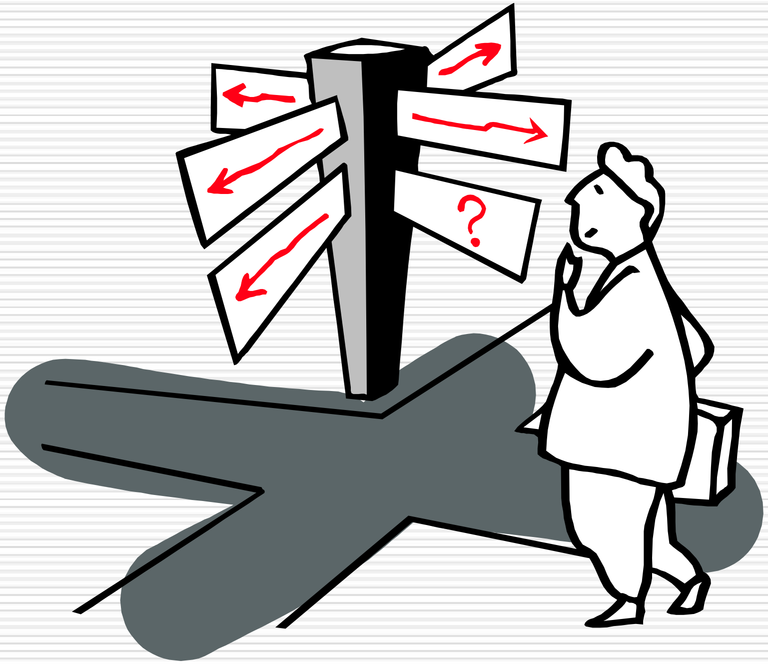
1.2 Regular Expressions

- Regular Expression
- RE → ε-NFA
- DFA → RE



Regular Expressions

- A FA is a “blueprint” for constructing a machine recognizing a regular language.
- A regular expression is a “user-friendly” declarative way of describing a language.
- Example: 01*+10*
- Used in e.g.
  - UNIX grep
  - Perl programming language

Inductive Definition of RE

- R is a regular expression if R is
  1. Ø, the empty set
  2. ε, the empty string
  3. a, for some $a \in \Sigma$
  4. R_1+R_2, where R_1 and R_2 are reg. exp.
  5. R_1R_2, where R_1 and R_2 are reg. exp.
  6. R_1*, where R_1 is a regular expression

Language of a RE

- A regular expression R describes the language L(R)
  - $L(\emptyset) = \emptyset$
  - $L(\varepsilon) = \{\varepsilon\}$
  - $L(a) = \{a\}$
  - $L(R_1+R_2) = L(R_1) \cup L(R_2)$ ← union
  - $L(R_1R_2) = L(R_1)L(R_2)$ ← concatenation
  - $L(R_1^*) = (L(R_1))^*$ ← closure

Concatenation of Languages

- If L_1 and L_2 are languages, we can define the concatenation

  $L_1L_2 = \{w \mid w=xy, x \in L_1, y \in L_2\}$
- Examples:
  - $\{ab, ba\}\{cd, dc\} =? \{abcd, abdc, bacd, badc\}$
  - $\emptyset\{ab\} =? \emptyset$

Exponentiation of a Language

- L^i is the language L concatenated with itself i times.
- Recursive definition:
  - Base: $L^0 = \{\varepsilon\}$
  - Induction: $L^{i+1} = LL^i$
- Example:
  - $\{ab, ba\}^2 =? \{abab, abba, baab, baba\}$
  - $\emptyset^0 =? \{\varepsilon\}$
  - $\emptyset^2 =? \{\varepsilon\}$

Kleene Closure

- $L^* = \cup^{\infty}_{i=0} L^i$

  $= L^0 \cup L^1 \cup L^2 \cup \ldots$

- Examples:
  - $\{ab, ba\}^* =? \{\varepsilon, ab, ba, abab, abba, \ldots\}$
  - $\emptyset^* =? \{\varepsilon\}$
  - $\{\varepsilon\}^* =? \{\varepsilon\}$

Example

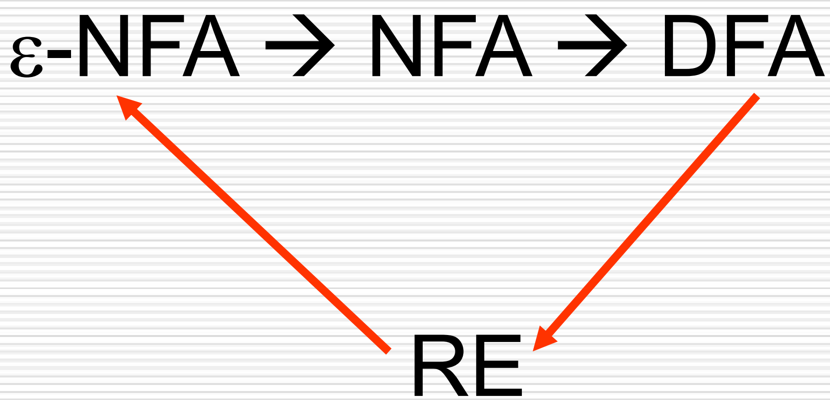
- Order of precedence for operators:
  - ( ) > Closure > Concatenation > Union

- L = {w | 0 and 1 alternate in w}, $\Sigma=\{0, 1\}$, how to describe L in a regular expression?
  - $(01)^*+(10)^*+0(10)^*+1(01)^*$
  - Or equivalently, $(\varepsilon+1)(01)^*(0+\varepsilon)$

Exercises

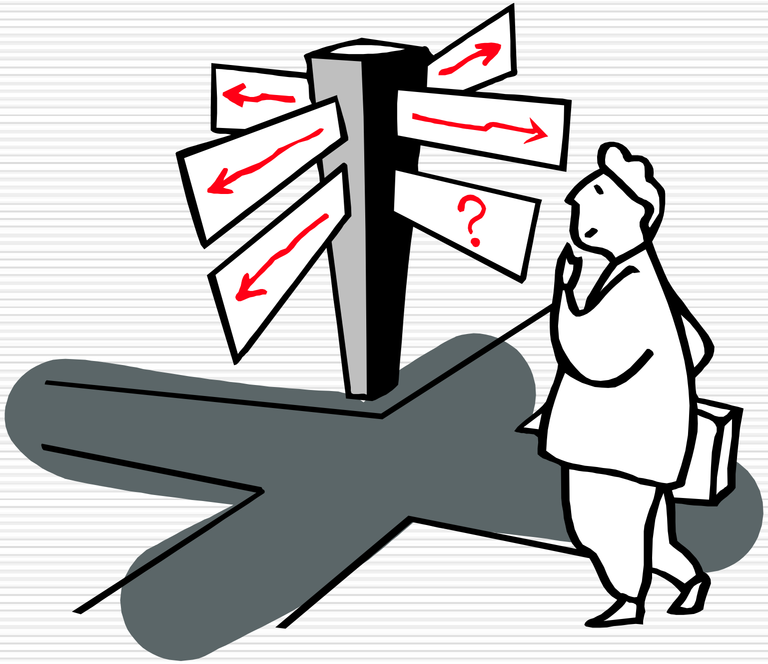
- $\Sigma = \{0, 1\}$
- What is the language for
  - 0^*1^*
- What is the regular expression for
  - $\{w \mid w$ has at least one $1\}$
  - $\{w \mid w$ starts and ends with same symbol$\}$
  - $\{w \mid |w| \leq 5\}$
  - $\{w \mid$ every 3^{rd} position of w is $1\}$
  - $L^+ = L^1 \cup L^2 \cup \ldots$
  - $L?$ (means an optional L)

Equivalence of RE and FA



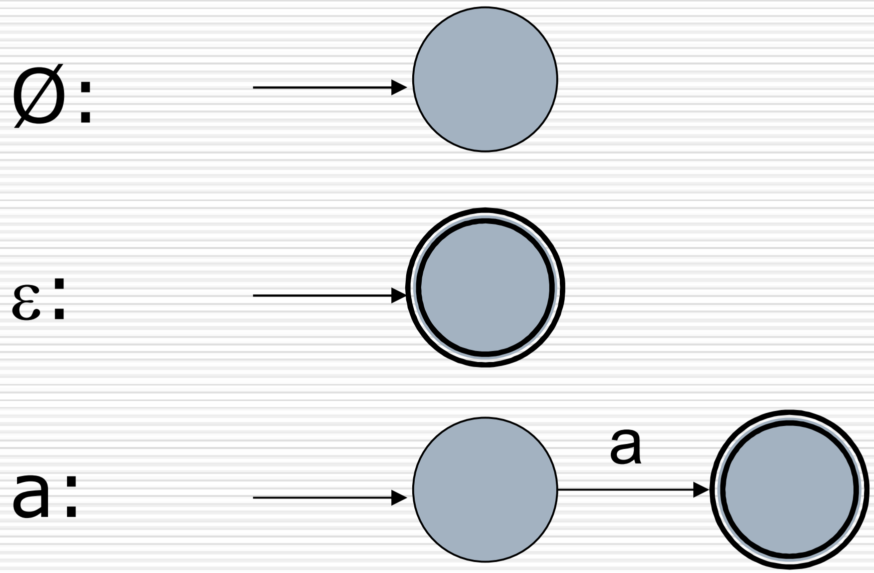
1.2 Regular Expressions

- Regular Expression
- RE → ε-NFA
- DFA → RE



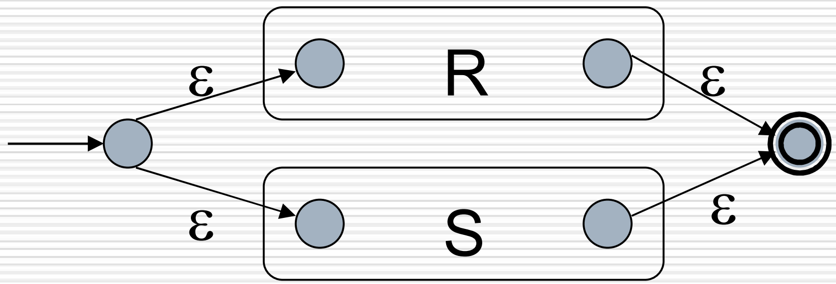
From RE to ε-NFA

- For every regular expression R, we can construct an ε-NFA A, s.t. L(A) = L(R).
- Proof by structural induction:

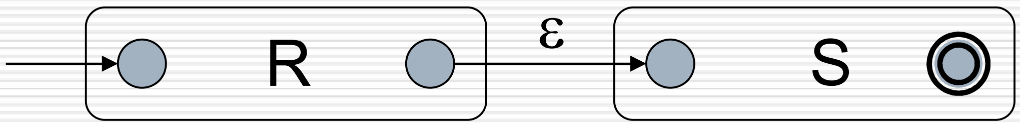


From RE to ε-NFA

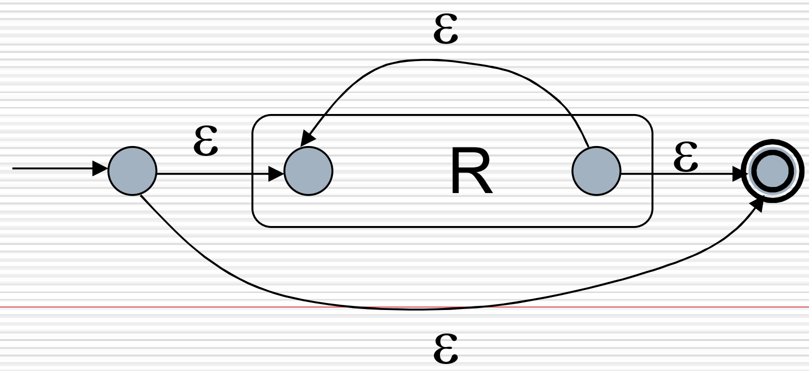
R+S:



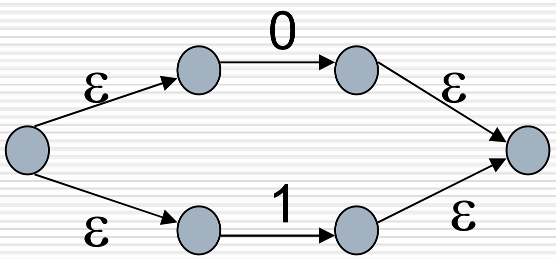
RS:

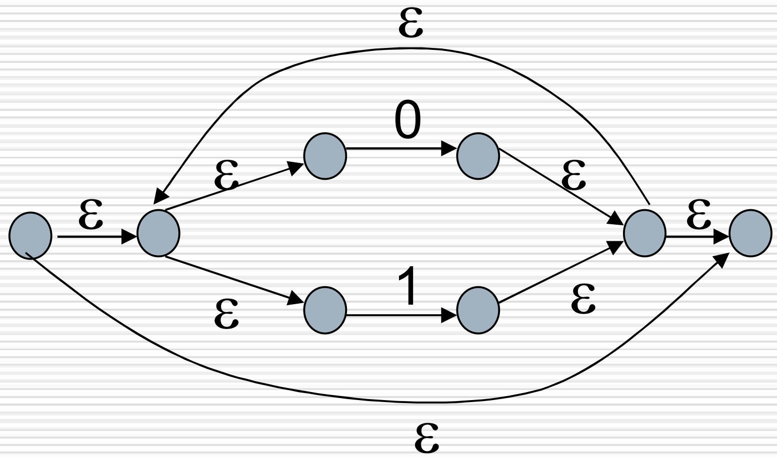


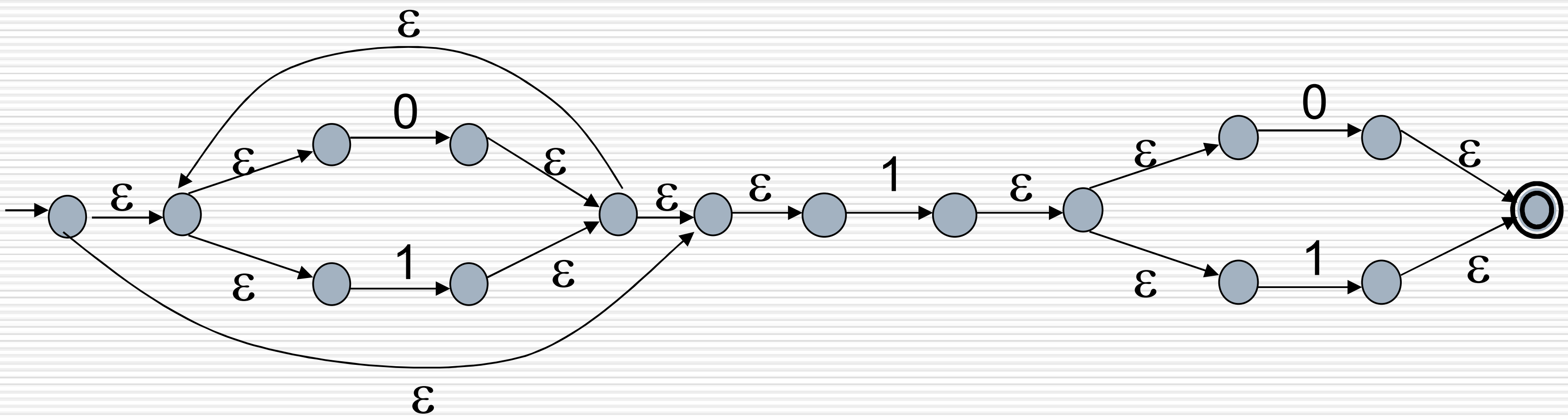
R*:



Example: (0+1)*1(0+1)



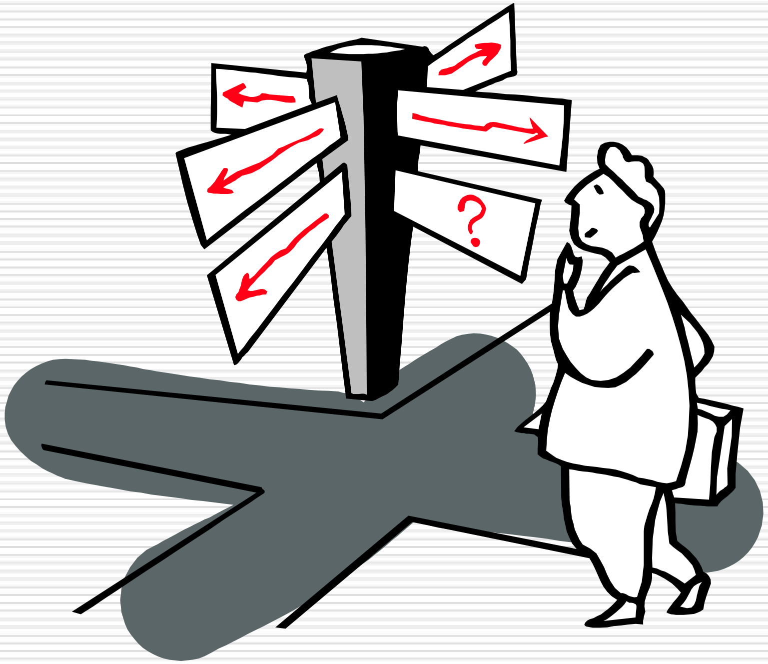




Example 1.31: (a+b)*aba

1.2 Regular Expressions

- Regular Expression
- RE → ε-NFA
- DFA → RE



From DFA to RE

- For every DFA $A=(Q, \Sigma, \delta, q_0, F)$, there is a regular expression R, s.t. $L(R) = L(A)$.
- Proof
  - Let $Q = \{1, 2, \ldots n\}$ and $q_0 = 1$;
  - Let $R[i, j]^k$ be a RE describing the set of all label paths in A from state i to state j going through the states $\{1,\ldots k\}$ only.
  - Then the union of all $R[1, j]^n$ $(j \in F)$ will be the final RE describing the language of A

From DFA to RE

We computes the $R[i, j]^k$ inductively

- Initially (k=0):
  - $R[i, i] = \varepsilon+a+b+\ldots$, whenever $\delta(i, a)=\delta(i, b)=i$, etc.
  - $R[i, j] = a+b+\ldots$, if $i \neq j$ and $\delta(i, a)=\delta(i, b)=j$, etc.
  - $R[i, j] = \emptyset$, if $i \neq j$ and there are no direct transitions from state i to state j.
- Induction
  - $R[i, j]^k = R[i, j]^{k-1} + R[i, k]^{k-1} (R[k, k]^{k-1})^* R[k, j]^{k-1}$

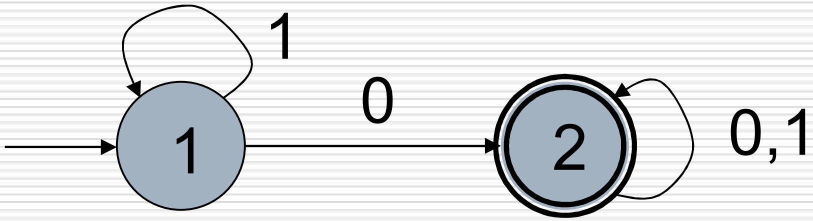
From DFA to RE

Proof that $R[i, j]^k$ is correctly computed:

- A path that goes through the states {1,…k} only either
  - never goes through state k, in which case the path's label is in the language of $R[i, j]^{k-1}$
  - or goes through k one or more times. In this case:
    - **$R[i, k]^{k-1}$** contains the portion of the path that goes from i to k for the first time.
    - **$(R[k, k]^{k-1})^*$** contains the portion of the path (possibly empty) from the first k visit to the last.
    - **$R[k, j]^{k-1}$** contains the portion of the path from the last k visit to state j.

Example

- $L(A) = \{x0y \mid x \in \{1\}^*, y \in \{0, 1\}^*\}$



k = 0

$R[1, 1]^0$	$\varepsilon+1$
$R[1, 2]^0$	0
$R[2, 1]^0$	$\emptyset$
$R[2, 2]^0$	$\varepsilon+0+1$

Example Cont'd

- We need the following simplification rules:
  - $(\varepsilon+R)^* = R^*$
  - $R+RS^* = RS^*$
  - $\emptyset R = R\emptyset = \emptyset$ (Annihilation)
  - $\emptyset+R = R+\emptyset = R$ (Identity)

Example Cont'd

$R[1, 1]^0$	$\varepsilon+1$
$R[1, 2]^0$	0
$R[2, 1]^0$	$\emptyset$
$R[2, 2]^0$	$\varepsilon+0+1$

$\Rightarrow$ $k = 1$

$R[1, 1]^1$	$\varepsilon+1+(\varepsilon+1)(\varepsilon+1)^*(\varepsilon+1) = 1^*$
$R[1, 2]^1$	$0+(\varepsilon+1)(\varepsilon+1)^*0 = 1^*0$
$R[2, 1]^1$	$\emptyset+\emptyset(\varepsilon+1)^*(\varepsilon+1) = \emptyset$
$R[2, 2]^1$	$\varepsilon+0+1+\emptyset(\varepsilon+1)^*0 = \varepsilon+0+1$

Example Cont'd

$R[1, 1]^1$	1^*
$R[1, 2]^1$	1^*0
$R[2, 1]^1$	$\varnothing$
$R[2, 2]^1$	$\varepsilon+0+1$

$\Rightarrow$ k = 2

$R[1, 1]^2$	$1^* + 1^*0(\varepsilon+0+1)^*\varnothing = 1^*$
$R[1, 2]^2$	$1^*0+1^*0(\varepsilon+0+1)^*(\varepsilon+0+1) = 1^*0(0+1)^*$
$R[2, 1]^2$	$\varnothing+(\varepsilon+0+1)(\varepsilon+0+1)^*\varnothing = \varnothing$
$R[2, 2]^2$	$\varepsilon+0+1+(\varepsilon+0+1)(\varepsilon+0+1)^*(\varepsilon+0+1) = 0+1$

Example Cont'd

- The final regular expression for A is

  $R[1, 2]^2 = 1^*0(0+1)^*$

Observation

```
for k = 1 to n
    for i = 1 to n
        for j = 1 to n
            B[i, j] = R[i, j] + R[i, k] (R[k, k])* R[k, j]
        end
    end
    R = B (copy all the updates to R).
End
```

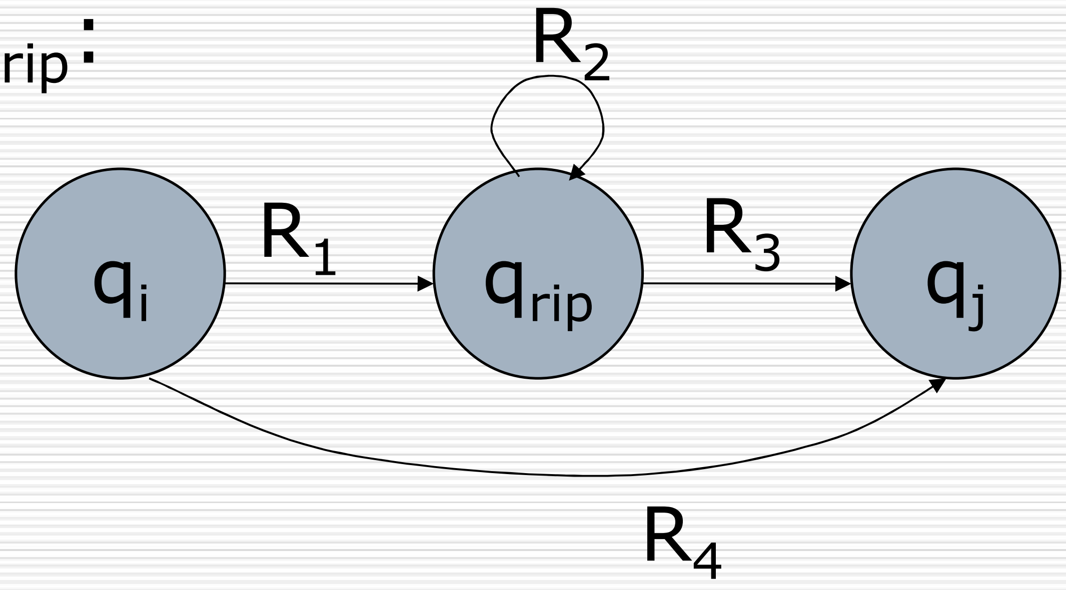
- n^3 expressions $R[i, j]^k$, and $R[i, j]^k$ could have size 4^n
- Alternative approach: state elimination

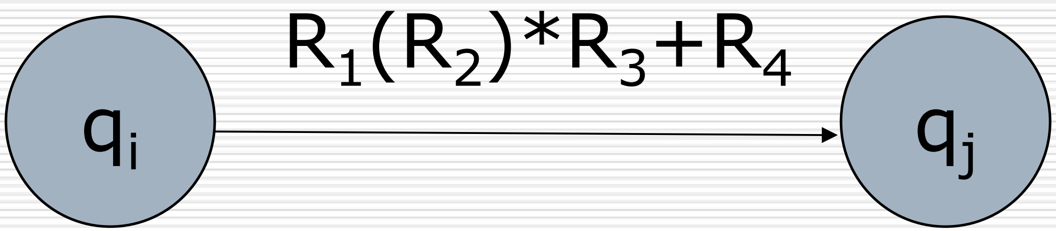
State Elimination Technique

- Label the edges of the FA with regular expressions instead of symbols.
- Add a new start state, q_{start}, with an ε edge to the old start state and a new accept state, q_{accept}, with ε arrows from the old accept states.
- eliminate all states except q_{start} and q_{accept}, the regular expression on the label is the answer.

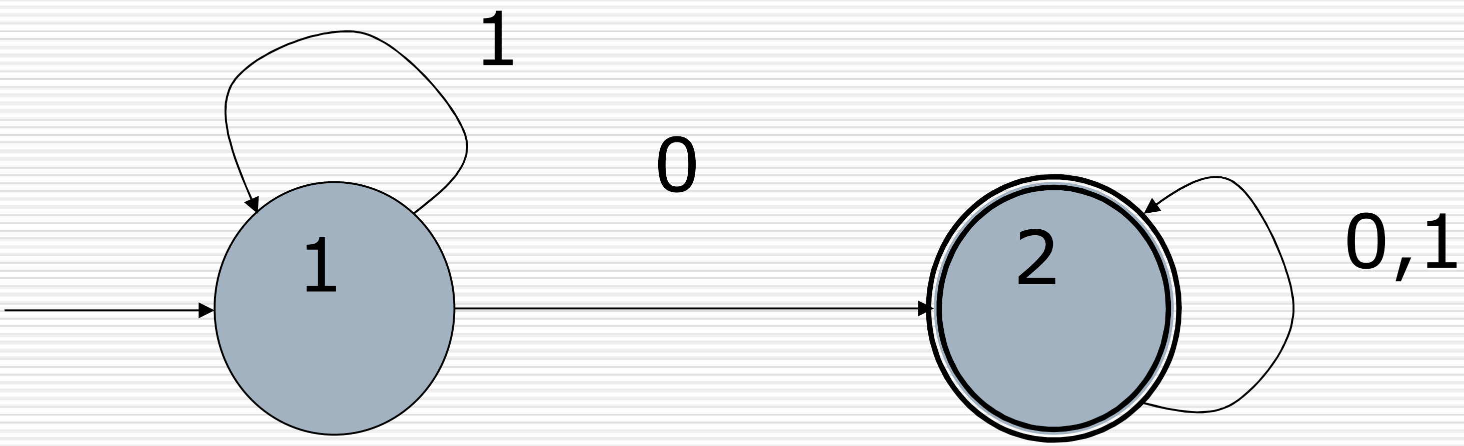
State Elimination Technique

- Eliminate q_{rip}:

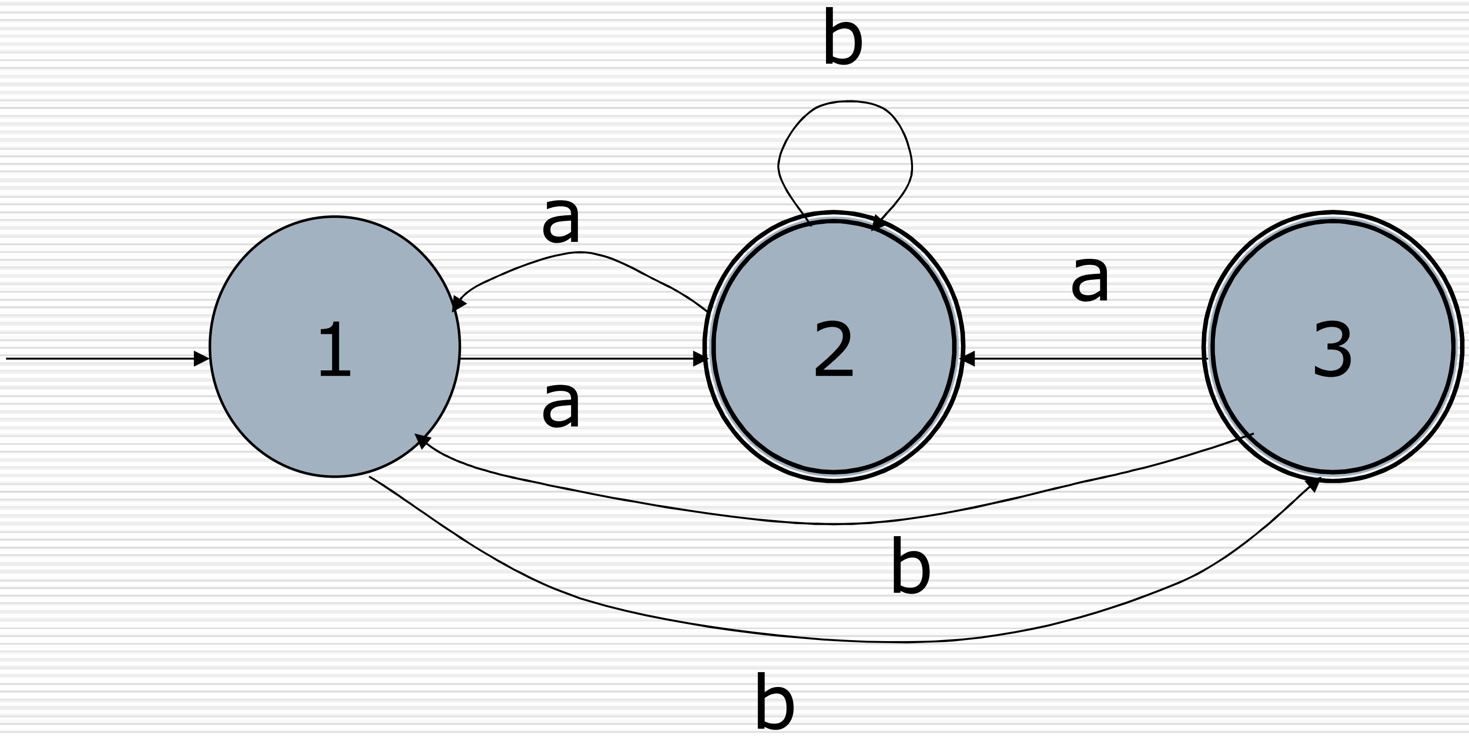




Example 1.35



Example 1.36



Algebraic Laws for RE

- Regexs E and F are equivalent: $L(E) = L(F)$.
- $(E + F) + G = E + (F + G)$
  - Union is associative
- $E + F = F + E$
  - Union is commutative
- $\emptyset + E = E + \emptyset = E$
  - $\emptyset$ is identity for union
- $E + E = E$
  - Union is idempotent

Algebraic Laws for RE

- $(E\ F)\ G = E\ (F\ G)$
  - Concatenation is associative
- $\varepsilon\ E = E\ \varepsilon = E$
  - ε is right and left identity for concatenation
- $\emptyset\ E = E\ \emptyset = \emptyset$
  - $\emptyset$ is right and left annihilator for concatenation
- $E\ (F + G) = E\ F + E\ G$
  - Concatenation is left distributive over union
- $(F + G)\ E = F\ E + G\ E$
  - Concatenation is right distributive over union

Algebraic Laws for RE

- $\emptyset^* = \varepsilon$
- $\varepsilon^* = \varepsilon$
- $(E^*)^* = E^*$
  - Closure is idempotent
- $E^* = EE^* + \varepsilon$