REGULAR EXPRESSION

ITEU133

AUTOMATA AND THEORY OF COMPUTATION

COURSE SYLLABUS

III. Regular Expression (RE) and Regular Language

- Definition of a Regular Expression
- Language Associated with Regular Expression
- Connection Between Regular Expressions and Regular Languages
- Equivalence of FA’s and Regular Expressions
- One State and Two State Machine.

OPERATIONS ON LANGUAGES

Let L, L_1, L_2 be subsets of Σ^*

Concatenation: $L_1L_2 = \{xy \mid x \text{ is in } L_1 \text{ and } y \text{ is in } L_2\}$

Concatenating a language with itself:

$L^0 = \{\varepsilon\}$ $L^i = LL^{i-1}$, for all i >= 1

Kleene Closure: $L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup \ldots$

Positive Closure: $L^+ = \bigcup_{i=1}^{\infty} L^i = L^1 \cup L^2 \cup \ldots$

KLEENE CLOSURE

Say, L^1 ={a, abc, ba}, on Σ ={a,b,c}

Then, L^2 = {aa, aabc, aba, abca, bcabc, abcba, baa, baabc, baba}

L^3= {a, abc, ba}. L^2

L^* = {ε, L^1, L^2, L^3, . . .}

REGULAR EXPRESSIONS

- Highlights:
  - A regular expression is used to specify a language, and it does so precisely.
  - Regular expressions are very intuitive.
  - Regular expressions are very useful in a variety of contexts.
  - Given a regular expression, an NFA-ε can be constructed from it automatically.
  - Thus, so can an NFA, a DFA, and a corresponding program, all automatically!

Definition of a Regular Expression

- Let Σ be an alphabet. The regular expressions over Σ are:
  - Ø Represents the empty set { }
  - ε Represents the set {ε}
  - a Represents the set {a}, for any symbol a in Σ

  Let r and s be regular expressions that represent the sets R and S, respectively.
  - r+s Represents the set R U S (precedence 3)
  - rs Represents the set RS (precedence 2)
  - r^* Represents the set R* (highest precedence)
  - (r) Represents the set R (not an op, provides precedence)

- If r is a regular expression, then L(r) is used to denote the corresponding language.

- **Examples:** Let $\Sigma = \{0, 1\}$

$(0 + 1)^*$	All strings of 0's and 1's
$0(0 + 1)^*$	All strings of 0's and 1's, beginning with a 0
$(0 + 1)^*1$	All strings of 0's and 1's, ending with a 1
$(0 + 1)^*0(0 + 1)^*$	All strings of 0's and 1's containing at least one 0
$(0 + 1)^*0(0 + 1)^*0(0 + 1)^*$	All strings of 0's and 1's containing at least two 0's
$(0 + 1)^*01^*01^*$	All strings of 0's and 1's containing at least two 0's
$(1 + 01^*0)^*$	All strings of 0's and 1's containing an even number of 0's
$1^*(01^*01^*)^*$	All strings of 0's and 1's containing an even number of 0's
$(1^*01^*0)^*1^*$	All strings of 0's and 1's containing an even number of 0's

Example:

1. Write the regular expression for the language
   $L=\{ab^n w : n \geq 3, w \varepsilon (a,b)^+\}$
2. . Write the regular expression for the language
   $L= \{w \varepsilon (a,b)^*: n_a (w) \bmod 3=0\}$
3. L = {strings of 0's and1's beginning with 0 and ending with 1}
4. For all strings containing exact one a, over the alphabet of {a,b,c}

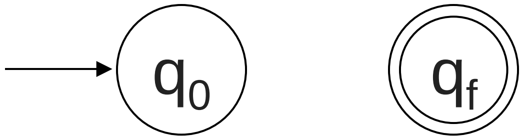
EQUIVALENCE OF REGULAR EXPRESSIONS AND NFA-ε's

- **Note:** Throughout the following, keep in mind that a string is accepted by an NFA-ε if there exists a path from the start state to a final state.
- **Lemma 1:** Let r be a regular expression. Then there exists an NFA-ε M such that L(M) = L(r). Furthermore, M has exactly one final state with no transitions out of it.
- **Proof:** (by induction on the number of operators, denoted by OP(r), in r).

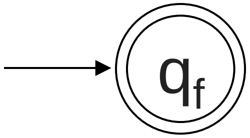
Basis: OP(r) = 0

Then r is either Ø, ε, or **a**, for some symbol **a** in Σ

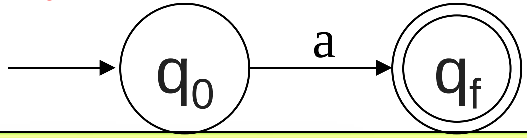
For Ø:



For ε:



For **a**:



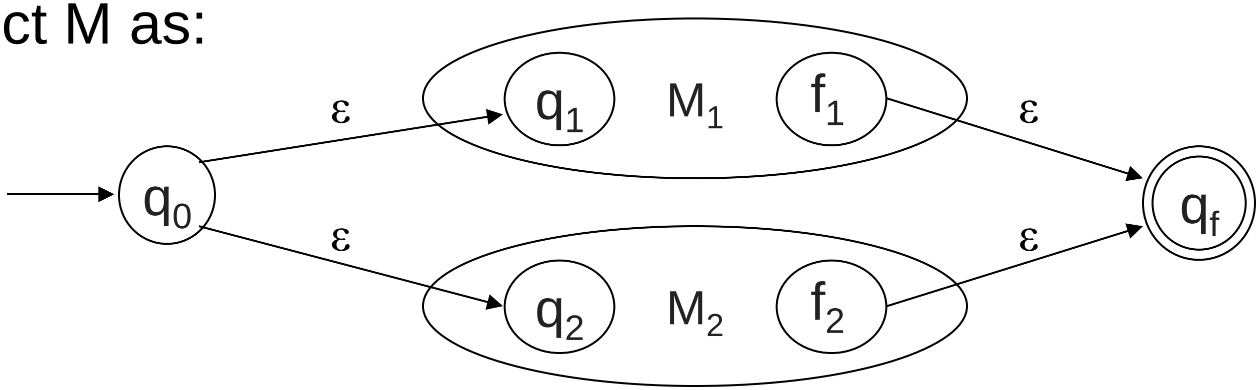
Inductive Hypothesis: Suppose there exists a $k \geq 0$ such that for any regular expression r where $0 \leq OP(r) \leq k$, there exists an NFA-ε such that L(M) = L(r). Furthermore, suppose that M has exactly one final state.

Inductive Step: Let r be a regular expression with k + 1 operators (OP(r) = k + 1), where k + 1 >= 1.

Case 1) $r = r_1 + r_2$

Since OP(r) = k +1, it follows that $0 <= OP(r_1), OP(r_2) <= k$. By the inductive hypothesis there exist NFA-ε machines M_1 and M_2 such that $L(M_1) = L(r_1)$ and $L(M_2) = L(r_2)$. Furthermore, both M_1 and M_2 have exactly one final state.

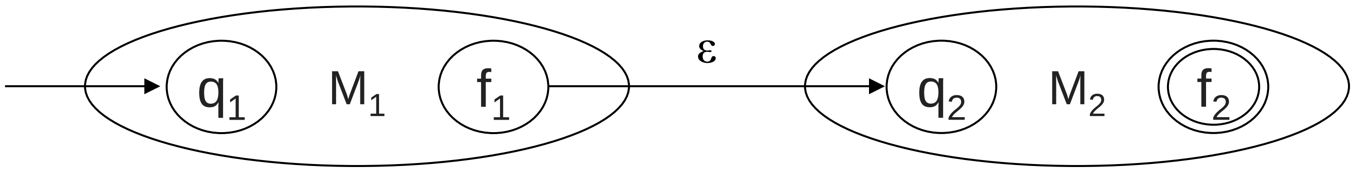
Construct M as:



Case 2) $r = r_1 r_2$

Since $OP(r) = k+1$, it follows that $0 <= OP(r_1), OP(r_2) <= k$. By the inductive hypothesis there exist NFA-ε machines M_1 and M_2 such that $L(M_1) = L(r_1)$ and $L(M_2) = L(r_2)$. Furthermore, both M_1 and M_2 have exactly one final state.

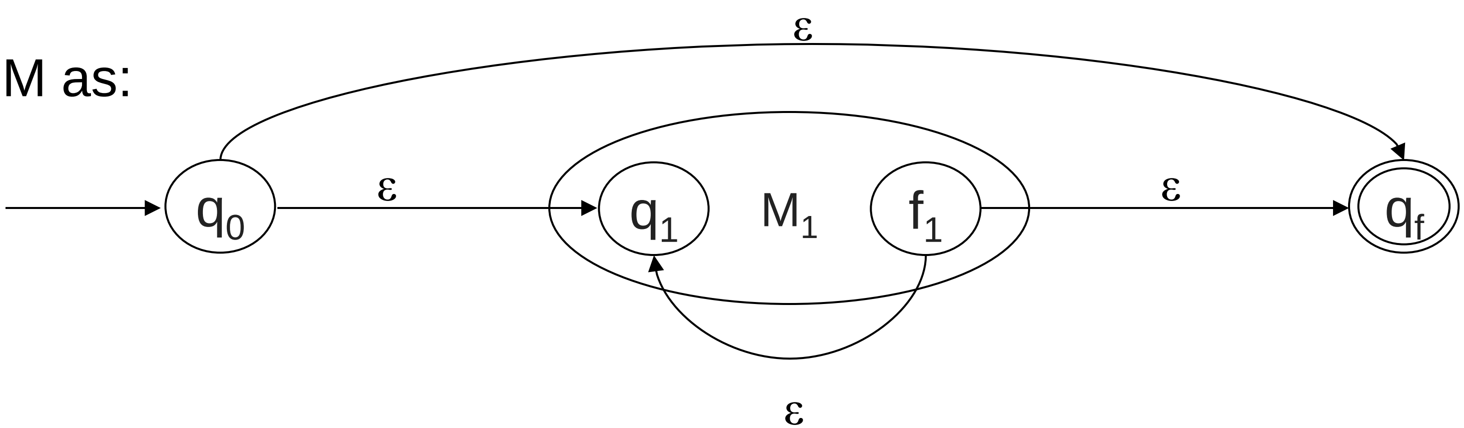
Construct M as:



Case 3) $r = r_1^*$

Since $OP(r) = k+1$, it follows that $0 <= OP(r_1) <= k$. By the inductive hypothesis there exists an NFA-ε machine M_1 such that $L(M_1) = L(r_1)$. Furthermore, M_1 has exactly one final state.

Construct M as:



- **Example:** $\mathbf{r = 0(0+1)^*}$

$r = r_1 r_2$

$r_1 = 0$

$r_2 = (0+1)^*$

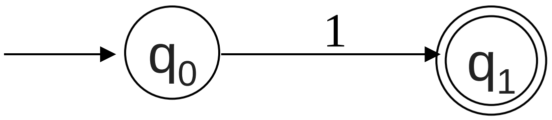
$r_2 = r_3{}^*$

$r_3 = 0+1$

$r_3 = r_4 + r_5$

$r_4 = 0$

$\mathbf{r_5 = 1}$



- **Example:** **$r = 0(0+1)^*$**

$r = r_1 r_2$

$r_1 = 0$

$r_2 = (0+1)^*$

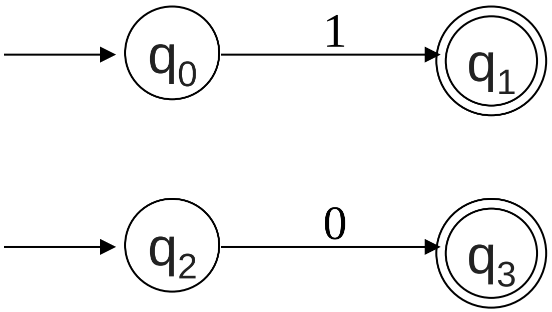
$r_2 = r_3^*$

$r_3 = 0+1$

$r_3 = r_4 + r_5$

$r_4 = 0$

$\mathbf{r_5 = 1}$



- **Example:** $\mathbf{r = 0(0+1)^*}$

$r = r_1 r_2$

$r_1 = 0$

$r_2 = (0+1)^*$

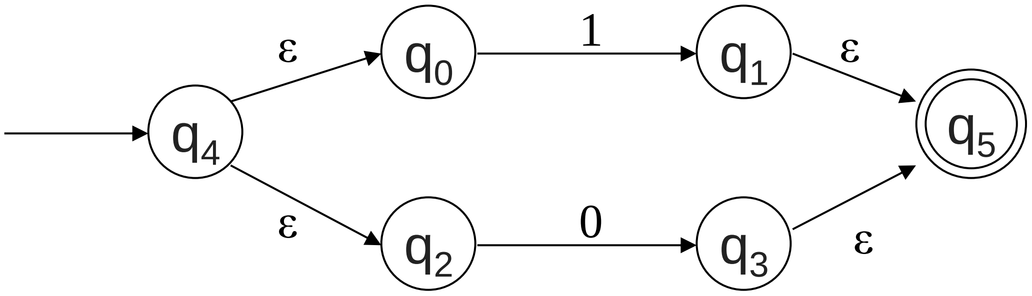
$r_2 = r_3^*$

$r_3 = 0+1$

$\mathbf{r_3 = r_4 + r_5}$

$r_4 = 0$

$r_5 = 1$



- **Example:** $\mathbf{r = 0(0+1)^*}$

$r = r_1 r_2$

$r_1 = 0$

$r_2 = (0+1)^*$

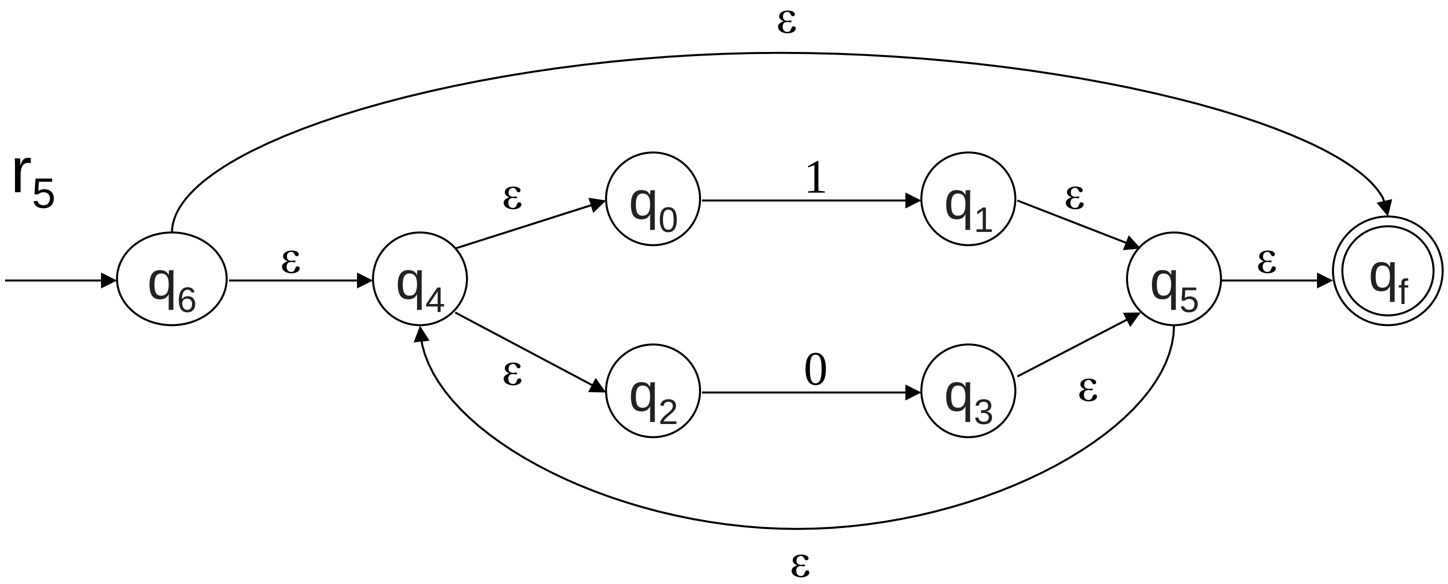
$\mathbf{r_2 = r_3^*}$

$r_3 = 0+1$

$r_3 = r_4 + r_5$

$r_4 = 0$

$r_5 = 1$



- **Example:** **$r = 0(0+1)^*$**

$r = r_1 r_2$

$r_1 = 0$

$r_2 = (0+1)^*$

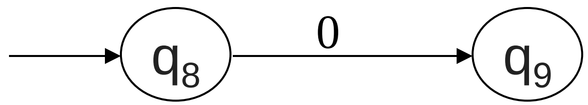
$r_2 = r_3^*$

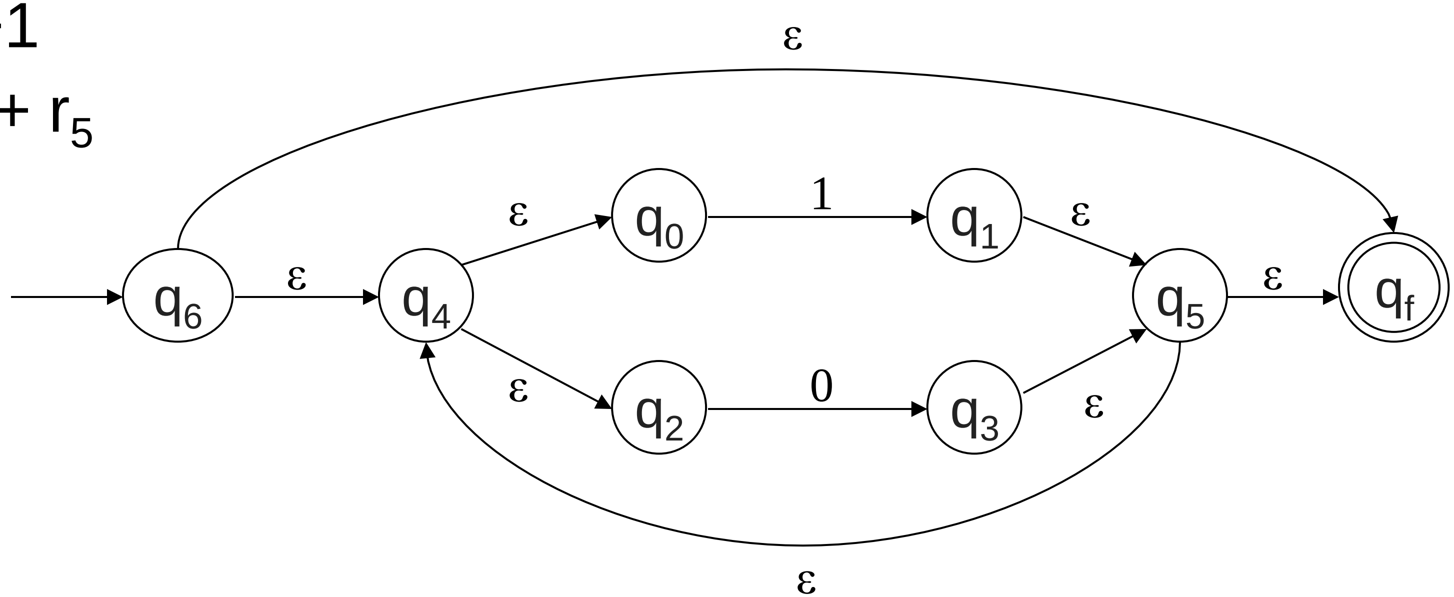
$r_3 = 0+1$

$r_3 = r_4 + r_5$

$r_4 = 0$

$r_5 = 1$





- **Example:** $r = 0(0+1)^*$

$r = r_1 r_2$

$r_1 = 0$

$r_2 = (0+1)^*$

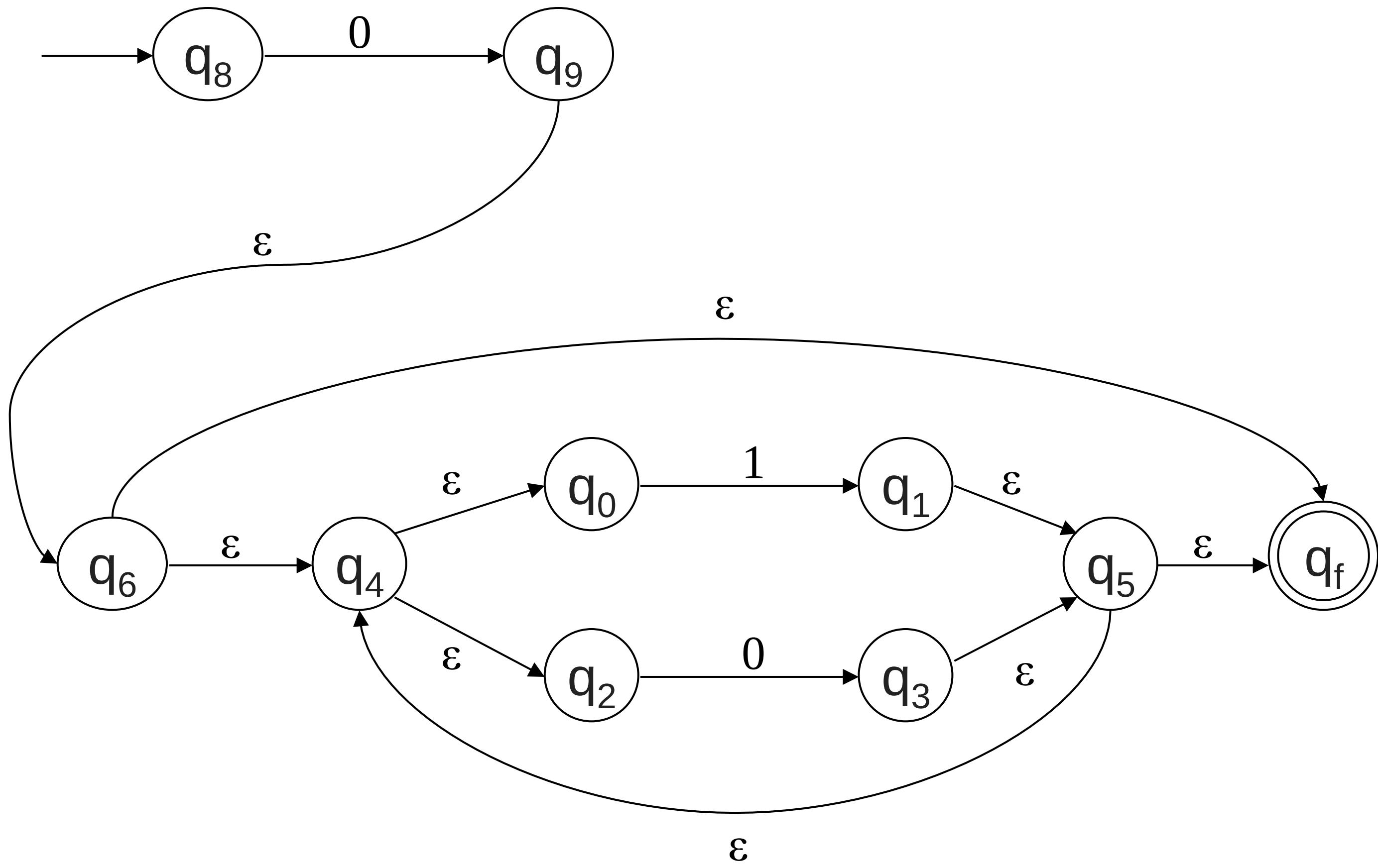
$r_2 = r_3^*$

$r_3 = 0+1$

$r_3 = r_4 + r_5$

$r_4 = 0$

$r_5 = 1$



TWO-WAY FINITE AUTOMATA

- Two-way finite automata are machines that can read input string in either direction. This type of machines have a “read head”, which can move left or right over the input string.