on arrow $\longrightarrow$ union of the labels
4) To complete δ_G, map each missing tuple (q_x, q_y) to ϕ, i.e., $\xrightarrow{\phi}$

Step B: $G' \longleftarrow \text{REDUCE}(G)$

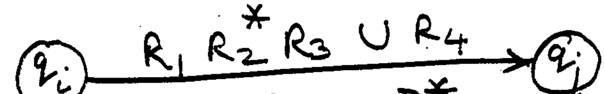


1) Select any $q_{rip} \in Q_G$
2) $Q'_G \longleftarrow Q_G - \{q_{rip}\}$
3) Let $q_i \in Q'_G - \{q_{acc}\}$, $q_j \in Q'_G - \{q_{st}\}$

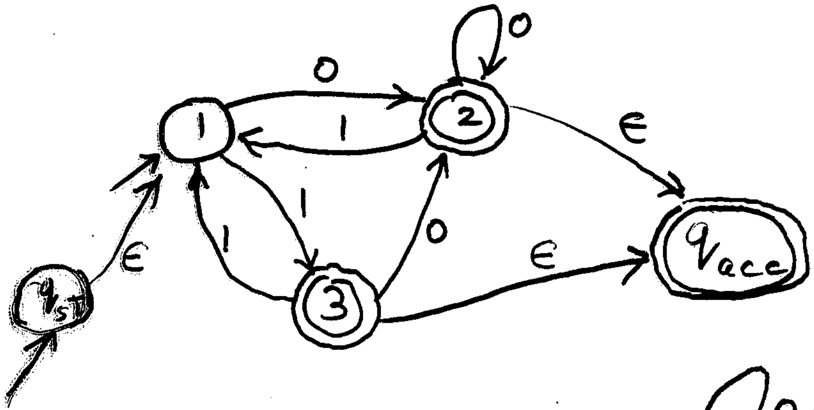
$R_1 = \delta_G(q_i\, q_{rip}) \qquad R_2 = \delta_G(q_{rip}\, q_{rip})$

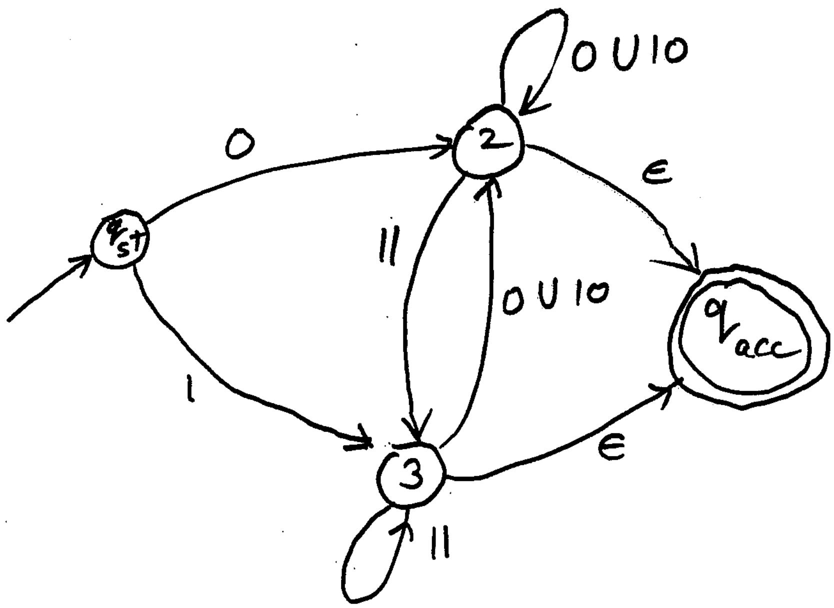
$R_3 = \delta_G(q_{rip}, q_j) \qquad R_4 = \delta_G(q_i\, q_j)$

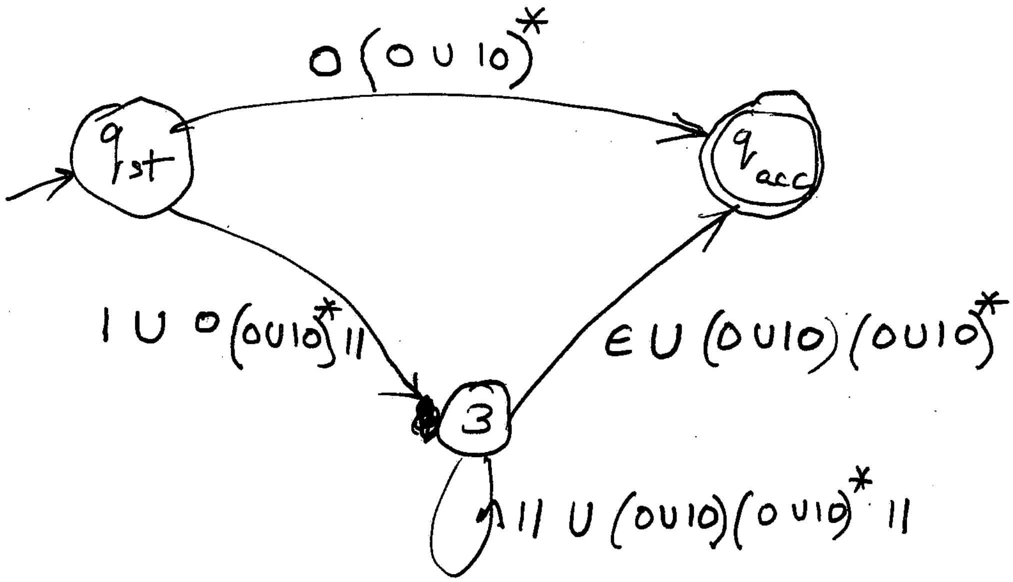
Then $\forall i\, \forall j$, $\delta'_G = R_1 R_2^* R_3 \cup R_4$

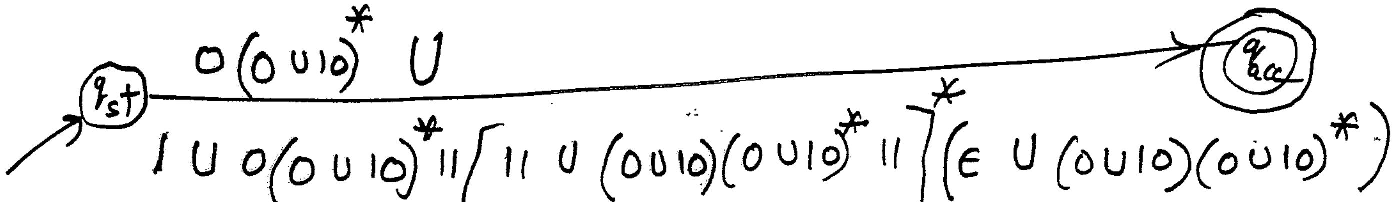


$\delta_G(q_i\, q_{rip}) \left[\delta_G(q_{rip}\, q_{rip})\right]^* \delta_G(q_{rip}\, q_j)$
$\cup\ \delta_G(q_i\, q_j)$


0
0
1
1
1
0
ϵ
ϵ
ϵ
1
2
3
q_{st}
q_{acc}


0 U 10
0
ϵ
11
0 U 10
1
ϵ
11
2
3
q_{st}
q_{acc}


0 (0 U 10)*
1 U 0 (0U10)* 11
ϵ U (0 U 10) (0 U 10)*
11 U (0U10)(0 U 10)* 11
3
q_{st}
q_{acc}


0 (0 U 10)* U
1 U 0(0 U 10)* 11 [11 U (0U10)(0 U 10)* 11]* (ϵ U (0U10)(0U10)*)
q_{st}
q_{acc}