automi

- Teoria della computazione
  - Teoria della calcolabilita
  - Teoria della complessita
  - Teoria degli automi e dei linguaggi formali
- Alfabeti e stringhe
- Automi
  - Automa a stati finiti deterministico (DFA)
  - Rappresentazioni
  - Automi a stati finiti non-deterministici (NFA)
  - Equivalenze DFA e NFA
  - NFA con ϵ-mosse

Teoria della computazione

- teoria della calcolabilita
- teoria degli automi e dei linguaggi formali
- teoria della complessita

Teoria della calcolabilita

nata dal **problema di decisione** ⇒ *esiste un algoritmo che, presi in input un insieme di assiomi e una proposizione matematica, e in grado di decidere se la proposizione e vera in ogni struttura che soddisfa gli assiomi? risposta negativa*

questione centrale ⇒

- problemi **solvable**
- problemi **unsolvable**

Teoria della complessita

solvable ⇒ problema per il quale esiste un *algoritmo* che permette di calcolare la soluzione del problema per ogni input.

- tempo di esecuzione
- memoria

hard problem ⇒ non esiste un *algoritmo* per risolverlo con una quantita ragionevole di risorse.

Questione centrale ⇒ classificazione dei problemi in base alla quantita di risorse necessarie per risolverli.

Teoria degli automi e dei linguaggi formali

automi

- automi a stati finiti
- automi a pila
- macchine di turing

Questione centrale ⇒ individuazione della classe di problemi (linguaggi riconoscibili) dai vari modelli e loro proprieta.

Alfabeti e stringhe

Un alfabeto Σ e un insieme finito e non vuoto di simboli.
Una stringa su Σ e una qualunque sequenza finita di simboli di Σ.
Dicendo che una stringa su Σ e una qualunque sequenza di simboli di Σ ammettiamo anche la sequenza vuota composta da zero simboli.
Chiamiamo tale sequenza stringa vuota, e la indichiamo con ϵ.

Automi

Automa a stati finiti deterministico (DFA)

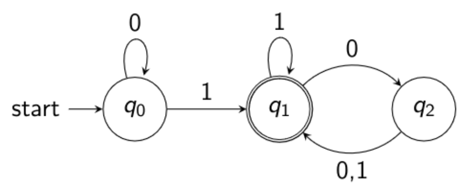
def $\Rightarrow$ e una 5-upla $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ in cui:
- $Q \Rightarrow$ e l insieme finito non vuoto degli *stati*.
- $\Sigma \Rightarrow$ e l alfabeto (input).
- $\delta : Q \times \Sigma \Rightarrow Q \Rightarrow$ e la funzione di transizione.
- $q_0 \in Q \Rightarrow$ e lo stato iniziale.
- $F \subseteq Q \Rightarrow$ e l insieme degli stati finali.

def $\Rightarrow$ il linguaggio $L(A)$ accettato da un DFA e l insieme di tutte le stringhe accettatte dall automa A (la computazione deve finire in un doppio cerchio, uno stato finale).
$L(A) = \{w \in \Sigma^* | A \text{ accetta } w\}$
$L(A) = \{w \in \Sigma^* | \hat{\delta}(q_0, w) \in F\}$

Note

- Si osservi che in un DFA, dati uno stato q e un simbolo di input a:
  - la mossa dell automa e sempre definita (δ associa un valore ad ogni coppia $(q, a) \in Q \times \Sigma$);
  - il risultato della mossa e univocamente determinato (e il valore della funzione $\delta(q, a)$).
- ne segue che dati uno stato $r_0 \in Q$ e una stringa $w = a_1...a_n \in \Sigma^*$, esiste una sola possibile sequenza di mosse dell'automa A su w.
- se lo stato iniziale e anche uno stato finale, l automa accetta la stringa vuota.

Rappresentazioni



Se nella rappresentazione tabellare evidenziamo lo stato iniziale ($\rightarrow$) e gli stati finali ($*$), l'automa è completamente descritto dalla tabella di transizione.

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

Figure 1: varie rappresentazioni automi

Alla fine della computazione l automa deve trovarsi in uno stato accettante (doppio cerchio).

Automi a stati finiti non-deterministici (NFA)

def $\Rightarrow$ e una 5-upla $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ in cui:
- $Q \Rightarrow$ e l insieme finito non vuoto degli *stati*.
- $\Sigma \Rightarrow$ e l alfabeto (input).
- $\delta : Q \times \Sigma \Rightarrow 2^Q \Rightarrow$ e la funzione di transizione. $2^Q = \{S | S \subseteq Q\}$
- $q_0 \in Q \Rightarrow$ e lo stato iniziale.
- $F \subseteq Q \Rightarrow$ e l insieme degli stati finali.

l unica differenza tra un DFA e NFA e la funzione di transizione δ. invece che associare ad una coppia stato simblo uno stato, associa ad una coppia stato simbolo un sottoinsieme di stati. 2^Q = insieme delle parti

Note

- A accetta w se $\delta(q_0, w) \cap F$ /
- Linguaggio riconosciuto da A. $L(A) = \{w \in \Sigma^* | A \text{ accetta } w\}$

Equivalenze DFA e NFA

DFA e NFA riconosocono la stessa classe di linguaggi:
- $\forall D \exists N_D | L(N_D) = L(D)$ - $\forall N \exists D_N | L(D_N) = L(N)$

Equivalenza da DFA a NFA (DFA $\Rightarrow$ NFA) i DFA sono casi particolari di NFA. ogni DFA e un NFA. un DFA e equivalente ad un NFA se essi riconoscono lo stesso linguaggio.

Lemma *Dato un DFA* $D = \langle Q, \Sigma, \delta, q_0, F \rangle$, $\forall w \in \Sigma^* : \hat{\delta}_N(q_0, w) = \{\hat{\delta}(q_0, w)\}$

Teorema *Dato un DFA* $D = \langle Q, \Sigma, \delta, q_0, F \rangle$, $L(N_D) = L(D)$

NFA con ϵ-mosse

$\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$ **def** $\Rightarrow$ e una 5-upla $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ in cui:
- $Q \Rightarrow$ e l insieme finito non vuoto degli *stati*.
- $\Sigma \Rightarrow$ e l alfabeto (input).
- $\delta : Q \times \Sigma\epsilon \Rightarrow 2^Q \Rightarrow$ e la funzione di transizione. $2^Q = \{S | S \subseteq Q\}$
- $q_0 \in Q \Rightarrow$ e lo stato iniziale.
- $F \subseteq Q \Rightarrow$ e l insieme degli stati finali.

Si noti che ϵ non consuma simboli dalla stringa di input quando computiamo nell automa.