Automi

- Gli automi (singolare automa) sono dispositivi matematici astratti che possono:
  - determinare l'appartenenza di una stringa ad un insieme di stringhe
  - trasformare una stringa in un'altra stringa
- Hanno tutti gli aspetti di un computer:
  - input e output
  - memoria
  - capacità di prendere decisioni
  - trasformare l'input in output

Linguaggi Formali

- Quindi in sostanza tutti i processi computazionali possono essere ridotti ad uno tra:
  - Determinazione dell'appartenenza a un insieme (di stringhe)
  - Mappatura tra insiemi (di stringhe)
- Formalizzeremo il concetto di computazione meccanica:
  - dando una definizione precisa del termine "algoritmo"
  - caratterizzando i problemi che sono o non sono adatti per essere risolti da un calcolatore.

- Concetti chiave:

Alfabeto: Insieme finito e non vuoto di simboli

- **Esempio:** $\Sigma = \{0, 1\}$ alfabeto binario
- **Esempio:** $\Sigma = \{a, b, c, \ldots, z\}$ insieme di tutte le lettere minuscole
- **Esempio:** Insieme di tutti i caratteri ASCII

Stringa: (o parola) Sequenza finita di simboli da un alfabeto Σ, e.g. 0011001

Stringa vuota: La stringa con zero occorrenze di simboli da Σ

- La stringa vuota è denotata con ε

Lunghezza di una stringa: Numero di simboli nella stringa.

- $|w|$ denota la lunghezza della stringa w
- $|0110| = 4$, $|\varepsilon| = 0$

- Potenze di un alfabeto: Σ^k = insieme delle stringhe di lunghezza k con simboli da Σ
  - Esempio: $\Sigma = \{0, 1\}$

$$\Sigma^0 = \{\varepsilon\}$$
$$\Sigma^1 = \{0, 1\}$$
$$\Sigma^2 = \{00, 01, 10, 11\}$$

  - Domanda: Quante stringhe ci sono in Σ^3?
- L'insieme di tutte le stringhe su Σ è denotato da Σ^*

$$\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i$$

- **Linguaggio:** dato un alfabeto Σ, chiamiamo linguaggio ogni sottoinsieme $L \subseteq \Sigma^*$

- Automi DFA

Un Automa a Stati Finiti Deterministico (DFA) è una quintupla

$$A = (Q, \Sigma, \delta, q_0, F)$$

- Q è un insieme finito di stati
- Σ è un alfabeto finito (= simboli in input)
- $\delta : Q \times \Sigma \mapsto Q$ è una funzione di transizione
- $q_0 \in Q$ è lo stato iniziale
- $F \subseteq Q$ è un insieme di stati finali

Possiamo rappresentare gli automi sia come diagramma di transizione che come tabella di transizione.

Si consideri che ci serve la formalità; i disegni sono utili per capire come combinare gli stati.

Sotto, si consideri un automa che accetta il linguaggio delle stringhe con 01 come sottostringa:

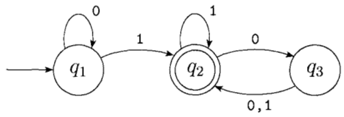


FIGURE 1.6
The finite automaton M_1

We can describe M_1 formally by writing $M_1 = (Q, \Sigma, \delta, q_1, F)$, where

1. $Q = \{q_1, q_2, q_3\}$,
2. $\Sigma = \{0,1\}$,
3. δ is described as

	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2

4. q_1 is the start state, and
5. $F = \{q_2\}$.

Servirà *poi* questo:

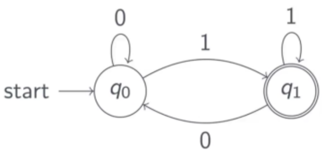
- Data una parola $w = w_1 w_2 \dots w_n$, la computazione dell'automa A con input w è una sequenza di stati $r_0 r_1 \dots r_n$ che rispetta due condizioni:
  1. $r_0 = q_0$ (inizia dallo stato iniziale)
  2. $\delta(r_i, w_{i+1}) = r_{i+1}$ per ogni $i = 0, \dots, n-1$ (rispetta la funzione di transizione)
- Diciamo che la computazione accetta la parola w se:
  3. $r_n \in F$ (la computazione termina in uno stato finale)
- Un DFA A accetta la parola w se la computazione accetta w
- Formalmente, il linguaggio accettato da A è

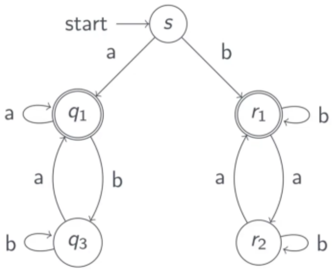
$$L(A) = \{w \in \Sigma^* \mid A \text{ accetta } w\}$$

- I linguaggi accettati da automi a stati finiti sono detti linguaggi regolari

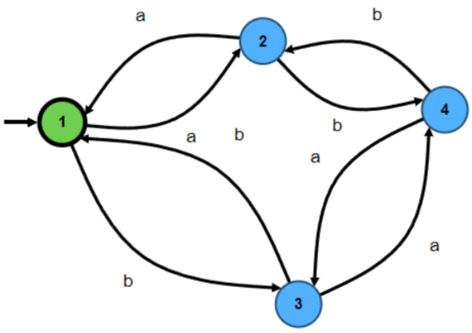
Eventuali esempi:

- Parole che terminano con 1 (left) e parola che inizia/finisce con "a" o con "b" (right)





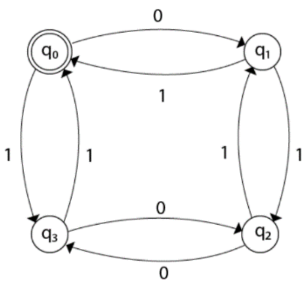
Automa DFA con alfabeto {0, 1} che ha come linguaggio: tutte e sole le stringhe con un numero pari di "a" e di "b"



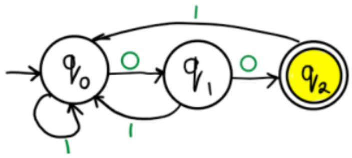
Esercizi:

DFA per i seguenti linguaggi sull'alfabeto $\{0, 1\}$:

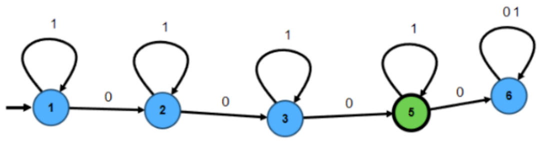
- Insieme di tutte e sole le stringhe con un numero pari di zeri e un numero pari di uni



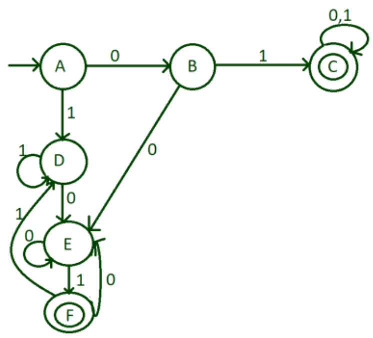
Insieme di tutte le stringhe che finiscono con 00



- Insieme di tutte le stringhe che contengono esattamente tre zeri (anche non consecutivi)



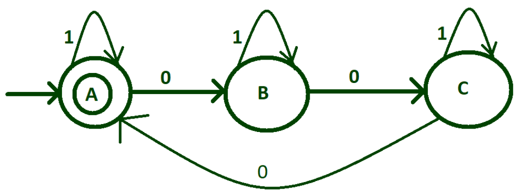
- Insieme delle stringhe che cominciano o finiscono (o entrambe le cose) con 01



Costruite un DFA che riconosce il linguaggio

$$L_1 = \{w \in \{0, 1\}^* \mid w \text{ contiene un numero di 0 multiplo di 3}\}$$

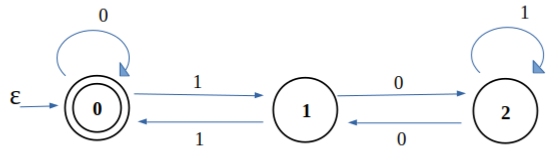
Per esempio, 000, 00110 e 010101010101 appartengono al linguaggio perché contengono rispettivamente 3, 3 e 6 zeri, mentre 00, 001010 e 0101010101, che contengono 2, 4 e 5 zeri, non appartengono al linguaggio.



2. Costruite un DFA che riconosce il linguaggio

$$L_2 = \{w \in \{0, 1\}^* \mid w \text{ è la codifica binaria di un mumero multiplo di 3}\}$$

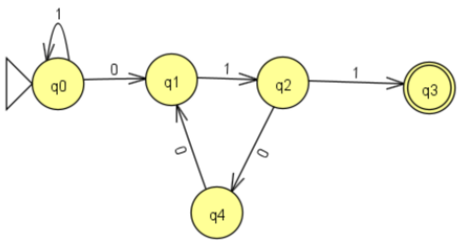
Per esempio, 11, 110 e 1001 appartengono al linguaggio perché sono le codifiche binarie di 3, 6 e 9, mentre 10, 111 e 1011 non appartengono al linguaggio perché sono le codifiche binarie di 2, 7 e 11. La stringa vuota non codifica nessun numero.



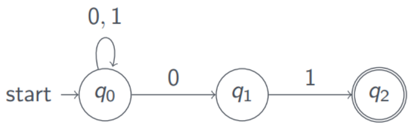
Infatti, quando un numero viene diviso per 3, ci sono solo 3 possibilità. Il resto può essere 0, 1 o 2. In questo caso, lo stato 0 rappresenta che il resto quando il numero è diviso per 3 è 0. Lo stato 1 rappresenta che il resto quando il numero è diviso per 3 è 1 e, analogamente, lo stato 2 rappresenta che il resto quando il numero è diviso per 3 è 2. Quindi, se una stringa raggiunge lo stato 0 alla fine, viene accettata altrimenti rifiutata.

4. $\{w \in \Sigma^* \mid \text{ogni 0 è seguito da 11}\}$

4)



Cosa fa questo automa?



È un esempio di automa a stati finiti non deterministico:

- può trovarsi contemporaneamente in più stati diversi

- Operazioni regolari

- **Intersezione:**
$$L \cap M = \{w : w \in L \text{ e } w \in M\}$$
- **Unione:**
$$L \cup M = \{w : w \in L \text{ oppure } w \in M\}$$
- **Complemento:**
$$\overline{L} = \{w : w \notin L\}$$
- **Concatenazione:**
$$L.M = \{uv : u \in L \text{ e } v \in M\}$$
- **Chiusura (o Star) di Kleene:**
$$L^* = \{w_1 w_2 \dots w_k : k \geq 0 \text{ e ogni } w_i \in L\}$$

Theorem

Se L e M sono regolari, allora anche $L \cap M$ è un linguaggio regolare.

Dimostrazione. Sia L il linguaggio di
$$A_L = (Q_L, \Sigma, \delta_L, q_L, F_L)$$
e M il linguaggio di
$$A_M = (Q_M, \Sigma, \delta_M, q_M, F_M)$$
Possiamo assumere che entrambi gli automi siano deterministici
Costruiremo un automa che simula A_L e A_M in parallelo, e accetta se e solo se sia A_L che A_M accettano.

Dimostrazione (continua).
Se A_L va dallo stato p allo stato s leggendo a, e A_M va dallo stato q allo stato t leggendo a, allora $A_{L\cap M}$ andrà dallo stato (p, q) allo stato (s, t) leggendo a.
$A_{L\cap M}$ accetta una parola solo quando sia A_L che A_M accettano
$$\Leftrightarrow$$
$A_{L\cap M}$ accetta solo quando (p, q) è una coppia di stati finali

Formalmente
$$A_{L\cap M} = (Q_L \times Q_M, \Sigma, \delta_{L\cap M}, (q_L, q_M), F_L \times F_M),$$
dove
$$\delta_{L\cap M}((p, q), a) = (\delta_L(p, a), \delta_M(q, a))$$

- Automi NFA

Un Automa a Stati Finiti Non Deterministico (NFA) è una quintupla
$$A = (Q, \Sigma, \delta, q_0, F)$$

- Q è un insieme finito di stati
- Σ è un alfabeto finito (= simboli in input)
- δ è una funzione di transizione che prende in input (q, a) e restituisce un sottoinsieme di Q
- $q_0 \in Q$ è lo stato iniziale
- $F \subseteq Q$ è un insieme di stati finali

L'NFA che riconosce le parole che terminano con 01 è

$$A = (Q, \{0, 1\}, \delta, q_0, \{q_2\})$$

dove δ è la funzione di transizione

	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
$*q_2$	$\emptyset$	$\emptyset$

Data una parola $w = w_1 w_2 \dots w_n$, una computazione di un NFA A con input w è una sequenza di stati $r_0 r_1 \dots r_n$ che rispetta due condizioni:

1. $r_0 = q_0$ (inizia dallo stato iniziale)
2. $\delta(r_i, w_{i+1}) = r_{i+1}$ per ogni $i = 0, \dots, n-1$ (rispetta la funzione di transizione)

Diciamo che una computazione accetta la parola w se:

3. $r_n \in F$ (la computazione termina in uno stato finale)

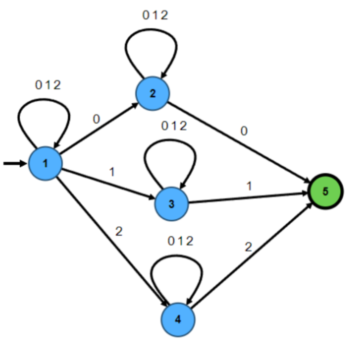
A causa del nondeterminismo, ci può essere più di una computazione per ogni parola!

- Un NFA A accetta la parola w se esiste una computazione che accetta w
- Un NFA A rifiuta la parola w se tutte le computazioni la rifiutano
- Formalmente, il linguaggio accettato da A è

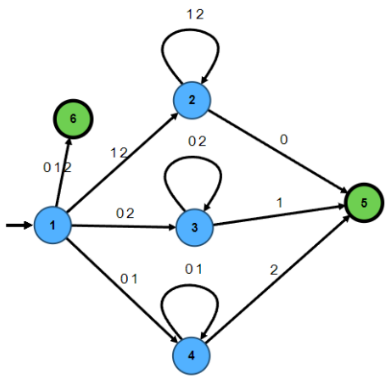
$$L(A) = \{w \in \Sigma^* \mid A \text{ accetta } w\}$$

Definire degli automi a stati finiti non deterministici che accettino i seguenti linguaggi:

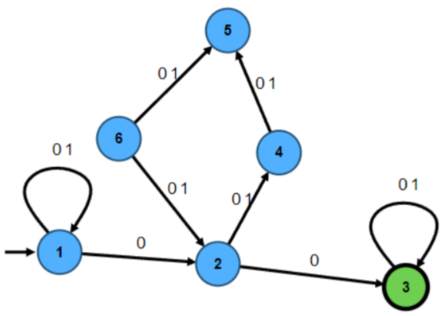
- L'insieme delle parole sull'alfabeto $\{0, 1, \dots, 9\}$ tali che la cifra finale sia comparsa in precedenza



L'insieme delle parole sull'alfabeto $\{0, 1, \dots, 9\}$ tali che la cifra finale *non* sia comparsa in precedenza



L'insieme delle parole di 0 e 1 tali che esistono due 0 separati da un numero di posizioni multiplo di 4 (0 è un multiplo di 4)



Sorprendentemente, NFA e DFA sono in grado di riconoscere gli stessi linugaggi

Per ogni NFA N c'è un DFA D tale che $L(D) = L(N)$, e viceversa

L'equivalenza di dimostra mediante una costruzione a sottoinsiemi:
Dato un NFA

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

costruiremo un DFA

$$D = (Q_D, \Sigma, S_0, \delta_D, F_D)$$

tale che

$$L(D) = L(N)$$

$Q_D = \{S : S \subseteq Q_N\}$
Ogni stato del DFA corrisponde ad un insieme di stati dell'NFA

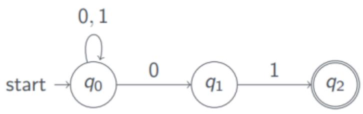
$S_0 = \{q_0\}$
Lo stato iniziale del DFA è l'insieme che contiene solo q_0

$F_D = \{S \subseteq Q_N : S \cap F_N \neq \emptyset\}$
Uno stato del DFA è finale se c'è almeno uno stato finale corrispondente nell'NFA

Per ogni $S \subseteq Q_N$ e per ogni $a \in \Sigma$

$$\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a)$$

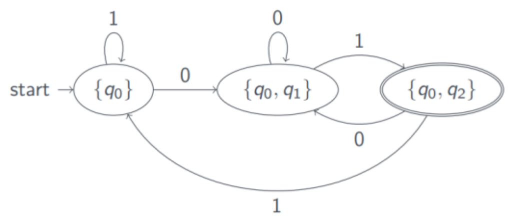
La funzione di transizione "percorre tutte le possibili strade"



Costruiamo δ_D per l'NFA qui sopra:

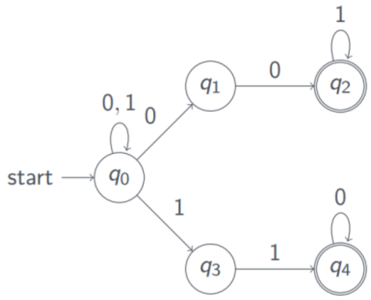
	0	1
$\emptyset$	$\emptyset$	$\emptyset$
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	$\emptyset$	$\{q_2\}$
$*\{q_2\}$	$\emptyset$	$\emptyset$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*\{q_1, q_2\}$	$\emptyset$	$\{q_2\}$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

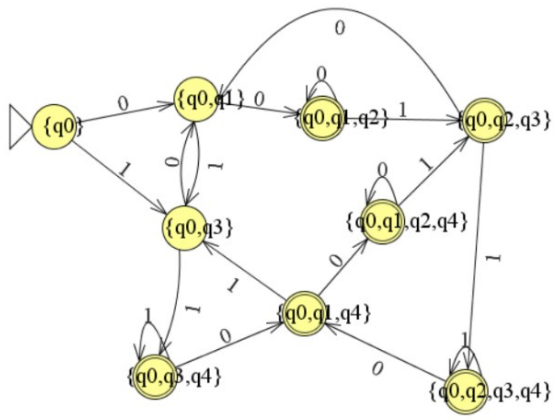
La tabella di transizione per D ci permette di ottenere il diagramma di transizione



Nota: $|Q_D| = 2^{|Q_N|}$, anche se alcuni degli stati in Q_D possono essere "inutili", cioè non raggiungibili dallo stato iniziale. In questo caso solo tre stati sono raggiungibili, e gli altri possono essere omessi.

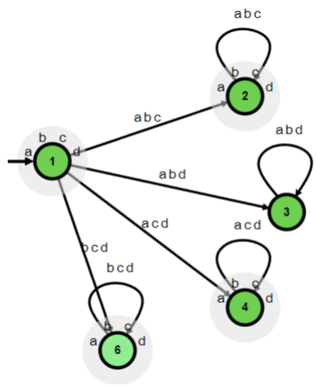
Trasformare il seguente NFA in DFA





Nota: con tabella si arriva là.

Automa NFA con alfabeto {a, b, c, d} che ha come linguaggio le stringhe in cui:
uno dei simboli dell'alfabeto non compare mai



Trasforma l'NFA in DFA usando la costruzione per sottoinsiemi.

- <u>Automi ϵ-NFA (occhio: non fa più distinzione tra epsilon-NFA e DFA)</u>

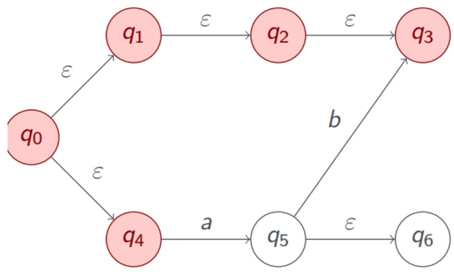
Un Automa a Stati Finiti Non Deterministico con ε-transizioni (ε-NFA) è una quintupla

$$A = (Q, \Sigma, \delta, q_0, F)$$

dove:

- Q, Σ, q_0, F sono definiti come al solito
- δ è una funzione di transizione che prende in input:
  - uno stato in Q
  - un simbolo nell'alfabeto $\Sigma \cup \{\varepsilon\}$

  e restituisce un sottoinsieme di Q



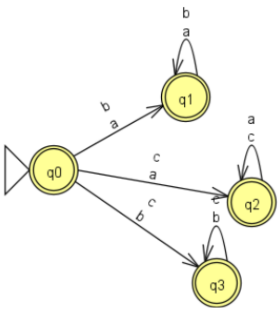
$$\mathrm{ECLOSE}(q_0) = \{q_0, q_1, q_4, q_2, q_3\}$$

NFA e ε-NFA

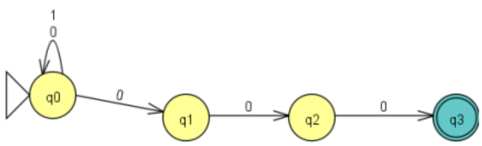
Per ognuno dei seguenti linguaggi, costruisci un ε-NFA che accetti il linguaggio.

1. $\{w \in \{a, b, c\}^* \mid$ non compaiono tutti i simboli$\}$
2. $\{w \in \{0, 1\}^* \mid$ contiene almeno tre 000 consecutivi$\}$
3. $\{w \in \{0, 1\}^* \mid$ contiene al suo interno la stringa 11 oppure 101$\}$

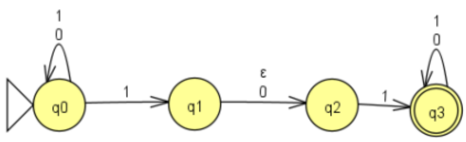
1)



2)



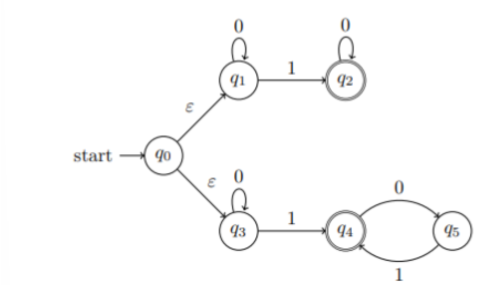
3)



Conversione NFA → DFA

Trasforma ciascuno dei seguenti ε-NFA in DFA usando la costruzione per sottoinsiemi.

1.



1) Primo step: calcolare le ε-chiusure
In questo caso avremmo (q0,q1) e (q0,q3). Lo stato iniziale sarà quindi (q0,q1,q3).
ENCLOSE (q0) = {q0,q1,q3}

2) Si calcola poi come sempre la tabella di transizione; sempre per unione e osservazione dello stato attuale e stati precedenti, in maniera complementare a quanto visto sopra.
Consiglio: mettere subito lo stato vuoto, perché come si vede servirà nel caso ci siano altri stati vuoti che lo raggiungono.

	0	1
Ø	Ø	Ø
-> {q0,q1,q3}	{q0,q1,q3}	{q2,q4}
{q1,q3}	{q1,q3}	{q2,q4}
*{q2,q4}	{q2,q5}	Ø
*{q2,q5}	{q2}	{q4}
{q2}	{q2}	Ø
{q4}	{q5}	Ø
{q5}	Ø	{q4}