Lesson 6: Regular Expressions

Marc Gaetano

Edition 2018

Operations on strings

- Given two strings $s = a_1 \ldots a_n$ and $t = b_1 \ldots b_m$, we define their concatenation $st = a_1 \ldots a_n b_1 \ldots b_m$

$$s = abb,\ t = cba \quad st = abbcba$$

- We define s^n as the concatenation $ss \ldots s$ n times

$$s = 011 \qquad\qquad s^3 = 011011011$$

Operations on languages

- The concatenation of languages L_1 and L_2 is

$$L_1L_2 = \{\text{st: s} \in L_1, \text{t} \in L_2\}$$

- Similarly, we write L^n for $LL\ldots L$ (n times)
- The union of languages $L_1 \cup L_2$ is the set of all strings that are in L_1 or in L_2

- Example: $L_1 = \{01, 0\}, L_2 = \{\varepsilon, 1, 11, 111, \ldots\}$. What is L_1L_2 and $L_1 \cup L_2$?

Operations on languages

- The star (Kleene closure) of L are all strings made up of zero or more chunks from L:

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

  – This is always infinite, and always contains ε

- Example: $L_1 = \{01, 0\}, L_2 = \{\varepsilon, 1, 11, 111, \ldots\}$. What is $L_1{}^*$ and $L_2{}^*$?

Constructing languages with operations

- Let's fix an alphabet, say $\Sigma = \{0, 1\}$
- We can construct languages by starting with simple ones, like $\{0\}$, $\{1\}$ and combining them

$\{0\}(\{0\}\cup\{1\})^*$ $\Rightarrow$ $0(0+1)^*$

all strings that start with 0

$(\{0\}\{1\}^*)\cup(\{1\}\{0\}^*)$ $\Rightarrow$ 01^*+10^*

Regular expressions

- A regular expression over Σ is an expression formed using the following rules:
  - The symbol $\emptyset$ is a regular expression
  - The symbol ε is a regular expression
  - For every $a \in \Sigma$, the symbol a is a regular expression
  - If R and S are regular expressions, so are RS, $R+S$ and R^*.

- Definition of regular language

A language is regular if it is represented by a regular expression

Examples

1. $01^* = \{0, 01, 011, 0111, \ldots..\}$
2. $(01^*)(01) = \{001, 0101, 01101, 011101, \ldots..\}$
3. $(0+1)^*$
4. $(0+1)^*01(0+1)^*$
5. $((0+1)(0+1)+(0+1)(0+1)(0+1))^*$
6. $((0+1)(0+1))^*+((0+1)(0+1)(0+1))^*$
7. $(1+01+001)^*(\varepsilon+0+00)$

Examples

- Construct a RE over $\Sigma = \{0,1\}$ that represents
  - All strings that have two consecutive 0s.

$$(0+1)^*00(0+1)^*$$

  - All strings except those with two consecutive 0s.

$$(1^*01)^*1^* + (1^*01)^*1^*0$$

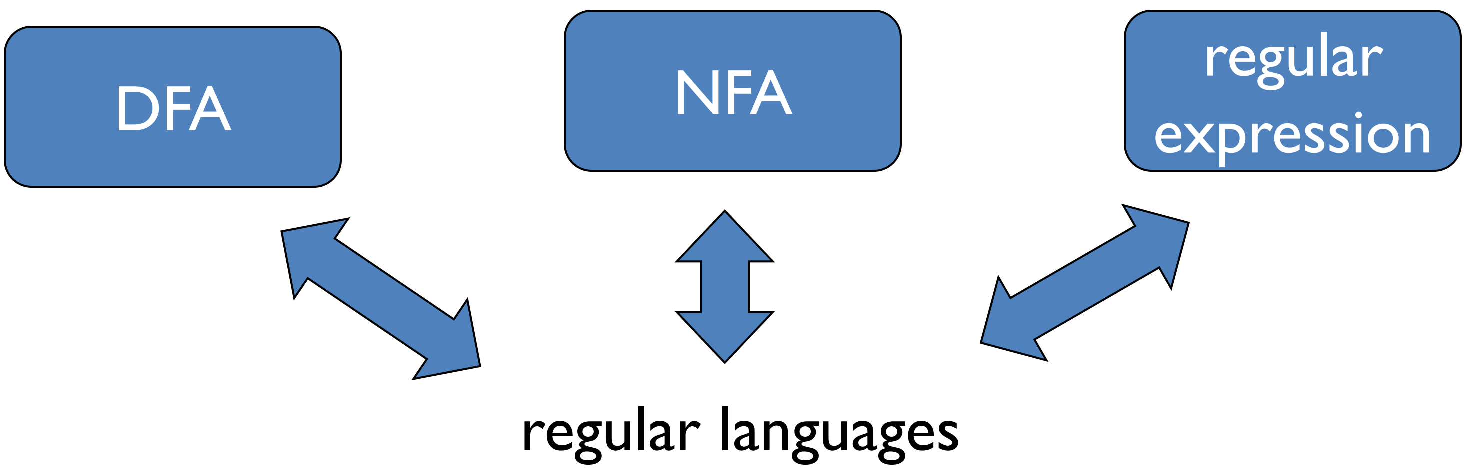
  - All strings with an even number of 0s.

$$(1^*01^*01^*)^*$$

Main theorem for regular languages

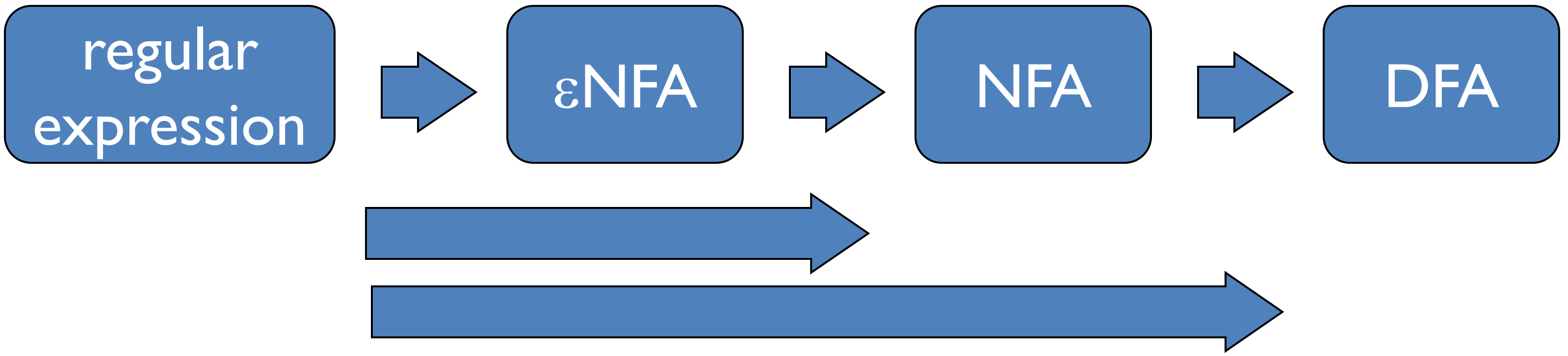
- Theorem

A language is regular if and only if it is the language of some DFA



Proof plan

- For every regular expression, we have to give a DFA for the same language



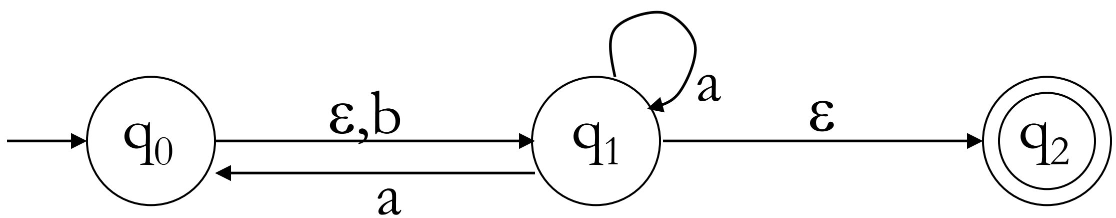
- For every DFA, we give a regular expression for the same language

εNFA reminder

- An εNFA is an extension of NFA where some transitions can be labeled by ε
  - Formally, the transition function of an εNFA is a function
    $\delta: \mathcal{Q} \times (\Sigma \cup \{\varepsilon\}) \rightarrow \text{subsets of } \mathcal{Q}$

- The automaton is allowed to follow ε-transitions without consuming an input symbol

Example of εNFA

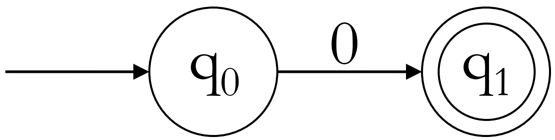


$\Sigma = \{a, b\}$

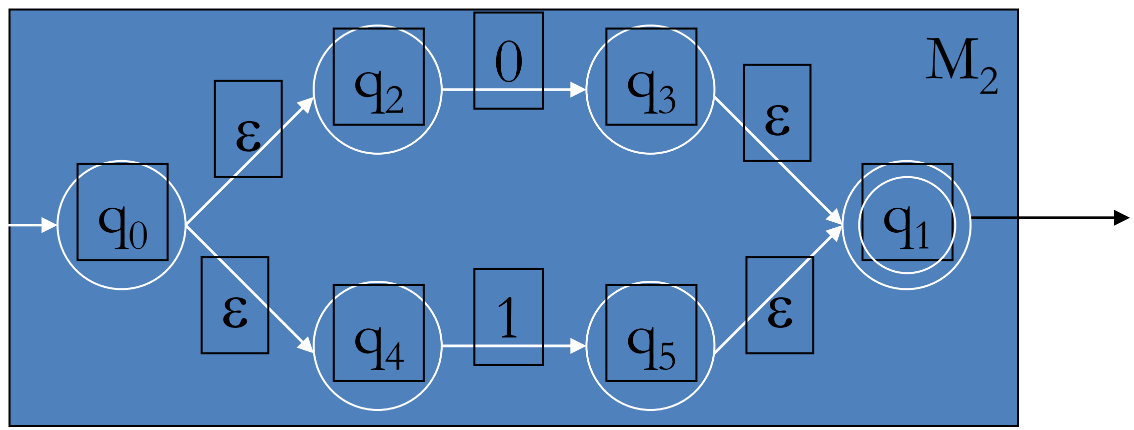
- Which of the following is accepted by this εNFA:
  - aab, bab, ab, bb, a, ε

Examples: regular expression $\rightarrow$ εNFA

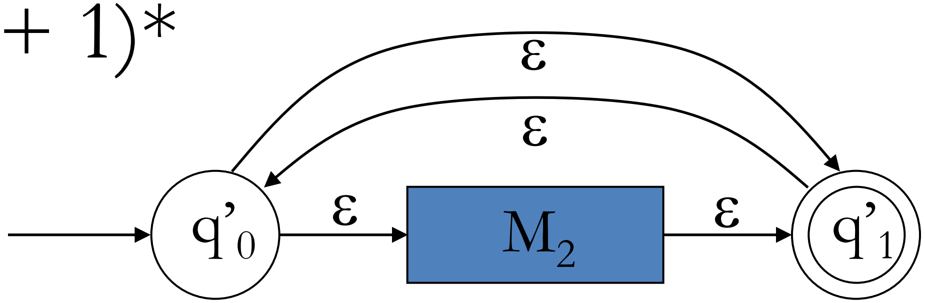
- $R_1 = 0$



- $R_2 = 0 + 1$



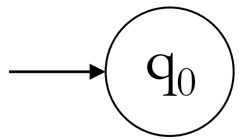
- $R_3 = (0 + 1)^*$



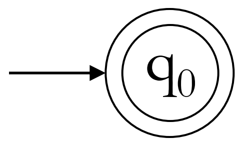
General method

regular expr ⇨ εNFA

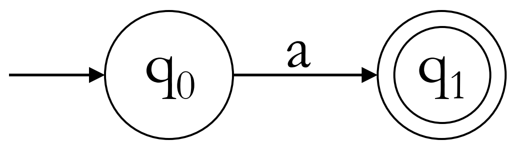
∅



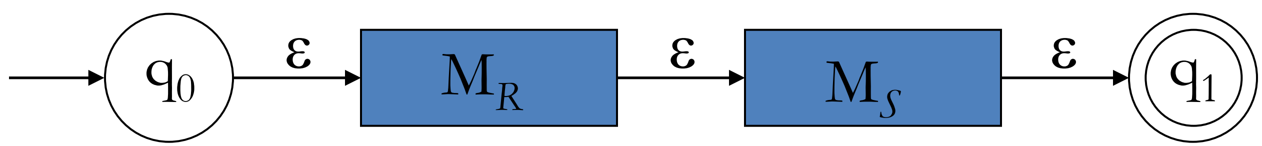
ε



symbol a



RS



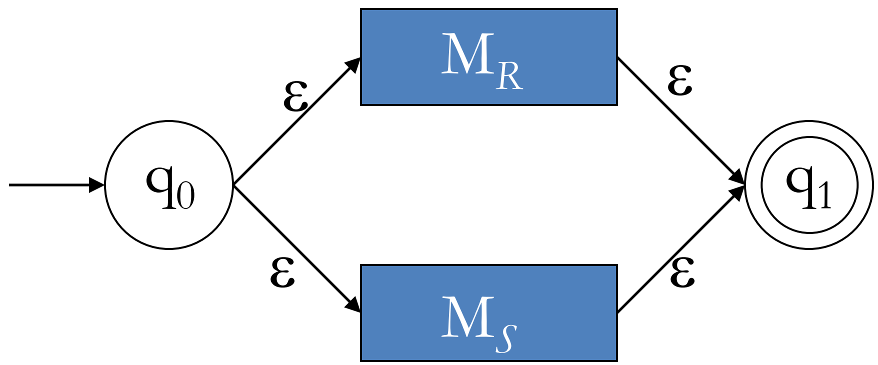
Convention

- When we draw a box around an εNFA:
  - The arrow going in points to the start state
  - The arrow going out represents all transitions going out of accepting states
  - None of the states inside the box is accepting
  - The labels of the states inside the box are distinct from all other states in the diagram

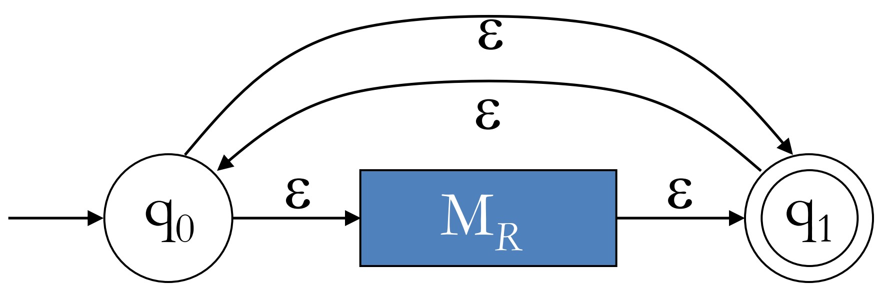
General method continued

regular expr $\Rightarrow$ εNFA

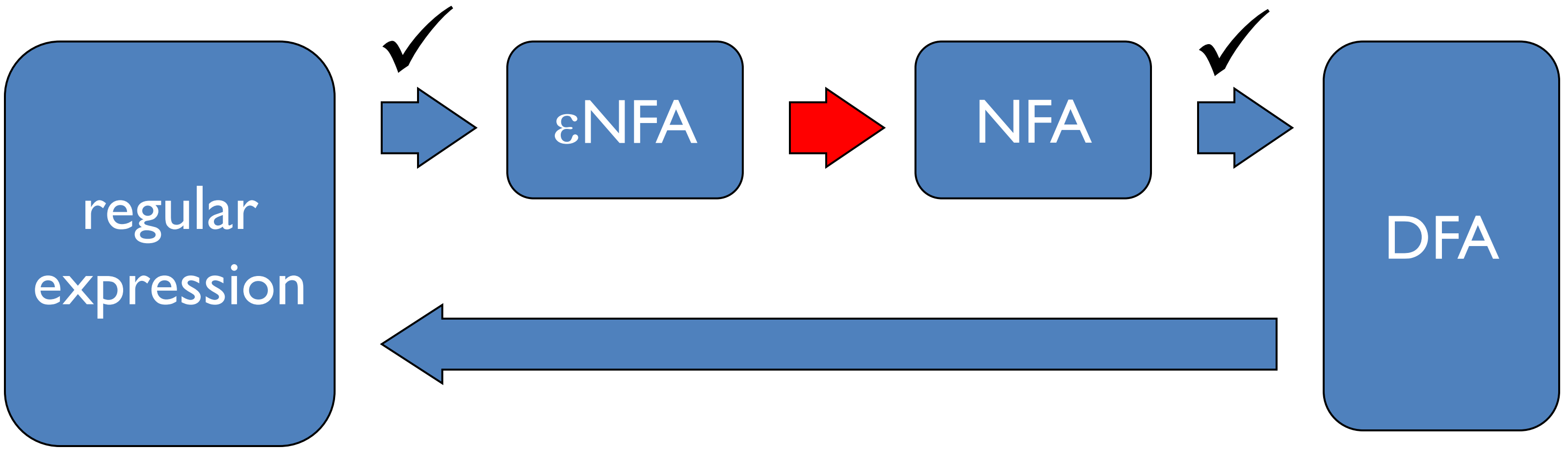
$R + S$



R^*

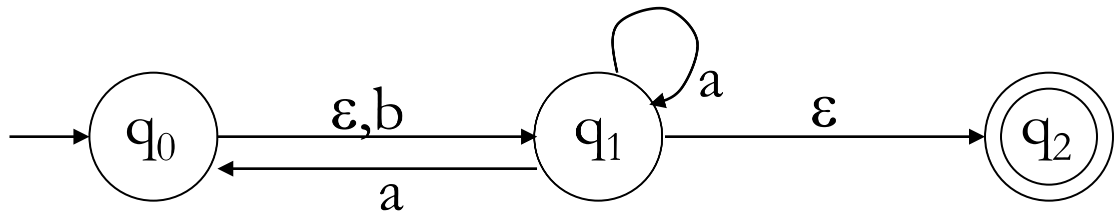


Road map



Example of εNFA to NFA conversion

εNFA:



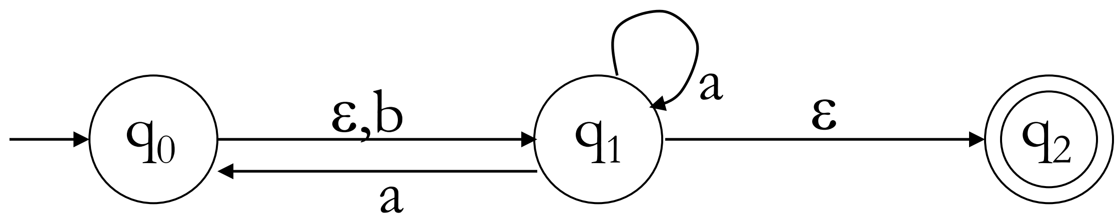
Transition table of corresponding NFA:

states	inputs: a	inputs: b
q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$
q_1	$\{q_0, q_1, q_2\}$	$\varnothing$
q_2	$\varnothing$	$\varnothing$

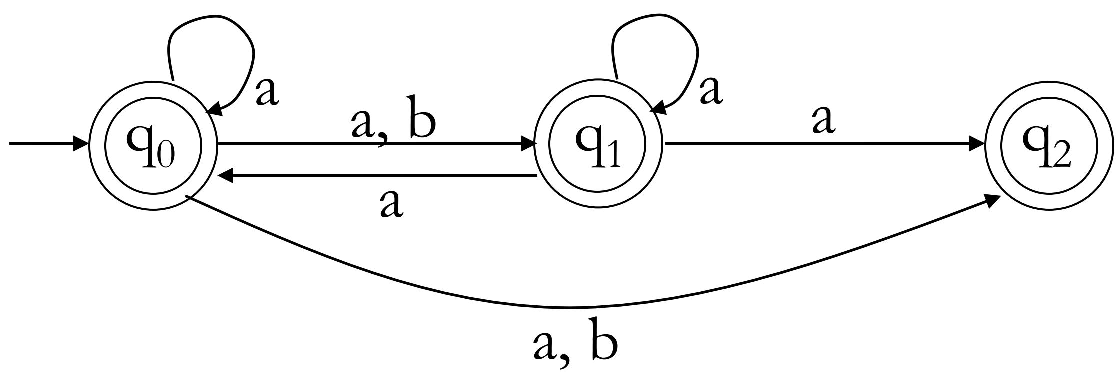
Accepting states of NFA: $\{q_0, q_1, q_2\}$

Example of εNFA to NFA conversion

εNFA:



NFA:



General method

- To convert an εNFA to an NFA:
  - States stay the same
  - Start state stays the same
  - The NFA has a transition from q_i to q_j labeled a iff the εNFA has a path from q_i to q_j that contains one transition labeled a and all other transitions labeled ε
  - The accepting states of the NFA are all states that can reach some accepting state of εNFA using only ε-transitions

Why the conversion works

In the original ε-NFA, when given input $a_1 a_2 \ldots a_n$ the automaton goes through a sequence of states:

$$q_0 \to q_1 \to q_2 \to \ldots \to q_m$$

Some ε-transitions may be in the sequence:

$$q_0 \xrightarrow{\varepsilon} \ldots \xrightarrow{a_1} \ldots \xrightarrow{\varepsilon} q_{i_1} \xrightarrow{\varepsilon} \ldots \xrightarrow{a_2} \ldots \xrightarrow{\varepsilon} q_{i_2} \xrightarrow{\varepsilon} \ldots \xrightarrow{\varepsilon} q_{i_n}$$

In the new NFA, each sequence of states of the form:

$$q_{i_k} \xrightarrow{\varepsilon} \ldots \xrightarrow{a_{k+1}} \ldots \xrightarrow{\varepsilon} q_{i_{k+1}}$$

will be represented by a single transition $q_{i_k} \xrightarrow{a_{k+1}} q_{i_{k+1}}$ because of the way we construct the NFA.

Proof that the conversion works

- More formally, we have the following invariant for any $k \geq 1$:

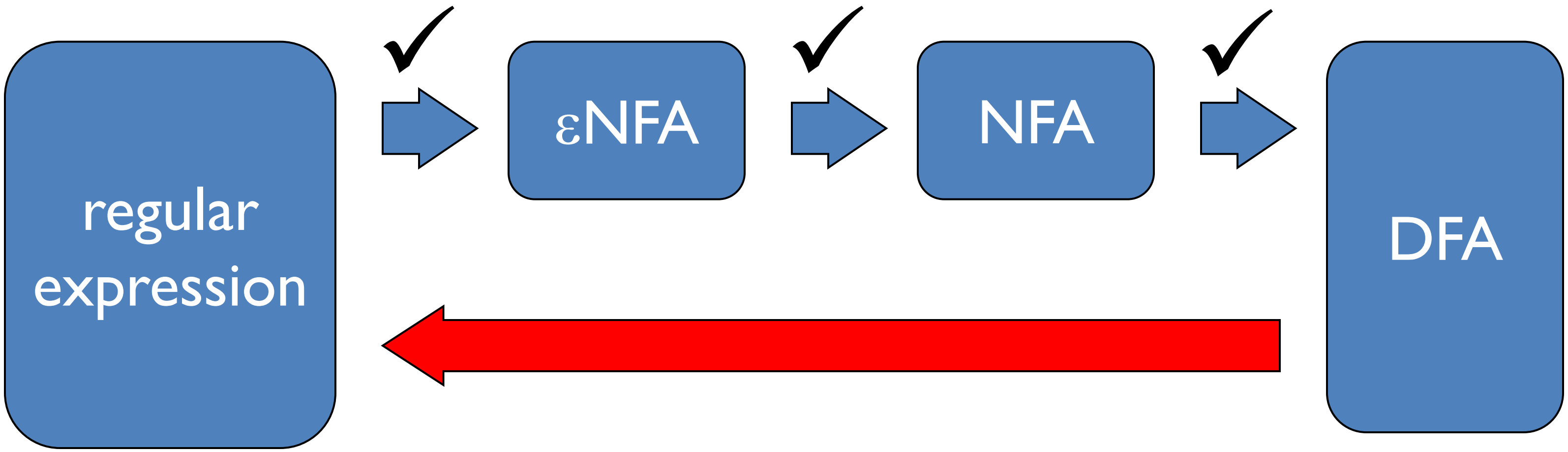
> After reading k input symbols, the set of states that the εNFA and NFA can be in are exactly the same

- We prove this by induction on k
- When $k = 0$, the εNFA can be in more states, while the NFA must be in q_0

Proof that the conversion works

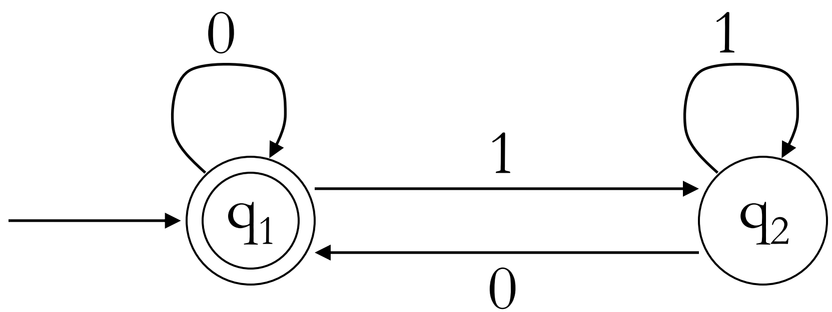
- When $k \geq 1$ (input is not the empty string)
  - If εNFA is in an accepting state, so is NFA
  - Conversely, if NFA is an accepting state q_i, then some accepting state of εNFA is reachable from q_i, so εNFA accepts also
- When $k = 0$ (input is the empty string)
  - The εNFA accepts iff one of its accepting states is reachable from q_0
  - This is true iff q_0 is an accepting state of the NFA

From DFA to regular expressions



Example

- Construct a regular expression for this DFA:



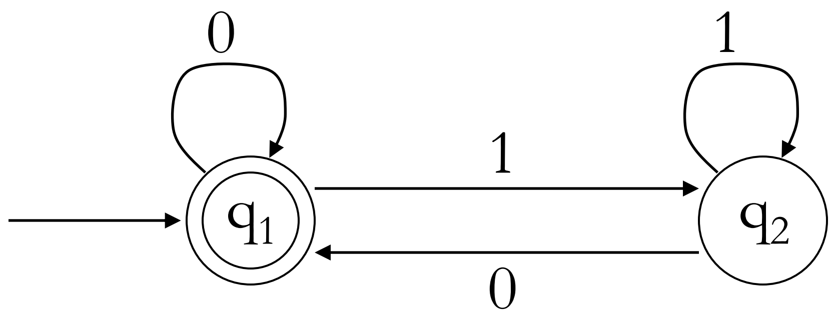
$$(0 + 1)^*0 + \varepsilon$$

General method

- We have a DFA M with states $q_1, q_2, \ldots q_n$
- We will inductively define regular expressions R_{ij}^{k}

R_{ij}^{k} will be the set of all strings that take M from q_i to q_j with intermediate states going through $q_1, q_2, \ldots$ or q_k only.

Example



$$R_{11}{}^{0} = \{\varepsilon, 0\} = \varepsilon + 0$$

$$R_{12}{}^{0} = \{1\} = 1$$

$$R_{22}{}^{0} = \{\varepsilon, 1\} = \varepsilon + 1$$

$$R_{11}{}^{1} = \{\varepsilon, 0, 00, 000, ...\} = 0^*$$

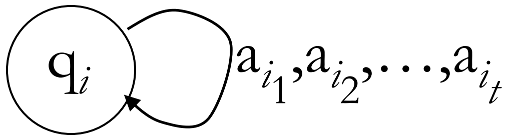
$$R_{12}{}^{1} = \{1, 01, 001, 0001, ...\} = 0^*1$$

General construction

- We inductively define R_{ij}^{k} as:

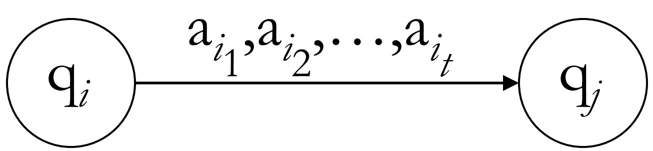
$$R_{ii}^{0} = a_{i_1} + a_{i_2} + \ldots + a_{i_t} + \varepsilon$$

(all loops around q_i and ε)



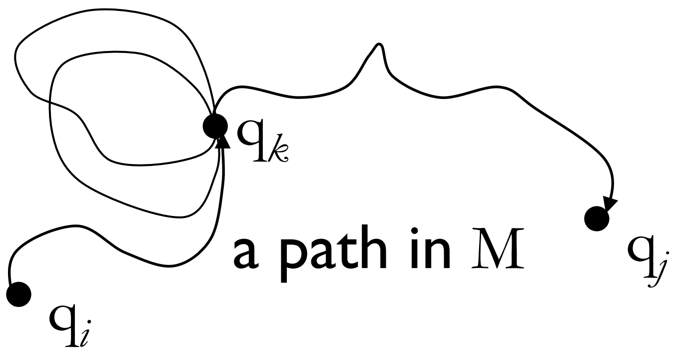
$$R_{ij}^{0} = a_{i_1} + a_{i_2} + \ldots + a_{i_t} \quad \text{if } i \neq j$$

(all $q_i \rightarrow q_j$)



$$R_{ij}^{k} = R_{ij}^{k-1} + R_{ik}^{k-1}(R_{kk}^{k-1})^{*}R_{kj}^{k-1}$$

(for $k > 0$)



Informal proof of correctness

- Each execution of the DFA using states $q_1, q_2, \ldots q_k$ will look like this:

state q_k is never visited

or

intermediate parts use only states $q_1, q_2, \ldots q_{k-1}$

$$q_i \to \ldots \to q_k \to \ldots \to q_k \to \ldots \to q_k \to \ldots \to q_j$$

$$R_{ij}^{k-1} \quad + \quad \underbrace{q_i \to \ldots \to q_k}_{R_{ik}^{k-1}} \underbrace{\to \ldots \to q_k \to \ldots \to q_k}_{(R_{kk}^{k-1})^*} \underbrace{\to \ldots \to q_j}_{R_{kj}^{k-1}}$$

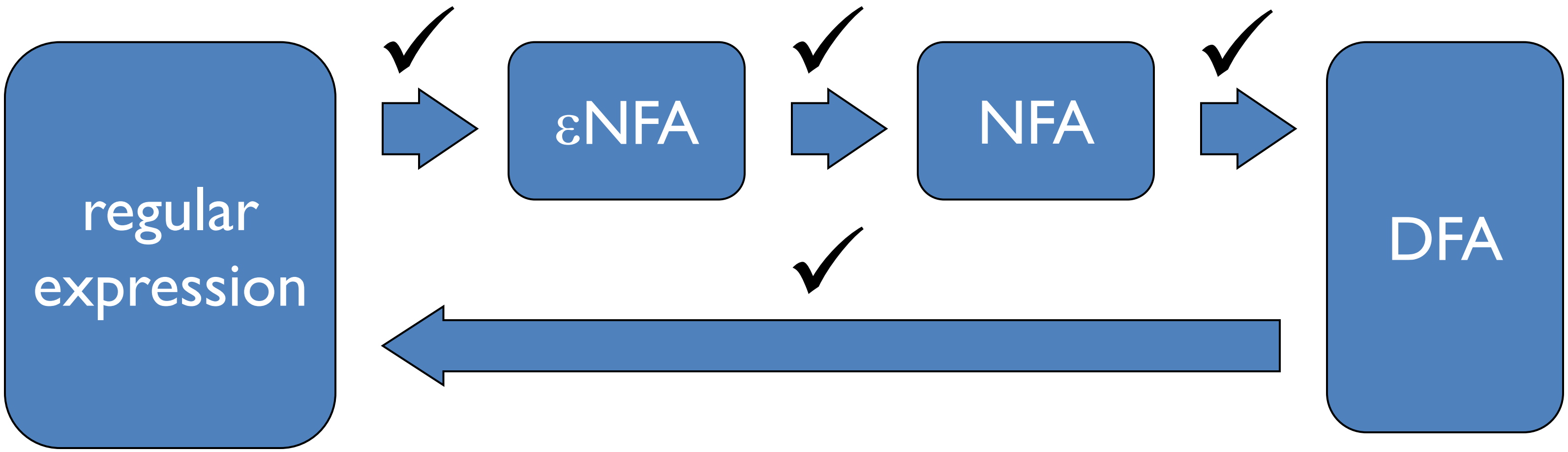
Final step

- Suppose the DFA start state is q_1, and the accepting states are $F = \{q_{j_1} \cup q_{j_2} \ldots \cup q_{j_t}\}$

- Then the regular expression for this DFA is

$$R_{1j_1}{}^{n} + R_{1j_2}{}^{n} + \ldots.. + R_{1j_t}{}^{n}$$

All models are equivalent



A language is regular iff it is accepted by a DFA, NFA, εNFA, or regular expression

Example

- Give a RE for the following DFA using this method: