Linguaggi formali

Let's start from the beginning

- A program is written in a programming language
- Every programming language (as every language in general) needs to obey its own rules
- We need to formally define languages…

Strings

- An alphabet is a finite set of symbols
- Examples

Σ_1 = {a, b, c, d, …, z}: the set of letters in Italian

Σ_2 = {0, 1}: the set of binary digits

Σ_3 = {(,)}: the set of open and closed brackets

A string over alphabet Σ is a finite sequence of symbols in Σ.

- Examples

abfbz is a string over Σ1 = {a, b, c, d, …, z}

11011 is a string over Σ2 = {0, 1}

))()(() is a string over Σ3 = {(,)}

The empty string is a string having no symbol, denoted by ε.

Strings

- The length of a string x is the number of symbols contained in the string x, denoted by |x|.

- Examples

  |abfbz|=5

  |110010|=6

  |))()(()|=7

  |ε|=0

Strings

- The concatenation of two strings x and y is a string xy,
  i.e., x is followed by y.
  it is an associative operation that admits the neutral element ε

- s is a substring of x if there exist strings y and z
  such that x = ysz.

Example:
the prefixes of abc are : ε, a, ab, abc

- In particular,
  when x = sz (substring with y=ε), s is called a prefix of x;
  when x = ys (substring with z=ε), s is called a suffix of x;

ε is a prefix and a suffix of ε and of all strings

Power of an alphabet

- We express the set of all strings over Σ of a given length

Σ^n denotes the strings of length n whose symbols are in Σ

Σ_2={0,1}

Σ_2^0={ε}

Σ_2^1={0,1}

Σ_2^2={00,01,11,10}

Σ_2^3 ={000,001,010,011, 100,101,110,111}

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \cup \ldots = \bigcup_{i>0} \Sigma^i \qquad \Sigma^* = \{\epsilon\} \cup \Sigma^+$$

Σ_2^+={0,1,00,01,11,10,000,001,010,011, 100,101,110,111....}

Languages

A language is a set of strings over an alphabet.

- Examples

L_1 = The set of all strings over Σ_1 that contain the substring "fool"

L_2 = The set of all strings over Σ_2 that are divisible by 7

= {7, 14, 21, …}

L_3 = The set of all strings over Σ_3 where every (is followed by 2 occurrences of)

= {(), ())),)(())), …}

Languages

- The following are operations on sets and hence also on languages.

  Union: $A \cup B$

  Intersection: $A \cap B$

  Difference: $A \setminus B$ ($A - B$ when $B \subseteq A$)

  Complement: $A = \Sigma^* - A$ where Σ^* is the set of all strings on alphabet Σ.

  Concatenation: $AB = \{ab \mid a \in A, b \in B\}$

Example: $\{0, 1\}\{1, 2\} = \{01, 02, 11, 12\}$.

Kleene Clousure

Kleene closure: $$A^* = \bigcup_{i=0}^{\infty} A^i$$

- Notation: $$A^+ = \bigcup_{i=1}^{\infty} A^i$$

Languages

Examples:

- The set of legal Italian words
- The set of C legal programs
- The set of strings with n 1's followed by n 0's
  {ε, 01, 0011, 000111, . . .}
- The set of strings with an equal number of 0's and 1's
  {ε, 01, 10, 0011, 0101, 1001, . . .}
- LP = The set of prime binary numbers
  {10, 11, 101, 111, 1011, . . .}
- The empty language ∅
- The language {ε} consisting of the empty string only

Footnotes : $\varnothing \neq \{\varepsilon\}$

Problems

- Does the string w belong to the language L?

  Example: 11101 $\in$ LP?

We want to define the ways to decide it!

We can try to generate all words….

We can try to recognise when a word belongs to a Language

Generating a language: Grammars

Starting from a particular initial symbol, using rewriting rules we generate the set of strings belonging to the language

Grammars

A Grammar (Σ, N, S, P) generates a language, where :

- the symbols are from alphabet Σ (called terminals)
- N is the set of nonterminals
- $S \in N$ is the starting symbol
- P is the set of productions, each of the form

$$U \rightarrow V$$

where $U \in (\Sigma \cup N)^+$ and $V \in (\Sigma \cup N)^*$.

A string (sequence of terminals) w is generated by G if there is a derivation of w using G, starting from the starting symbol S.

A language generated by grammar G, denoted L(G), is the set of strings derived using G

Grammar Example

L1 is the language of strings with an even number of 1's

L1 can be generated by a grammar ({0,1},{S,T},S,P) with P equal to

$S \rightarrow \varepsilon$

$S \rightarrow 0S$

$S \rightarrow 1T$

$T \rightarrow 0T$

$T \rightarrow 1S$

A string belongs to L1 iff it can be generated by the grammar

Grammar Example

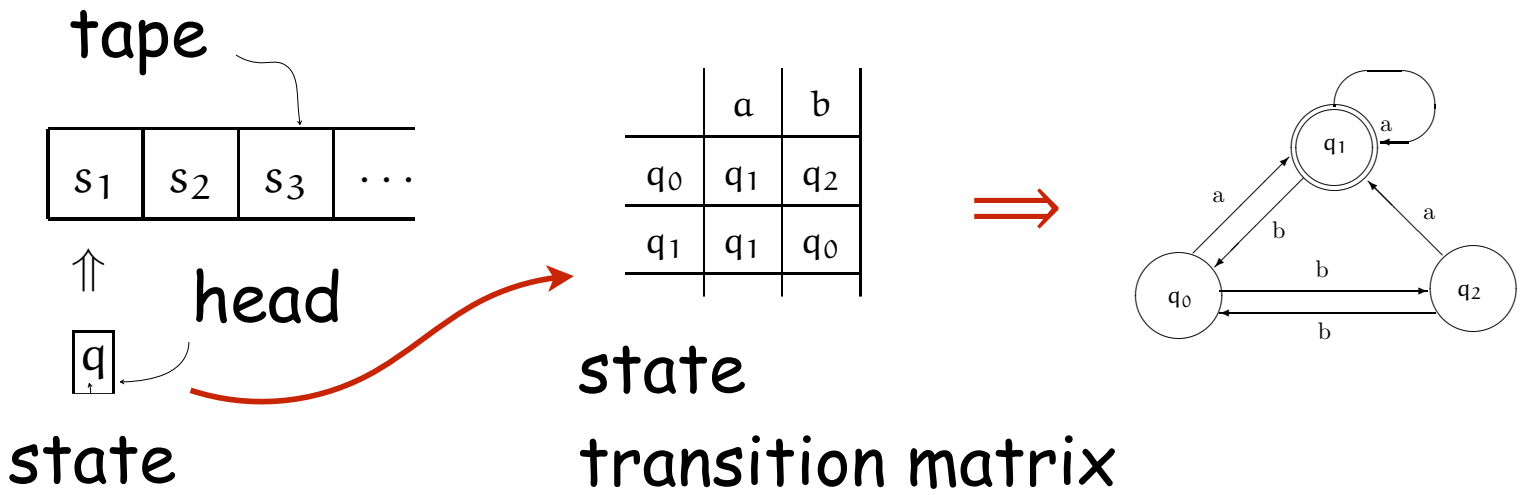
Does the string 01010 belong to L1?
We need to construct a derivation

$S \rightarrow \varepsilon \mid 0S \mid 1T$
$T \rightarrow 0T \mid 1S$

S

Recognising a language: Automata

- A finite state automaton is finite state machine with an input (eventually an output) of discrete values.
- The system may be in a state among a finite set of possible states. Being in a state allows him to keep track of previous history.



Back to our Problems

- Does the string w belong to the language L?

  We want to define ways to decide it!

- Which is the computational complexity necessary to answer to the previous question ?

It depends on the language!!

Grammars and Languages

Restrictions on productions give different types of grammars :

- Regular (type 3)
- Context-free (type 2)
- Context-sensitive (type 1)
- Phrase-structure (type 0)

For context-free, e.g., left side must be single nonterminal

No restrictions for phrase-structure

A language is of type i iff
there is a grammar of type i which generates it

Complexity of Languages Problems

	Regular Grammar Type 3	Context Free Grammar Type 2	Context Sensitive Grammar Type 1	Unrestricted Grammar Type 0
Is $W \in L(G)$?	P	P	PSPACE	U
Is L(G) empty?	P	P	U	U
Is $L(G1) \equiv L(G2)$?	PSPACE	U	U	U

P: decidable in polynomial time

PSPACE: decidable in polynomial space (at least as hard as NP-complete)

U: undecidable

Regular languages

All the following ways to represent regular languages are equivalent:

- Regular grammars (RG, type 3)
- Deterministic finite automata (DFA)
- Non-deterministic finite automata (NFA)
- Non-deterministic finite automata with ε transitions (ε-NFA)
- Regular expressions (RE)

Regular Grammars

A Right (or, analogously, Left) Grammar is a generative grammar, where

- every production has the form A-> aB| a
- only for the starting symbol S, we can have $S \rightarrow \varepsilon$

every non terminal symbol B is always preceded by a terminal one.

Example

G=({a,b}, {S,B},S,P) where productions P are:

S-> aS|aB

B->bB|b

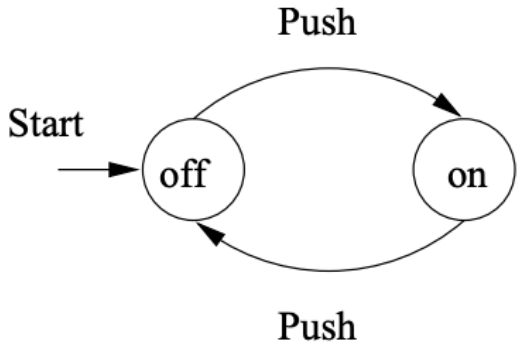
aaabb $\in$ L(G)

L(G)={ $a^n b^m$ | n,m>0}

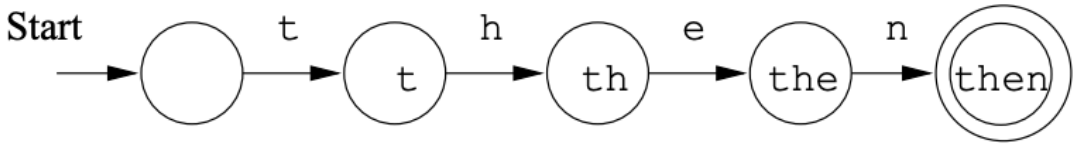
S

Deterministic Finite Automata

The states of a switch:



An automaton recognising the keyword then:



Deterministic Finite Automata

A deterministic finite automaton (DFA) $(Q, \Sigma, \delta, qo, F)$

Q a finite set of states

Σ a finite set Σ of symbols

δ : Q x Σ-> Q a transition function that takes as argument a state and a symbol and returns a state

qo the starting state

$F \subseteq Q$ the set of final or accepting states

Deterministic Finite Automata

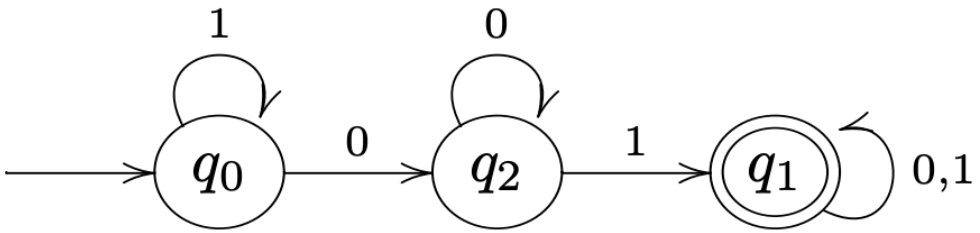
How to represent a DFA? With a transition table

	0	1
$\rightarrow q_0$	q_2	q_0
$*q_1$	q_1	q_1
q_2	q_2	q_1

-> indicates the starting state

* indicates the final states

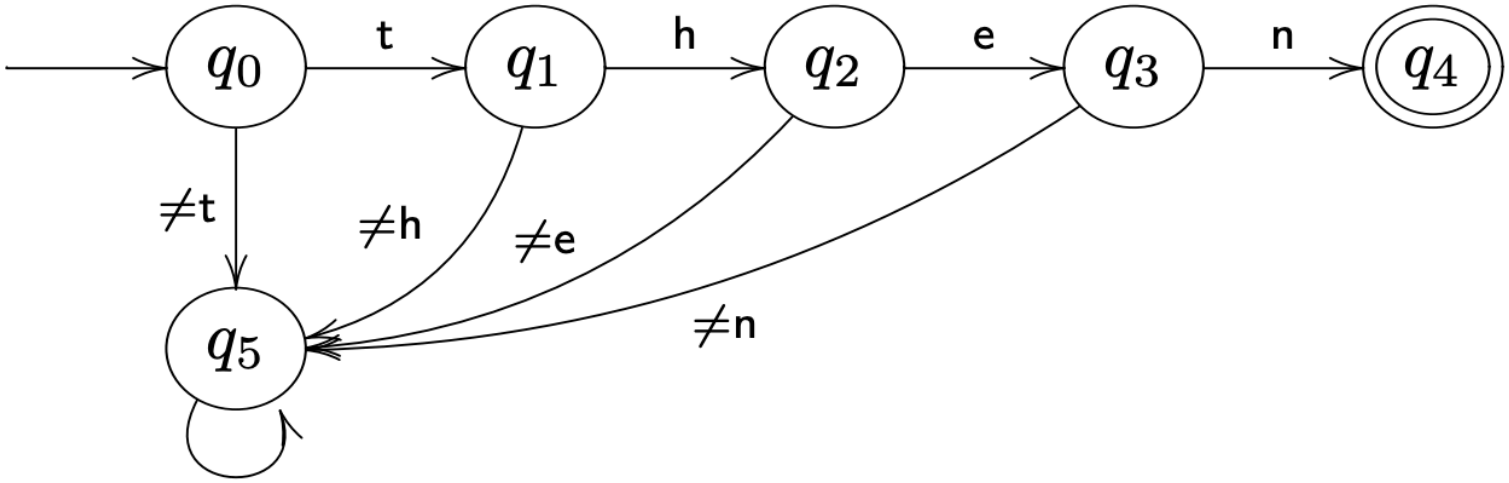
This defines the following transition diagram



Deterministic Finite Automata

When does an automaton accept a word?

It reads a word and accept it if it stops in an accepting state



here Q= $\{q_0, q_1, q_2, q_3, q_4, q_5\}$ F= $\{q_4\}$

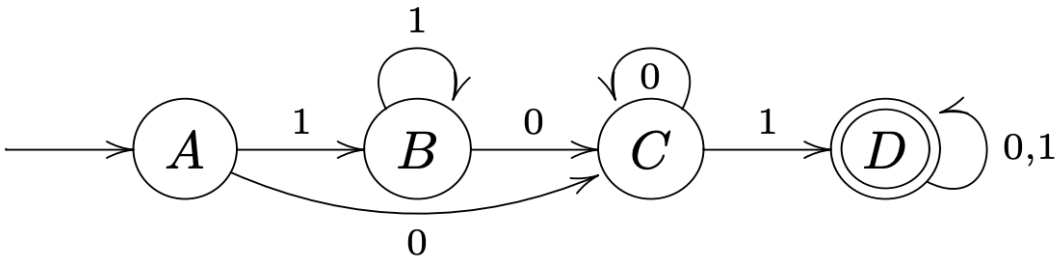
Only the word then is accepted

How DFA processes Strings

We build an automaton that accepts string containing the substring 01

Σ={0,1}
L={x01y| x,y$\in\Sigma^*$}

We get



	0	1
→A	C	B
B	C	B
C	C	D
*D	D	D

Extending the transition function to Strings

We define the transitive closure of δ

$$\hat{\delta} : Q \times \Sigma^* \longrightarrow Q$$

$$\begin{cases} \hat{\delta}(q, \varepsilon) = q \\ \hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a) \end{cases}$$

A string x is accepted by M=(Q, Σ, δ ,qo,F) iff $\hat{\delta}(q_0, x) \in F$

$$L(M) = \{x \in \Sigma^* | \hat{\delta}(q_0, x) \in F\}$$

Nondeterministic Finite Automata

A nondeterministic finite automaton (NFA) (Q, Σ, δ ,qo,F)

δ: Q x Σ-> $\wp(Q)$ a transition function that takes as argument a state and a symbol and returns a set of state s

Note that δ(q,a) can be equal $\emptyset$ to for some q and a.

We define the transitive closure of δ

$$\begin{cases} \hat{\delta}(q, \varepsilon) = \{q\} \\ \hat{\delta}(q, wa) = \bigcup_{p \in \hat{\delta}(q,w)} \delta(p, a) \end{cases}$$

□

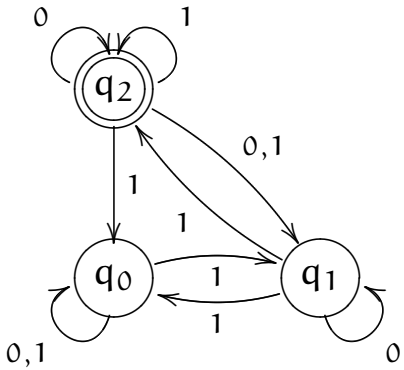
A string x is accepted by M=(Q, Σ, δ ,qo,F) iff $\widehat{\delta}(q_0, x) \cap F \neq \emptyset$

$$L(M) = \{x \in \Sigma^* | \widehat{\delta}(q_0, x) \cap F \neq \emptyset\}$$

Example

	0	1
→ q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
* q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$

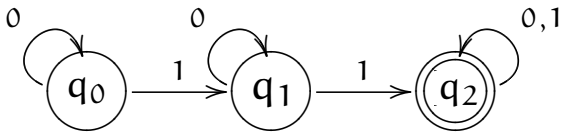
$F = \{q_2\}$.



NFA

L= $\{x \in \{0,1\}^* \mid$ x contains at least 2 occurrences of 1 $\}$

	0	1
→ q_0	q_0	q_1
q_1	q_1	q_2
* q_2	q_2	q_2



DFA

Different characterisation of Regular Languages

Remember? there are ways to characterise a regular language

- Regular grammars
- Deterministic Finite Automata
- Non Deterministic Finite Automata
- Epsilon Non deterministic Finite Automata
- Regular expression

Roadmap

DFA

NFA  RG

RE

ε-NFA

Equivalence between Regular Grammars and NFA

Theorem 1.
For each right grammar RG (or left grammar LG), there is NFA N such that L(RG)=L(N).

Construction Algorithm
For a given right grammar RG=(Σ, N, S, P) there is a corresponding NFA (N U{F}, Σ, δ,S, F') where F is a newly added state.
The transition function δ is defined by the following rules.

1) For any A->a belonging to P, with a in Σ, set δ(A,a)=F
2) For any A-> aB belonging to P, with a in Σ and B in N, set δ(A,a)=B

If S-> ε belongs to P then F'= {F}U{S} else F'= {F}.

Example

G=({a,b}, {S,B},S,P) where productions P are:

S-> aS|aB

B->bB|b L(G)={ $a^n b^m$ | n,m>0}

Equivalence between Regular Grammars and NFA

Theorem 2

For each NFA N, there is one right grammar RG (or left grammar LG) where L(RG)=L(N).

For a given finite automata N= $(Q, \Sigma, \delta, q_0, F)$, a corresponding right grammar RG=(Σ, Q, q_0', P) can be constructed using the following steps

1) for any $\delta(A,a)=B$ add $A \rightarrow aB$ to P,

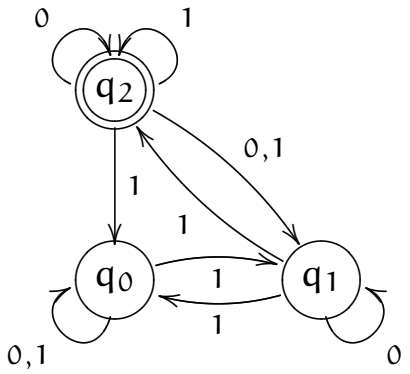
2) if B belongs to F add also $A \rightarrow a$ to P;

If q_0 belongs to F then add $q \rightarrow q_0 \mid \varepsilon$ to P and $q_0'=q$ else $q_0'=q_0$.

Example

	0	1
→ q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
* q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$

$F = \{q_2\}$.



NFA

L= $\{x \in \{0,1\}^* \mid$ x contains at least 2 occurrences of 1 $\}$

Roadmap

RE

ε-NFA

Equivalence between DFA and NFA

The NFA are usually easier to "program".

For each NFA N there is a DFA D, such that L (D) = L (N), and vice versa.

This involves a subset construction.

Given an

NFA N = $(Q_N, \Sigma, \delta_N, q_0, F_N)$

we will build a

DFA D = $(Q_D, \Sigma, \delta_D, q_0, F_D)$

such that

L (D) = L (N)

Equivalence between DFA and NFA

$$Q_D = \wp(Q_N),$$

Note that not all these state are necessary, most of them will be unreachable.

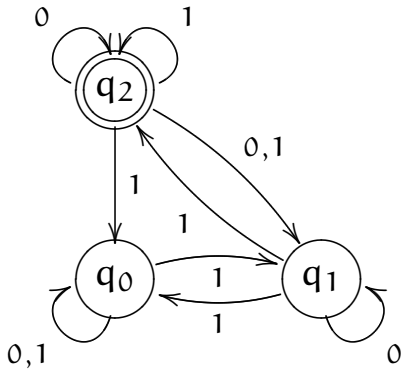
$$\delta_N(P, a) = \bigcup_{p \in P} \delta(p, a) \text{ for } P \in \wp(Q_N)$$

$$F_D = \{P \subseteq Q : P \cap F \neq \emptyset\}$$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



Consider all the subsets

$\wp(Q)$ $\quad$ $\emptyset$

$\{q_0\}$ $\quad$ $\{q_1\}$ $\quad$ $\{q_2\}$

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

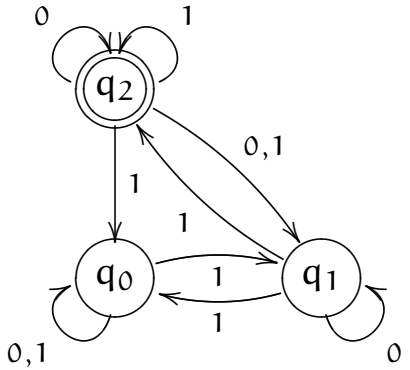
$\{q_0, q_1, q_2\}$

Which ones are final?

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



$\wp(Q)$ $\quad \emptyset$

$\{q_0\}$ $\quad \{q_1\}$ $\quad \{q_2\}$

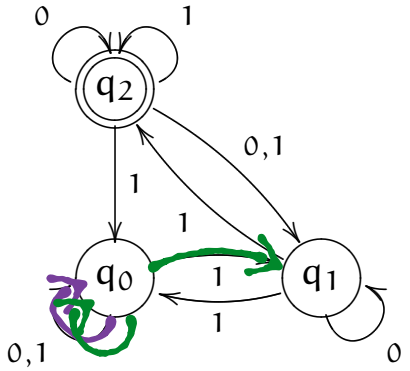
$\{q_0, q_1\}$ $\quad \{q_0, q_2\}$ $\quad \{q_1, q_2\}$

$\{q_0, q_1, q_2\}$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



$\wp(Q)$

$\emptyset$

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$

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

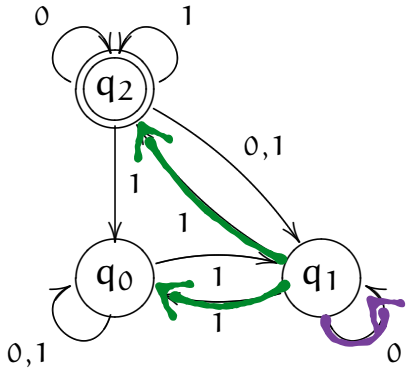
$\{q_0, q_1, q_2\}$

		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
q_0'	$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



$\wp(Q)$

$\emptyset$

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$

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

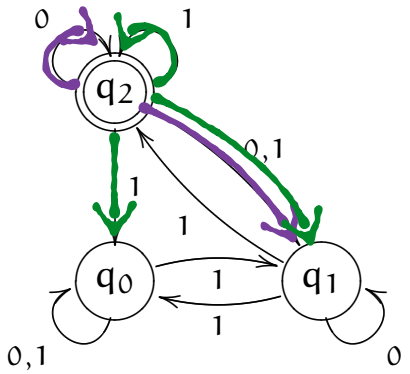
$\{q_0, q_1, q_2\}$

		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
q_0'	$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
q_1'	$\{q_1\}$	$\{q_1\}$	$\{q_0, q_2\}$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



$\wp(Q)$

$\emptyset$

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$

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

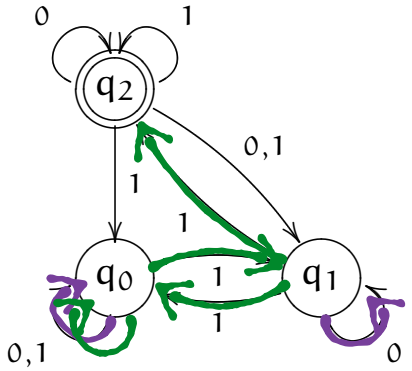
$\{q_0, q_1, q_2\}$

		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
q'_0	$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
q'_1	$\{q_1\}$	$\{q_1\}$	$\{q_0, q_2\}$
q'_2	$\{q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



$\wp(Q)$

$\emptyset$

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$

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

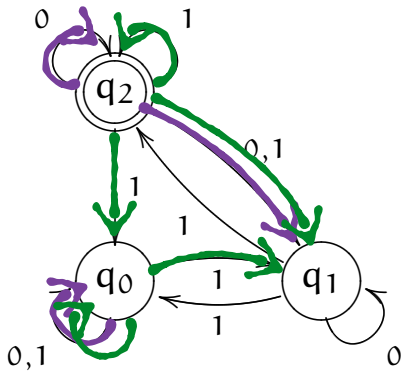
$\{q_0, q_1, q_2\}$

		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
q'_0	$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
q'_1	$\{q_1\}$	$\{q_1\}$	$\{q_0, q_2\}$
q'_2	$\{q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_3	$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



$\wp(Q)$

$\emptyset$

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$

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

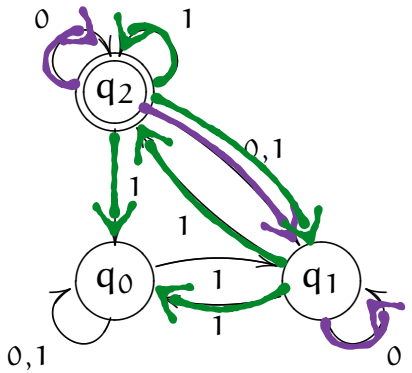
$\{q_0, q_1, q_2\}$

		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
q'_0	$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
q'_1	$\{q_1\}$	$\{q_1\}$	$\{q_0, q_2\}$
q'_2	$\{q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_3	$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$
q'_4	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



$\wp(Q)$

$\emptyset$

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$

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

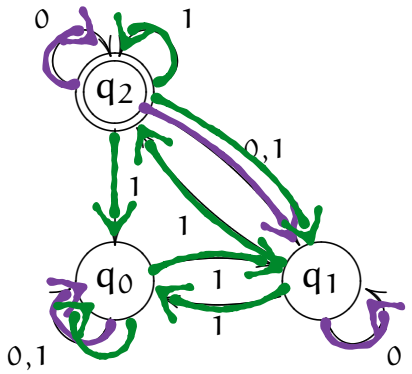
$\{q_0, q_1, q_2\}$

		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
q'_0	$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
q'_1	$\{q_1\}$	$\{q_1\}$	$\{q_0, q_2\}$
q'_2	$\{q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_3	$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$
q'_4	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_5	$\{q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$



$\wp(Q)$

$\emptyset$

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$

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

$\{q_0, q_1, q_2\}$

		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
q'_0	$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
q'_1	$\{q_1\}$	$\{q_1\}$	$\{q_0, q_2\}$
q'_2	$\{q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_3	$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$
q'_4	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_5	$\{q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_6	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$

Example

NFA

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_1\}$	$\{q_0, q_2\}$
q_2	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$

$\wp(Q)$

$\emptyset$

$\{q_0\}$ $\{q_1\}$ $\{q_2\}$

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

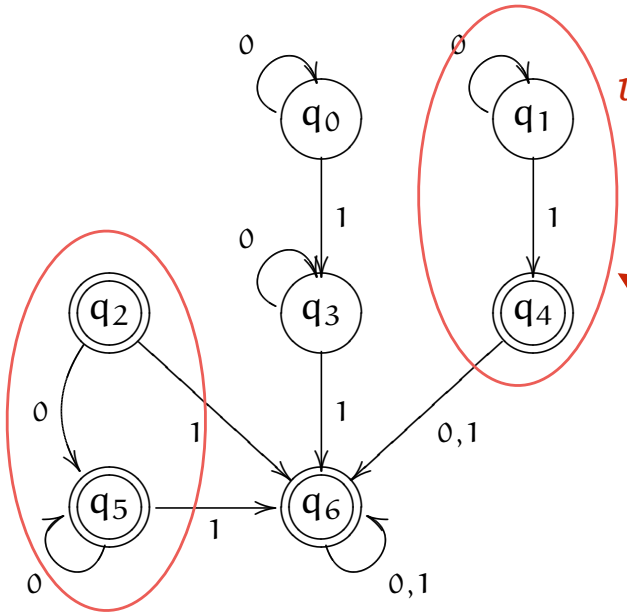
$\{q_0, q_1, q_2\}$

		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
q'_0	$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
q'_1	$\{q_1\}$	$\{q_1\}$	$\{q_0, q_2\}$
q'_2	$\{q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_3	$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$
q'_4	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_5	$\{q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q'_6	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$

DFA

	0	1
q_0	q_0	q_3
q_1	q_1	q_4
q_2	q_5	q_6
q_3	q_3	q_6
q_4	q_6	q_6
q_5	q_5	q_6
q_6	q_6	q_6

Example

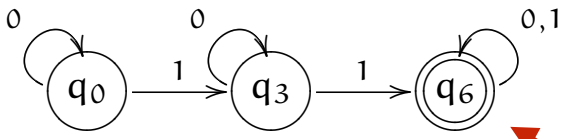


DFA with unreachable states

DFA

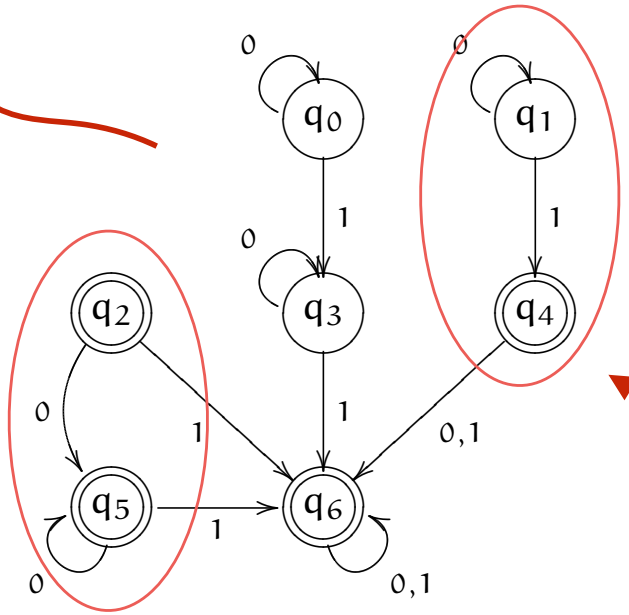
	0	1
q_0	q_0	q_3
q_1	q_1	q_4
q_2	q_5	q_6
q_3	q_3	q_6
q_4	q_6	q_6
q_5	q_5	q_6
q_6	q_6	q_6

Example



minimum DFA

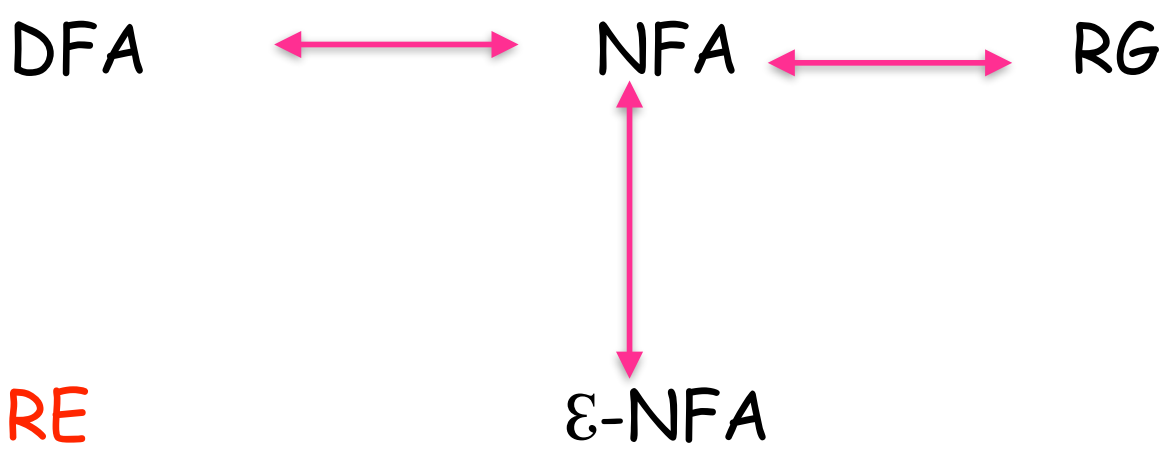
DFA with unreachable states



DFA

	0	1
q0	q0	q3
q1	q1	q4
q2	q5	q6
q3	q3	q6
q4	q6	q6
q5	q5	q6
q6	q6	q6

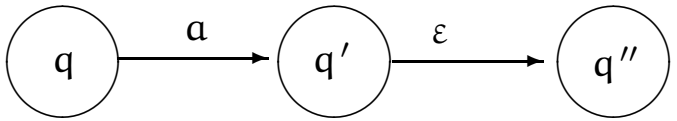
Roadmap



ε-NFA

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow \wp(Q)$$

$\varepsilon\text{-}closure$ applied to a state gives all the states reachable with ε-transitions



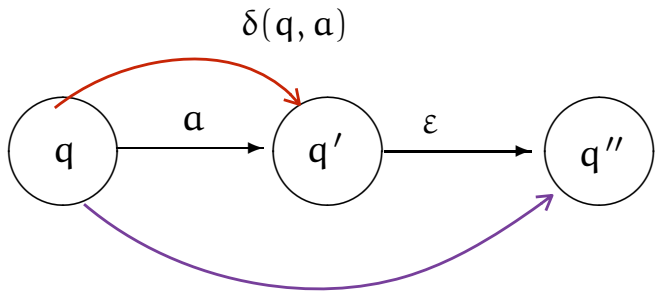
ϵ-closure(q)=$\{q\}$ ϵ-closure(q')=$\{q', q''\}$

$$\varepsilon\text{-closure}(\mathrm{P}) = \bigcup_{\mathrm{p} \in \mathrm{P}} \varepsilon\text{-closure}(\mathrm{p})$$

The transitive closure of δ

$$\hat{\delta} : Q \times \Sigma^* \longrightarrow \wp(Q)$$

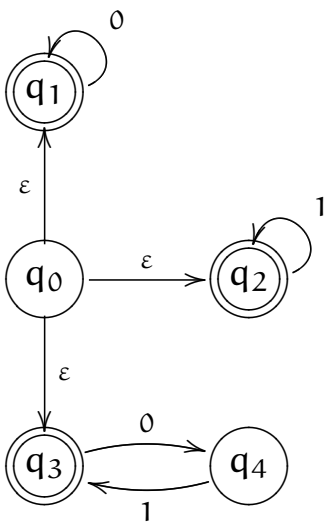
$$\begin{cases} \hat{\delta}(q, \varepsilon) &= \varepsilon\text{-closure}(q) \\ \hat{\delta}(q, wa) &= \bigcup_{p \in \hat{\delta}(q,w)} \varepsilon\text{-closure}(\delta(p, a)) \end{cases}$$



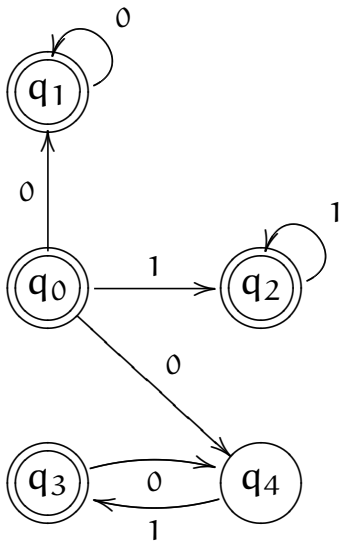
$$\hat{\delta}(q, a) = \bigcup_{p \in \hat{\delta}(q,\varepsilon)} \varepsilon\text{-closure}(\delta(p, a)) = \{q', q''\}$$

Example

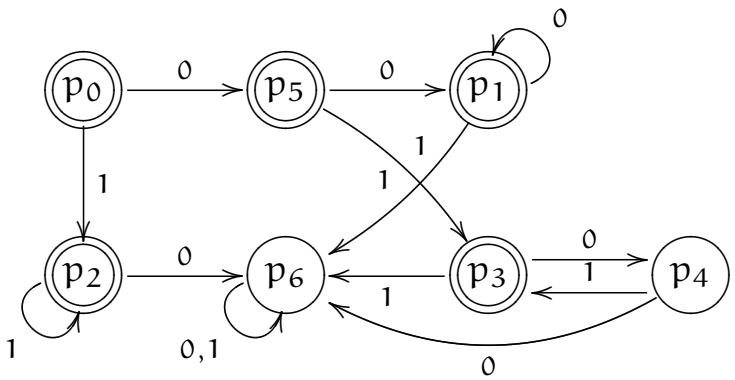
ε-NFA



NFA



DFA



$$L = \{\ x \mid \exists n \in \mathbb{N}.\ x = 0^n \ \vee\ x = 1^n \ \vee\ x = (01)^n\ \}$$

Equivalence between ε-NFA and NFA

For each ε-NFA E there is a NFA N, such that L (E) = L (N), and vice versa.

Given an

ε-NFA E = $(Q, \Sigma, \delta_E, q_0, F_E)$

we will build a

NFA N = $(Q, \Sigma, \delta_N, q_0, F_N)$

such that

L (D) = L (N)

Equivalence between ε-NFA and NFA

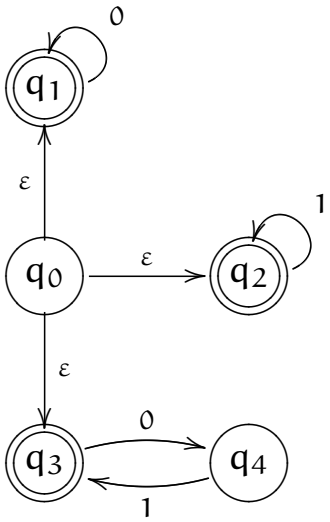
$$\delta_N(q,a) = \widehat{\delta}_E(q,a)$$

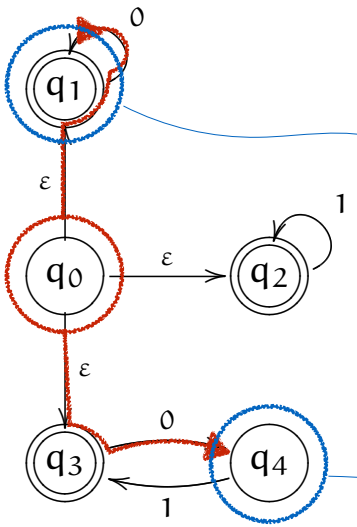
$$F_N = \begin{cases} F_E \cup \{q_0\} & \text{if } \epsilon\text{-closure}(q_0) \cap F_E \neq \emptyset \\ F_E & \text{otherwise} \end{cases}$$

(if a final state can be reached with an epsilon transition from the initial state)

Example

ε-NFA

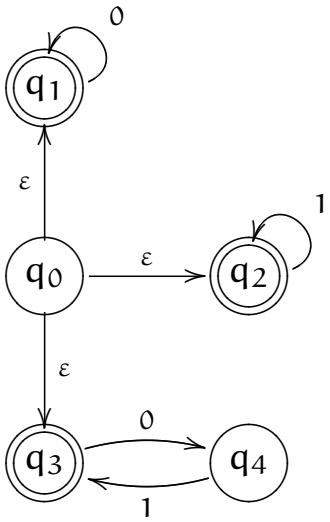


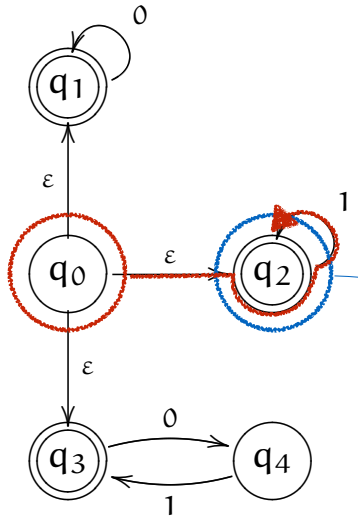


	0	1
q_0	$\{q_1, q_4\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\emptyset$
q_2	$\emptyset$	$\{q_2\}$
q_3	$\{q_4\}$	$\emptyset$
q_4	$\emptyset$	$\{q_3\}$

Example

ε-NFA

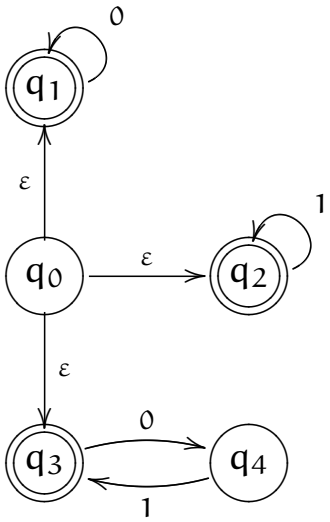


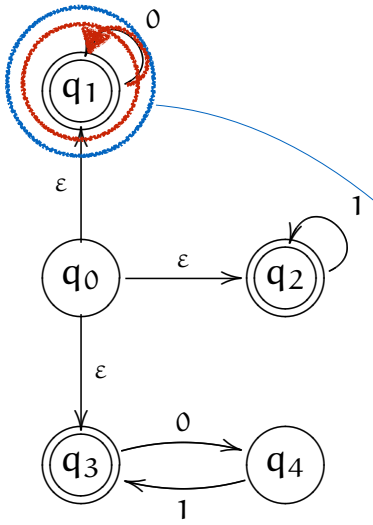


	0	1
q_0	$\{q_1, q_4\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\emptyset$
q_2	$\emptyset$	$\{q_2\}$
q_3	$\{q_4\}$	$\emptyset$
q_4	$\emptyset$	$\{q_3\}$

Example

ε-NFA

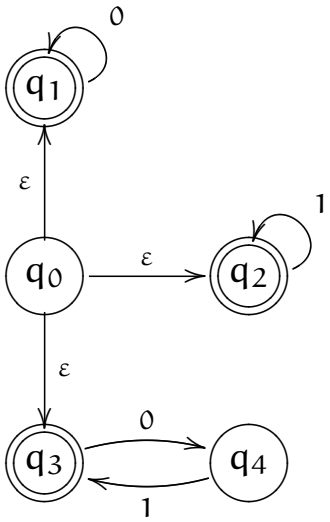


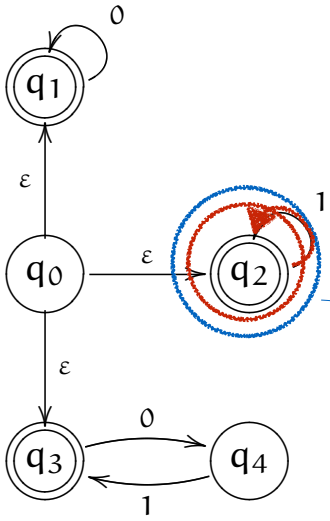


	0	1
q_0	$\{q_1, q_4\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\emptyset$
q_2	$\emptyset$	$\{q_2\}$
q_3	$\{q_4\}$	$\emptyset$
q_4	$\emptyset$	$\{q_3\}$

Example

ε-NFA

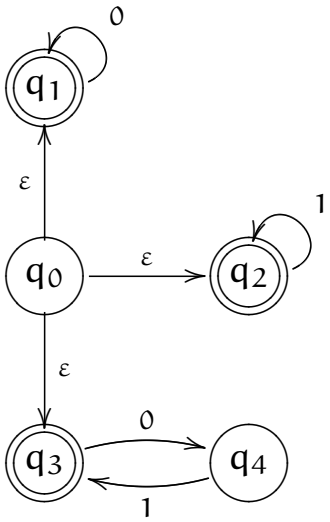


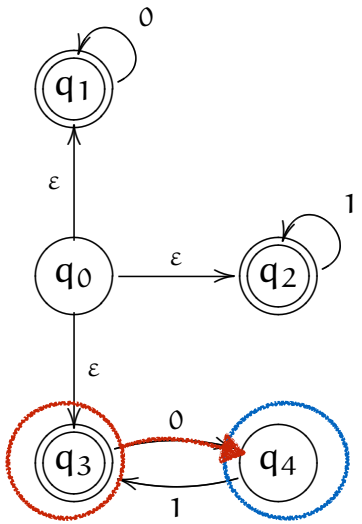


	0	1
q_0	$\{q_1, q_4\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\emptyset$
q_2	$\emptyset$	$\{q_2\}$
q_3	$\{q_4\}$	$\emptyset$
q_4	$\emptyset$	$\{q_3\}$

Example

ε-NFA

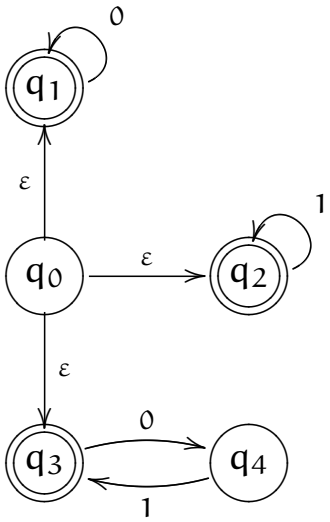


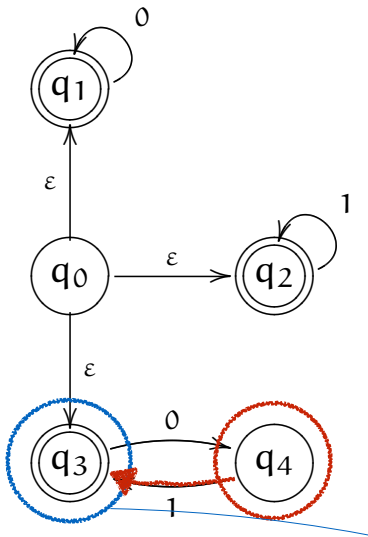


	0	1
q_0	$\{q_1, q_4\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\emptyset$
q_2	$\emptyset$	$\{q_2\}$
q_3	$\{q_4\}$	$\emptyset$
q_4	$\emptyset$	$\{q_3\}$

Example

ε-NFA

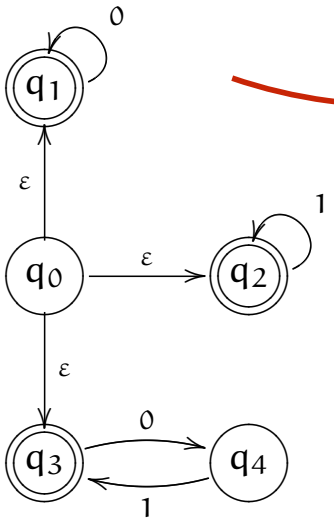




	0	1
q_0	$\{q_1, q_4\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\emptyset$
q_2	$\emptyset$	$\{q_2\}$
q_3	$\{q_4\}$	$\emptyset$
q_4	$\emptyset$	$\{q_3\}$

Example

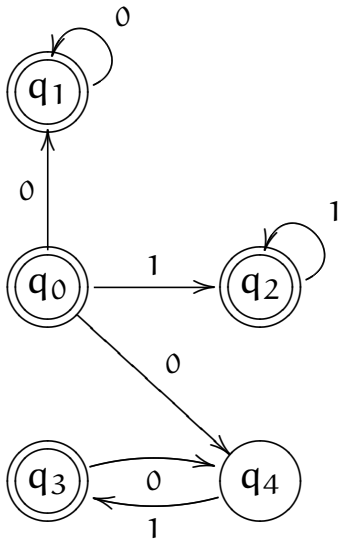
ε-NFA



NFA

	0	1
q_0	$\{q_1, q_4\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\emptyset$
q_2	$\emptyset$	$\{q_2\}$
q_3	$\{q_4\}$	$\emptyset$
q_4	$\emptyset$	$\{q_3\}$

NFA



Operations on languages: recap.

Union: $A \cup B$

Intersection: $A \cap B$

Difference: $A \setminus B$

Complement: $A = \Sigma^* - A$

Concatenation: $AB = \{ab \mid a \in A, b \in B\}$

Kleene Clousure: $$A^* = \bigcup_{i=0}^{\infty} A^i$$