Riconoscitori e ASFD

Created	@March 18, 2025 3:28 PM
Class	Fondamenti di informatica: linguaggi formali

Permettono di rappresentare finitamente linguaggi eventualmente infiniti.

I riconoscitori sono strumenti finiti che prendono la stringa in input e danno in output se la stringa appartiene o meno al linguaggio. (ASF, Macchine di Turing).

Tra i riconoscitori, studieremo in particolari gli automi, strumenti in grado di rappresentare finitamente linguaggi che sono eventualmente infiniti.

Metodi generativi, simbolo iniziale → regole → applicano regole → generano le grammatiche. A seconda della forma delle regole, andiamo a definire diversi tipi di grammatiche (regolari, context-free, context-sensitive ecc...)

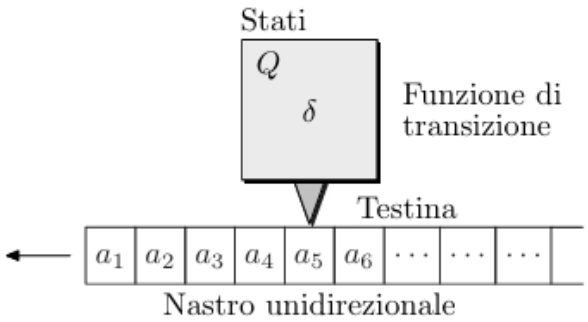
Una stessa classe di linguaggi può essere caratterizzata sia da riconoscitori che da metodi generativi.

Problema dell'alt della macchina di Turing è indecidibile. ossia non esiste un algoritmo che risolva il problema, è stato dimostrato dal problema della soddisfacibilità di formule logiche

Nastro infinito in entrambe le direzioni con celle contenenti i simboli dell'alfabeto, e simbolo Blank. Alcuni riconoscitori hanno un nastro semi-finito.

Ha una testina di lettura/scrittura che si può muovere a destra o a sinistra e può modificare/leggere una cella alla volta.

> δ è la **funzione di transizione** $\delta : Q \times \Sigma \to Q$. Tale funzione è **finita**, in quanto è definita per un insieme finito di valori, ossia $Q \times \Sigma$.



Un automa esegue una computazione applicando iterativamente, ad ogni istante, la propria funzione di transizione alla configurazione attuale, a partire dalla configurazione iniziale.

Possiamo quindi vedere una computazione eseguita da un automa $\mathcal{A}$ a partire da una configurazione iniziale c_0 come una sequenza di configurazioni $c_0, c_1, c_2, \ldots$ tale che, per $i = 0, 1, \ldots$, si ha $c_i \vdash_{\mathcal{A}} c_{i+1}$. Indicheremo anche nel seguito con la notazione $\vdash_{\mathcal{A}}^{*}$ la chiusura transitiva e riflessiva della relazione $\vdash_{\mathcal{A}}$. È facile allora rendersi conto che, dato un automa $\mathcal{A}$ e due configurazioni c_i, c_j di $\mathcal{A}$, si ha $c_i \vdash_{\mathcal{A}}^{*} c_j$ se e solo se esiste una computazione che porta $\mathcal{A}$ da c_i a c_j.

Se la sequenza di configurazioni $c_0, c_1, c_2, \ldots, c_n$ ha lunghezza finita e se è massimale, nel senso che non esiste nessuna configurazione c tale che $c_n \vdash_{\mathcal{A}} c$, allora diciamo che la computazione *termina*: in tal caso, diciamo che essa è una *computazione di accettazione* se termina in una configurazione di accettazione, mentre, nel caso contrario in cui termini in una configurazione non di accettazione, diremo che è una *computazione di rifiuto*.

q_o è lo stato da cui inizia la computazione o configurazione iniziale. Una configurazione è una “fotografia” del riconoscitore in un determinato instante della computazione, che ci fornisce informazioni su come essa proseguirà.

Se lo stato finale $q_n \in Q$ non è contenuto nell’insieme degli stati finali F vorrà dire che la stringa non è riconosciuta dall’automa.

Se esiste un algoritmo che consente di calcolare una funzione, allora esiste una macchina di Turing che risolve quella funzione. Ogni funziona calcolabile è Turing-calcolabile, ossia calcolabile da una macchina di Turing (Congettura di Church.

Fino adesso abbiamo trattato il modello deterministico per i riconoscitori.

Automi a stati finiti deterministici (ASFD)

Proprietà

Ricordiamo che un automa a stati finiti:

- ha una **testina** che si muove **solo a destra di una cella per volta**;
- può solo **leggere ma non scrivere**;
- la sua **funzione di transizione** modifica lo stato **in virtù dello stato della macchina e del simbolo letto**.

La funzione di transizione è del tipo:

$$\delta : Q \times \Sigma \to Q$$

Un **automa a stati finiti deterministico**, abbreviato **ASFD**, è un sistema:

$$\mathbb{A} =< Q, \Sigma, \Delta, q_0, F >$$

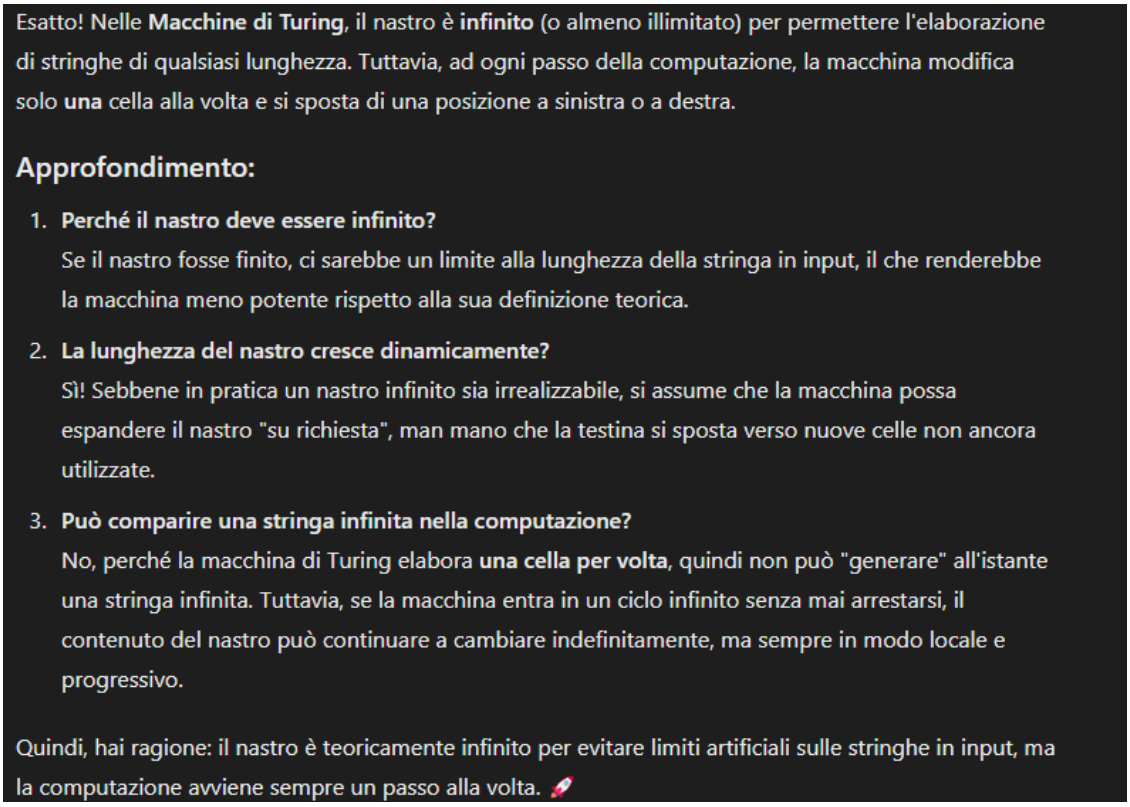
dove:

- Q è l'insieme degli stati, un **insieme finito**;
- Σ è l'**alfabeto di input**, anche questo un **insieme finito**;
- $q_0 \in Q$ è lo **stato iniziale dell'automa**;
- $F \subseteq Q$ è l'**insieme degli stati finali**, che serve per **determinare le stringhe accettate dall'automa** (in un certo senso è il modo in cui termina la computazione);
- δ è la **funzione di transizione** $\delta : Q \times \Sigma \to Q$. Tale funzione è **finita**, in quanto è definita per un insieme finito di valori, ossia $Q \times \Sigma$.

Non può capitare che durante la computazione compaia una stringa infinita, poiché ad ogni movimento della testina si può cambiare il contenuto di una unica cella per volta. La lunghezza del nastro quindi è infinita poiché se fosse finita avremmo limiti per la lunghezza in input delle stringhe.

Stato finale non è uguale a configurazione finale, quest'ultima comprende tutte le componenti dell'automa, compreso comunque lo stato finale, che appartiene all'insieme degli stati dell'automa Q.

La funzione di transizione dipende sia dal simbolo della cella in cui mi trovo, sia dallo stato precedente.



Esatto! Nelle **Macchine di Turing**, il nastro è **infinito** (o almeno illimitato) per permettere l'elaborazione di stringhe di qualsiasi lunghezza. Tuttavia, ad ogni passo della computazione, la macchina modifica solo **una** cella alla volta e si sposta di una posizione a sinistra o a destra.

Approfondimento:

1. **Perché il nastro deve essere infinito?**
   Se il nastro fosse finito, ci sarebbe un limite alla lunghezza della stringa in input, il che renderebbe la macchina meno potente rispetto alla sua definizione teorica.
2. **La lunghezza del nastro cresce dinamicamente?**
   Sì! Sebbene in pratica un nastro infinito sia irrealizzabile, si assume che la macchina possa espandere il nastro "su richiesta", man mano che la testina si sposta verso nuove celle non ancora utilizzate.
3. **Può comparire una stringa infinita nella computazione?**
   No, perché la macchina di Turing elabora **una cella per volta**, quindi non può "generare" all'istante una stringa infinita. Tuttavia, se la macchina entra in un ciclo infinito senza mai arrestarsi, il contenuto del nastro può continuare a cambiare indefinitamente, ma sempre in modo locale e progressivo.

Quindi, hai ragione: il nastro è teoricamente infinito per evitare limiti artificiali sulle stringhe in input, ma la computazione avviene sempre un passo alla volta. 🚀

Ad esempio, sia $\mathbb{A} =< Q, \Sigma, \delta, q_0, F >$ definito da:

ati finiti

- $Q = \{ q_0, q_1, q_2 \}$;
- $\Sigma = \{ a, b \}$;
- $F = \{ q_1 \}$;

La funzione δ è rappresentata dalla tabella:

δ	a	b
q_0	q_0	q_1
q_1	q_2	q_2
q_2	q_2	q_2

Analizziamo la stringa aab. Vediamo se l'automa riconosce questa stringa. Allora allo stato iniziale sarà a q_0. Legge il carattere a, e rimane in q_0. Si muove a destra, legge la lettera a e quindi rimane a q_0. Infine legge b, ed essendo nello stato q_0, esegue la transizione a q_1. Infine legge $blank$, pertanto la computazione termina allo stato q_1. La stringa è riconosciuta in quanto $q_1 \in F$. Il **linguaggio riconosciuto dall'automa** $L(\mathbb{A}) = \{ a^n b : n \geq 0 \}$.

Definizione 3.5 *La* funzione di transizione estesa *di un automa a stati fin deterministico* $\mathcal{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$ *è la funzione* $\overline{\delta} : Q \times \Sigma^* \mapsto Q$, *definita n seguente modo:*

$$\begin{aligned} \overline{\delta}(q, \varepsilon) &= q \\ \overline{\delta}(q, xa) &= \delta\left(\overline{\delta}(q, x), a\right), \end{aligned}$$

dove $a \in \Sigma$, $x \in \Sigma^*$.

Se la sequenza di configurazioni $c_0, c_1, c_2, \ldots, c_n$ ha lunghezza finita e se è massimale, nel senso che non esiste nessuna configurazione c tale che $c_n \vdash_{\mathcal{A}} c$, allora diciamo che la computazione *termina*: in tal caso, diciamo che essa è una *computazione di accettazione* se termina in una configurazione di accettazione, mentre, nel caso contrario in cui termini in una configurazione non di accettazione, diremo che è una *computazione di rifiuto*.

Un automa, in particolare, è detto *deterministico* se ad ogni stringa di input associa una sola computazione, e quindi una singola sequenza di configurazioni (vedi Figura 2.6). Possiamo anche dire che un automa deterministico, data una stringa di input, può eseguire una sola computazione: se tale computazione termina in una configurazione di accettazione, allora la stringa viene accettata.

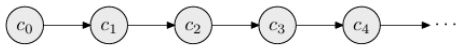


FIGURA 2.6 Computazione di un automa deterministico.

Esempio 2.29 Nel caso di automi a stati finiti viene definito, dato un automa, un insieme F di stati, detti *finali*. Una configurazione di accettazione corrisponderà allora alla situazione in cui l'automa, letta tutta la stringa di input, si trova in uno stato finale. Ne deriva quindi che una configurazione di accettazione ha la struttura:

1. la stringa x;
2. la prima posizione successiva alla stringa;
3. uno stato finale $q \in F$.

Una configurazione non di accettazione corrisponderà, al contrario, alla situazione in cui l'automa non ha letto tutta la stringa di input oppure, se l'ha letta, non si trova in uno stato finale. Una configurazione di rifiuto ha quindi la struttura

1. la stringa x
2. posizione su un carattere della stringa
3. uno stato qualunque $q \in Q$,

o la struttura

1. la stringa x
2. la prima posizione successiva alla stringa
3. uno stato non finale $q \notin F$.

Un automa esegue una computazione applicando iterativamente, ad ogni istante, la propria funzione di transizione alla configurazione attuale, a partire dalla configurazione iniziale.

Possiamo quindi vedere una computazione eseguita da un automa $\mathcal{A}$ a partire da una configurazione iniziale c_0 come una sequenza di configurazioni $c_0, c_1, c_2, \ldots$ tale che, per $i = 0, 1, \ldots$, si ha $c_i \vdash_{\mathcal{A}} c_{i+1}$. Indicheremo anche nel seguito con la notazione $\vdash^*_{\mathcal{A}}$ la chiusura transitiva e riflessiva della relazione $\vdash_{\mathcal{A}}$. È facile allora rendersi conto che, dato un automa $\mathcal{A}$ e due configurazioni c_i, c_j di $\mathcal{A}$, si ha $c_i \vdash^*_{\mathcal{A}} c_j$ se e solo se esiste una computazione che porta $\mathcal{A}$ da c_i a c_j.

Appunti Formali sugli Automi a Stati Finiti Deterministici (ASFD)

Definizione Formale di un ASFD

Un automa a stati finiti deterministico è formalmente definito come la quintupla:

$$A = \langle Q, \Sigma, \delta, q_0, F \rangle$$

dove:

- Q: Insieme finito e non vuoto di **stati**.
- Σ: Alfabeto finito di **input** (simboli leggibili).
- δ: **Funzione di transizione** totale, $\delta : Q \times \Sigma \to Q$.
- q_0: **Stato iniziale**, con $q_0 \in Q$.
- F: Insieme di **stati finali** (o accettanti), con $F \subseteq Q$.

Configurazioni e Transizioni

Definizione di Configurazione

Una configurazione c è una coppia ordinata:

$$c = (q, w)$$

- $q \in Q$: Stato corrente dell'automa.
- $w \in \Sigma^*$: Porzione residua del nastro di input (stringa non ancora letta).

Relazione di Transizione $\vdash_A$

La transizione tra configurazioni è definita dalla relazione $\vdash_A \subseteq (Q \times \Sigma^*) \times (Q \times \Sigma^*)$:

$$(q, aw) \vdash_A (q', w) \quad \text{se e solo se} \quad \delta(q, a) = q'$$

dove $a \in \Sigma$, $w \in \Sigma^*$.

Chiusura Transitiva e Riflessiva $\vdash_A^*$

La relazione $\vdash_A^*$ è la chiusura riflessiva e transitiva di $\vdash_A$:

- **Riflessività**: $(q, w) \vdash_A^* (q, w)$ per ogni (q, w).
- **Transitività**: Se $(q, w) \vdash_A^* (q', w')$ e $(q', w') \vdash_A^* (q'', w'')$, allora $(q, w) \vdash_A^* (q'', w'')$.

Funzione di Transizione Estesa $\overline{\delta}$

La funzione $\overline{\delta} : Q \times \Sigma^* \to Q$ estende δ a stringhe arbitrarie. È definita induttivamente come segue:

Casi Base e Ricorsione

1. **Caso base (stringa vuota)**:

   $$\overline{\delta}(q, \varepsilon) = q$$

   Interpretazione: Senza simboli da leggere, l'automa rimane nello stato corrente.

2. **Passo ricorsivo (stringa non vuota)**:
   Sia $w = xa$, con $x \in \Sigma^*$, $a \in \Sigma$. Allora:

   $$\overline{\delta}(q, xa) = \delta\left(\overline{\delta}(q, x), a\right)$$

   Interpretazione: L'automa processa x, raggiunge uno stato q', poi legge a e transisce a $\delta(q', a)$.

Esempio Formale

Dato l'automa con $\delta(q_0, a) = q_0$, $\delta(q_0, b) = q_1$, e $w = aab$:

$$\begin{aligned}
\overline{\delta}(q_0, aab) &= \delta\left(\overline{\delta}(q_0, aa), b\right) \\
\overline{\delta}(q_0, aa) &= \delta\left(\overline{\delta}(q_0, a), a\right) \\
\overline{\delta}(q_0, a) &= \delta\left(\overline{\delta}(q_0, \varepsilon), a\right) = \delta(q_0, a) = q_0 \\
\overline{\delta}(q_0, aa) &= \delta(q_0, a) = q_0 \\
\overline{\delta}(q_0, aab) &= \delta(q_0, b) = q_1
\end{aligned}$$

Risultato: $\overline{\delta}(q_0, aab) = q_1$, stato finale $\in F$.

Computazione come Sequenza di Configurazioni

Una computazione è una sequenza massimale di configurazioni $c_0, c_1, \ldots, c_n$, dove:

- $c_0 = (q_0, w)$: Configurazione iniziale.
- $c_i \vdash_A c_{i+1}$: Ogni passo segue δ.
- **Terminazione**: La computazione termina se $c_n = (q, \varepsilon)$ (input esaurito) o non esiste c_{n+1}.

Accettazione e Rifiuto Formali

- **Accettazione**: A accetta w se $\exists q \in F$ tale che $(q_0, w) \vdash_A^* (q, \varepsilon)$.
- **Rifiuto**: A rifiuta w se:
  - Termina in (q, ε) con $q \notin F$.
  - Si blocca prima di esaurire l'input (transizione non definita).

Esempio di Computazione Dettagliata

Automa: $A = \langle \{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_1\} \rangle$, con δ definita da:

δ	a	b
q_0	q_0	q_1
q_1	q_2	q_2
q_2	q_2	q_2

Stringa: $w = aab$.

Passi della Computazione

1. $c_0 = (q_0, aab) \vdash_A (q_0, ab)$ (legge a, rimane in q_0).
2. $c_1 = (q_0, ab) \vdash_A (q_0, b)$ (legge a, rimane in q_0).
3. $c_2 = (q_0, b) \vdash_A (q_1, \varepsilon)$ (legge b, transisce a q_1).

Esito: $q_1 \in F \rightarrow w$ è accettata.

Proprietà Formali degli ASFD

1. **Determinismo**: Per ogni $(q, a) \in Q \times \Sigma$, $\delta(q, a)$ è univocamente definita.
2. **Limite della Memoria**: Gli ASFD non possono "ricordare" informazioni arbitrarie (es. conteggi).
3. **Equivalenza con Grammatiche Regolari**: Per il **Teorema di Kleene**, un linguaggio è regolare se e solo se è riconosciuto da un ASFD.

Il linguaggio riconosciuto da questo automa è L(A) = {a^n concatenato b | n ≥ 0}

Stringa accetta è quella che può essere computata, mentre quella riconosciuta è quella che termina con uno stato finale in Q o in F. Una funzione di transizione per una coppia stato-simbolo può anche non essere definita, facendo fermare quindi la computazione. funziona totale, ossia ogni posizione conterrà uno stato definito. Se la stringa x di una coppia stato simbolo è uguale a epsilon, allora si troverà nell'ultima cella della stringa, ovvero punta a blank.

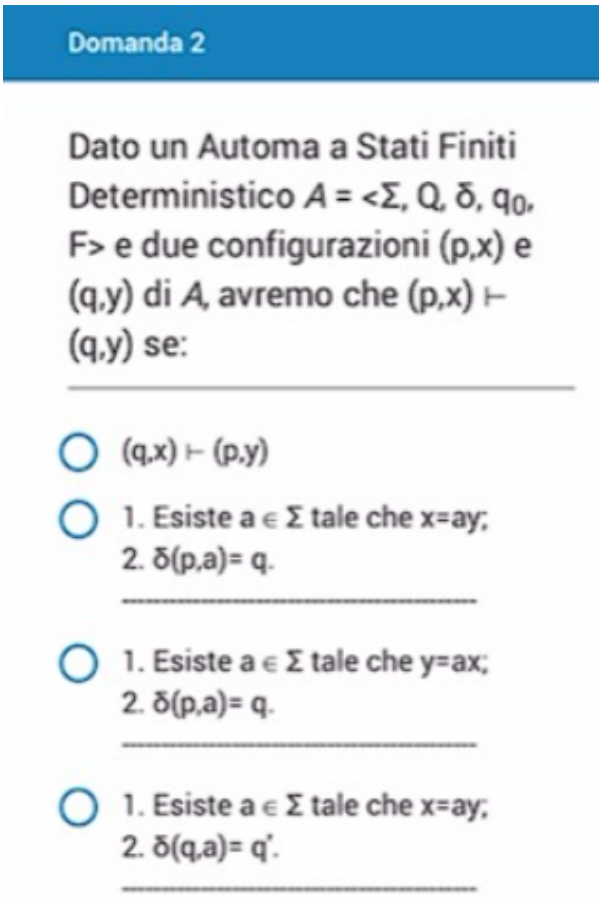
Per esprimere che da una configurazione iniziale, si può arrivare a una qualsiasi configurazione, si utilizza il seguente simbolo:

Blank è un simbolo di nastro e non è uguale alla parola vuota, e non appartiene mai all'insieme di alfabeto Sigma.

1. **Automa a stati finiti deterministico (DFA)**
   - Un DFA deve avere una transizione definita per ogni coppia (q, a), dove q è uno stato e a è un simbolo dell'alfabeto.
   - Se manca una transizione, l'automa non è valido secondo la definizione formale.
   - Spesso, per gestire queste situazioni, si introduce uno **stato di errore (stato "trappola")** in cui l'automa entra quando legge un simbolo senza transizione e da cui non può più uscire.

Confronto con Modelli Non-Deterministici

Aspetto	ASFD	ASFN (Non-Deterministico)
Funzione di Transizione	$\delta : Q \times \Sigma \to Q$	$\delta : Q \times \Sigma \to \mathcal{P}(Q)$
Transizioni Multiple	No	Sì (possibili scelte parallele)
Potenza Computazionale	Equivalente agli ASFN	Equivalente agli ASFD (per subset construction)

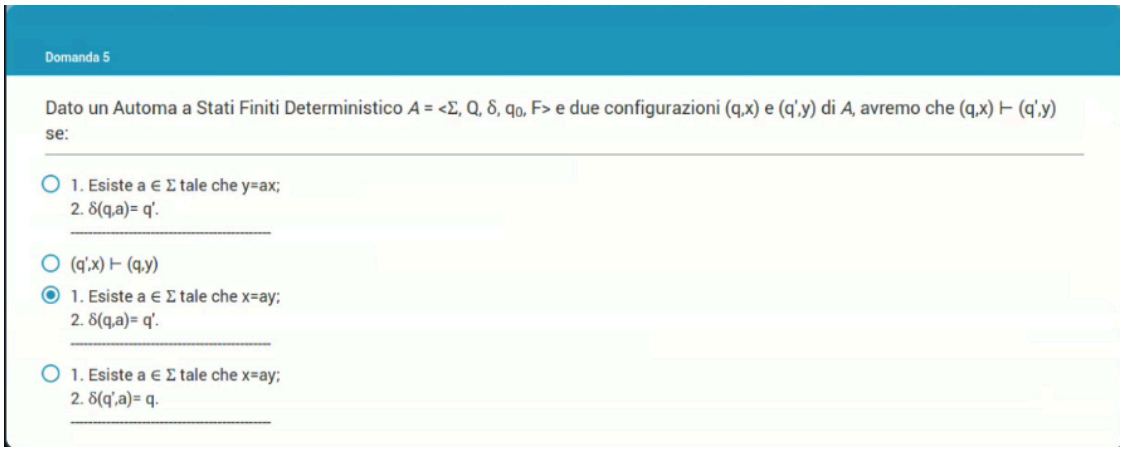


Risposta corretta: 2

Spiegazione:
Per transizione $(p, x) \vdash (q, y)$, l'automa deve leggere un simbolo a dall'input x (quindi $x = ay$) e spostarsi dallo stato p allo stato q secondo la funzione di transizione $\delta(p, a) = q$. Le altre opzioni presentano errori logici (es. inversione di stati o stringhe, definizioni incoerenti).

In un DFA, la notazione $(p, x) \vdash (q, y)$ significa che l'automa, partendo dallo stato p e con input x, legge un simbolo $a \in \Sigma$, passa allo stato q, e l'input rimanente diventa y.

- **Perché (x = ay ?**
  - x è la stringa di input **iniziale**. Per effettuare una transizione, l'automa deve leggere il **primo simbolo** di x, che chiamiamo a.
  - Dopo aver letto a, la parte rimanente di x (cioè tutto tranne a) diventa y.
  - Quindi, x può essere scomposta come a seguito da y, ovvero $x = a \cdot y$.



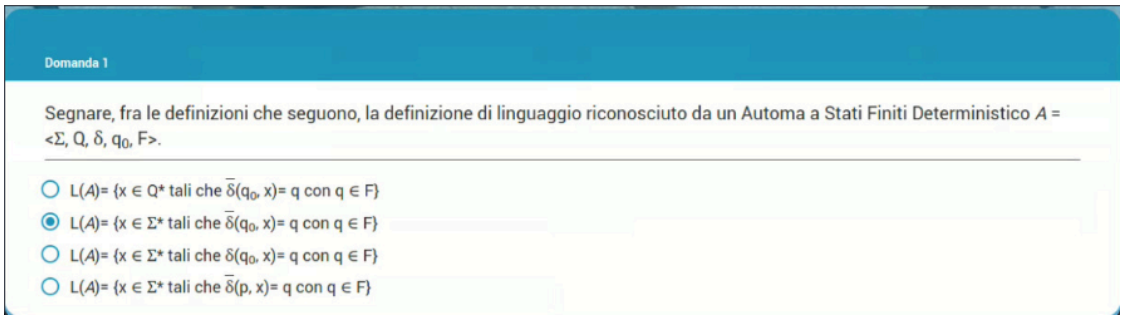
q' è lo **stato successivo** raggiunto dall'automa dopo aver letto un simbolo $a \in \Sigma$ dallo stato corrente q.

Esempio:

- Se $\delta(q, a) = q'$, l'automa passa da q a q' leggendo a.
- Se $x = ay$, l'input x inizia con a, e y è la parte rimanente.

In breve:

- q' è il risultato della transizione $\delta(q, a)$.
- È semplicemente un nome per indicare il nuovo stato dopo aver processato a.



Spiegazione:
Il linguaggio $L(A)$ è l'insieme di tutte le stringhe $x \in \Sigma^*$ (simboli di input) per cui l'automa, partendo dallo stato iniziale q_0, termina la lettura di x in uno stato finale $q \in F$. La notazione $\overline{\delta}(q_0, x)$ indica l'estensione della funzione di transizione a stringhe, mentre δ si riferisce a singoli simboli. Le altre opzioni contengono errori:

- **Opzione 1**: Usa Q^* (stringhe di stati) invece di Σ^*.
- **Opzione 3**: Usa δ (transizione per simboli singoli) invece di $\overline{\delta}$.
- **Opzione 4**: Parte da uno stato generico p, non dallo stato iniziale q_0.

✅ **Risposta corretta**:

- $L(A) = \{x \in \Sigma^* \text{ tali che } \overline{\delta}(q_0, x) = q \text{ con } q \in F\}$.

- δ descrive il comportamento **istantaneo** dell'automa (un solo passo).
- $\overline{\delta}$ descrive il comportamento **complessivo** su intere stringhe.
- Senza $\overline{\delta}$, non potremmo definire il linguaggio riconosciuto dall'automa $L(A)$, che dipende dallo stato finale raggiunto dopo aver letto tutta la stringa.

Errori comuni

- Usare δ con una stringa: $\delta(q_0, "ab")$ è **sbagliato** (va usato $\overline{\delta}$).
- Confondere Σ^* (stringhe di simboli) con Q^* (stringhe di stati).

university/1202 Fondamenti di Informatica/itinere/20230503 - AN - Madonia/Q1.jpeg at main · MatteoGalletta/university · GitHub

https://github.com/MatteoGalletta/university/tree/main/1202 Fondamenti di Informatica/itinere/20230504 - OZ - Madonia

Diagrammi di Transizione

Sono diagrammi che permettono di rappresentare la successione di configurazioni.

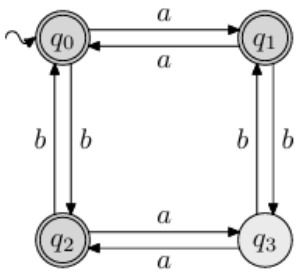


FIGURA 3.3 Diagramma degli stati di un ASFD che riconosce parole con un numero pari di a o di b.

La stringa vuota appartiene al linguaggio riconosciuto da un automa se e solo se lo stato iniziale è anche stato finale, dal momento che serve un carattere da leggere diverso da epsilon per la transizione.
esercizio x me, derivare un diagramma di transizione da una tabella di transizione.

Una transizione del tipo Delta(q0, epsilon), rappresentata con diagramma è inclusa sempre in qualsiasi configurazione che ha come stato finale q0.