



SAPIENZA  
UNIVERSITÀ DI ROMA

“SAPIENZA” UNIVERSITÀ DI ROMA  
INGEGNERIA DELL'INFORMAZIONE,  
INFORMATICA E STATISTICA  
DIPARTIMENTO DI INFORMATICA

---

# Automi, Calcolabilità e Complessità

---

Appunti integrati con il libro "Introduzione alla teoria  
della computazione", Michael Sipser

*Author*  
Simone Bianco

30 settembre 2023

# Indice

Informazioni e Contatti	1
1 Automi e Linguaggi	2

# Informazioni e Contatti

Appunti e riassunti personali raccolti in ambito del corso di *Automi, Calcolabilità e Complessità* offerto dal corso di laurea in Informatica dell'Università degli Studi di Roma "La Sapienza".

Ulteriori informazioni ed appunti possono essere trovati al seguente link:

<https://github.com/Exyss/university-notes>. Chiunque si senta libero di segnalare incorrettezze, migliorie o richieste tramite il sistema di Issues fornito da GitHub stesso o contattando in privato l'autore :

- Email: [bianco.simone@outlook.it](mailto:bianco.simone@outlook.it)
- LinkedIn: [Simone Bianco](#)

Gli appunti sono in continuo aggiornamento, pertanto, previa segnalazione, si prega di controllare se le modifiche siano già state apportate nella versione più recente.

## Prerequisiti consigliati per lo studio:

Apprendimento del materiale relativo al corso *Progettazione di Algoritmi*.

## Licence:

These documents are distributed under the [GNU Free Documentation License](#), a form of copyleft intended for use on a manual, textbook or other documents. Material licensed under the current version of the license can be used for any purpose, as long as the use meets certain conditions:

- All previous authors of the work must be **attributed**.
- All changes to the work must be **logged**.
- All derivative works must be **licensed under the same license**.
- The full text of the license, unmodified invariant sections as defined by the author if any, and any other added warranty disclaimers (such as a general disclaimer alerting readers that the document may not be accurate for example) and copyright notices from previous versions must be maintained.
- Technical measures such as DRM may not be used to control or obstruct distribution or editing of the document.

# 1

## Automi e Linguaggi

### Definizione 1: Alfabeto

Definiamo come **alfabeto** un insieme finito di elementi detti **caratteri**

**Esempio:**

- L'insieme  $\Sigma = \{0, 1, x, y, z\}$  è un alfabeto
- L'insieme  $\Sigma = \{0, 1\}$  è un alfabeto. In particolare, tale alfabeto viene detto **alfabeto binario**

### Definizione 2: Stringa

Dato un alfabeto  $\Sigma$ , definiamo come **stringa di  $\Sigma$**  una sequenza di caratteri  $x_1x_2 \dots x_n$  dove  $x_1, \dots, x_n \in \Sigma$  e  $n \in \mathbb{N}$ .

In particolare, indichiamo come  $\varepsilon$  la **stringa vuota**

**Esempio:**

- Dato l'alfabeto  $\Sigma = \{0, 1, x, y, z\}$ , una stringa di  $\Sigma$  è  $0x1yyy0$

### Definizione 3: Linguaggio

Dato un alfabeto  $\Sigma$ , definiamo come **linguaggio di  $\Sigma$** , indicato come  $\Sigma^*$ , l'insieme delle stringhe di  $\Sigma$ .

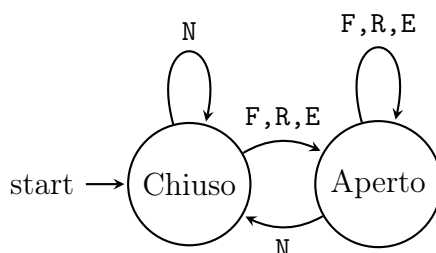
In particolare, notiamo che  $\varepsilon \in \Sigma^*$  per qualsiasi linguaggio  $\Sigma^*$

#### Definizione 4: Automa

Un **automa** è un meccanismo di controllo (o macchina) progettato per seguire automaticamente una sequenza di operazioni o rispondere a istruzioni predeterminate, mantenendo informazioni relative allo **stato** attuale dell'automa stesso ed agendo di conseguenza, **passando da uno stato all'altro**.

#### Esempio:

- Un sensore che apre e chiude una porta può essere descritto tramite il seguente automa, dove **Chiuso** e **Aperto** sono gli stati dell'automa e N, F, R e E sono le operazioni di transizione tra i due stati indicanti rispettivamente:
  - N: il sensore non rileva alcuna persona da entrambi i lati della porta
  - F: il sensore rileva qualcuno nel lato frontale della porta
  - R: il sensore rileva qualcuno nel lato retrostante della porta
  - E: il sensore rileva qualcuno da entrambi i lati della porta



- L'automa appena descritto è in grado di interpretare una **stringa in input** che ne descriva la sequenza di operazioni da svolgere
- Ad esempio, la stringa NFNNFRR terminerà l'esecuzione dell'automa sullo stato **Aperto**

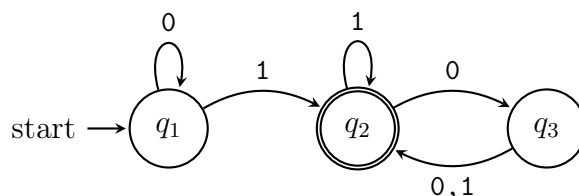
#### Definizione 5: Deterministic Finite Automaton (DFA)

Un **Deterministic Finite Automaton (DFA)** (o *Automa Deterministico a Stati Finiti*) è una quintupla  $(Q, \Sigma, \delta, q_0, F)$  dove:

- $Q$  è l'**insieme finito degli stati** dell'automa
- $\Sigma$  è l'**alfabeto** dell'automa
- $\delta : Q \times \Sigma \rightarrow Q$  è la **funzione di transizione degli stati** dell'automa
- $q_0 \in Q$  è lo **stato iniziale** dell'automa
- $F \subseteq Q$  è l'**insieme degli stati accettanti** dell'automa, ossia l'insieme degli stati su cui, a seguito della lettura di una stringa in input, l'automa accetta la corretta terminazione

### Esempio:

- Consideriamo il seguente DFA



dove:

- $Q = \{q_1, q_2, q_3\}$  è l'insieme degli stati dell'automa
- $\Sigma = \{0, 1\}$  è l'alfabeto dell'automa
- $\delta : Q \times \Sigma \rightarrow Q$  definita come

$\delta$	$q_1$	$q_2$	$q_3$
0	$q_1$	$q_3$	$q_2$
1	$q_2$	$q_2$	$q_2$

è la funzione di transizione degli stati dell'automa

- $q_0 := q_1 \in Q$  è lo stato iniziale dell'automa
- $F = \{q_2\}$  è l'insieme degli stati accettanti

### Definizione 6: Funzione di transizione estesa

Sia  $M := (Q, \Sigma, \delta, q_0, F)$  un DFA. Definiamo  $\delta^* : Q \times \Sigma^* \rightarrow Q$  come **funzione di transizione estesa di  $M$**  la funzione definita ricorsivamente come:

- $\delta^*(q, \varepsilon) = \delta(q, \varepsilon) = q$
- $\delta^*(q, ax) = \delta^*(\delta(q, a), x)$ , dove  $a \in \Sigma, x \in \Sigma^*$

### Definizione 7: Stringa accettata

Sia  $M := (Q, \Sigma, \delta, q_0, F)$  un DFA. Data una stringa  $x \in \Sigma^*$ , diciamo che  $x$  è **accettata da  $M$**  se  $\delta^*(q_0, x) \in F$ , ossia l'interpretazione di tale stringa **termina su uno stato accettante**

### Esempio:

- Consideriamo ancora il DFA dell'esempio precedente.
- La stringa 0101 è accettata da tale DFA, poiché:

$$\begin{aligned}\delta^*(q_1, 0101) &= \delta^*(\delta(q_1, 0), 101) = \delta^*(q_2, 101) = \delta^*(\delta(q_2, 1), 01) = \delta^*(q_2, 01) = \\ &= \delta^*(\delta(q_2, 0), 1) = \delta^*(q_3, 1) = \delta^*(\delta(q_3, 1), \varepsilon) = \delta^*(q_2, \varepsilon) = q_2 \in F\end{aligned}$$

- La stringa 1010, invece, non è accettata dal DFA, poiché:

$$\delta^*(q_1, 1010) = \delta^*(q_2, 010) = \delta^*(q_3, 10) = \delta^*(q_2, 0) = \delta^*(q_3, \varepsilon) = q_3 \notin F$$

### Definizione 8: Linguaggio di un DFA

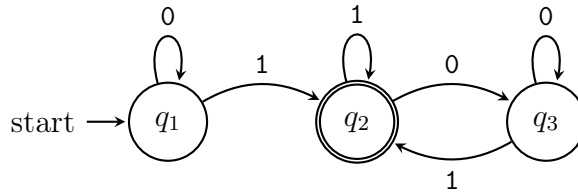
Sia  $M := (Q, \Sigma, \delta, q_0, F)$  un DFA. Definiamo come **linguaggio di  $M$** , indicato come  $L(M)$ , l'insieme di stringhe accettate da  $M$

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

Inoltre, diciamo che  $M$  **riconosce**  $L(M)$

### Esempi:

- Consideriamo il seguente DFA  $M$



- Il linguaggio riconosciuto da tale DFA corrisponde a

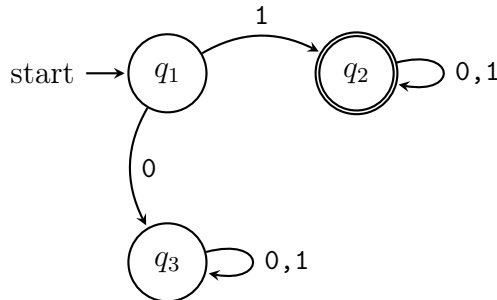
$$L(M) = \{x \in \{0, 1\}^* \mid x := y1, \exists y \in \{0, 1\}^*\}$$

ossia al linguaggio composto da tutte le stringhe terminanti con 1

- Consideriamo il seguente linguaggio

$$L = \{x \in \{0, 1\}^* \mid 1y, \exists y \in \{0, 1\}^*\}$$

- Un DFA in grado di riconoscere tale linguaggio corrisponde a

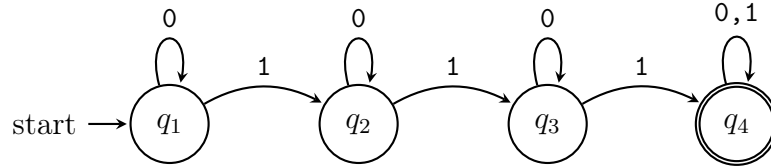


3. • Consideriamo il seguente linguaggio

$$L = \{x \in \{0, 1\}^* \mid w_H(x) \geq 3\}$$

dove  $w_H$  è il **peso di Hamming** (ossia  $w_H(x)$  = numero di "1" in  $x$ )

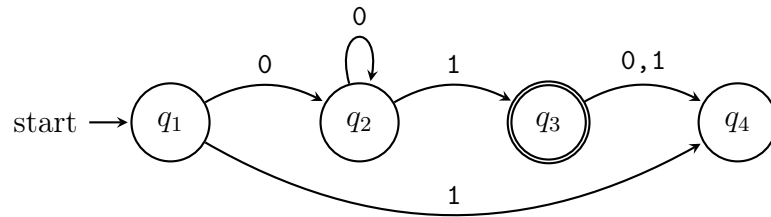
- Un DFA in grado di riconoscere tale linguaggio corrisponde a



4. • Consideriamo il seguente linguaggio

$$L = \{x \in \{0, 1\}^* \mid 0^n 1, n \in \mathbb{N} - \{0\}\}$$

- Un DFA in grado di riconoscere tale linguaggio corrisponde a



### Definizione 9: Linguaggi regolare

Dato un linguaggio  $\Sigma^*$ , definiamo come **insieme dei linguaggi regolari di  $\Sigma^*$** , indicato con  $REG$ , l'insieme dei sottolinguaggi per cui esiste un DFA che riconosce tale linguaggi

$$REG = \{L \subseteq \Sigma^* \mid \exists \text{DFA } M \text{ tale che } L = L(M)\}$$

### Definizione 10: Configurazione di un DFA

Sia  $M := (Q, \Sigma, \delta, q_0, F)$  un DFA. Definiamo la coppia  $(q, x) \in Q \times \Sigma^*$  come **configurazione di  $M$**

### Definizione 11: Passo di computazione

Definiamo come **passo di computazione** la relazione binaria definita come

$$(p, ax) \vdash_M (q, x) \iff \delta(p, a) = q$$



---

### Proposizione 1: Chiusura del passo di computazione

Sia  $M := (Q, \Sigma, \delta, q_0, F)$  un DFA. La **chiusura riflessiva e transitiva** di  $\vdash_M$ , indicata come  $\vdash_M^*$ , gode delle seguenti proprietà:

- $(p, ax) \vdash_M (q, x) \implies (p, ax) \vdash_M^* (q, x)$
- $\forall q \in Q, x \in \Sigma^* \quad (q, x) \vdash_M^* (q, x)$
- $(p, aby) \vdash_M (q, by) \wedge (q, by) \vdash_M (r, y) \implies (p, aby) \vdash_M^* (r, y)$

### Osservazione 1

Sia  $M := (Q, \Sigma, \delta, q_0, F)$  un DFA. Dati  $q_i, q_f \in Q, x \in \Sigma^*$ , si ha che

$$\delta^*(q_i, x) = q_f \iff (q_i, x) \vdash_M^* (q_f, \varepsilon)$$

(*dimostrazione omessa*)