aglaia norza

Automi, Calcolabilità e Complessità

appunti delle lezioni
libro del corso: ”*Introduzione alla teoria della computazione*”, Micheal Sipser

28/09/2025

Contents

1. Linguaggi regolari

1.1. Automi a stati finiti

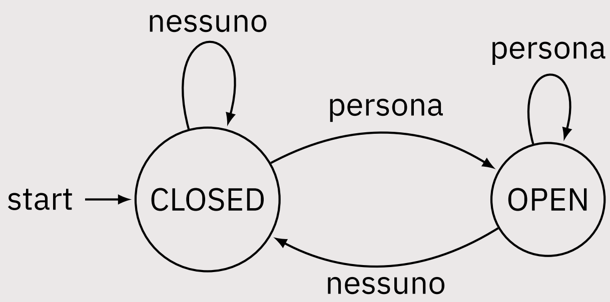
def. 1: automa

Un automa è un modello matematico di calcolo e una macchina teorica o reale che può elaborare informazioni e agire in modo automatico, seguendo una sequenza di **stati** predefiniti.

Un automa che possiede un numero finito di stati è detto **automa a stati finiti**, o **DFA** (*Deterministic Finite Automaton*).

esempio

Un classico esempio di automa è il sistema di controllo per una porta automatica. Le porte automatiche si aprono quando il sistema di controllo avverte che una persona si sta avvicinando, e si chiudono quando esso non rileva più la presenza di una persona. Il sistema di controllo è in uno di due stati: OPEN e CLOSED.



def. 2: DFA, definizione formale

Un DFA è una quintupla $(Q, \Sigma, \delta, q_0, F)$, dove:

- Q è l'insieme finito degli **stati**
- Σ è l'insieme finito di simboli di input, chiamato **alfabeto**
- $\delta : Q \times \Sigma \to Q$ è la **funzione di transizione**
- q_0 è lo **stato iniziale**
- $F \subseteq Q$ è l'insieme degli **stati di accettazione** (o "finali").

  ogni automa, ricevuto un input, lo elabora e restituisce un output. L'output è *accetta* se l'automa termina in uno stato di accettazione, o *rifiuta* altrimenti.

Prendiamo come esempio un DFA M:

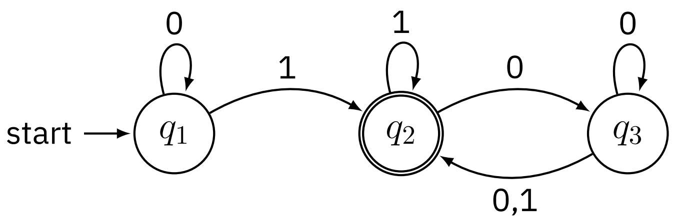


Figure 1.1: automa M

La figura 1.1 è chiamata *diagramma di stato* di M. Lo *stato iniziale* q_1 è indicato dall'arco "start". Lo *stato accettante* q_2 è rappresentato con un doppio cerchio.

Quando l'automa riceve una stringa in input, come per esempio 1101, legge un simbolo alla volta e si sposta da uno stato ad un altro seguendo le transizioni (archi). Quando legge l'ultimo simbolo, produce un output di tipo *accetta/rifiuta*.
Quando forniamo 1101 all'automa in figura, l'elaborazione procede così:

$$q_1 \to q_2 \to q_2 \to q_3 \to q_2$$

L'automa *accetta*, perché si trova nello stato accettante q_2 alla fine dell'input.

Possiamo descrivere formalmente $M = (Q, \Sigma, \delta, q_1, F)$, dove:

- $Q = \{q_1, q_2, q_3\}$
- $\Sigma = \{0, 1\}$
- $\delta = \begin{array}{c|cc} & 0 & 1 \\ \hline q_1 & q_1 & q_2 \\ q_2 & q_3 & q_2 \\ q_3 & q_2 & q_2 \end{array}$
- $F = \{q_2\}$

1.2. Linguaggi

def. 3: linguaggio di un DFA

Se M è un DFA, l'**insieme di stringhe riconosciute** ("accettate") da M si denota come $L(M)$.

esempio

Per esempio, il DFA M della figura 1.1 ha come linguaggio:

$A = \{w|w$ contiene almeno un 1 e un numero pari di 0 segue l'ultimo 1$\}$

. Quindi, $L(M) = A$ o, equivalentemente, M riconosce A.

Per definire il linguaggio di un automa, si introduce il concetto di **funzione di transizione estesa**.

def. 4: funzione di transizione estesa

La **funzione di transizione estesa** δ^* estende δ a più di un carattere in questo modo:

- $\delta^* : Q \times \Sigma^* \to Q$
- $\delta^*(q, \varepsilon) = \delta(q, \varepsilon)$
- $\delta^*(q, ax) = \delta^*(\delta(q, a), x)$
  (si "computa" il primo simbolo, e poi tutto il resto (ricorsivamente))

con Σ^* = insieme di stringhe in input, $x \in \Sigma^*, a \in \Sigma$.

Si introduce anche il concetto di **configurazione**.

def. 5: configurazione

Una **configurazone** è data da una tupla $\in Q \times \Sigma^*$ con elementi:

1. lo stato
2. quello che resta da leggere

Dato $x \in \Sigma^*$, la *configurazione iniziale* è q_0, x.

Un passo di computazione porta da una configurazione ad un'altra.

Si introduce anche il concetto di **relazione binaria tara configurazioni** ($\vdash$).

def. 6: passo di computazione (relazione binaria)

Si ha che:

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

con $p, q \in Q, a \in \Sigma, x \in \Sigma^*$.

(ovvero, dalla configurazione di sinistra si passa, con un passo di computazione, a quella a destra.)

come deterministica s

La relazione binaria M si può estendere considerando la sua **chiusure riflessiva e transitiva** M^*:

1. $q, x \vdash_{M^*} (q, x)$ (se non legge nessun input)
2. $(q, aby) \vdash_M (p, by) \wedge (p, by) \vdash_M (r, y) \Rightarrow (q, aby) \vdash_{M^*} (r, y)$ (transitività)

Possiamo usare questo concetto per formalizzare la nostra definizione di **linguaggio accettato**.

def. 7: linguaggio accettato (definizione formale)

Diciamo che $x \in \Sigma^*$ è **accettato** da $M = (Q, \Sigma, \delta, q_0, F)$ se $\delta^*(q_0, x) \in F$, oppure $(q_0, x) \vdash_{M^*} (q, \varepsilon)$.

dove $q \in F$ e ε rappresenta la stringa vuota.

In altre parole, $L(M) = \{x \in \Sigma^* : \delta^*(q_0, x) \in F\}$.

Data la definizione di linguaggio accettato, possiamo definire un **linguaggio regolare**.

def. 8: linguaggio regolare

Un linguaggio è chiamato **linguaggio regolare** se un automa finito lo riconosce.

Ovvero,

$$\text{REG} = \{L \subseteq \Sigma^* \mid \exists \text{ DFA } M \ \ t.c. \ L(M) = L\}$$

Vogliamo capire come progettare un DFA per un dato linguaggio.

esempio

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

$(1||y = 1$ OR y, ovvero stringhe che iniziano per $1)$

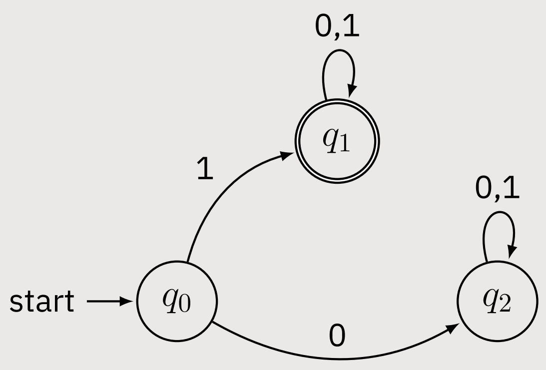


Figure 1.2: automa A

nota

Lo stato q_2 è un cosiddetto "stato pozzo" (non è lo stato di accettazione e non ha archi uscenti verso altri stati). L'automa è quindi equivalente ad un automa A', senza gli archi di q_2

Vogliamo anche capire come sviluppare una **prova di correttezza**, ovvero dimostrare che il DFA accetta $x \iff x \in L$.

Proviamo con l'automa nella figura 1.2.

Notiamo che:

- $\delta^*(q_1, u) = q_1 \, \forall u \in \{0, 1\}^*$
- $\delta^*(q_2, u) = q_2 \, \forall u \in \{0, 1\}^*$

Vogliamo costruire questa dimostrazione per induzione.

- **Caso base**: $|x| = 0$, quindi $x = \varepsilon$;
  si ha $\delta^*(q_0, \varepsilon) = \delta(q_0, \varepsilon) = q_0 \notin F$.
- **Passo induttivo**: Sia $n > 0$. Supponiamo $|w| \leq n$. Si ha:

$$\delta^*(q_0, w) = \begin{cases} q_1 & \text{se } w \text{ inizia con 1} \\ q_2 & \text{se } w \text{ inizia con 0} \end{cases}$$

  (perché sia q_1 che q_2 non hanno archi uscenti verso altri stati, quindi non importa quali siano le cifre successive)

- Prendo x t.c. $|x| = n + 1$, e lo immagino come $x = au$, con $a \in \{0, 1\}^*, u \in \{0, 1\}^*$.
  Ho quindi:

$$\delta^*(q_0, x) = \delta^*(q_0, au) = \delta^*(\delta(q_0, a), u)$$

e, visto che $\delta(q_0, a) = \begin{cases} q_0 & \text{se } a = 0 \\ q_1 & \text{se } a = 1 \end{cases}$

si ha $\delta^*(q_0, x) = q_1 \iff a = 1$