La Macchina di Turing: linguaggi riconosciuti e linguaggi decisi

Oggi

- Obiettivo: diversi «usi» della MdT
  - MdT riconosce un linguaggio
  - MdT decide un linguaggio
  - MdT calcola una funzione

Una MdT

Una Turing Machine è

- una macchina a stati finiti con un nastro semi–infinito
- La testina può muoversi in entrambe le direzioni.
- Può leggere, scrivere in ogni cella del nastro
- Quando la MdT raggiunge uno stato accept/reject allora accetta/rifiuta immediatamente.

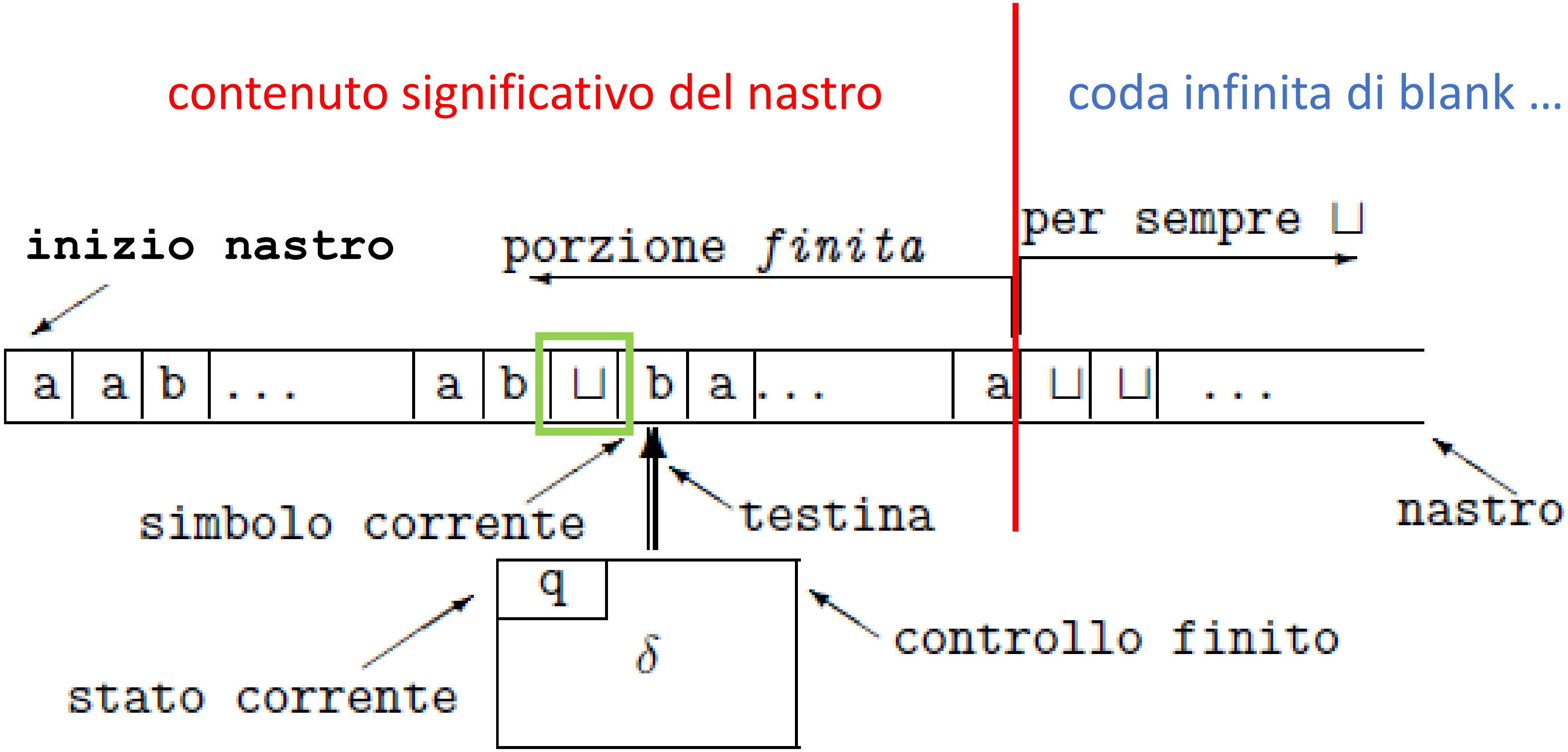
Descrizione formale MdT

Una Macchina di Turing è una settupla

$$(Q, \Sigma, \Gamma, \delta, q_o, q_{accept}, q_{reject})$$

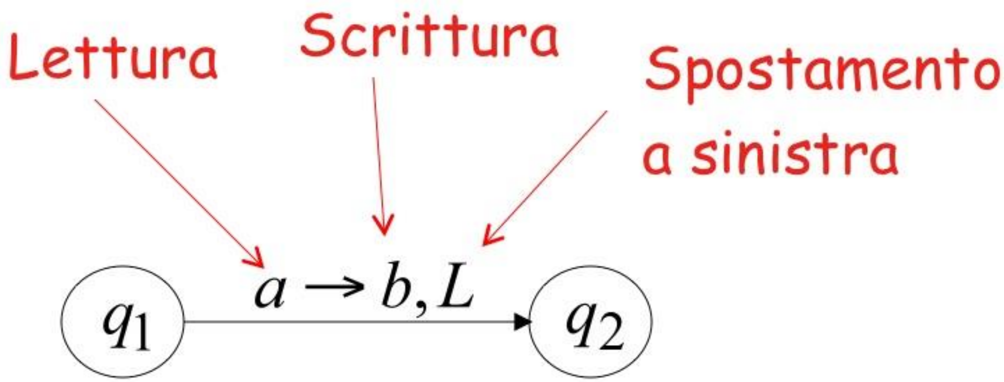
- **Insieme Stati** Q
- **Alfabeto di lavoro** Σ ($\_ \notin \Sigma$)
- Γ: **Alfabeto del nastro** ($\_ \in \Gamma$, $\Sigma \subset \Gamma$)
- $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$: funzione transizione
- q_0: stato **iniziale**
- q_{accept}: stato **accept**
- q_{reject}: stato **reject**

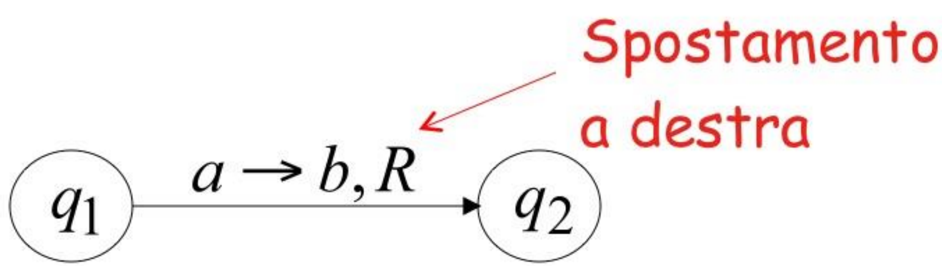
Una MdT



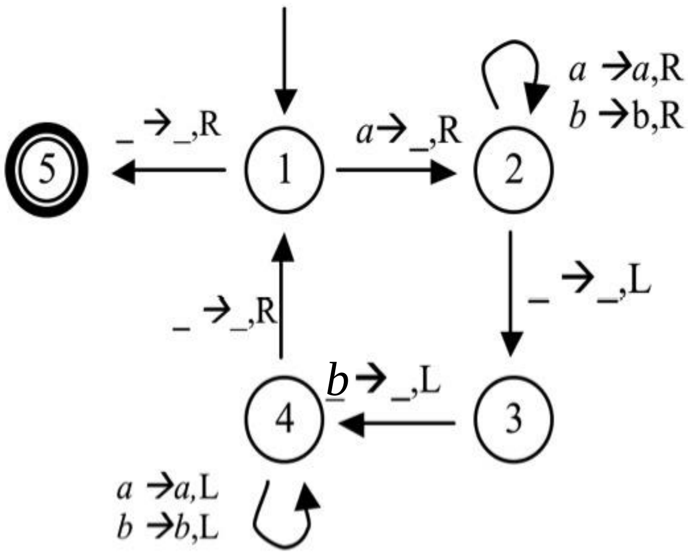
Il contenuto significativo del nastro è una stringa $w \in \Gamma^*$, con la convenzione che il suo ultimo carattere (se $w \neq \varepsilon$) non sia blank.
La stringa w PUO' contenere blank al suo interno.
A volte nel progetto di una MdT si definiscono delle transizioni per scrivere un carattere speciale nella prima cella del nastro, per meglio individuarla.

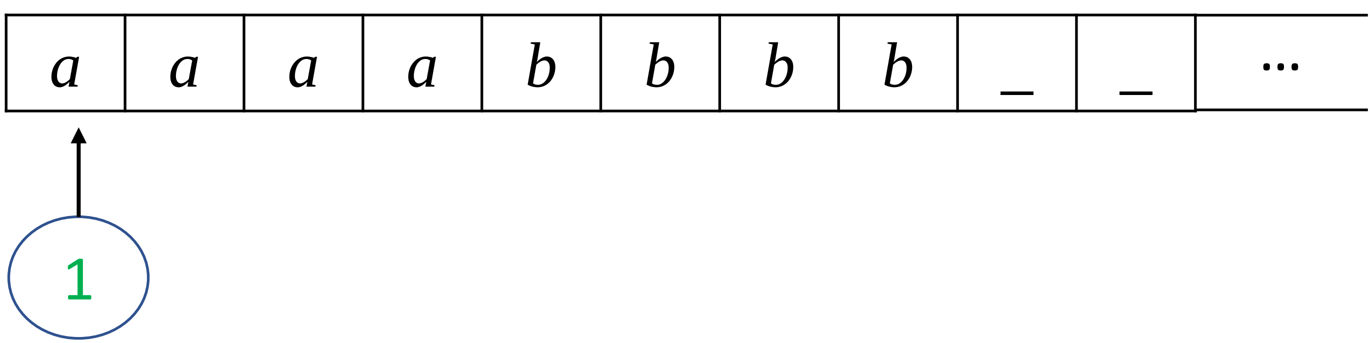
Stati e Transizioni





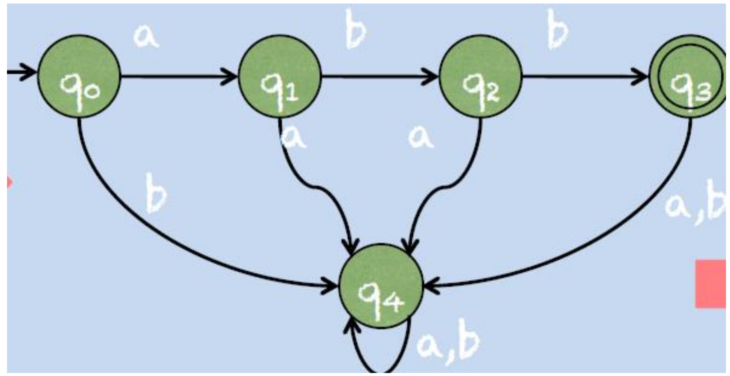
MdT all’opera su *aaaabbbb*

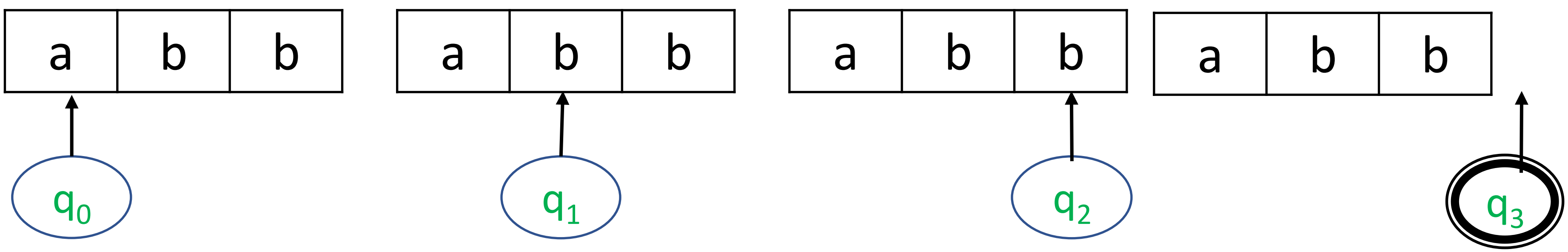




Una computazione del DFA

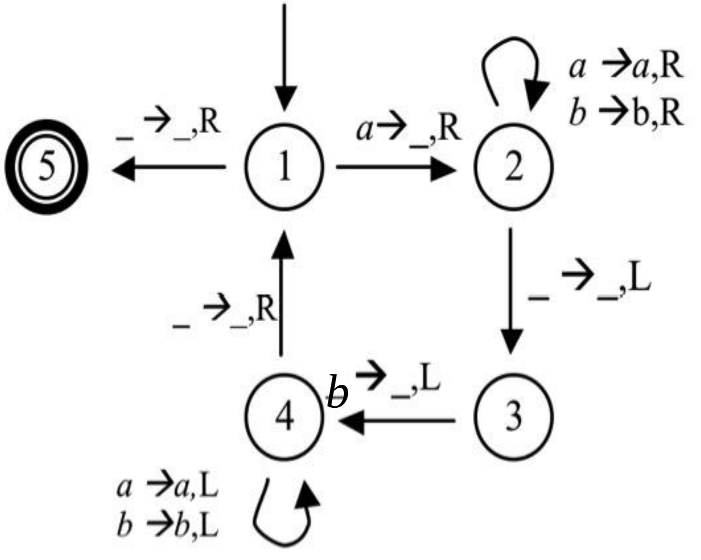
Su input abb





E’ sufficiente elencare gli stati: q_0 , q_1 , q_2 , q_3 .
E possiamo ricostruire tutta la computazione: il resto lo sappiamo.

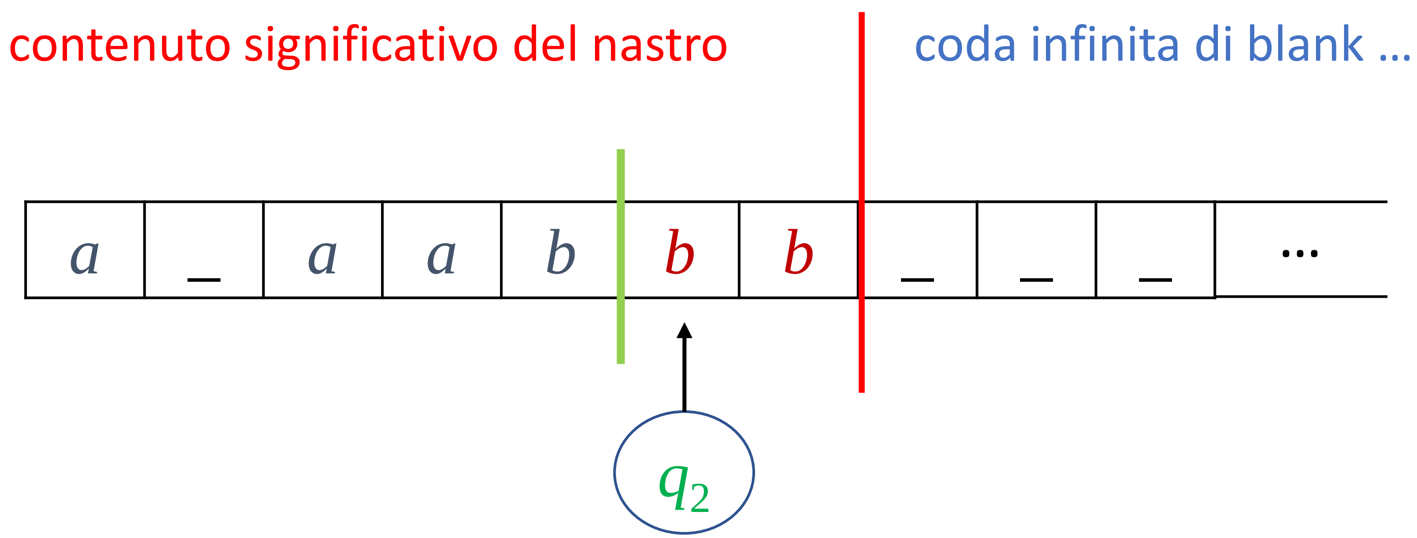
Una computazione della MdT



Esempio: la sequenza 1 , 2 , 2 , 2 , 3 , 4 , 1 , 5 è una computazione valida del MdT su *ab*?
Devo tenere traccia di altre informazioni per poter verificare i passi.

Configurazione di una MdT

Occorre fare un'istantanea di stato, posizione e contenuto significativo del nastro correnti



$$a\_\ a\ a\ b\ q_2\ b\ b$$

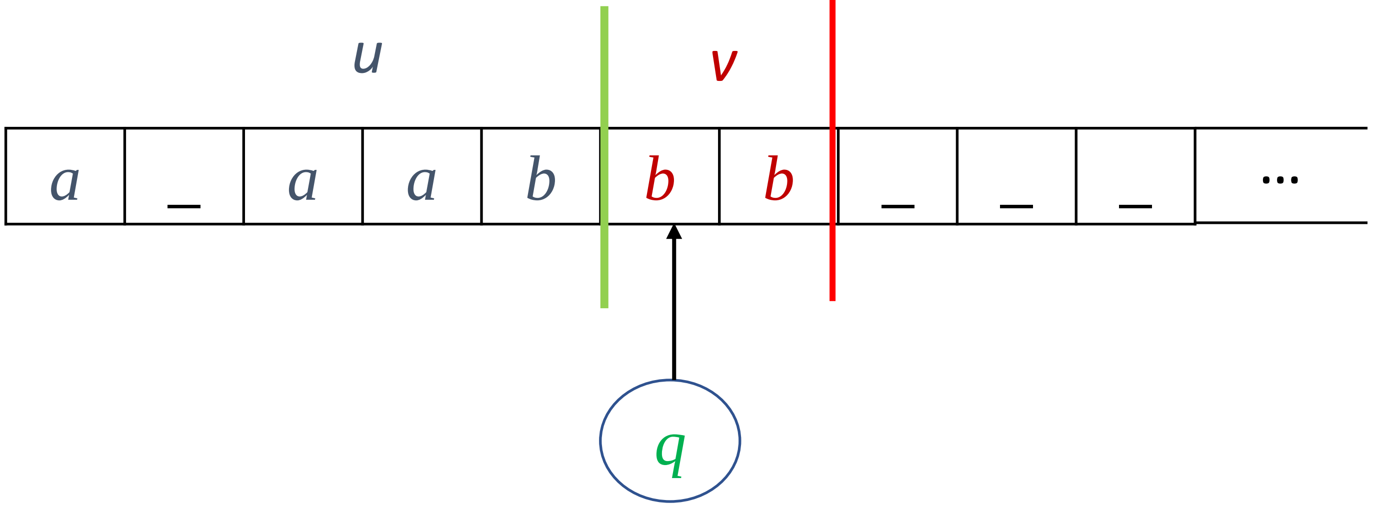
Configurazione di una MdT

La configurazione $C = u\ q\ v$ corrisponde a

contenuto significativo del nastro =

$u\ v$

coda infinita di blank …



Configurazione di una MdT

Descrizione concisa della situazione del calcolo di una MdT ad un certo istante, anche detta descrizione istantanea.

Configurazione di una MdT $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$

$$C = u\ q\ v$$

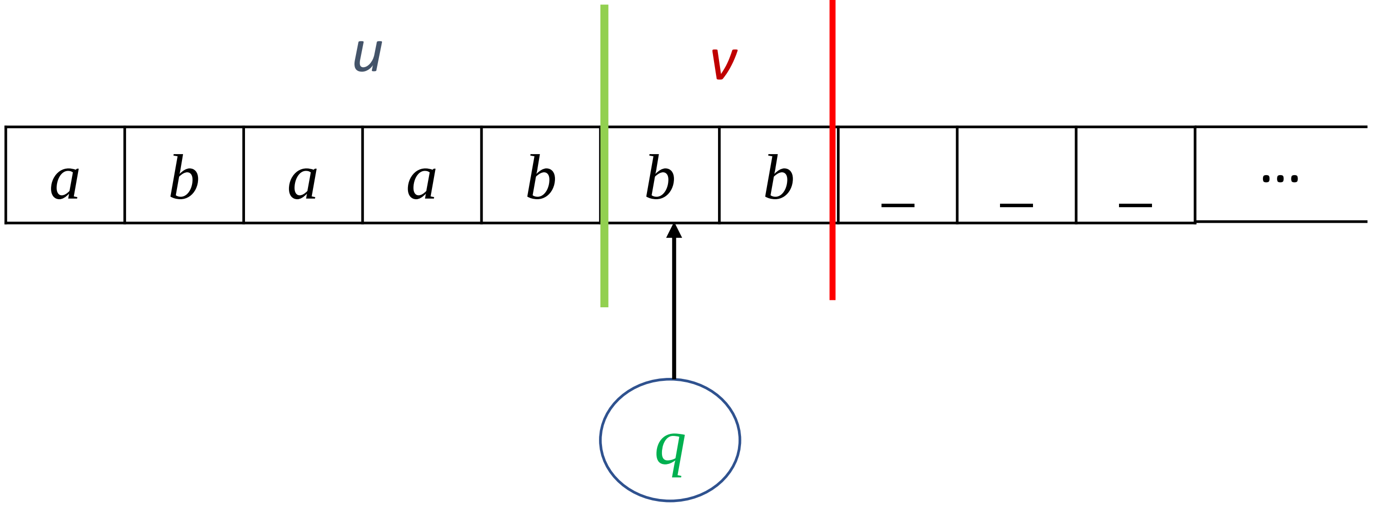
- $q \in$ Q è lo stato corrente
- $u\ v \in \Gamma^*$ è il contenuto significativo del nastro (senza _ finali, se $u\ v \neq \varepsilon$)
- La testina è posizionata sul primo simbolo di v, se $v \neq \varepsilon$, su _ altrimenti

Configurazione di una MdT: esempi

Qual è la configurazione corrispondente?

contenuto significativo del nastro = *u v*

coda infinita di blank ...



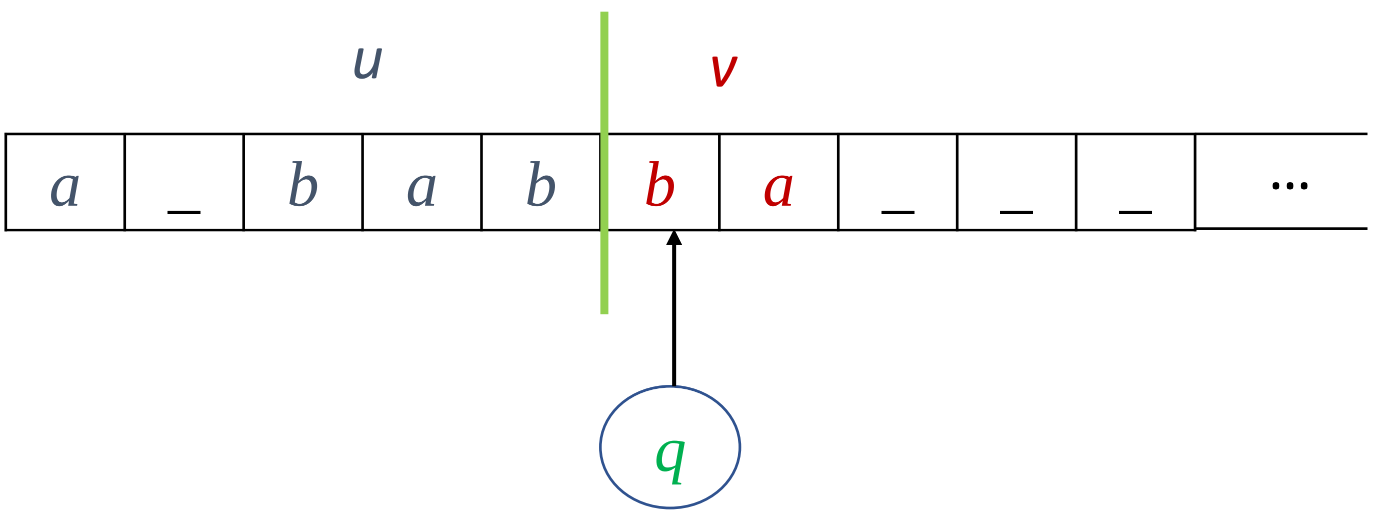
La configurazione corrispondente è: $u\ q\ v = abaab\ q\ bb$

Configurazione di una MdT: esempi

Quale situazione rappresenta la configurazione

$$u\ q\ v = a\_bab\ q\ ba\ \ ?$$

Contenuto significativo del nastro è $u\ v = a\_bab\ ba$



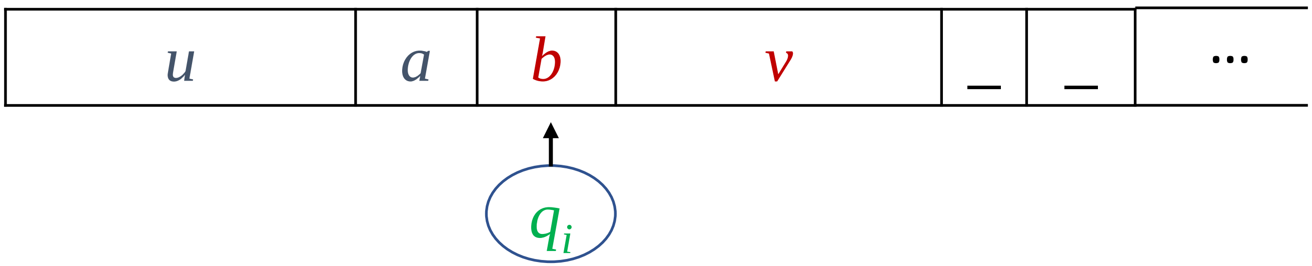
Configurazioni particolari

In una configurazione $C = u\ q\ v$, sia u che v possono essere ε

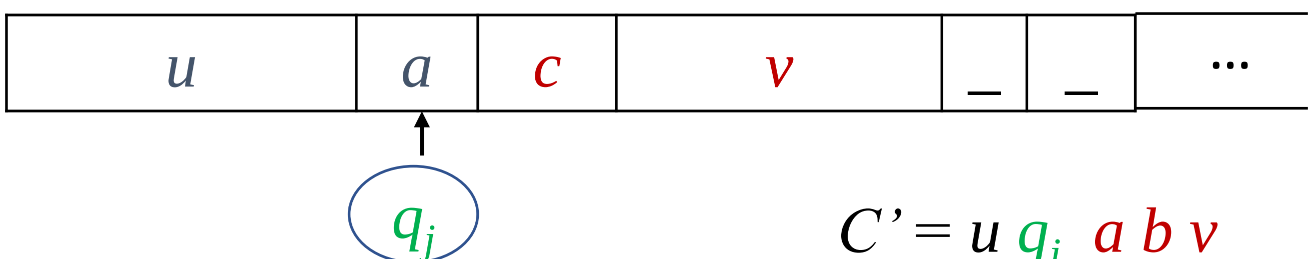
- Se $u = \varepsilon$, $C = q\ v$, allora la testina è posizionata sulla prima lettera di v nella prima cella del nastro (contenuto significativo nastro è ε v = v)
- Se $v = \varepsilon$, $C = u\ q$, allora la testina è posizionata sulla prima cella della porzione di nastro contenente solo _ (ricorda che uv=u è la porzione significativa del nastro, senza la coda infinita di _)
- $u\ q$ è equivalente a $u\ q$ _ ; la parte vuota del nastro è riempita con tutti _

Computazione di una MdT: passo verso sinistra

Supponiamo che $C = u\ a\ q_i\ b\ v$



Se $\delta\ (q_i, b) = (\ q_j\ , c, L)$
quale sarà la successiva configurazione C' ?

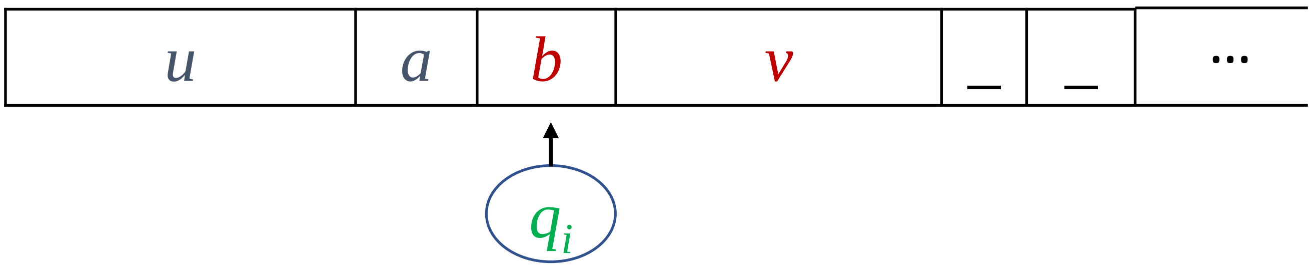


$C' = u\ q_j\ \ a\ b\ v$

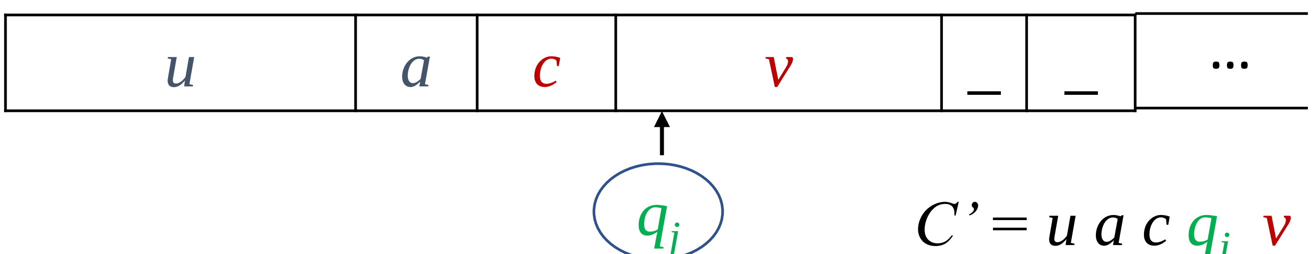
Diremo che C produce C', in simboli $C \rightarrow C'$

Computazione di una MdT: passo verso destra

Supponiamo che $C = u\ a\ q_i\ b\ v$



Se $\delta\ (q_i, b) = (\ q_j\ , c, R)$
quale sarà la successiva configurazione C' ?



$C' = u\ a\ c\ q_j\ \ v$

Diremo che C produce C', in simboli $C \rightarrow C'$

Casi particolari

La definizione generale è più complessa perché bisogna considerare anche i casi particolari ($C = qv$, $C = uq$ con u, v eventualmente uguali a ϵ).

Ad esempio $q_i bv$ produce $q_j cv$
se $\delta(q_i, b) = (q_j, c, L)$.

$q_i bv$ produce $cq_j v$ se $\delta(q_i, b) = (q_j, c, R)$.

Passo di computazione

Siano C_1, C_2 due configurazioni di una MdT M.

Se C_1 produce C_2, scriveremo

$$C_1 \to C_2$$

La trasformazione $\to$ di C_1 in C_2 prende il nome di **passo di computazione**.

Corrisponde a un'applicazione della funzione di transizione di M.

Esempio

$$
\begin{aligned}
\delta(q_0, 0) &= (q_0, 0, R), \quad \delta(q_0, 1) = (q_0, 1, R), \\
\delta(q_0, \sqcup) &= (q_1, \sqcup, L), \\
\delta(q_1, 1) &= (q_2, 1, L), \quad \delta(q_2, 0) = (q_3, 0, L), \\
\delta(q_3, 1) &= (q_{accept}, 1, L)
\end{aligned}
$$

$q_0 11 \to 1 q_0 1 \to 11 q_0 \to 1 q_1 1 \to q_2 11 \to q_{reject} 11$

$q_0 101 \to 1 q_0 01 \to 10 q_0 1 \to 101 q_0 \to 10 q_1 1 \to 1 q_2 01 \to$
$q_3 101 \to q_{accept} 101$

Computazione di una MdT

Siano C, C' *configurazioni.*
$C \rightarrow^* C'$ *se esistono configurazioni* $C_1, \ldots, C_k$, $k \geq 1$ *tali che*

1. $C_1 = C$,
2. $C_i \rightarrow C_{i+1}$, *per* $i \in \{1, \ldots, k-1\}$,
   (ogni C_i *produce* C_{i+1}*)*
3. $C_k = C'$.

Diremo che $C \rightarrow^* C'$ *è una* **computazione** *(di lunghezza* $k-1$*).*

Configurazioni

Una configurazione C si dice:

- iniziale su input w se $C = q_0 w$, con w $\in \Sigma^*$
- di accettazione se $C = u\ q_{accept}\ v$
- di rifiuto se $C = u\ q_{reject}\ v$

Poiché non esistono transizioni da q_{accept} e da q_{reject}, allora le configurazioni di accettazione e di rifiuto sono dette configurazioni di arresto.

Parola accettata o rifiutata

Definizione

Una MdT M accetta una parola $w \in \Sigma^*$ se esiste una computazione $C \to^* C'$, dove $C = q_0 w$ è la configurazione iniziale di M con input w e $C' = uq_{accept}v$ è una configurazione di accettazione.

Una MdT M rifiuta una parola $w \in \Sigma^*$ se esiste una computazione $C \to^* C'$, dove $C = q_0 w$ è la configurazione iniziale di M con input w e $C' = uq_{reject}v$ è una configurazione di rifiuto.

Risultati di una computazione

Tre possibili Risultati computazione:

1. M accetta – se si ferma in q_{accept}
2. M rifiuta – se si ferma in q_{reject}
3. M cicla/loop – se non si ferma mai

Mentre M funziona non si può dire se è in loop; si potrebbe fermare in seguito oppure no.

Esempio

$$\begin{aligned} \delta(q_0, 0) &= (q_0, 0, R), \quad \delta(q_0, 1) = (q_0, 1, R), \\ \delta(q_0, \sqcup) &= (q_1, \sqcup, L), \\ \delta(q_1, 1) &= (q_2, 1, L), \quad \delta(q_2, 0) = (q_3, 0, L), \\ \delta(q_3, 1) &= (q_{accept}, 1, L) \end{aligned}$$

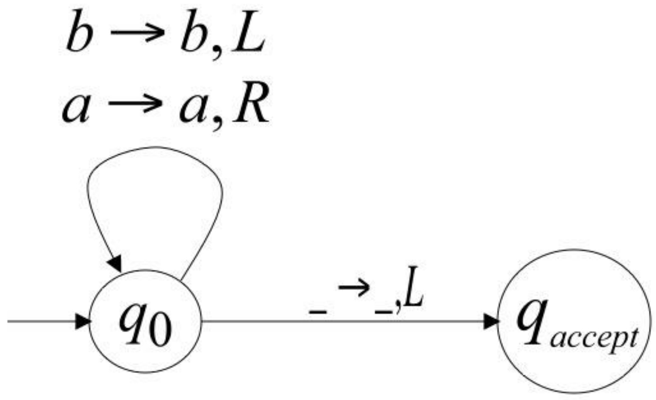
$q_0 101 \to 1 q_0 01 \to 10 q_0 1 \to 101 q_0 \to 10 q_1 1 \to 1 q_2 01 \to$
$q_3 101 \to q_{accept} 101$

$q_0 101 \to^* q_{accept} 101$: 101 è accettata.

$q_0 11 \to 1 q_0 1 \to 11 q_0 \to 1 q_1 1 \to q_2 11 \to q_{reject} 11$

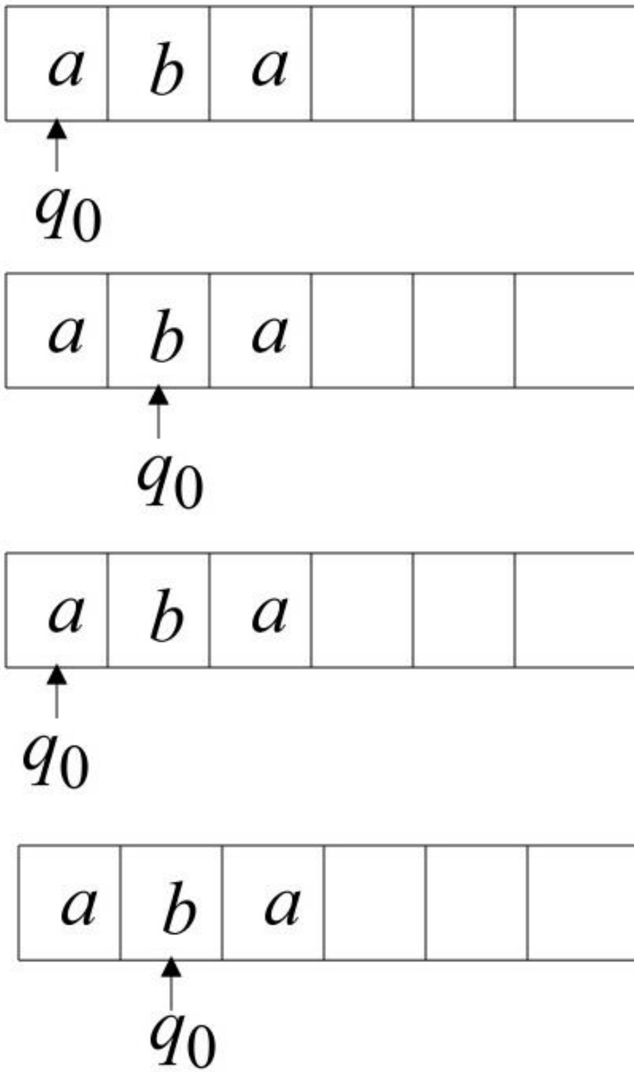
$q_0 11 \to^* q_{reject} 11$: 11 è rifiutata.

Esempio di non terminazione



$$q_0\ aba \to^* q_0\ aba$$

10



cicla e non si ferma mai

$\boxed{\textit{aba} \text{ non è accettata.}}$

È errato dire che *aba* è rifiutata.

Esempio di non terminazione

Esempio: $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, con
$Q = \{q_0, q_{accept}, q_{reject}\}$, $\Sigma = \{a, b\}$, $\Gamma = \{a, b, \sqcup\}$,
$\delta(q_0, a) = (q_0, a, R)$, $\delta(q_0, b) = (q_0, b, L)$,
$\delta(q_0, \sqcup) = (q_{accept}, \sqcup, L)$.

$$q_0 aba \to a q_0 ba \to q_0 aba \to a q_0 ba \to \ldots$$

$$q_0\, aba \to^* q_0\, aba$$

cicla e non si ferma mai

$\boxed{aba \text{ non è accettata.}}$

È errato dire che *aba* è rifiutata.

Linguaggio riconosciuto da una MdT

Definizione

Sia $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ *una MdT. Il linguaggio* $L(M)$ *riconosciuto da* M*, è l'insieme delle stringhe che* M *accetta:*

$$L(M) = \{w \in \Sigma^* \mid \exists u, v \in \Gamma^* \; q_0 w \to^* u q_{accept} v\}.$$

Quindi

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

Decidere

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

$$R(M) = \{w \in \Sigma^* \mid M \text{ rifiuta } w\}$$

In generale $L(M) \cup R(M)$ non coincide con Σ^*.

Se $L(M) \cup R(M) = \Sigma^*$, allora M si arresta su ogni input.

In tal caso M è chiamata un decisore (o decider) ed $L(M)$ è il linguaggio deciso da M.

Dal punto di vista delle macchine

Definizione

Una MdT $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ *è un* **decisore** *(o* **decider***) se, per ogni* $w \in \Sigma^*$*, esistono* $u, v \in \Gamma^*$ *e* $q \in \{q_{accept}, q_{reject}\}$ *tali che*

$$q_0 w \to^* uqv$$

Definizione

Una MdT M **decide** *un linguaggio L se M è un decisore e L = L(M).*

In tal caso L è **deciso** *da M.*

Dal punto di vista dei linguaggi

Definizione

Un linguaggio $L \subseteq \Sigma^*$ è Turing riconoscibile se esiste una macchina di Turing $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ tale che:

1. *M riconosce L*
   (cioè $L = L(M) = \{w \in \Sigma^* \mid \exists u, v \in \Gamma^* \ q_0 w \rightarrow^* u q_{accept} v\}$).

Definizione

Un linguaggio $L \subseteq \Sigma^*$ è decidibile se esiste una macchina di Turing $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ tale che:

1. *M riconosce L*
   (cioè $L = L(M) = \{w \in \Sigma^* \mid \exists u, v \in \Gamma^* \ q_0 w \rightarrow^* u q_{accept} v\}$).
2. *M si arresta su ogni input (cioè per ogni $w \in \Sigma^*$, $q_0 w \rightarrow^* uqv$ con $q \in \{q_{accept}, q_{reject}\}$).*

Vedremo che l'insieme dei linguaggi decidibili è un sottoinsieme proprio dell'insieme dei linguaggi Turing riconoscibili.

Come conseguenza delle definizioni, un linguaggio L è Turing riconoscibile ma non decidibile se:

1. esiste una MdT che riconosce L (quindi accetta tutte e sole le stringhe di L)
2. non esiste nessuna MdT M tale che M accetta tutte le stringhe in L e rifiuta tutte le stringhe che appartengono al complemento $\overline{L}$ di L.

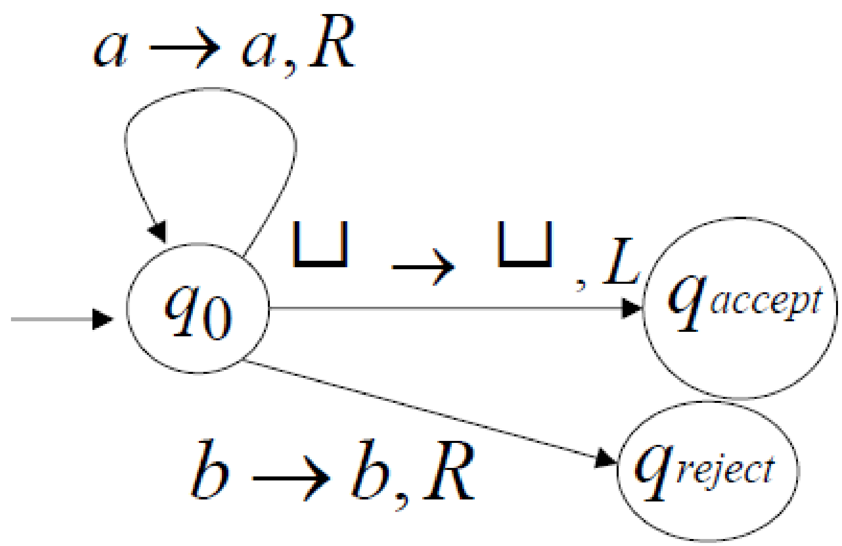
Non confondere la proprietà di un linguaggio (essere o non essere Turing riconoscibile, essere o non essere decidibile) con la proprietà di una MdT (essere o non essere un decider).

Esempio: $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, con
$Q = \{q_0, q_{accept}, q_{reject}\}$, $\Sigma = \{a, b\}$, $\Gamma = \{a, b, \sqcup\}$,
$\delta(q_0, a) = (q_0, a, R)$, $\delta(q_0, b) = (q_0, b, L)$,
$\delta(q_0, \sqcup) = (q_{accept}, \sqcup, L)$.

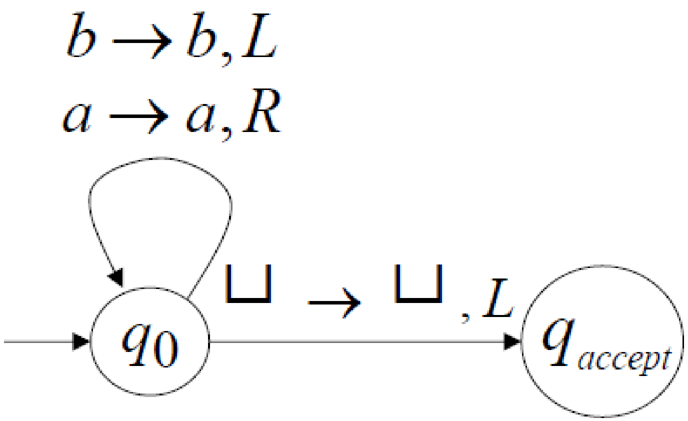
M non è un decider ma $L(M) = a^*$ è decidibile.

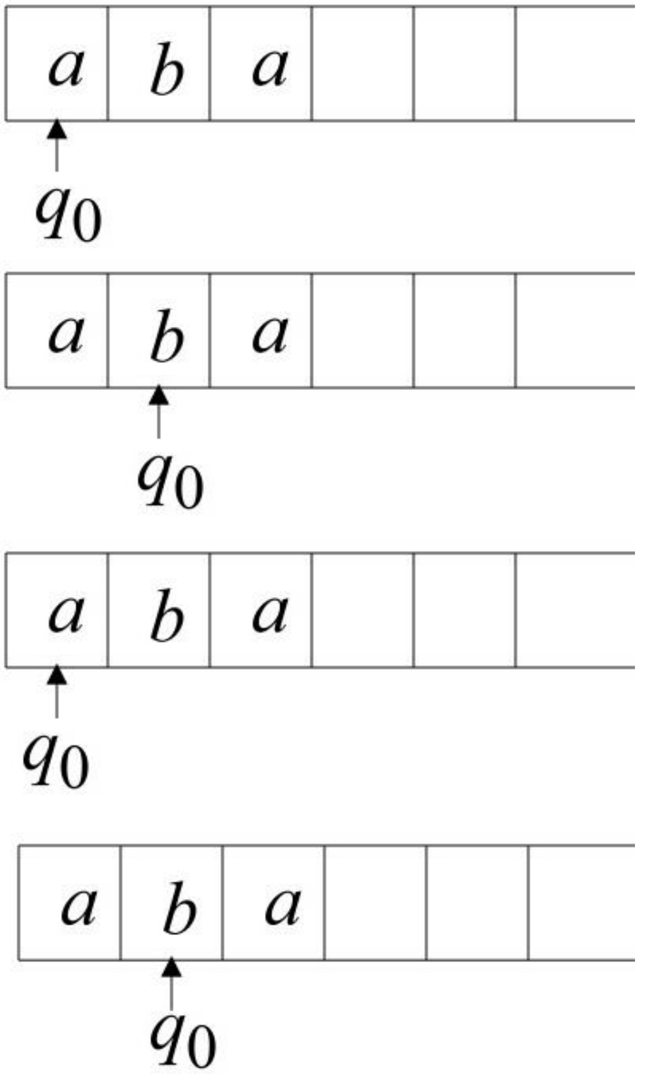
Il linguaggio a* è decidibile

Una macchina di Turing che decide a^*



Una macchina di Turing che non è un decisore e che riconosce a^*





Non è un decisore perché su *aba* (e non solo) cicla e non si ferma mai

Linguaggi T-riconoscibili, decidibili e regolari

Ogni linguaggio regolare è Turing riconoscibile?

Ogni linguaggio regolare è decidibile?

Ogni linguaggio decidibile è regolare?

Ogni linguaggio Turing riconoscibile è regolare?

Esempio

Consideriamo il linguaggio

$$L = \{0^{2^n} \mid n > 0\}$$

insieme stringhe di 0 la cui lunghezza è potenza di 2

Nota. Il linguaggio non è regolare

Vogliamo costruire una MdT M_2 che lo decide.

Esempio

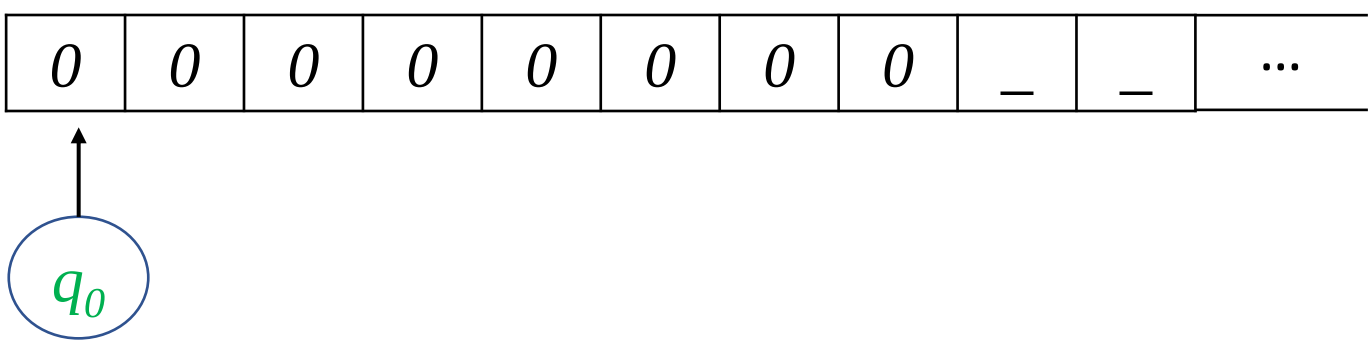
Come riconoscere se il numero di 0 è una potenza di 2?

1, 2, 4, 8, 16, 32, ...

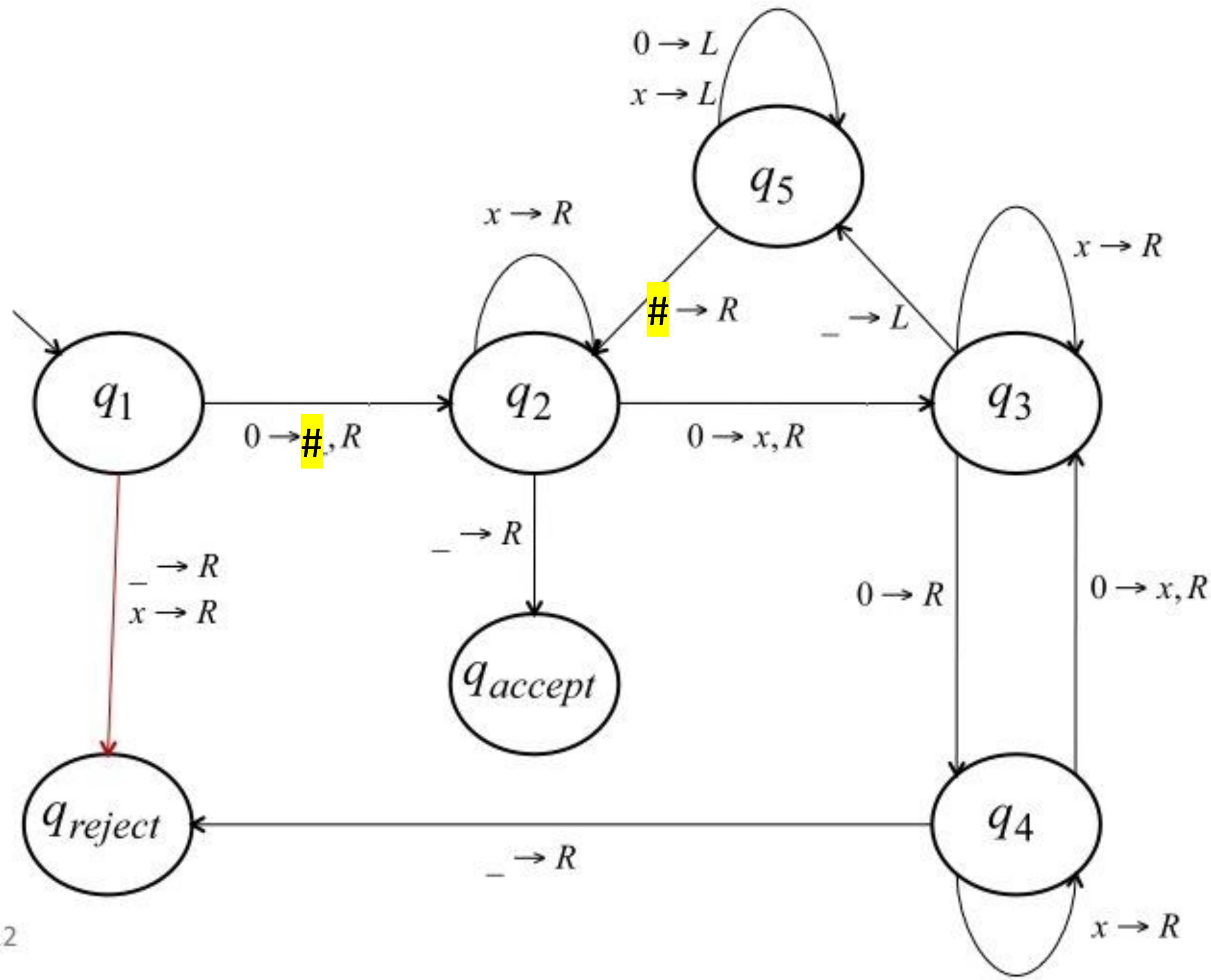
Innanzitutto deve essere pari; se dispari rifiuto.

Se pari?

Le potenze di 2 hanno la caratteristica che dividendo ripetutamente per 2 trovo sempre numeri pari, fino ad arrivare ad 1.



MdT per 0^{2^n}


$0 \to L$
$x \to L$
q_5
$x \to R$
$\# \to R$
$\_ \to L$
$x \to R$
q_1
q_2
q_3
$0 \to \#, R$
$0 \to x, R$
$\_ \to R$
$\_ \to R$
$x \to R$
$0 \to R$
$0 \to x, R$
q_{accept}
q_{reject}
q_4
$\_ \to R$
$x \to R$

Esercizio svolto

Ogni linguaggio regolare L è decidibile.

Dato un $(Q, \Sigma, \delta, q_0, F)$ un DFA che riconosce L, costruire una MdT $(Q', \Sigma, \Gamma, \delta', q'_0, q_{accept}, q_{reject})$ che decide L.