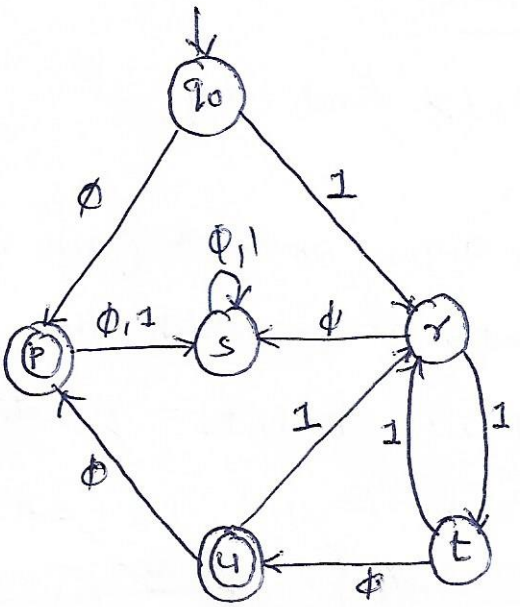
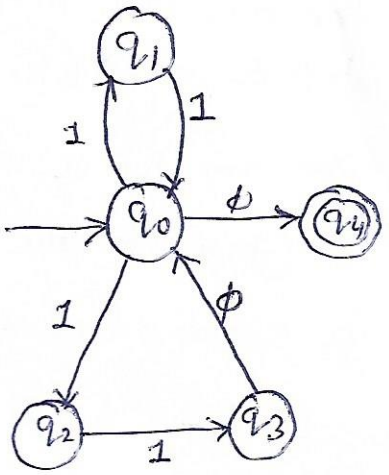
CH 4 — Nondeterminism and Kleene's Theorem

FA



NFA



A simpler approach for accepting $\{11, 110\}^*\{0\}$

NFA

- For some state-input combination, no transition may be specified
- Multiple transitions from a state for an input symbol. eg. 2 transitions from q_0 on input 1.

- Acceptance of a string : Some path corresponding to the string should lead to an accepting state.

- NFA is allowed to guess and make mistakes.
- Structure of NFA is simple, less states and closely matches the RE.

Definition – [A Non-Deterministic Finite Automaton]

An NFA $M = \langle Q, \Sigma, q_0, A, \delta \rangle$, where Q and Σ are non-empty finite sets, $q_0 \in Q$, $A \subseteq Q$ and

$$\delta : Q \times \Sigma \to 2^Q$$

↑ Set of subsets of Q.

Similar to FA, δ^* is:

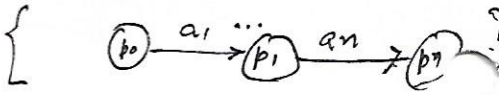
$$\delta^*(p, xa) = \delta(\delta^*(p, x), a)$$

Non-Recursive Def^n of δ^* for an NFA

For an NFA $M = \langle Q, \Sigma, q_0, A, \delta \rangle$ and any $p \in Q$, $\delta^*(p, \lambda) = \{p\}$.

For any $p \in Q$ and any $x = a_1 a_2 \ldots a_n \in \Sigma^*$ (with $n \geq 1$), $\delta^*(p, x)$ is the set of all states q for which there is a sequence of states $p = p_0, p_1, \ldots p_{n-1}, p_n = q$ satisfying

$$p_i \in \delta(p_{i-1}, a_i) \text{ for each } i \text{ with } 1 \leq i \leq n.$$



Recursive Def^n of δ^* for an NFA

Let $M = (Q, \Sigma, q_0, A, \delta)$ be an NFA. The function $\delta^*: Q \times \Sigma^* \to 2^Q$ is defined as follows:

1. For any $q \in Q$, $\delta^*(q, \lambda) = \{q\}$
2. For any $q \in Q$, $y \in \Sigma^*$ and $a \in \Sigma$

$$\delta^*(q, ya) = \bigcup_{p \in \delta^*(q, y)} \delta(p, a)$$

Acceptance by an NFA

Let $M = (Q, \Sigma, q_0, A, \delta)$ be an NFA.

The string $x \in \Sigma^*$ is accepted by M if

$$\delta^*(q_0, x) \cap A \neq \phi$$

Example

State Transition Table for M

δ :

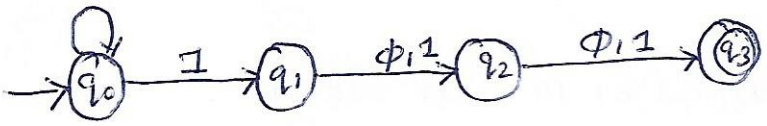
q	$\delta(q,0)$	$\delta(q,1)$
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_2\}$	$\{q_2\}$
q_2	$\{q_3\}$	$\{q_3\}$
(q_3)	ϕ	ϕ

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

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

$A = \{q_3\}$

State Transition Diagram for M



Let's compute δ^* for some strings using the recursive definition.

i) $\delta^*(q_0, 111)$ ii) $\delta^*(q_0, 011)$

$$\delta^*(q_0, 11) = \bigcup_{p \in \delta^*(q_0,1)} \delta(p,1)$$

$$= \bigcup_{p \in \{q_0, q_1\}} \delta(p,1)$$

$$= \delta(q_0,1) \cup \delta(q_1,1)$$

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

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

$$\delta^*(q_0, 111) = \bigcup_{p \in \delta^*(q_0,11)} \delta(p,1)$$

$$= \bigcup_{p \in \{q_0, q_1, q_2\}} \delta(p,1)$$

$$= \delta(q_0,1) \cup \delta(q_1,1) \cup \delta(q_2,1)$$

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

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

$$\delta^*(q_0, 01) = \bigcup_{p \in \delta^*(q_0,0)} \delta(p,1)$$

$$= \bigcup_{p \in \{q_0\}} \delta(p,1)$$

$$= \delta(q_0,1)$$

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

$$\delta^*(q_0, 011) = \bigcup_{p \in \delta^*(q_0, 01)} \delta(p, 1)$$

$$= \bigcup_{p \in \{q_0, q_1\}} \delta(p, 1)$$

$$= \delta(q_0, 1) \cup \delta(q_1, 1)$$

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

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

$\Rightarrow$ 111 is accepted by M but not 011.

The language recognized by M is

$$\{0,1\}^* \{1\} \{0,1\}^* \{0,1\}$$

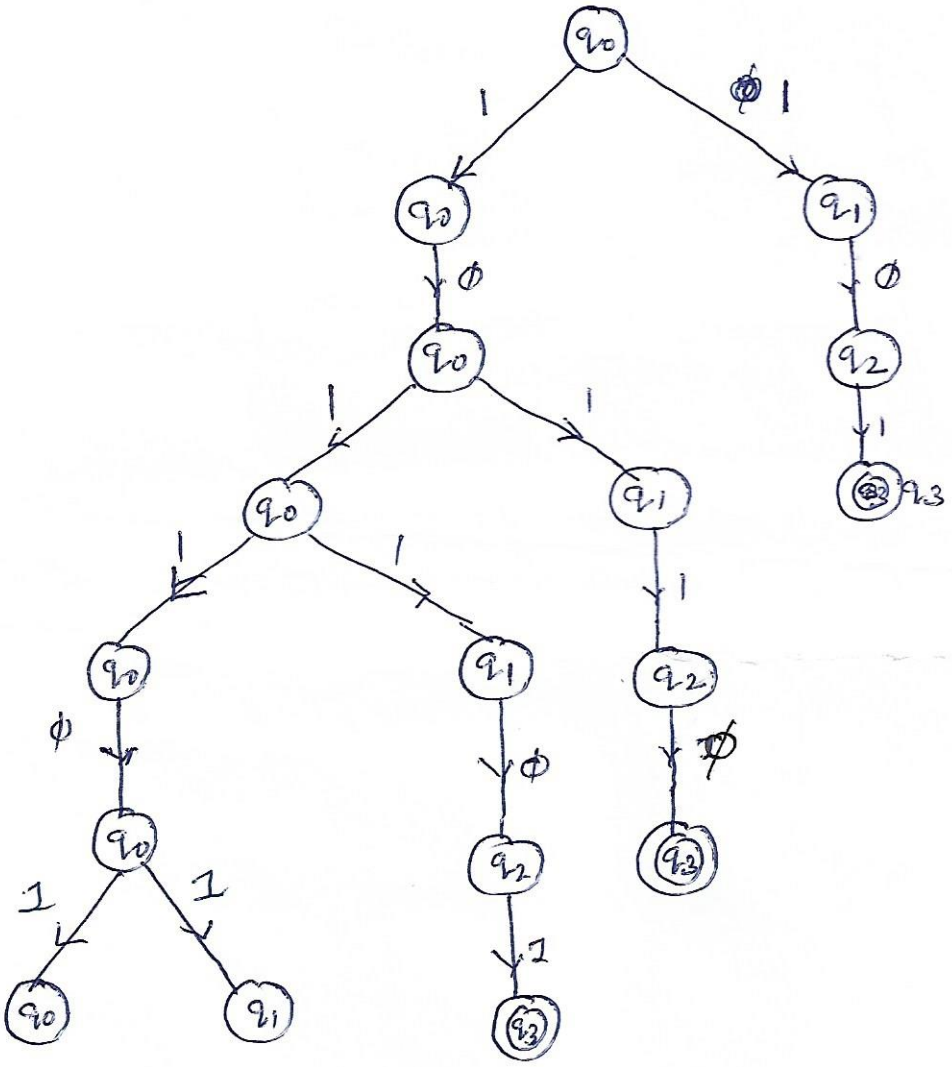
i.e $\{0,1\}^* \{1\} \{0,1\}^{*2}$ which is L_3

i.e. Set of strings with length atleast 3 and having 1 in the third position from the end.

For any $n \geq 1$, an NFA with $n+1$ states recognizes L_n, whereas an ordinary FA accepting L_n needs atleast 2^n states.

Computational Tree for processing a string by an NFA.

Computational Tree for the input 10110 for the NFA of last example.



Deciding whether x is accepted is by checking whether any accepting states appear in the tree at level $|x|$ (Initial state is level ϕ)

Theorem 4.1

For any NFA $M = \langle Q, \Sigma, q_0, A, \delta \rangle$ accepting language $L \subseteq \Sigma^*$, there is an FA $M_1 = \langle Q_1, \Sigma, q_1, A_1, \delta_1 \rangle$ that also accepts L.

Proof

M_1 is defined as follows:

$$Q_1 = 2^Q \qquad q_1 = \{q_0\}$$

For $q \in Q_1$ and $a \in \Sigma$, $\delta_1(q, a) = \bigcup_{r \in q} \delta(r, a)$

$$A_1 = \{ q \in Q_1 \mid q \cap A \neq \phi \}$$

↑ { If a subset has an accepting state, then the subset becomes accepting state in M_1 }

The last def^n/ is the right one because a string x should be accepted by M_1 if, starting in q_0, the set of states in which M might end up as a result of processing x contains at least one element of A.

The fact that M_1 accepts the same language as M follows from the fact that for any $x \in \Sigma^*$,

$$\delta_1^*(q_1, x) = \delta^*(q_0, x)$$

which we now prove using structural induction on x.

If $x = \Lambda$

$$\delta_1^*(q_1, x) = \delta_1^*(q_1, \Lambda)$$

$$= q_1 \text{ (by def}^n\text{/ of } \delta_1^*)$$

$$= \{q_0\} \text{ (by def}^n\text{/ of } q_1)$$

$$= \delta^*(q_0, \Lambda) \text{ (by def}^n\text{/ of } \delta^*)$$

$$= \delta^*(q_0, x)$$

The induction hypothesis is that x is a string satisfying $\delta_1^*(q_1, x) = \delta^*(q_0, x)$, and we wish to prove that for any $a \in \Sigma$, $\delta_1^*(q_1, xa) = \delta^*(q_0, xa)$:

$$\delta_1^*(q_1, xa) = \delta_1(\delta_1^*(q_1, x), a)$$

(by defn of δ_1^*)

$$= \delta_1(\underbrace{\delta^*(q_0, x)}_{q}, a) \quad \text{(by Ind. Hypo.)}$$

$$\delta_1(q, a) = \bigcup_{r \in q} \delta(r, a) = \bigcup_{r \in \delta^*(q_0, x)} \delta(r, a) \quad \text{(by our defn of } \delta_1\text{)}$$

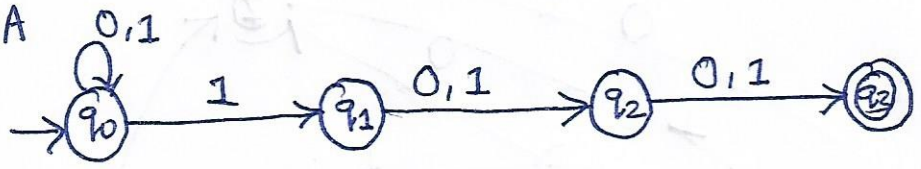
$$= \delta^*(q_0, xa) \quad \text{(by defn of } \delta^*\text{)}$$

That M & M_1 recognize the same lang. is now easy to see. A string x is accepted by M_1 if $\delta_1^*(q_1, x) \in A_1$; we can now say that this is true if and only if $\underbrace{\delta^*(q_0, x)}_{q} \in A_1$; and using the definition of A_1, we conclude that this is true if and only if $\underbrace{\delta^*(q_0, x)}_{q} \cap A \neq \emptyset$. In other words, x is accepted by M_1 if and only if x is accepted by M.

Applying Subset Construction Algorithm to convert an NFA to an FA

(2^N Subsets Possible for N States of NFA)

NFA

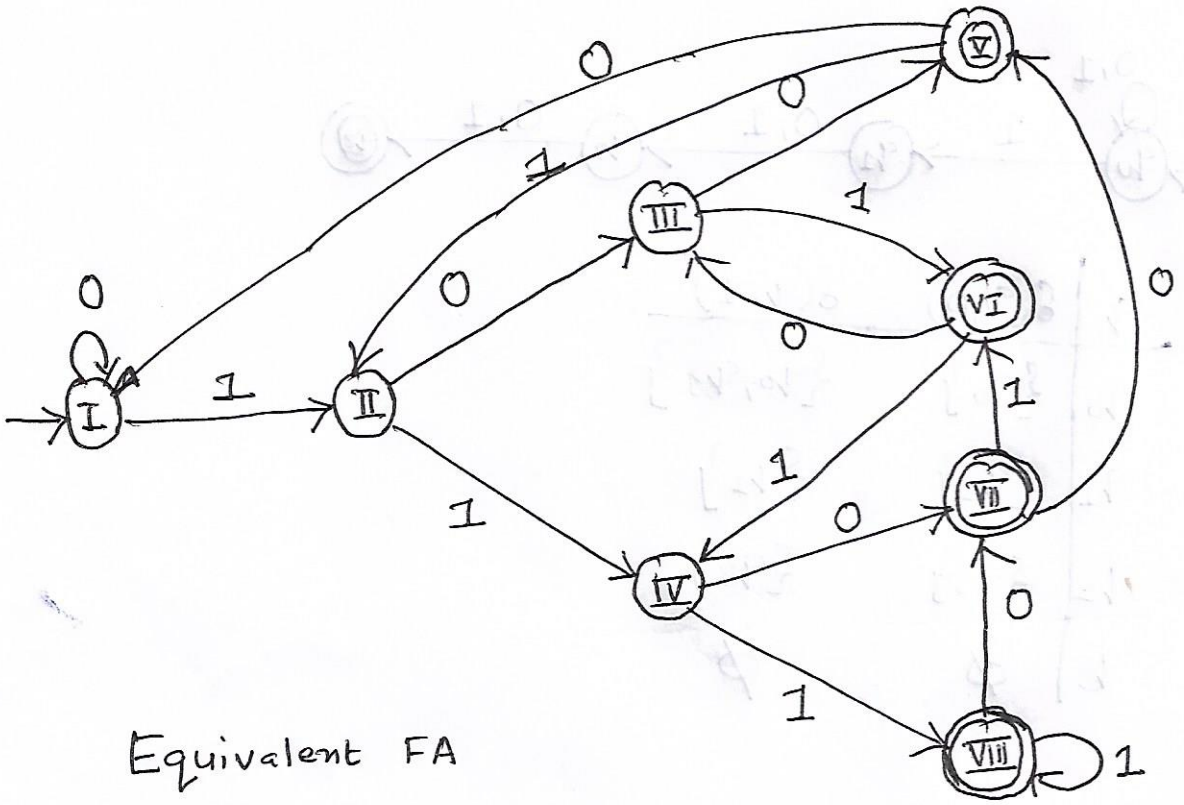


q	$\delta(q,0)$	$\delta(q,1)$
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_2\}$	$\{q_2\}$
q_2	$\{q_3\}$	$\{q_3\}$
q_3	ϕ	ϕ

FA

	Subset s	$\delta_1(s,0)$	$\delta_1(s,1)$
→ I	$\{q_0\}$	$\{q_0\}$ I	$\{q_0, q_1\}$ II
II	$\{q_0, q_1\}$	$\{q_0, q_2\}$ III	$\{q_0, q_1, q_2\}$ IV
III	$\{q_0, q_2\}$	$\{q_0, q_3\}$ V	$\{q_0, q_1, q_3\}$ VI
IV	$\{q_0, q_1, q_2\}$	$\{q_0, q_2, q_3\}$ VII	$\{q_0, q_1, q_2, q_3\}$ VIII
(V)	$\{q_0, q_3\}$	$\{q_0\}$ I	$\{q_0, q_1\}$ II
(VI)	$\{q_0, q_1, q_3\}$	$\{q_0, q_2\}$ III	$\{q_0, q_1, q_2\}$ IV
(VII)	$\{q_0, q_2, q_3\}$	$\{q_0, q_3\}$ V	$\{q_0, q_1, q_3\}$ VI
(VIII)	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_2, q_3\}$ VII	$\{q_0, q_1, q_2, q_3\}$ VIII

Out of 16 states corresponding to 16 subsets, only 8 are reachable from the initial state. Therefore, 8 states in the final FA.



Equivalent FA

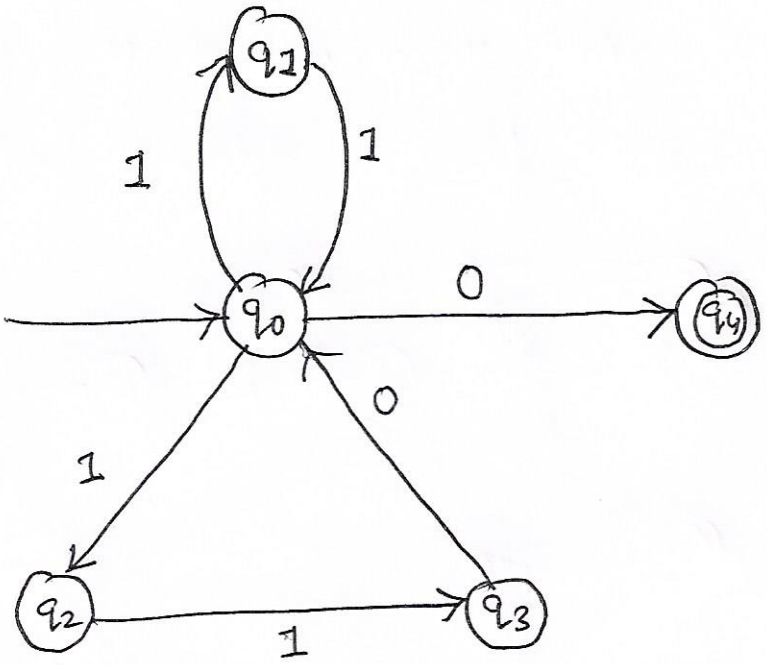
EX. 2 NFA to FA Conversion, using subset construction algorithm

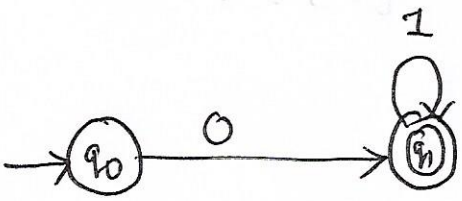
NFA (S.T. Table)

q	δ(q,0)	δ(q,1)
q_0	q_4	$\{q_1, q_2\}$
q_1	ϕ	$\{q_0\}$
q_2	ϕ	$\{q_3\}$
q_3	$\{q_0\}$	ϕ
q_4	ϕ	ϕ

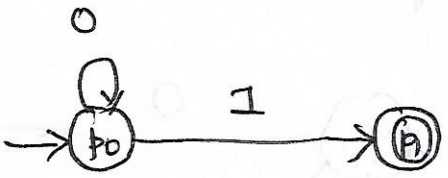
q	$\delta_1(q,0)$	$\delta_1(q,1)$
I $\{q_0\}$	$\{q_4\}$ II	$\{q_1, q_2\}$ III
II $\{q_4\}$	ϕ IV	ϕ IV
III $\{q_1 q_2\}$	ϕ IV	$\{q_0, q_3\}$ V
IV ϕ	ϕ IV	ϕ IV
V $\{q_0 q_3\}$	$\{q_0, q_4\}$ VI	$\{q_1, q_2\}$ III
VI $\{q_0, q_4\}$	$\{q_4\}$ II	$\{q_1, q_2\}$ III

State Transition Diagram of NFA of Ex. 2



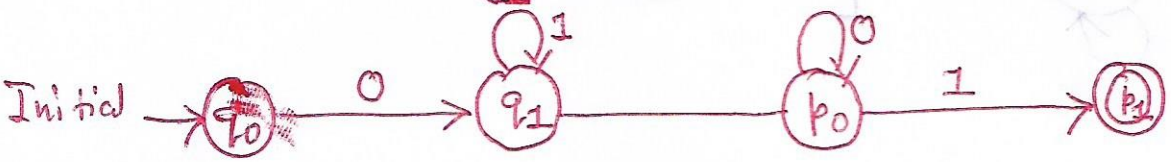


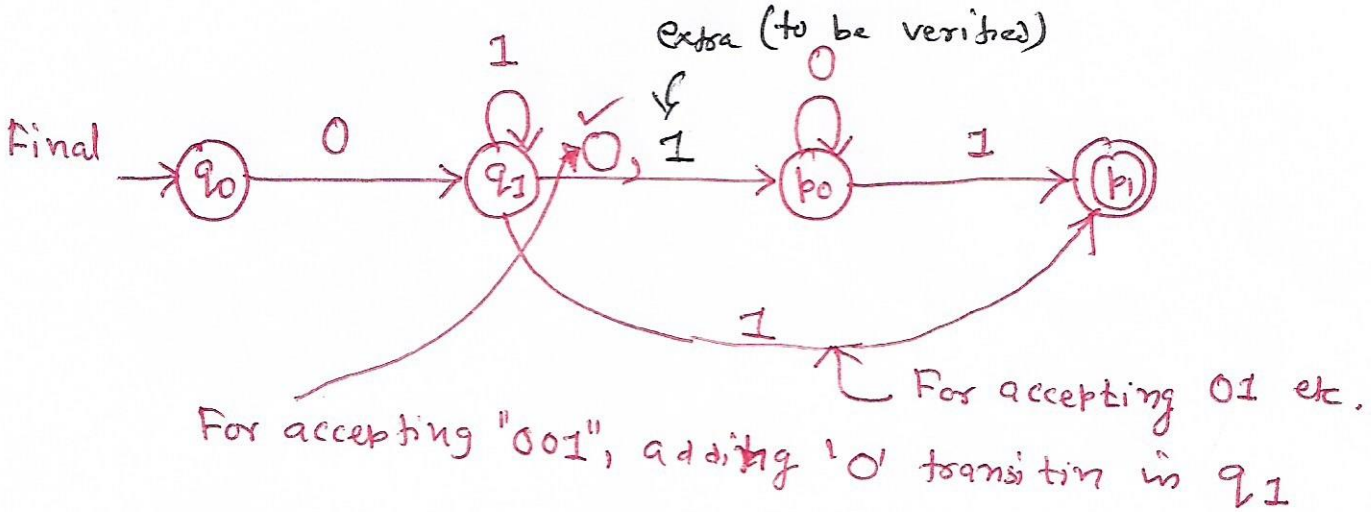
(a) An NFA accepting $\{0\}\{1\}^*$



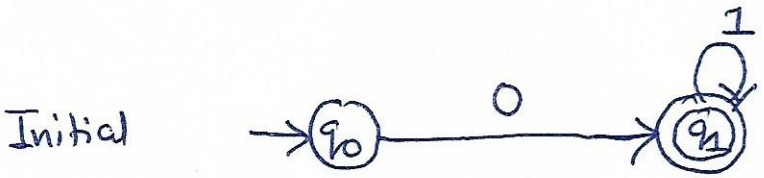
(b) An NFA accepting $\{0\}^*\{1\}$

(c) NFA for $\{0\}\{1\}^*\{0\}^*\{1\}$





(d) An NFA accepting $\{0\}\{1\}^* \cup \{0\}^*\{1\}$



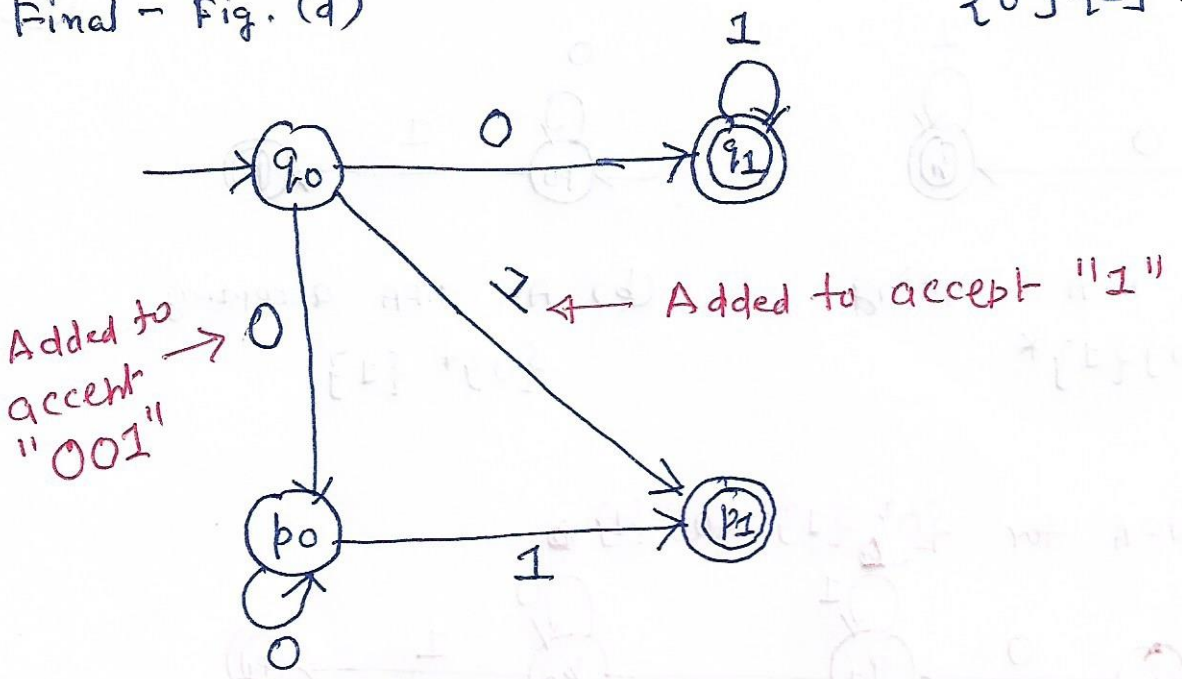
$\{0\}\{1\}^*$

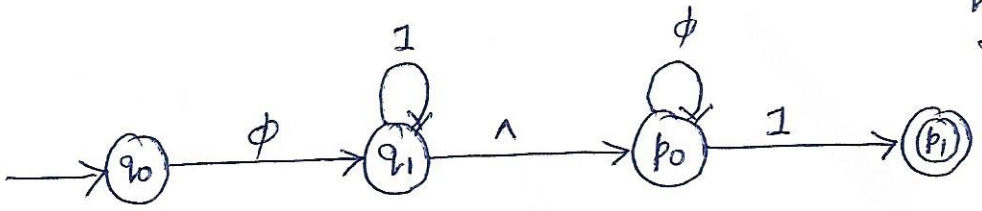
$\{0\}^*\{1\}$

Find - Fig. (d)

$\{0\}\{1\}^* \cup \{0\}^*\{1\}$

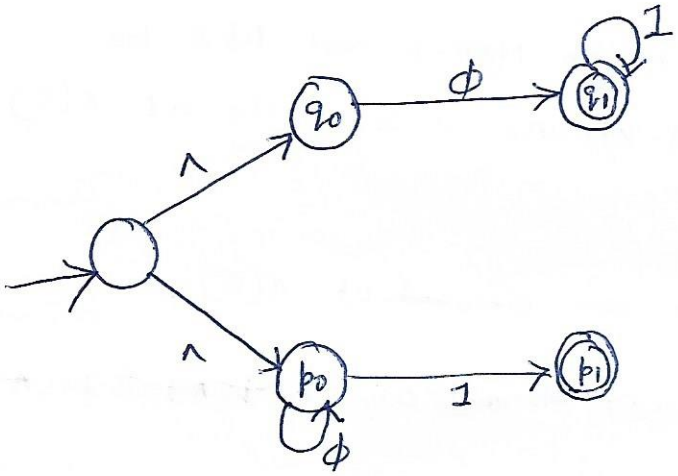
(8C)





→ M guesses in state q_1 that no more i/p for L_1 and proceeds to process L_2.

(e) An NFA-∧ accepting $\{0\}\{1\}^*\{\phi\}^*\{1\}$



→ Machine M makes guess and operates like M_1 or M_2

(f) An NFA-∧ accepting $\{0\}\{1\}^* \cup \{0\}^*\{1\}$

⇒ NFA-∧s are more general than NFAs as they are allowed to make transitions, not only on input symbols from the alphabet, but also on null inputs.

* NFA-∧ is a 5-tuple $(Q, \Sigma, q_0, A, \delta)$, where Q and Σ are finite ~~states~~ sets, $q_0 \in Q$, $A \subseteq Q$ and

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

Non-Recursive defⁿ of δ^* for NFA-Λ

Very similar to NFA.

Λ-closure of a set of states

Let $M = \langle Q, \Sigma, q_0, A, \delta \rangle$ be an NFA-Λ and let S be any subset of Q. The Λ-closure of S is the set $\Lambda(S)$ defined as follows:

1. Every element of S is an element of $\Lambda(S)$.
2. For any $q \in \Lambda(S)$, every element of $\delta(q, \Lambda)$ is in $\Lambda(S)$.
3. No other elements of Q are in $\Lambda(S)$.

Recursive defⁿ of δ^* for an NFA-Λ

Let $M = \langle Q, \Sigma, q_0, A, \delta \rangle$ be an NFA-Λ.

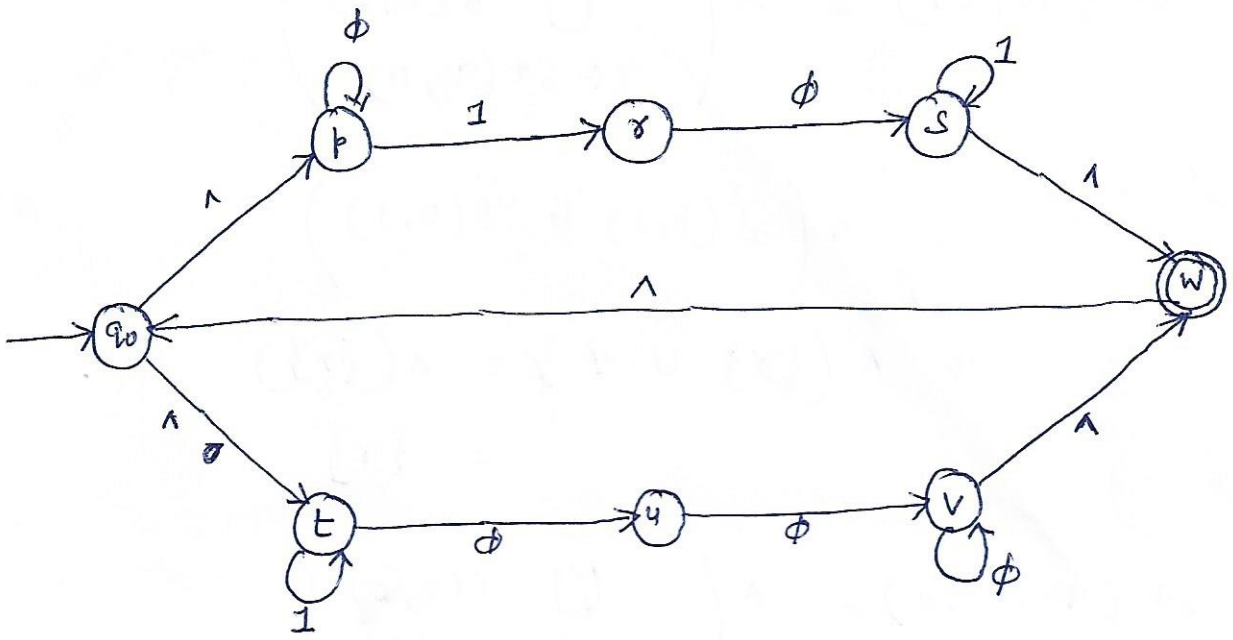
The $\delta^* : Q \times \Sigma^* \to 2^Q$ is defined as follows:

1. For any $q \in Q$, $\delta^*(q, \Lambda) = \Lambda(\{q\})$
2. For any $q \in Q$, $y \in \Sigma^*$ and $a \in \Sigma$,

$$\delta^*(q, ya) = \Lambda\left(\bigcup_{r \in \delta^*(q,y)} \delta(r, a) \right)$$

→ A string x is accepted by M if $\delta^*(q_0, x) \cap A \neq \phi$

Can we write $\Lambda(A)$ here?
No. Because, it could lead to



$$\delta^*(q_0, \wedge) = \wedge(\{q_0\})$$

$$= \{q_0, p, t\}$$

$$\delta^*(q_0, 0) = \wedge\left(\bigcup_{z \in \delta^*(q_0, \wedge)} \delta(z, 0) \right)$$

$$= \wedge\left(\bigcup_{z \in \{q_0, p, t\}} \delta(z_0, 0) \right)$$

$$= \wedge\left(\delta(q_0, 0) \cup \delta(p, 0) \cup \delta(t, 0) \right)$$

$$= \wedge\left(\phi \cup \{p\} \cup \{u\} \right)$$

$$= \wedge\left(\{p, u\} \right)$$

$$= \{p, u\}$$

$$\delta^*(q_0, 01) = \wedge\left(\bigcup_{z \in \delta^*(q_0,0)} \delta(z,1)\right)$$

$$= \wedge\left(\delta(p,1) \cup \delta(u,1)\right)$$

$$= \wedge\left(\{r\} \cup \phi\right) = \wedge(\{r\})$$

$$= \{r\}$$

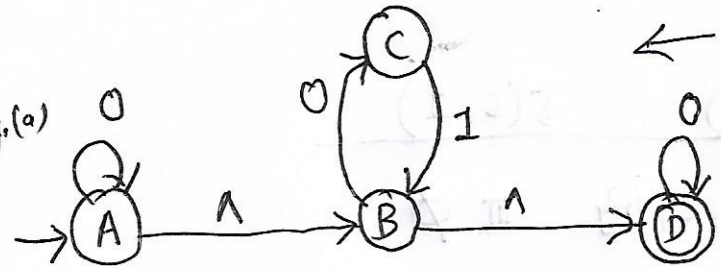
$$\delta^*(q_0, 010) = \wedge\left(\bigcup_{z \in \delta^*(q_0,01)} \delta(z,0)\right)$$

$$= \wedge\left(\delta(r,0)\right)$$

$$= \wedge(\{s\}) = \{s, w, q_0, p, t\}$$

Converting NFA-Λ to an NFA

Fig.(a) ← NFA-Λ for $\{0\}^*\{01\}^*\{0\}^*$



Λ-closure def^n of δ^* applied here ↓

Q	$\delta(q,\Lambda)$	$\delta(q,0)$	$\delta(q,1)$	$\delta^*(q,0)$	$\delta^*(q,1)$
A	$\{B\}$	$\{AB\}$	ϕ	$\{A,B,C,D\}$	ϕ
B	$\{D\}$	$\{C\}$	ϕ	$\{C,D\}$	ϕ
C	ϕ	ϕ	$\{B\}$	ϕ	$\{B,D\}$
D	ϕ	$\{D\}$	ϕ	$\{D\}$	ϕ

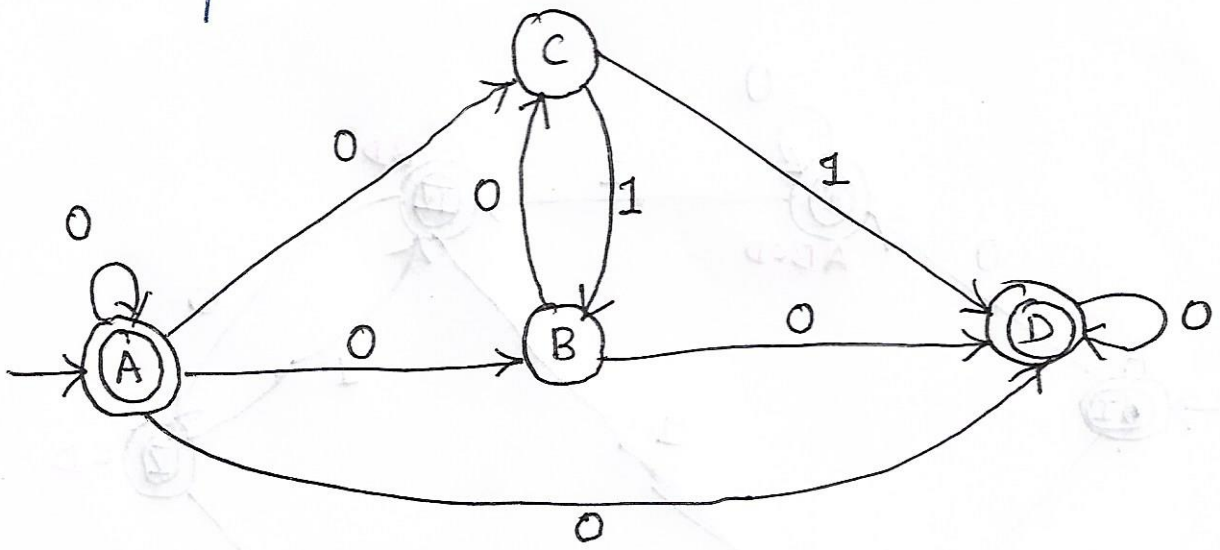
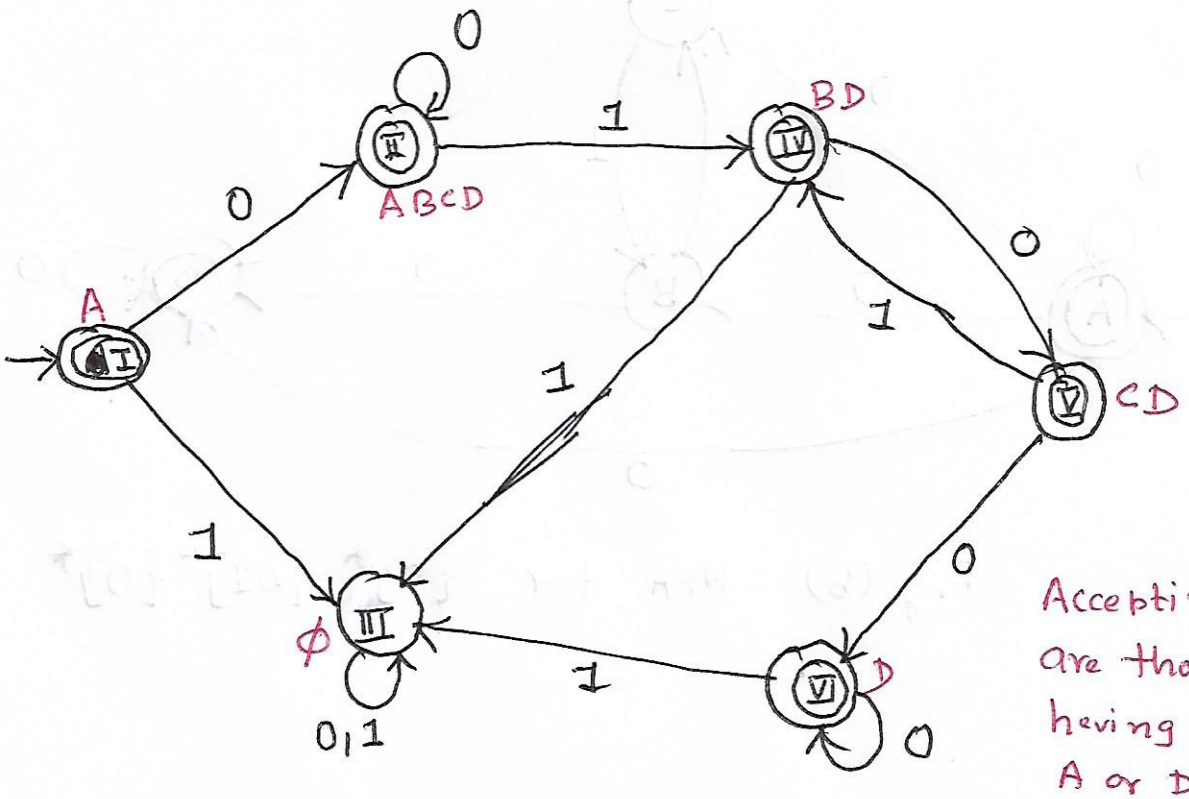


Fig. (b) NFA for $\{0\}^*\{01\}^*\{0\}^*$

Subset S	δ(S, 0)	δ(S, 1)
I {A}	II {A, B, C, D}	III ϕ
II {A, B, C, D}	II {A, B, C, D}	IV {B, D}
III ϕ	III ϕ	III ϕ
IV {B, D}	V {C, D}	III ϕ
V {C, D}	VI {D}	IV {B, D}
VI {D}	VI {D}	III ϕ



Accepting states are those having either A or D in the subset

An NFA for {0}* {01}* {0}*

Another example for converting NFA-$\wedge$ to NFA

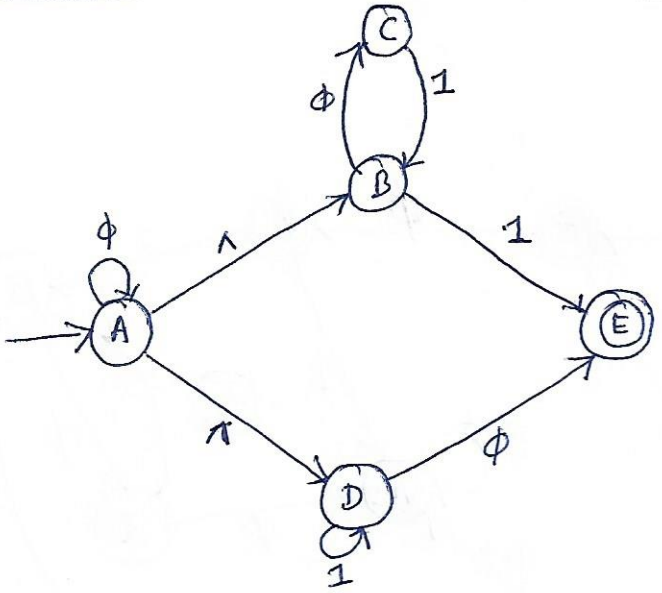


Fig.(a)

NFA-$\wedge$ for

$\{\phi\}^* \{\phi 1\}^* \{1\}$

$\cup \{1\}^* \{0\}$

$\{0\}^*$

010 is accepted but not belong to the language if we omit.

q	$\delta(q, \wedge)$	$\delta(q, 0)$	$\delta(q, 1)$	$\delta^*(q, 0)$	$\delta^*(q, 1)$
A	{B, D}	{A}	ϕ	{A, B, C, D, E}	{D, E}
B	ϕ	{C}	{E}	{C}	{E}
C	ϕ	ϕ	{B}	ϕ	{B}
D	ϕ	{E}	{D}	{E}	{D}
E	ϕ	ϕ	ϕ	ϕ	ϕ

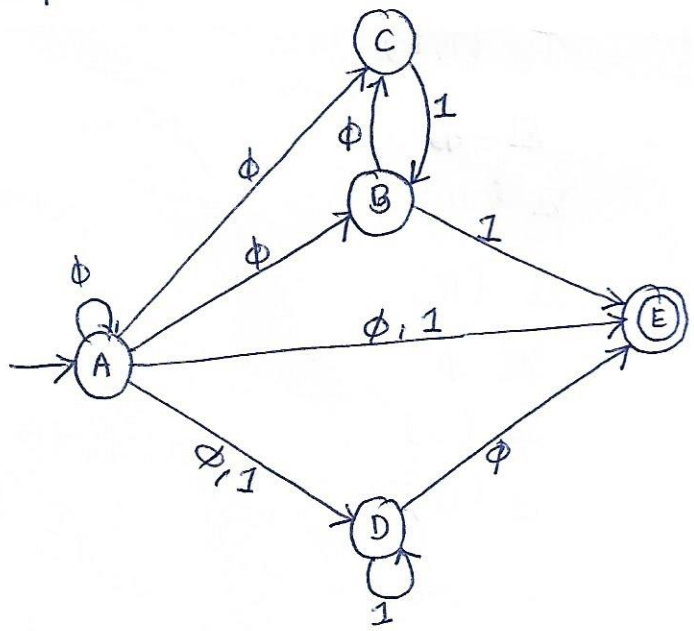
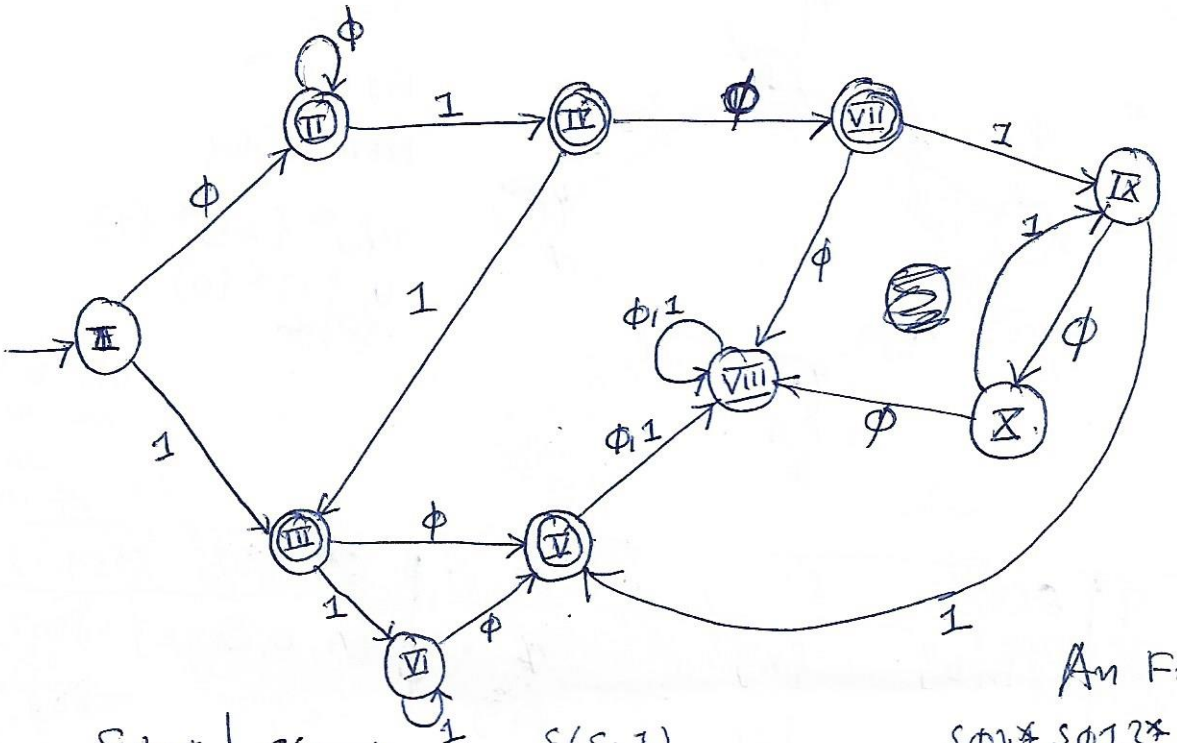


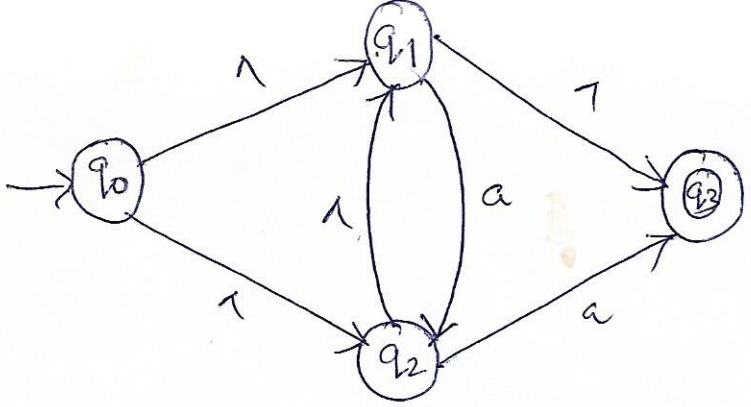
Fig. (b)

NFA For above Lang.



An FA for
$\{\phi\}^*\{\phi 1\}^*\{1\} \cup \{\phi\}^*\{1\}^*\{\phi\}$

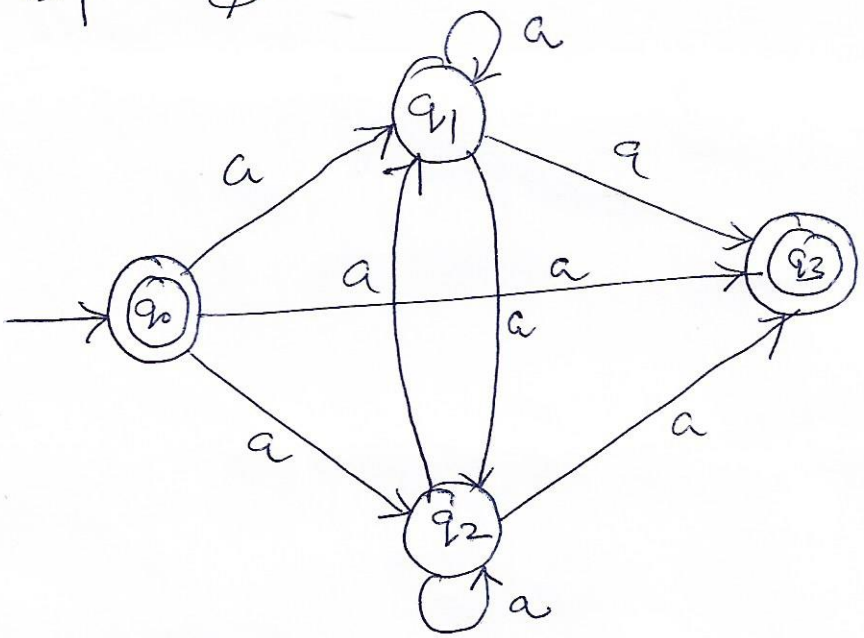
	Subsets S	δ(S, 0)	δ(S, 1)
I	{A}	II {A, B, C, D, E}	III {D, E}
(II)	{A, B, C, D, E}	II {A, B, C, D, E}	IV {B, D, E}
(III)	{D, E}	V {E}	VI {D}
(IV)	{B, D, E}	VII {C, E}	III {D, E}
(V)	{E}	VIII ϕ	VIII ϕ
VI	{D}	V {E}	VI {D}
(VII)	{C, E}	VIII ϕ	IX {B}
VIII	ϕ	VIII ϕ	VIII ϕ
IX	{B}	X {C}	V {E}
X	{C}	VIII ϕ	IX {B}



{RE is q^*}

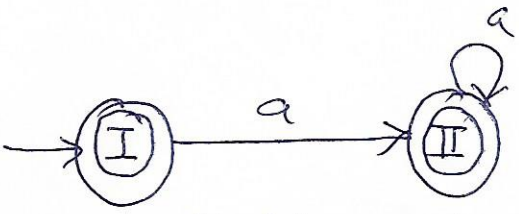
q	$\delta^*(q,a)$
q_0	$\{q_1, q_2, q_3\}$
q_1	$\{q_1, q_2, q_3\}$
q_2	$\{q_1, q_2, q_3\}$
q_3	ϕ

NFA



DFA

	S	$\delta(S,a)$	
I	$\{q_0\}$	$\{q_1, q_2, q_3\}$	II
II	$\{q_1, q_2, q_3\}$	$\{q_1, q_2, q_3\}$	II



⇓ Minimized

Kleene's Theorem – Part I

Any Regular language can be accepted by a finite automaton.

We will prove this using the structural induction.

(~~The~~ Reg. lang. are closed under union, concatenation and Kleene's *)

The basic step of the proof is to show that the three basic languages can be accepted by NFA-Λs.

i.e. ϕ, $\{\Lambda\}$, $\{a\}$ for $a \in \Sigma$.

NFA-Λ's (a) ϕ (b) $\{\Lambda\}$ (c). $\{a\}$

Induction Hypo.: L_1 and L_2 are languages that can be accepted by NFA-Λs.

Induction step: $L_1 \cup L_2$, $L_1 L_2$, L_1^* can be accepted by some NFA-Λs.

- Suppose that L_1 and L_2 are recognized by NFA-Λs M_1 and M_2 respectively, i.e. $M_i = (Q_i, \Sigma, q_i, A_i, \delta_i)$ for $i = 1$ & 2.
- By renaming states, if necessary, we assume that $Q_1 \cap Q_2 = \phi$.
- We will construct NFA-Λs $M_\cup$, M_c, M_k for recognizing the languages $L_1 \cup L_2$, $L_1 L_2$, L_1^* respectively.

Construction of $M_4 = (Q_4, \Sigma, q_4, A_4, \delta_4)$,

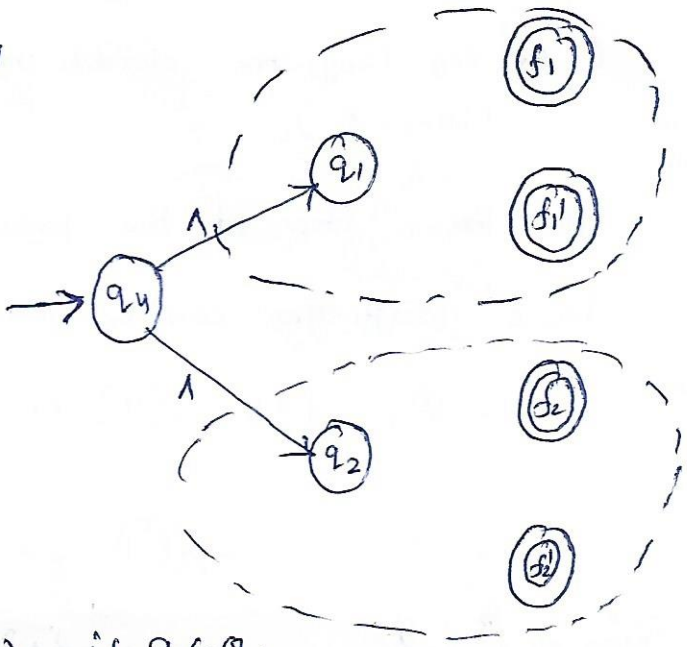
Let q_4 be a new state not in either Q_1 or Q_2.

$$Q_4 = Q_1 \cup Q_2 \cup \{q_4\}$$

$$A_4 = A_1 \cup A_2$$

$$\delta_4(q_4, \wedge) = \{q_1, q_2\}$$

$$\delta_4(q_4, a) = \phi \text{ for every } a \in \Sigma$$



And for each $q \in Q_1 \cup Q_2$

and $a \in \Sigma \cup \{\wedge\}$

{or $\alpha \in \Sigma \cup \{\wedge\}$}

$$\delta_4(q_4, a) = \begin{cases} \delta_1(q_4, a) = \alpha & \text{if } q \in Q_1 \\ \delta_2(q_4, a) = \alpha & \text{if } q \in Q_2 \end{cases}$$

$\{\alpha\}$

To prove that M_4 accepts $L_1 \cup L_2$

i) if $x \in L_1 \cup L_2$, M_4 accepts it.

M_4 moves to q_1 or q_2 from q_4 through $\wedge$-transition.

Then process x the way M_1 or M_2 would do.

Thus x would be accepted by M_4.

ii) If M_4 accepts x, then $x \in L_1 \cup L_2$

As M_4 accepts x, there is a sequence of transition for x from q_4 to one of the accepting states of either M_1 or M_2.

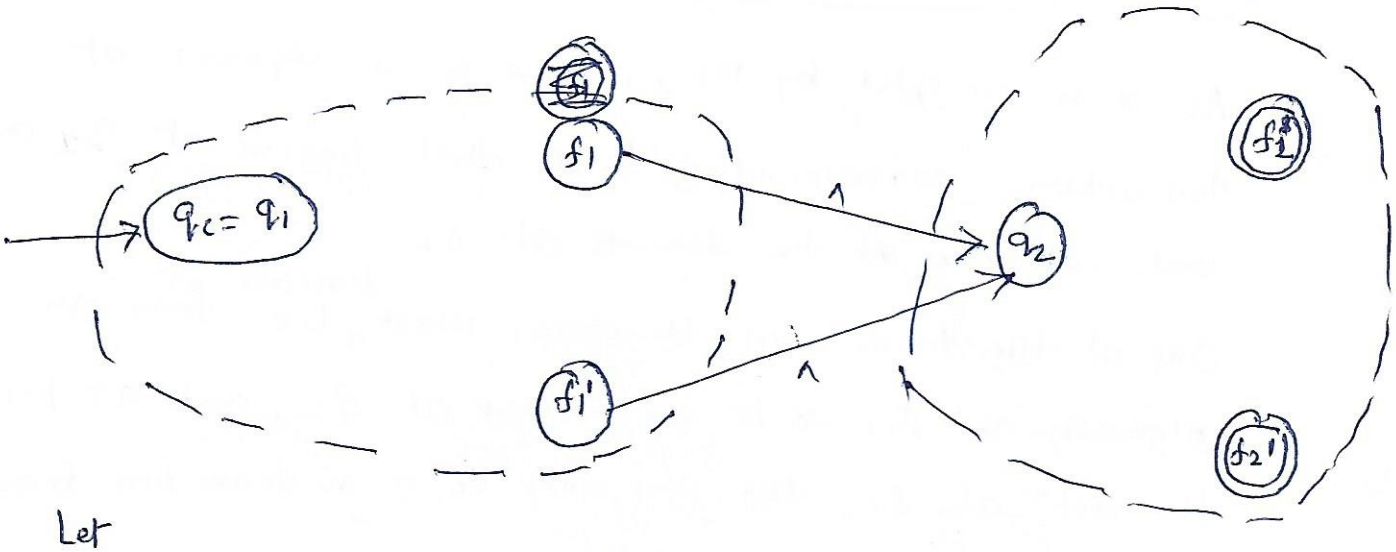
The first of transition must be to q_1 or q_2 from q_4 on $\wedge$ as no other transition is present from q_4.

Thereafter, all other transitions are from q_1 to elements of Q_1 or from q_2 to elements of Q_2, as $Q_1 \cap Q_2 = \phi$

Therefore x is accepted either by M_1 or M_2.

$\therefore x \in L_1 \cup L_2$

Construction of $M_c = (Q_c, \Sigma, q_c, A_c, \delta_c)$



Let

$Q_c = Q_1 \cup Q_2$

$q_c = q_1$

$A_c = A_2$

δ_c → or $\{q \in Q_1 \cup Q_2 - A_1\}$

For any q not in A_1 and $\alpha \in \Sigma \cup \{\Lambda\}$

Case I $q \in Q_1 \cup Q_2 - A_1$

i.e. $\delta_c(q, \alpha) = \begin{cases} \delta_1(q, \alpha) & \text{if } q \in Q_1 - A_1 \\ \delta_2(q, \alpha) & \text{if } q \in Q_2 \end{cases}$

Case II $q \in A_1$

For $q \in A_1$

$\delta_c(q, a) = \delta_1(q, a)$ for every $a \in \Sigma$

$\delta_c(q, \Lambda) = \delta_1(q, \Lambda) \cup \{q_2\}$

Proof

If $x \in L_1 L_2$, M_c would accept it

On i/p string $x_1 x_2$, where $x_i \in L_i$ for both values of i, M_c can process x_1, arriving at a state in A_1; jump from this state to q_2 by a Λ-transition; and

then process x_2 the way M_2 would, so that $x_1 x_2$ is accepted.

Conversely, if x is accepted by M_C then $x \in L_1 L_2$.

As x is accepted by M_C, there is a sequence of transitions corresponding to x that begins at q_1 and ends at one of the elements of A_2.

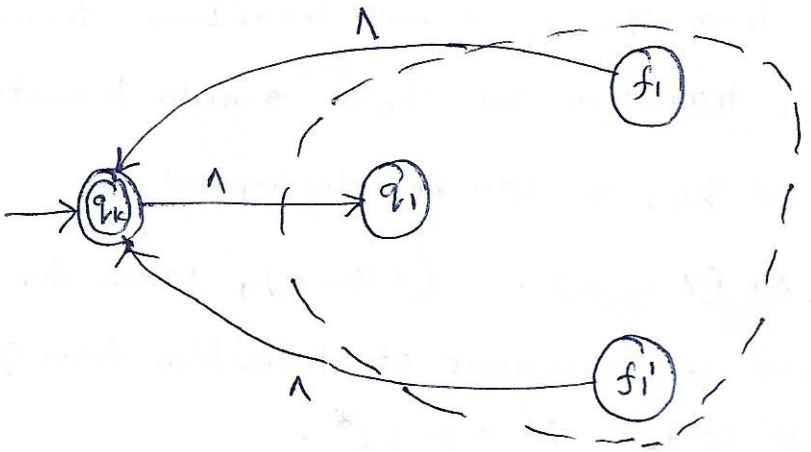
One of the transition, ~~therefore~~, must therefore be from an elements of Q_1 to an element of Q_2, and as per the def^n of δ_C, this can only be a $\wedge$-transition from an element of A_1 to q_2.

Because $Q_1 \cap Q_2 = \phi$, all the previous transitions are from q_1 to element of A_1 and all the subsequent transitions are from q_2 to any elements of A_2.

It follows that $x = x_1 \wedge x_2 = x_1 x_2$, where x_1 is accepted by M_1 and x_2 is accepted by M_2.

$\therefore \quad x \in L_1 L_2$

Construction of M_k :



$$M_k = (Q_k, \Sigma, q_k, A_k, \delta_k)$$

$$Q_k = Q_1 \cup \{q_k\}$$

$$A_k = \{q_k\}$$

$$\delta_k(q_k, \lambda) = \{q_1\}$$

$$\delta_k(q_k, a) = \phi \quad \text{for } a \in \Sigma$$

{Simple defn of δ_k}

(A) For $q \in Q_1 - A_1$ and $\alpha \in \Sigma \cup \{\lambda\}$

$$\delta_k(q, \alpha) = \delta_1(q, \alpha)$$

(B) For $q \in A_1$

$$\delta_k(q, a) = \delta_1(q, a) \quad \text{for } a \in \Sigma$$

$$\delta_k(q, \lambda) = \delta_1(q, \lambda) \cup \{q_k\}$$

Use (A) instead of this

For $q \in Q_1$ and $\alpha \in \Sigma \cup \{\lambda\}$,

$$\delta_k(q, \alpha) = \delta_1(q, \alpha)$$ except when $q \in A_1$ and α is λ.

(B)

For $q \in A_1$ and α is λ

$$\delta_k(q, \lambda) = \delta_1(q, \lambda) \cup \{q_k\}$$

To Prove : If $x \in L_1^*$, then M_k accepts it.

Proof : If x is λ, Then it is accepted by M_k.

Otherwise for $m \geq 1$, $x = x_1 x_2 \ldots x_m$ where each $x_i \in L_1$

M_k then makes a null-transition from q_k to q_1 and then process x_i like M_1 and take a λ-transition from an element of A_1 to q_k.

$\therefore$ $x = (\lambda x_1 \lambda)(\lambda x_2 \lambda) \ldots (\lambda x_m \lambda)$ is accepted by M_k (where each $x_i \in L_1$)

Converse, if x is accepted by M_k, then $x \in L_1^*$.

Because x is accepted by M_k, there is a sequence of transition corresponding to x that begins and ends at q_k.

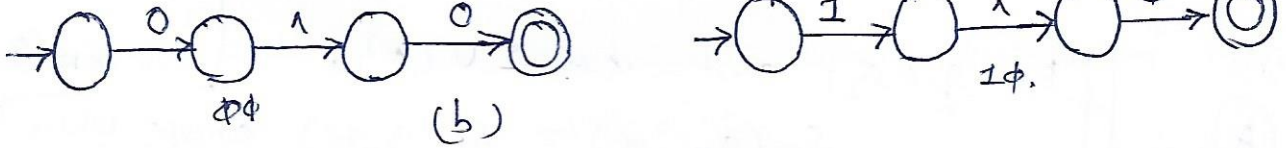
Since, only transition from q_k is a null-transition from q_k to q_1 and only transition to q_k are null-transitions from elements of A_1 to q_k, x can be decomposed as:

$x = (\wedge x_1 \wedge)(\wedge x_2 \wedge) \cdots (\wedge x_m \wedge)$, where for each x_i, there is a sequence of transition from q_1 to an element of A_1. $\therefore x \in L_1^*$.

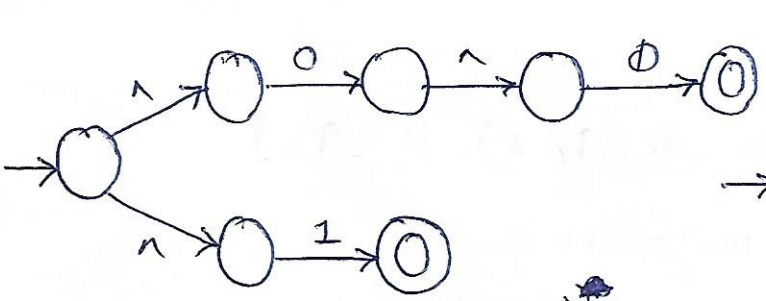
Constructing an NFA-$\wedge$ for $(00+1)^*(10)^*$.

(Using the Algo.)

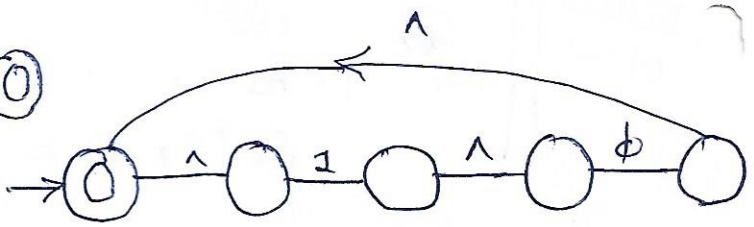
(a)



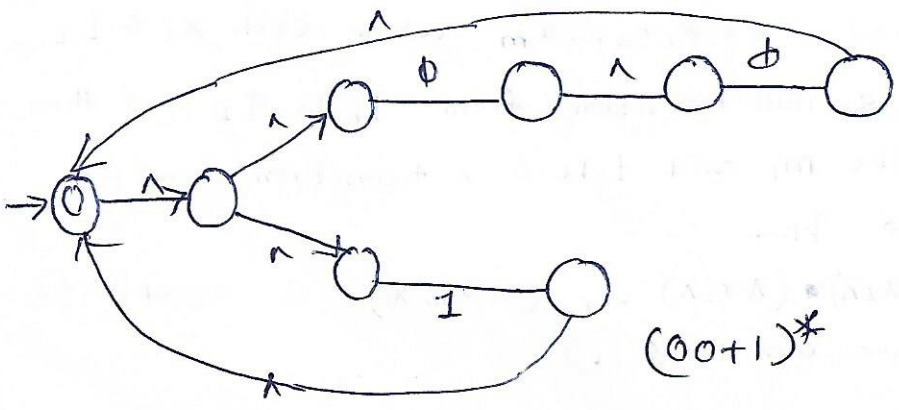
(b)



(c) $(\phi\phi+1)^*$



$(1\phi)^*$



$(00+1)^*$