REGULAR EXPRESSIONS

— The Second Model for Defining Languages

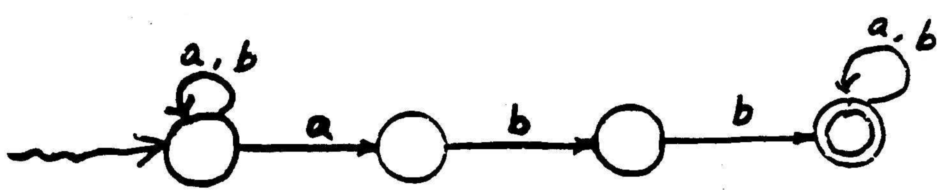
Example:

Consider the language of all the words that consist of a's and b's and have abb as a subword.

We can formally define this language by the following:

(1) $L = \{x \mid x \in \{a, b\}^*$ **and** x **has** abb **as a subword**$\}$**;**

(2) $L = L(M)$ **where** M **is an NFA given by the following diagram:**



Both of the above definitions are lengthy. It can also be expressed by

$$L([a \cup b]^* abb[a \cup b]^*)$$

<u>Definition</u> Let Σ be an alphabet.
A <u>regular expression</u> over Σ is defined recursively:

Basis: **(1)** $\emptyset$,
(2) λ,
(3) a, **where** $a \in \Sigma$
are R.E.'s over Σ

Induction Step:
If E_1 **and** E_2 **are R.E.'s over** Σ**, then**
(4) $[E_1 \cup E_2]$,
(5) $[E_1 \cdot E_2]$,
(6) E_1^*
are R.E.'s over Σ

We usually omit $\cdot$**.**

The set of all regular expressions over Σ **is denoted by**

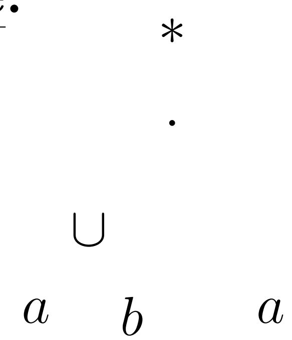
$$\mathcal{R}_\Sigma$$

$[[a+b]a]^*$ **is the same as** $[[a \cup b]a]^*$

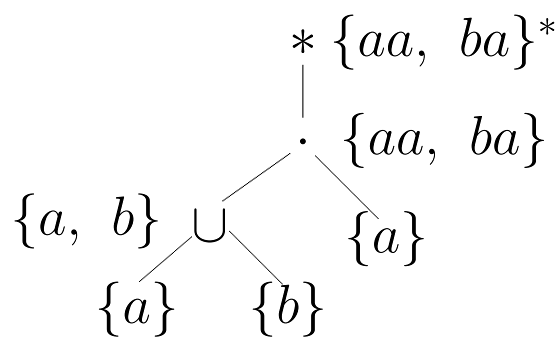
Example:

$$[\underbrace{\underbrace{[\overbrace{a}^{E_1} \cup \overbrace{b}^{E_2}]}_{E_3}\ \underbrace{a}_{E_4}}_{E_5}]^* \textbf{ is a R.E. over } \{a,\ b\}$$

We display the "parsing" by an expression tree:



What language does an R.E. define?



<u>Definition</u> Given a regular expression E, the language $L(E)$ denoted by E is defined as follows:

<u>Basis</u>: (1) if $E = \emptyset$, then $\emptyset$;

(2) if $E = \lambda$, then $\{\lambda\}$;

(3) if $E = a$, then $\{a\}$;

<u>Induction Step</u>:

(4) if $E = [E_1 \cup E_2]$, then $L(E_1) \cup L(E_2)$;

(5) if $E = [E_1 E_2]$, then $L(E_1)L(E_2)$;

(6) if $E = E_1^*$, then $(L(E_1))^*$.

<u>Properties of R.E.'s</u>

$\cup$**:** $E_1 \cup E_2 \equiv E_2 \cup E_1$;

$[E_1 \cup E_2] \cup E_3 \equiv E_1 \cup [E_2 \cup E_3]$;

So, $[E_1 \cup E_2] \cup E_3 \Rightarrow E_1 \cup E_2 \cup E_3$.

$\cdot$**:** $[E_1 E_2]E_3 \equiv E_1[E_2 E_3]$;

So, $[E_1 E_2]E_3 \Rightarrow E_1 E_2 E_3$.

. . . .

Definition **A language** L **is** **regular** **iff there is a regular expression** E **such that** $L = L(E)$**.**

The family of (all) regular languages is denoted by $\mathcal{L}_{REG}$**.**

Example $E = [b^*ab^*a]^*b^*$**.**
$L_{even} = \{x \mid x$ **in** $\{a, b\}^*$ **and** x **contains an even number of** $a's\}$**.**

Claim: $L(E) = L_{even}$

Proof:

(i) $L(E) \subseteq L_{even}$**, since every word in** $L(E)$ **contains an even number of** a**'s.**

(ii)Let $x \in L_{even}$**. Then** x **can be written as** $b^{i_0}ab^{i_1}a \ldots\ldots ab^{i_{2n}},\ i_0, i_1, \ldots, i_{2n} \geq 0$**.**
So, $x = (b^{i_0}ab^{i_1}a)(b^{i_2}ab^{i_3}a) \ldots\ldots (b^{i_{2n-2}}ab^{i_{2n-1}}a)b^{i_{2n}}$**.**
$x \in L([b^*ab^*a]^*)L(b^*) \ = \ L(E)$**.** *q.e.d.*

Examples $\Sigma = \{a,\ b\}$**.**

$L_1 = \{x \mid x = au,\ u \in \Sigma^*\}$**.**

$L_2 = \{x \mid |x|_a \equiv 0$ **mod** $3\}$**.**

$[a[aa]]^*$

$L_3 = \{x \mid x$ **has 2 or 3** $a's$ **with the last two appearances nonconsecutive** $\}$

$L_4 = \{x \mid x = a^n b^n,\ n \geq 1\}$

Examples

What are the languages denoted by the following R.E.'s ?

$$E_1 = a^*ba^*$$

$$E_2 = [a \cup ab]^*$$

$$E_3 = a[a \cup b]^*a$$

$$E_4 = [aa \cup bb \cup ba \cup ab]^*$$

How many languages over Σ do R.E.'s define?

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

(1) Infinitely many ?

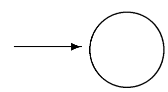
(2) Countable ?

Regular Expression into Finite Automata

Let E be a regular expression over Σ. Then we can construct a λ-NFA M such that $L(M) = L(E)$, using the following rules:

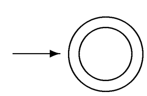
(i) $E = \emptyset$**.**

Construct M **such that** $L(M) = \emptyset$



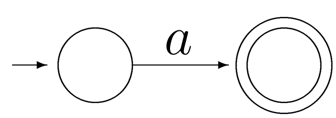
(ii) $E = \lambda$**.**

Construct M **such that** $L(M) = \{\lambda\}$

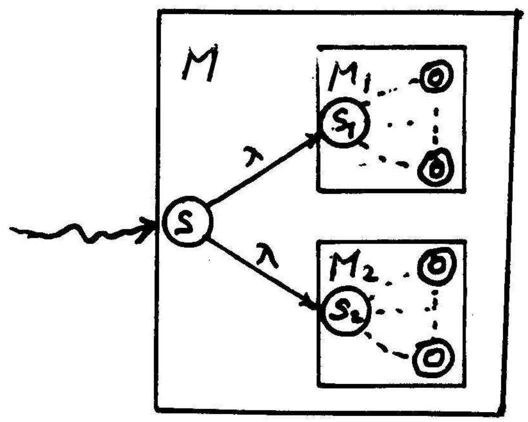


(iii) $E = a$**,** $a \in \Sigma$**.**

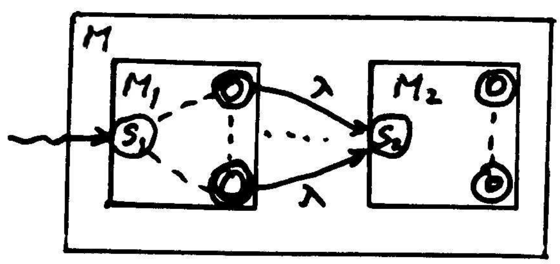
Construct M **such that** $L(M) = \{a\}$



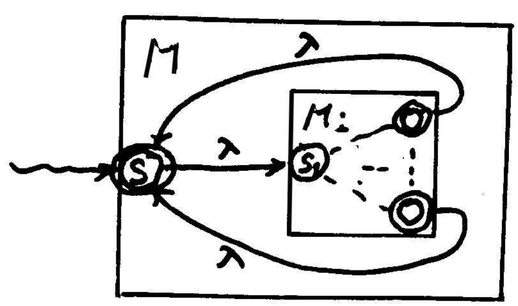
(iv) $E = [E_1 \cup E_2]$.
Construct M **such that** $L(M) = L(M_1) \cup L(M_2)$.



(v) $E = [E_1 E_2]$.
Construct M **such that** $L(M) = L(M_1)L(M_2)$.



(vi)$E = E_1^*$.
Construct M **such that** $L(M) = L(M_1)^*$.



Example

$E = [c^*[a \cup [bc^*]]^*]$

Construct a FA M such that $L(M) = L(E)$.

$\triangle$ $a, \quad b, \quad c$ **by (iii)**

$\triangle$ c^* **by (vi)**

$\triangle$ $[bc^*]$ **by (v)**

$\triangle$ $[a \cup bc^*]$ **by (iv)**

$\triangle$ $[a \cup bc^*]^*$ **by (vi)**

$\triangle$ $[c^*[a \cup bc^*]^*]$ **by (v)**

Theorem For E, an arbitrary regular expression over Σ, the λ-NFA, M, constructed as above satisfies $L(M) = L(E)$.

Proof: Let $Op(E)$ be the total number of $\cup$, $\cdot$, and * operations in E. We prove this theorem by induction on $Op(E)$.

Basis: $Op(E) = 0$. Then $E = \emptyset$, λ, or $a \in \Sigma$. Then clearly we have $L(M) = L(E)$.

Induction Hypothesis:
Assume the claim holds for all E with $Op(E) \leq k$, for some $k \geq 0$.

Induction Step:
Consider an arbitrary regular expression E with $Op(E) = k+1$. Since $k+1 \geq 1$, E contains at least one operator $\cup$, $\cdot$, or *.

Case I: $E = E_1 \cup E_2$. Then $Op(E_1) \leq k$ and $Op(E_2) \leq k$. So, $L(M_1) = L(E_1)$ and $L(M_2) = L(E_2)$ by I.H.. We know the construction of $M = M_1 \cup M_2$ satisfies $L(M) = L(M_1) \cup L(M_2)$, and $L(E) = L(E_1) \cup L(E_2)$

Therefore, $L(M) = L(E)$**.**

<u>Case II</u>: $E = E_1E_2$

. . . . .

<u>Case III</u>: $E = E_1^*$

. . . . .

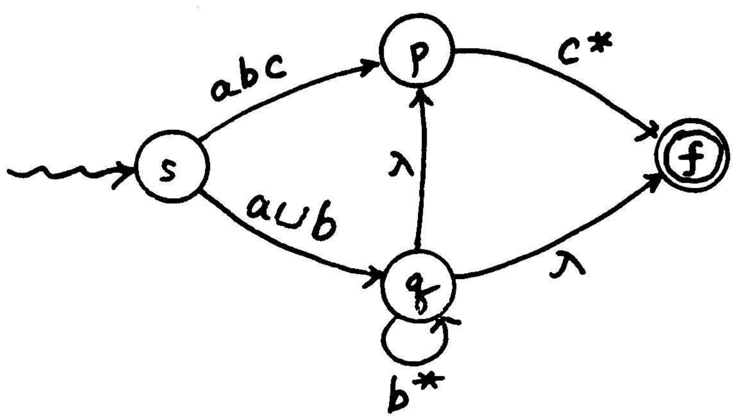
In each of the three cases, we have shown that $L(M) = L(E)$**. Therefore this holds for all regular expressions by the principle of induction.** *q.e.d.*

Finite Automata into Regular Expressions

To prove that every DFA language is regular we introduce an extension of finite automata.

Definition An extended finite automaton (EFA), M, is a quintuple $(Q,\ \Sigma,\ \delta,\ s,\ f)$ where
$Q,\ \Sigma,\ s$ are as in λ-NFA,
f is the only final state, $f \neq s$,
$\delta : Q \times Q \to R_\Sigma$ is a total extended transition function.

Example of an EFA:



$\delta(p, s) = \emptyset$
$\delta(s, f) = \emptyset$
. . . . . .
One final state $f \neq s$

△ **A configuration is in** $Q\Sigma^*$

△ **Move**

$$px \ \vdash \ qy \text{ **if**}$$

(i) $x = wy, \ w \in \Sigma^*$,

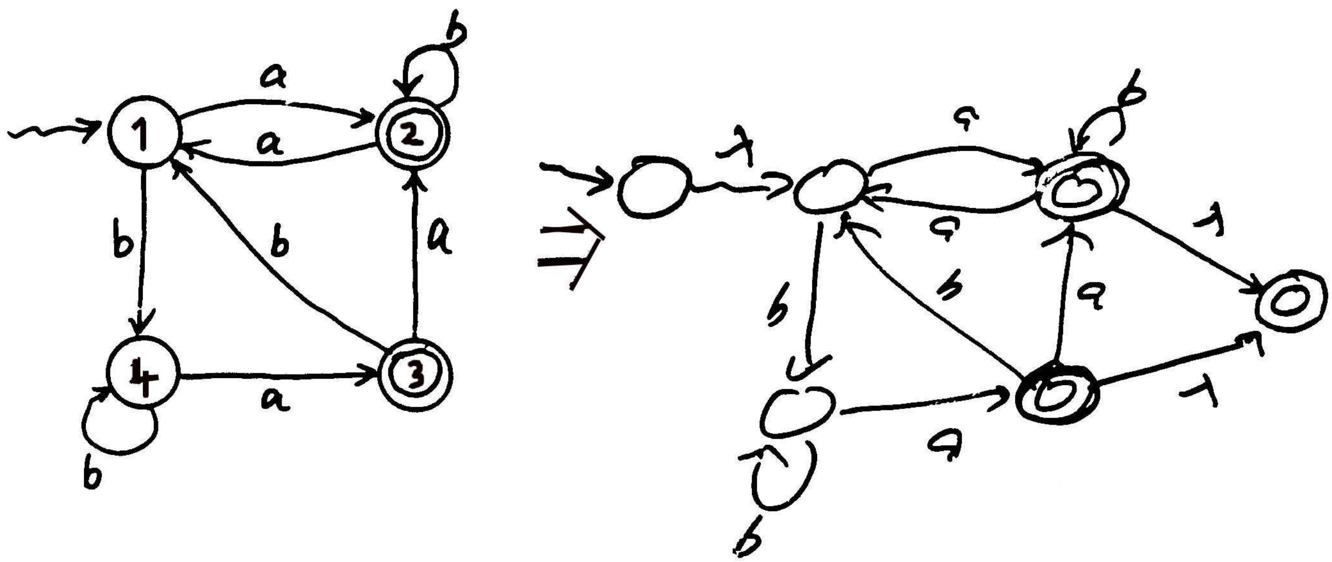
(ii) $\delta(p, q) = E$, **and**

(iii) $w \in L(E)$.

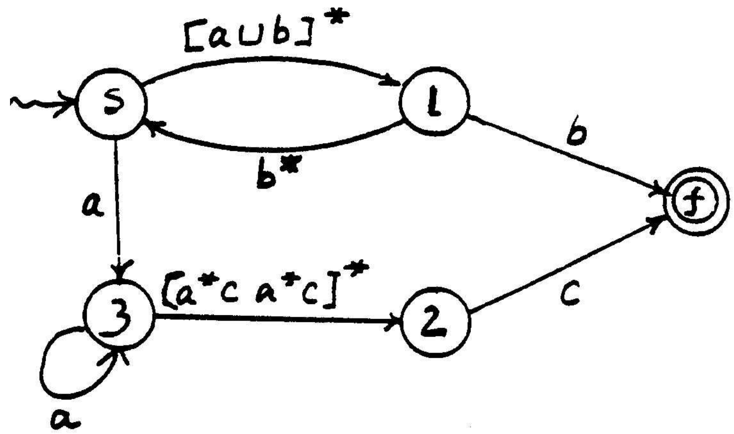
△ $\vdash^*$, $\vdash^+$ **are defined similarly as before.**

Lemma If M **is a DFA, Then there is an EFA** M' **with** $L(M') = L(M)$**.**

Example DFA into EFA.



<u>Example</u>:
An extended finite Automaton (EFA).
M:



Check if the following words are in $L(M)$

(1) $bbabab$

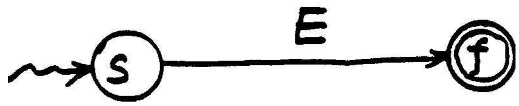
(2) $aabbc$

(3) $acccc$

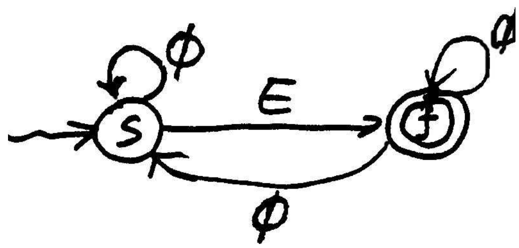
(4) $aaaaac$

State Elimination Technique

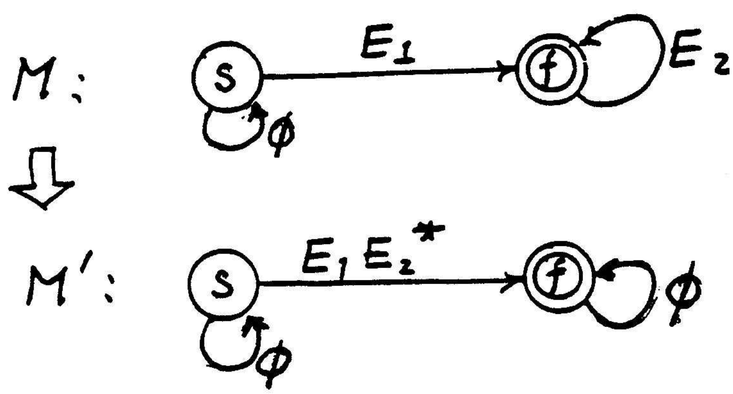
Goal of the technique:



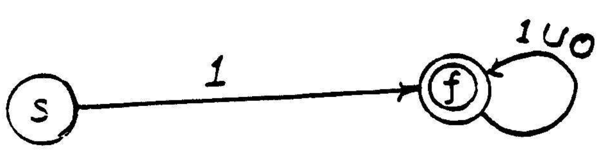
i.e.:



(1) EFA has 2 states

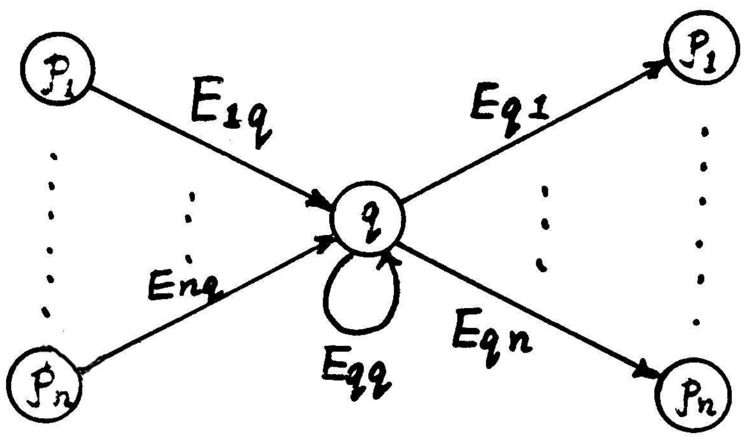


Example



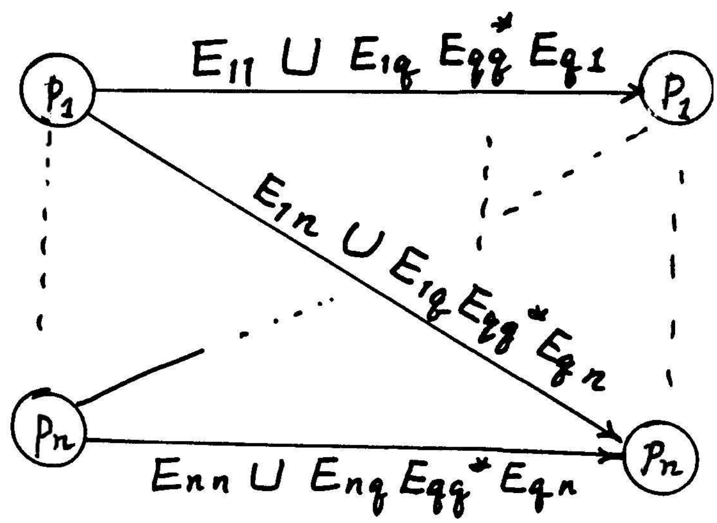
(2) EFA M has $k+1$ states, $k \geq 2$.
Then eliminate a state from M to form M':

M :



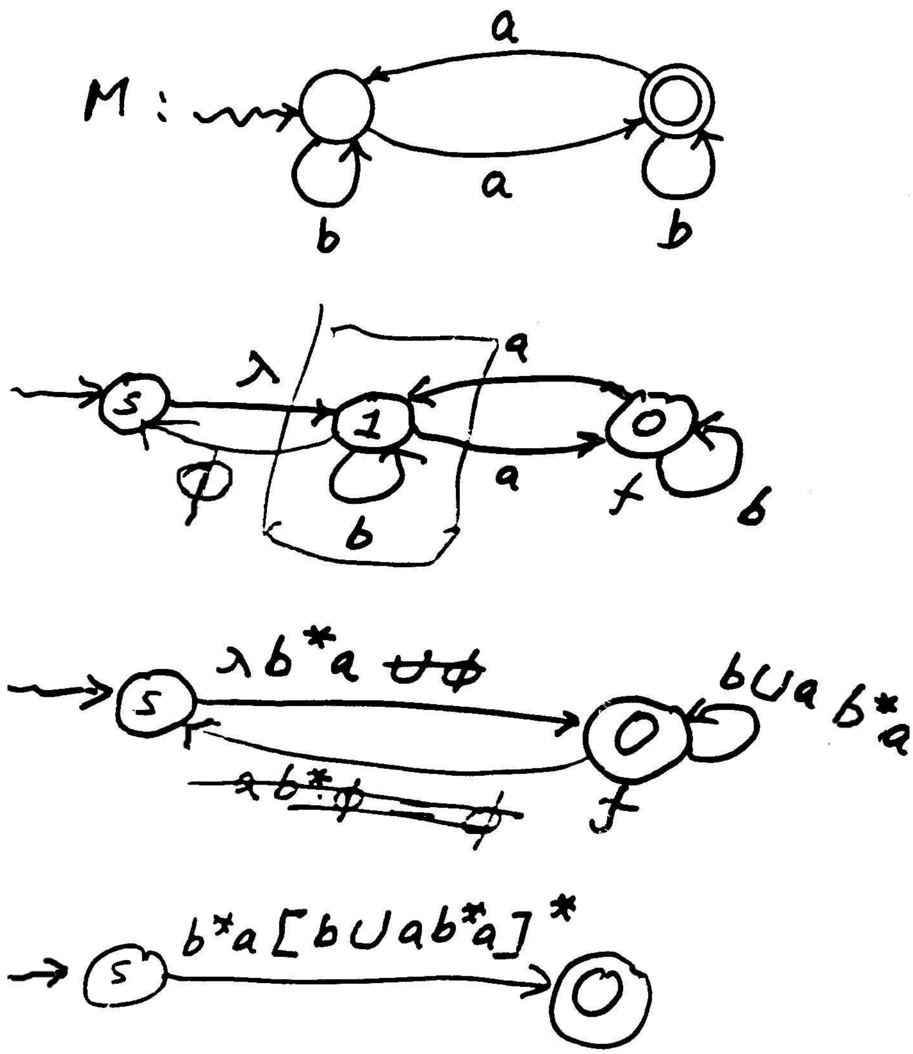
Note: $q \in Q - \{s,\ f\}$
Consider all transitions $(p_i,\ E_{iq},\ q)$
and $(q,\ E_{qi},\ p_i)$

M' :



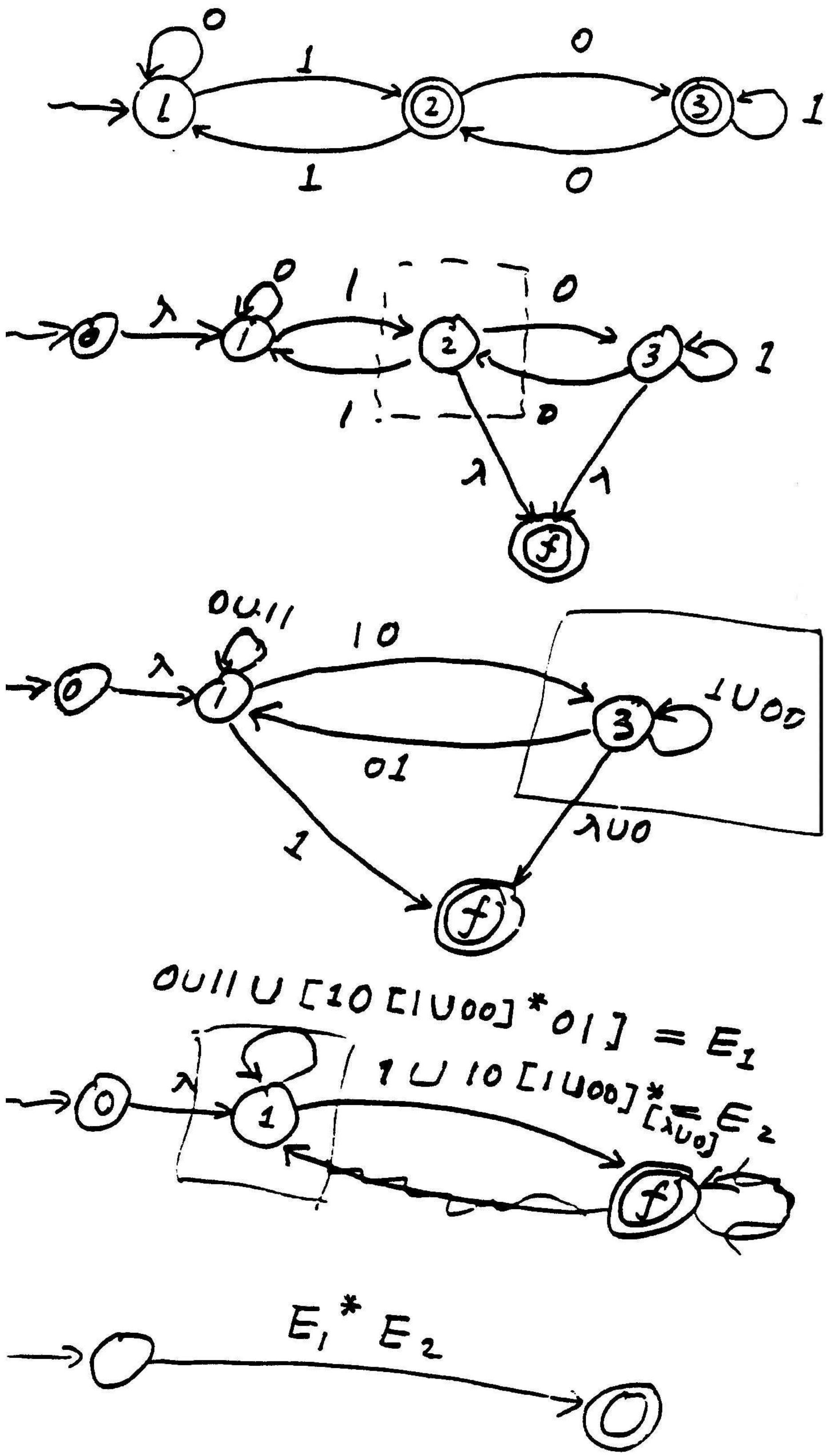
$$\delta'(p_i, p_j) = \delta(p_i, p_j) \cup \delta(p_i, q)(\delta(q, q))^* \delta(q, p_j)$$

Example



$b^*a[b \cup ab^*a]^*$

Example



Summary of the State Elimination Technique

(0) Change FA into EFA

(1) Add a new start state if the original one has incoming transitions.

(2) Add a new final state if there are more than one final states originally. Old final states become non-final states.

(3) Eliminate the states in $Q - \{s,\ f\}$ one by one.

(4) Eliminate the transition $\delta(f, f)$.