Computabilità
Appunti_06 e 07 (14 e 16/12/2022)

Mario Verdicchio
Università degli Studi di Bergamo
Anno Accademico 2022-2023

Esercizio 1

- Creare una macchina di Turing (MT) che accetti il seguente linguaggio:

  0^i, con i intero pari

Esercizio 1

- Creare una macchina di Turing (MT) che accetti il seguente linguaggio:

  0^i, con i intero pari

  Concepire l'algoritmo che la governa e descriverne il funzionamento per mezzo di una tavola di istruzioni

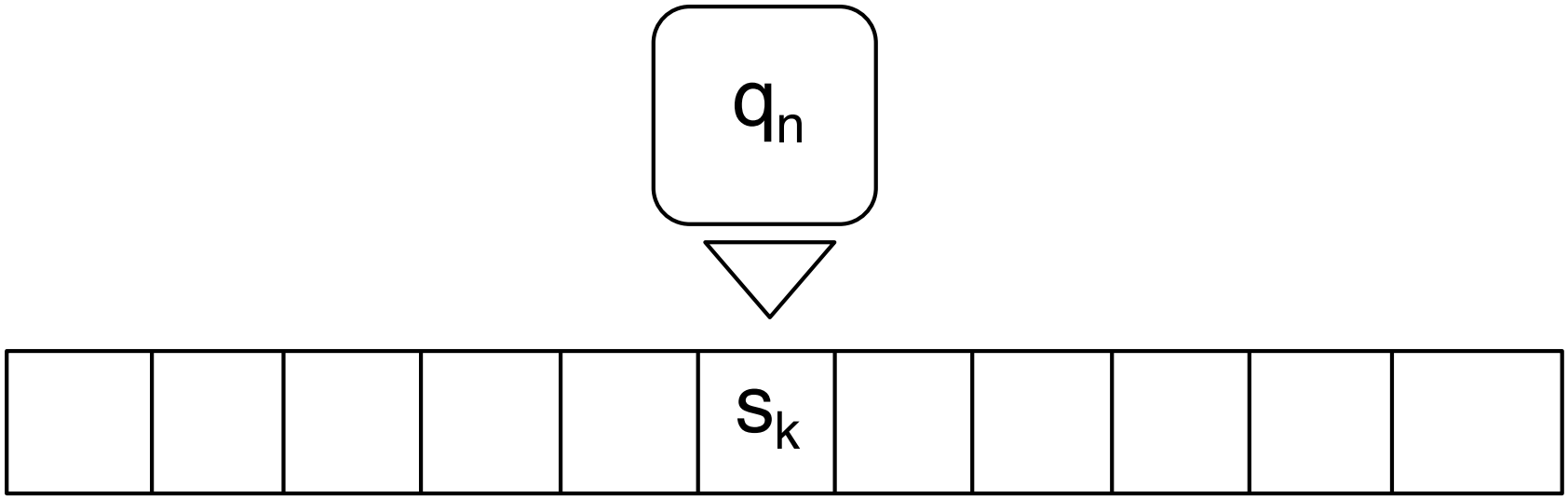
Esercizio 1

- Creare una macchina di Turing (MT) che accetti il seguente linguaggio:

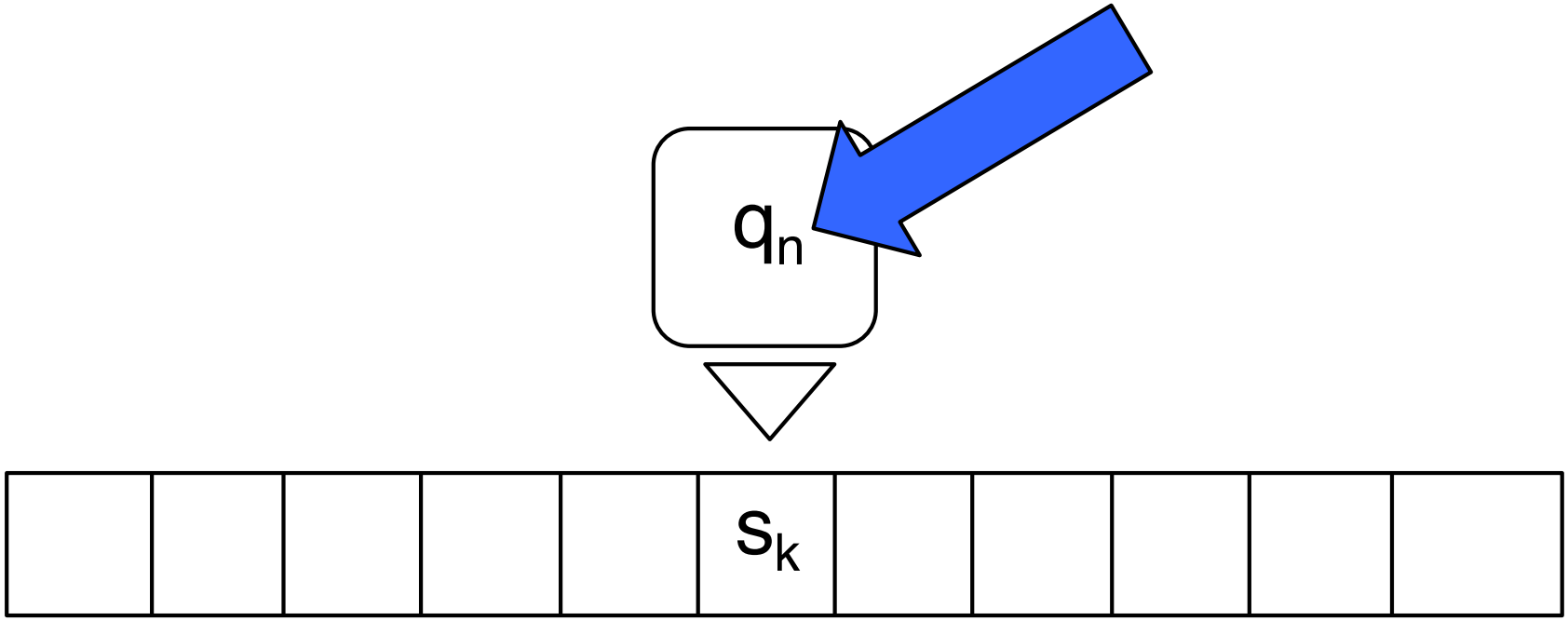
  0^i, con i intero pari

  Se la MT trova sul nastro una sequenza di simboli che rispetta la definizione del linguaggio, l'output sul nastro è 0 (= sì).
  Se la MT trova sul nastro una sequenza di simboli che NON rispetta la definizione del linguaggio, l'output sul nastro è 1 (= no).
  Attenzione: sul nastro deve rimanere solo lo 0 oppure l'1. La sequenza di input va cancellata.

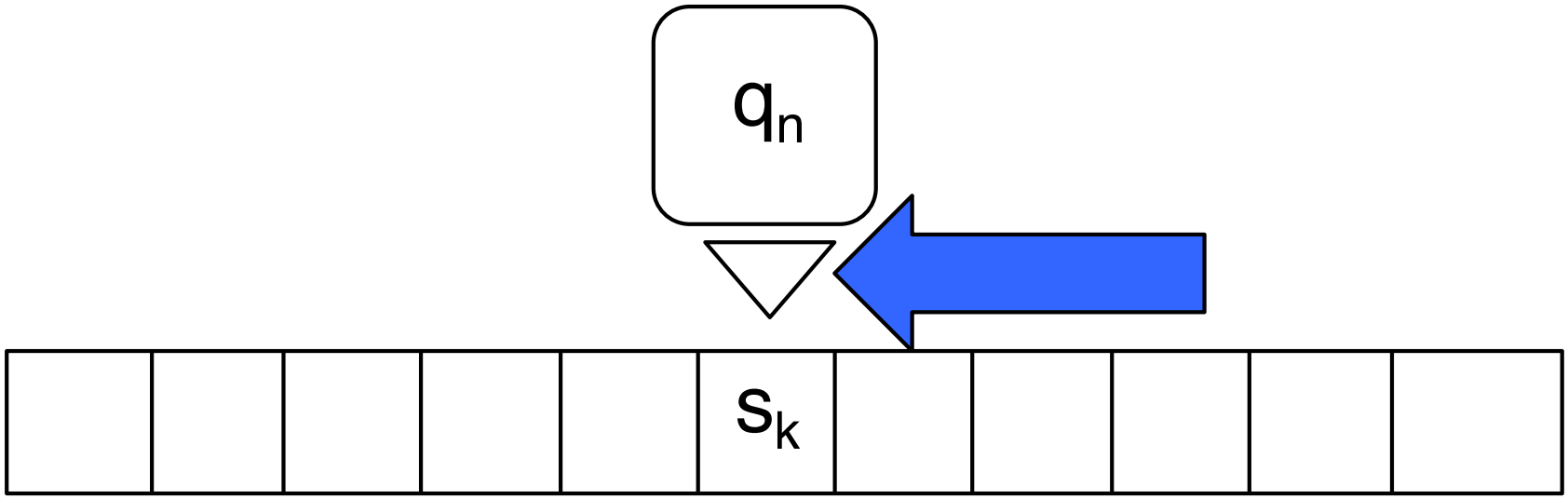
La Macchina di Turing



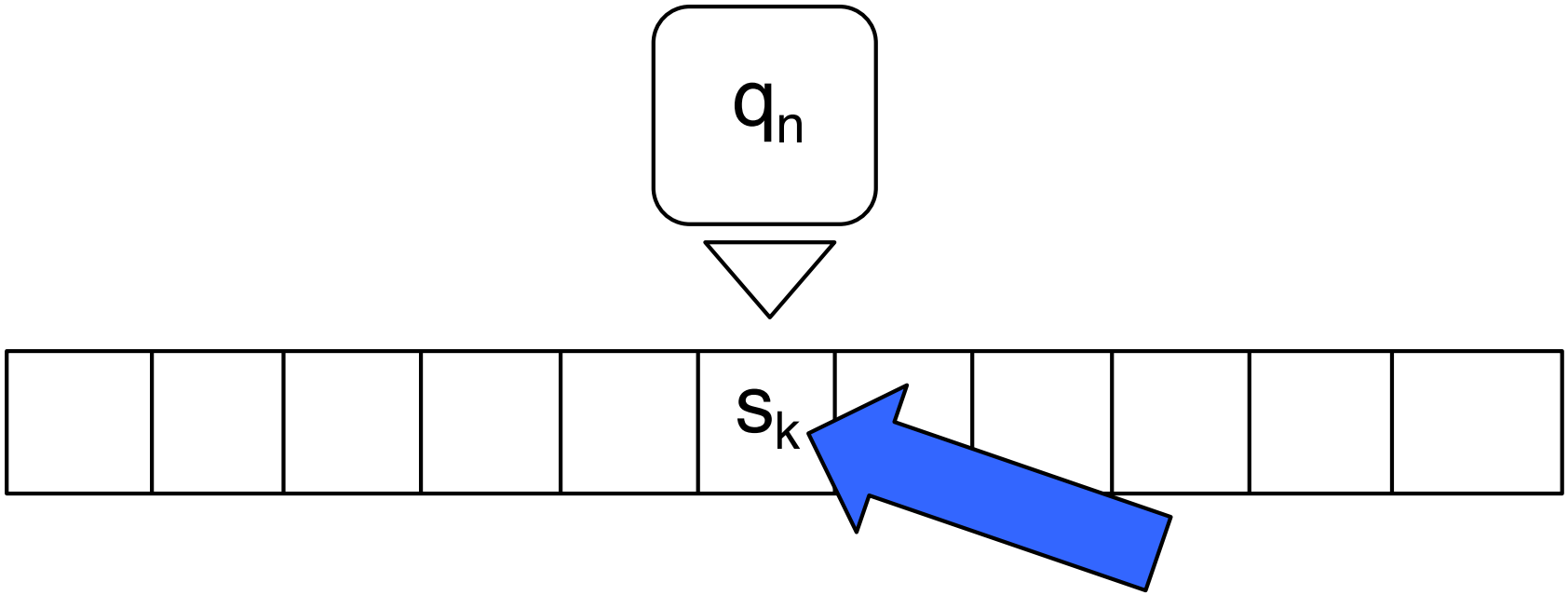
Stato interno



Testina di lettura e scrittura



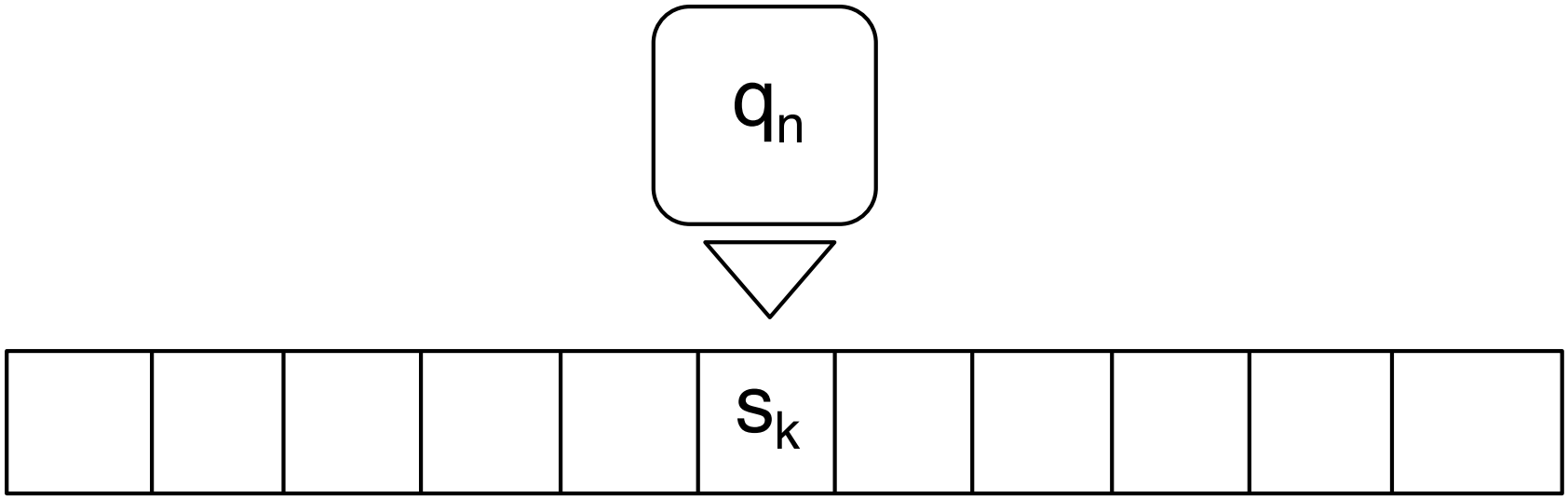
Simbolo sul nastro biinfinito



Funzionamento della MT

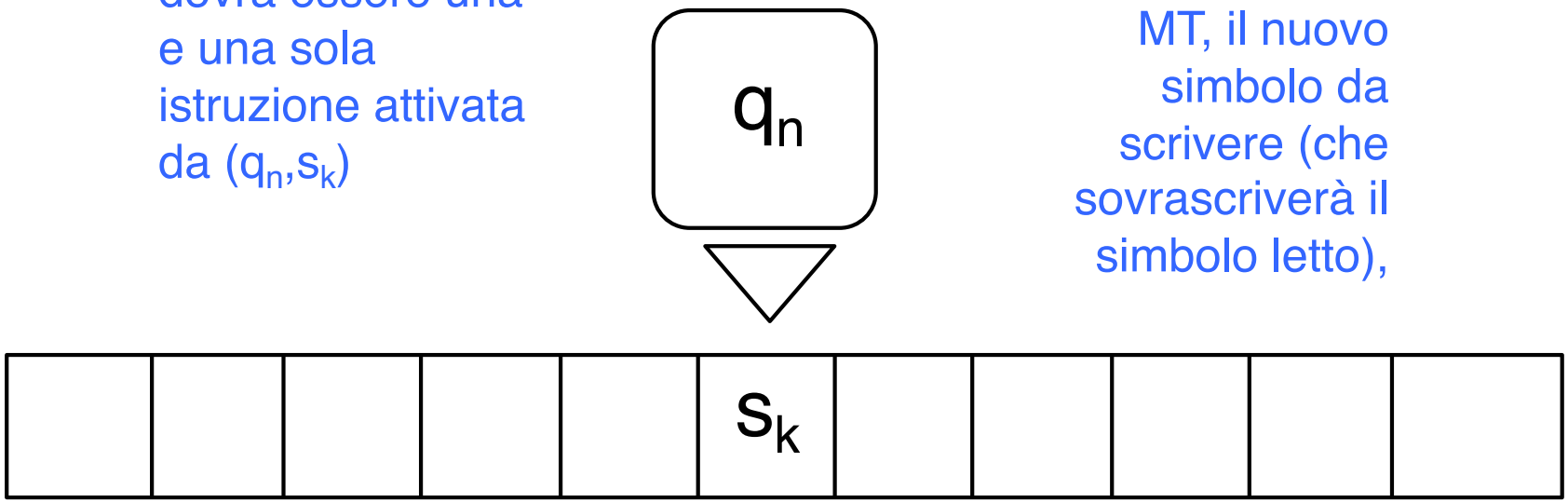
- La MT esegue l'istruzione all'interno del suo programma che è attivata dalla configurazione in cui la MT si trova
- Tale configurazione è determinata da:
  - lo stato interno della MT
  - il simbolo letto dalla testina

La MT è nella configurazione (q_n,s_k)



Funzionamento della MT

Nel suo programma ci dovrà essere una e una sola istruzione attivata da (q_n,s_k)

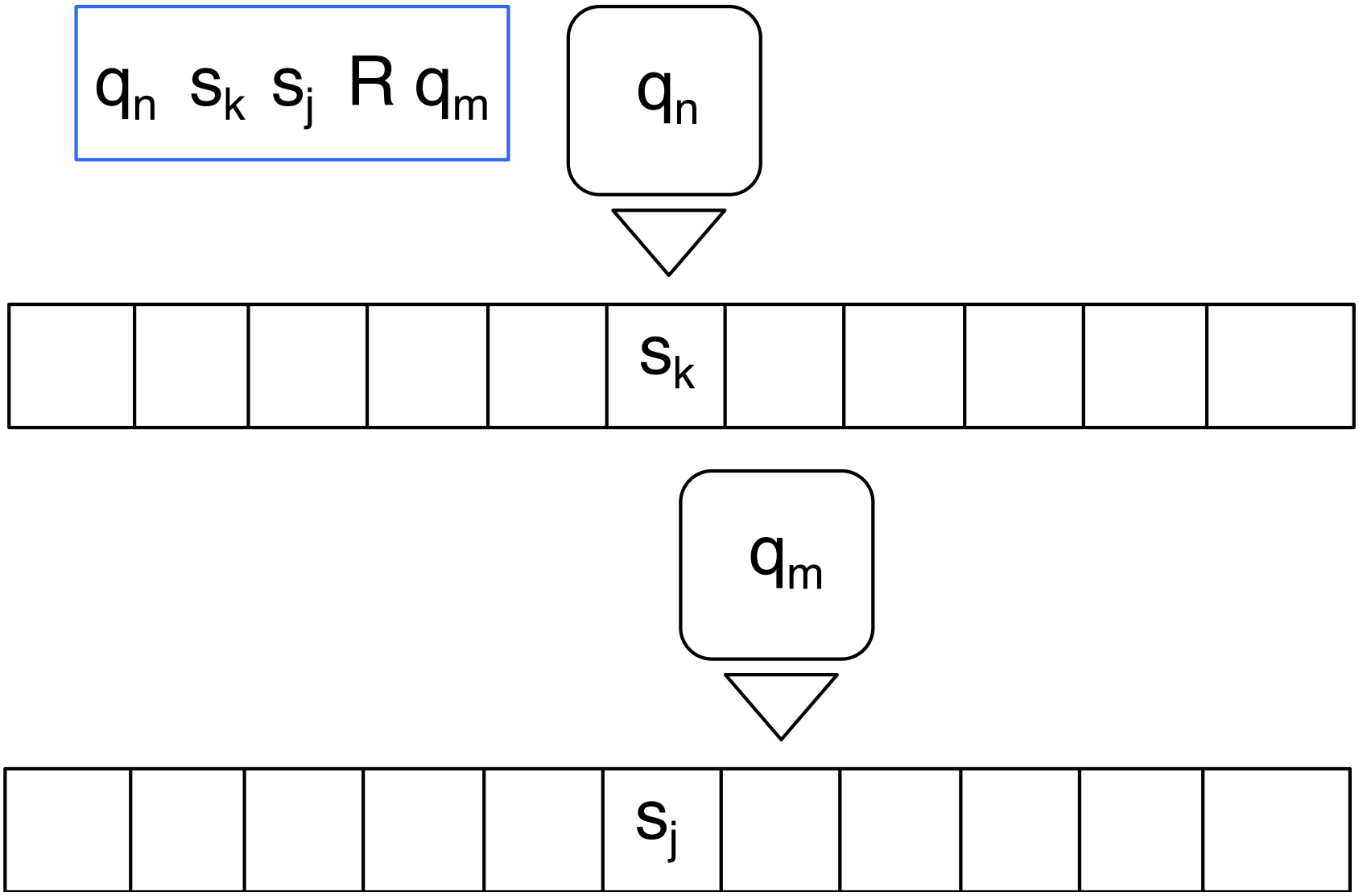


Tale istruzione indicherà il nuovo stato interno della MT, il nuovo simbolo da scrivere (che sovrascriverà il simbolo letto),

e il movimento da eseguire (L: left, R: right, C: center)

$q_n \ s_k \ s_j \ R \ q_m$

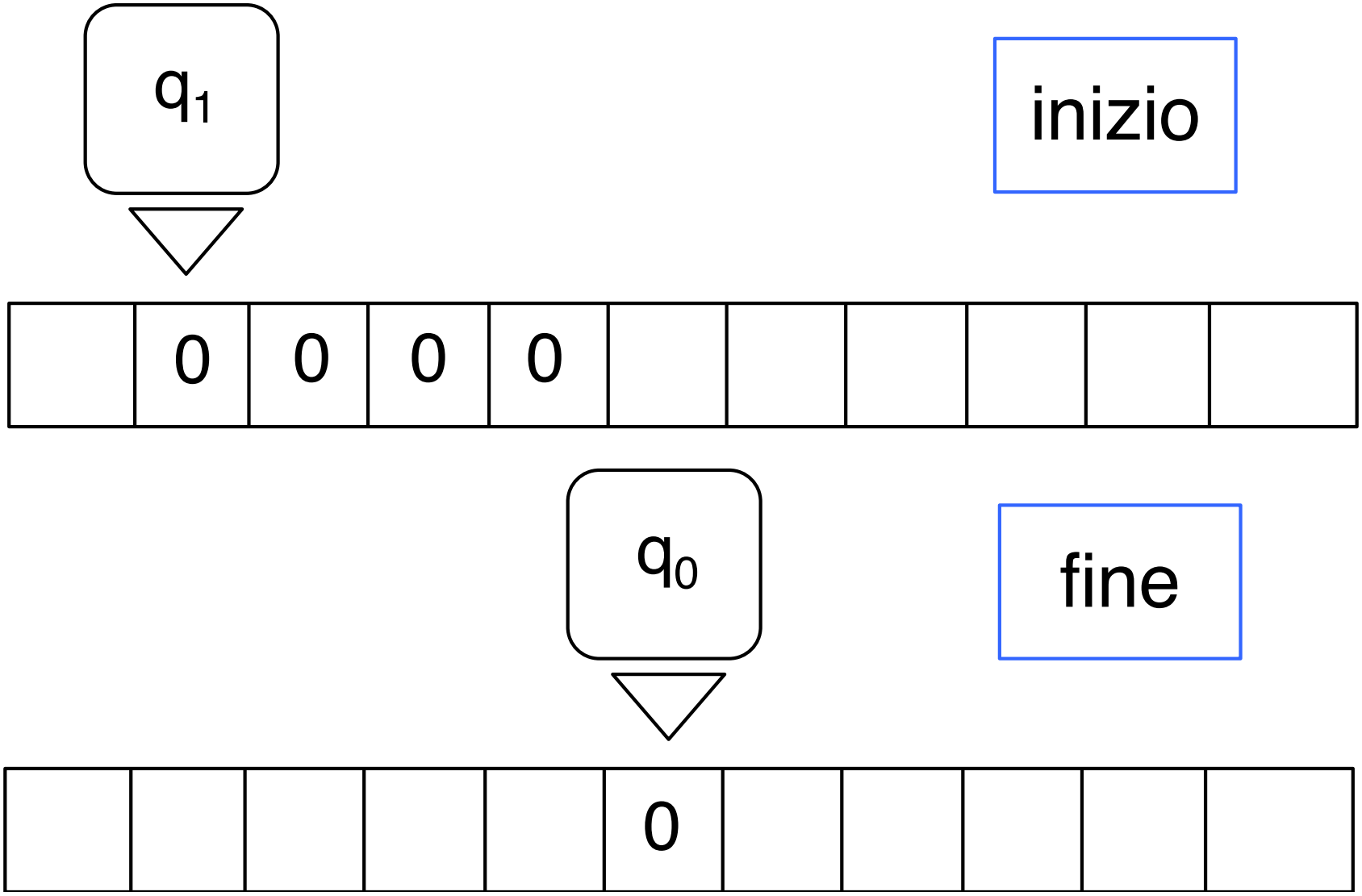
- Questa istruzione vuole dire:
  - se la MT ha stato interno q_n
  - e se legge sul nastro il simbolo s_k
  - allora scrive sul nastro il simbolo s_j
  - esegue il movimento R
  - e passa allo stato interno q_m

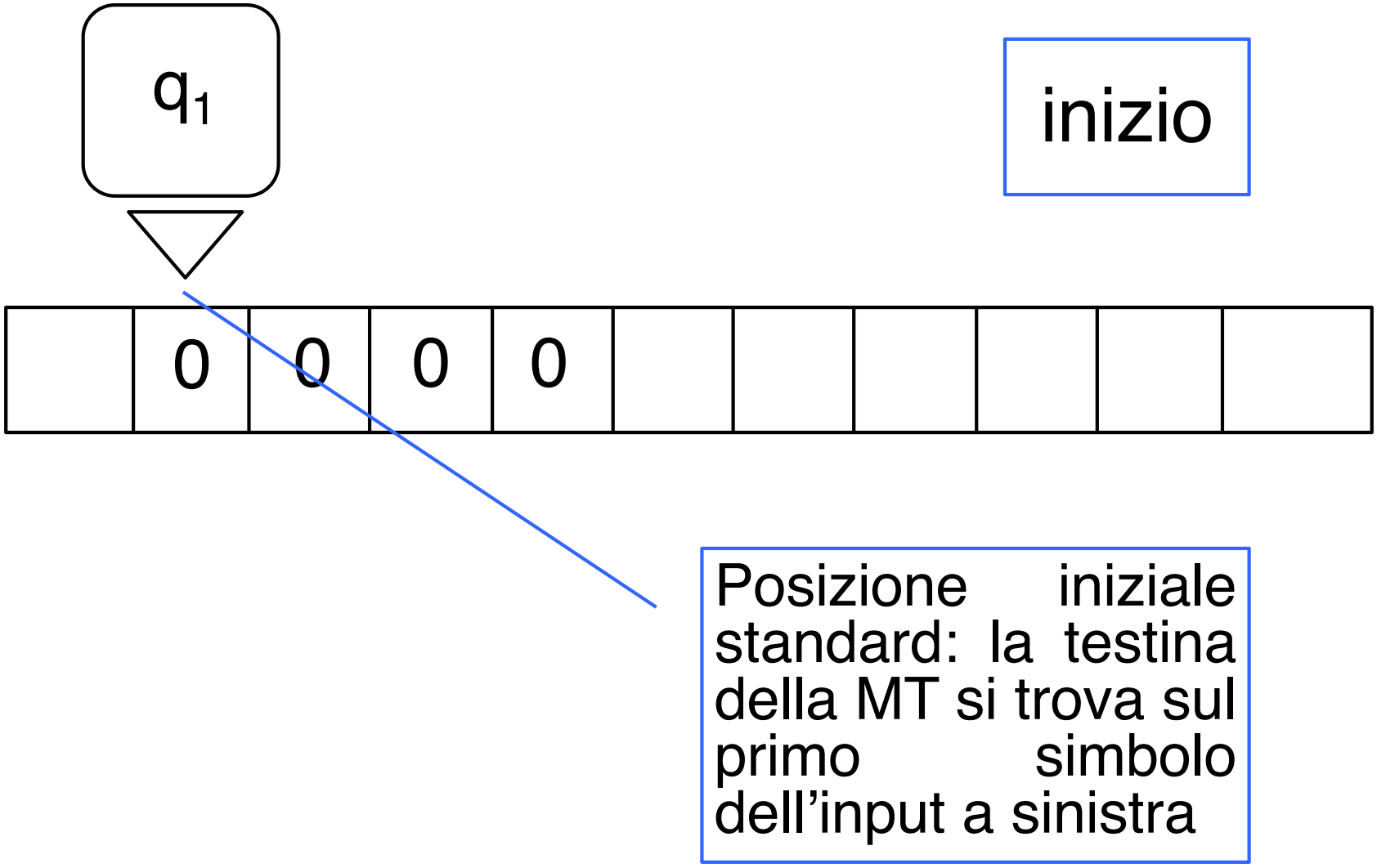

q_n s_k s_j R q_m
q_n
s_k
q_m
s_j

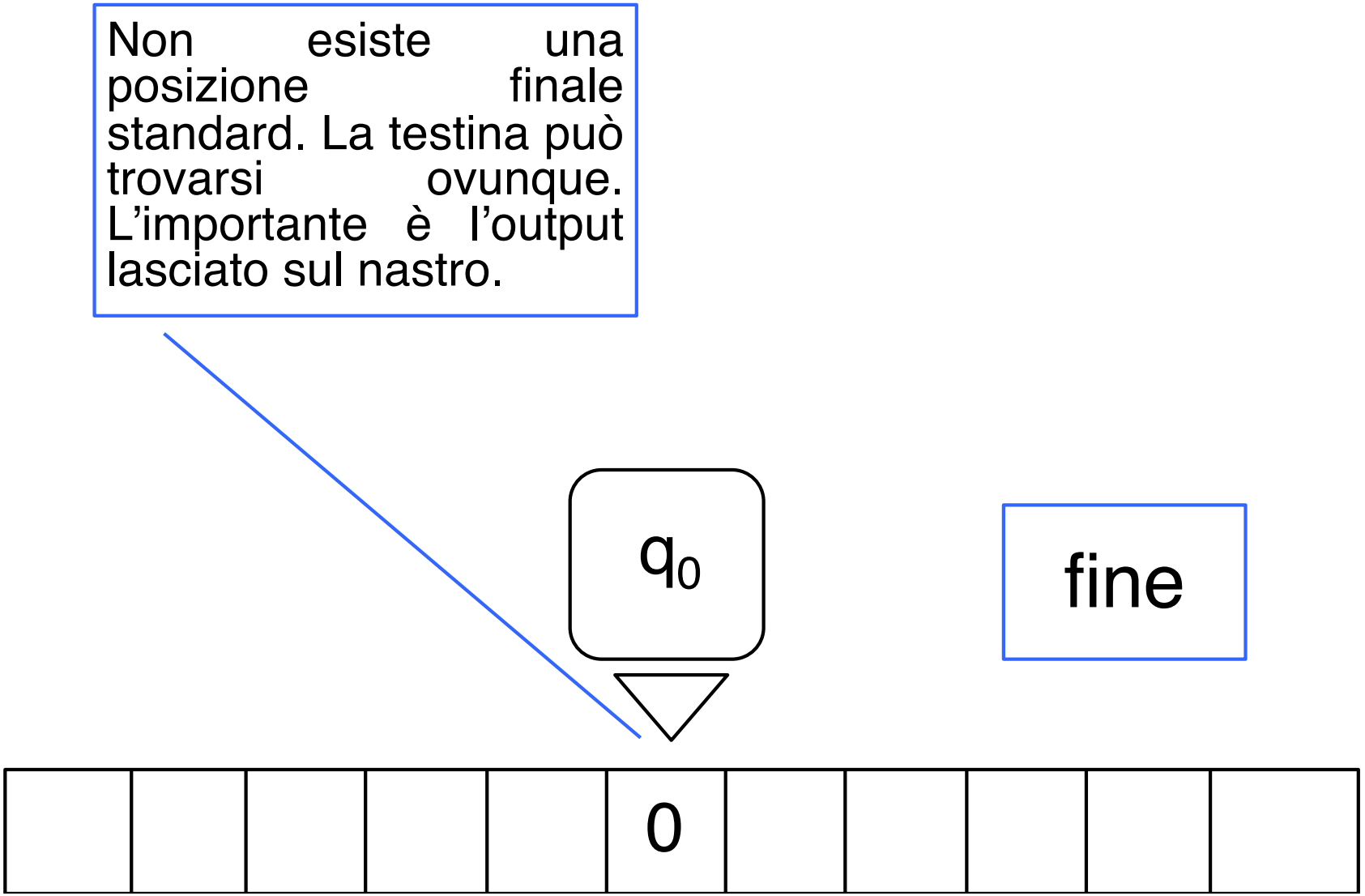
Esercizio 1

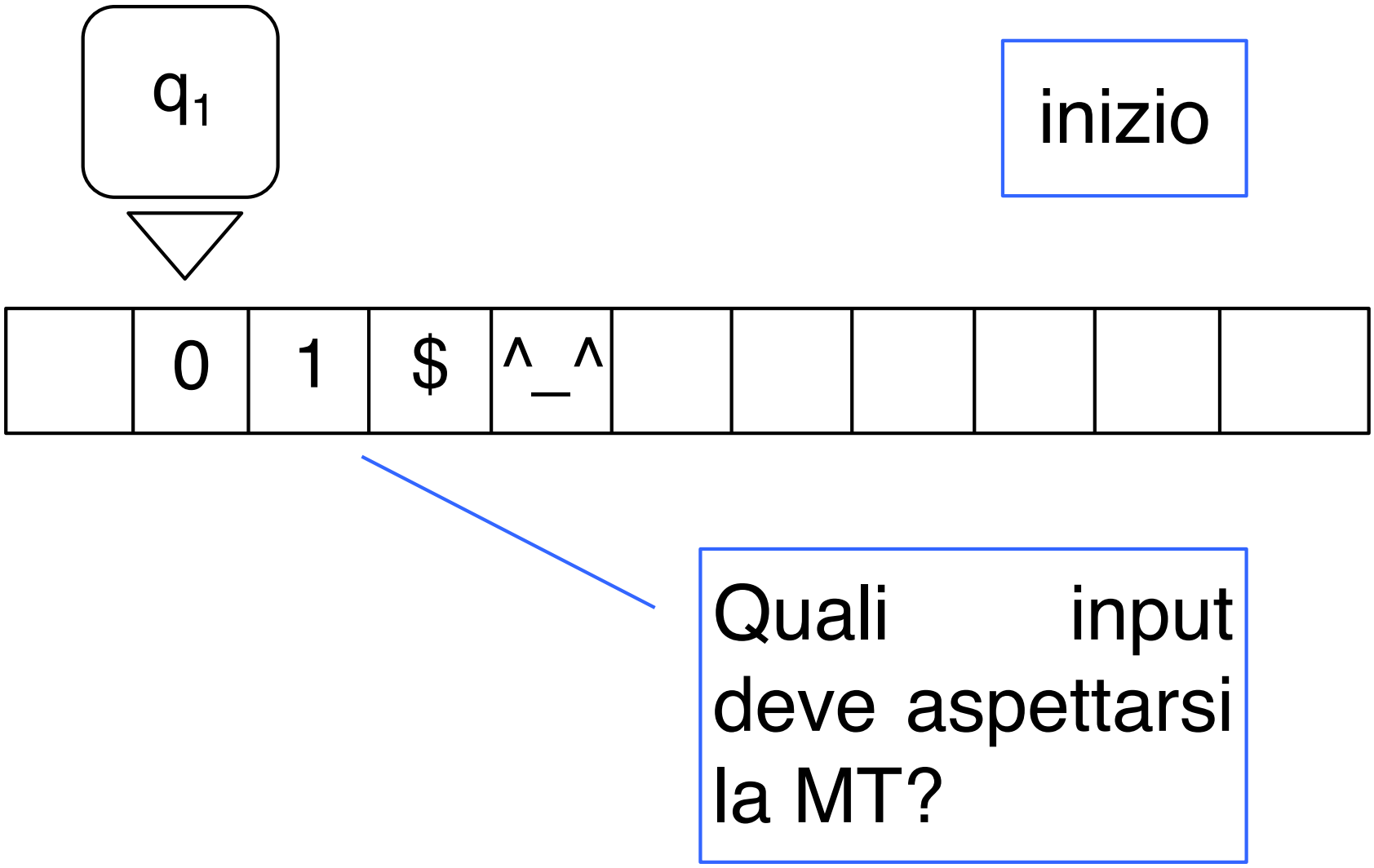
- Creare una macchina di Turing (MT) che accetti il seguente linguaggio:

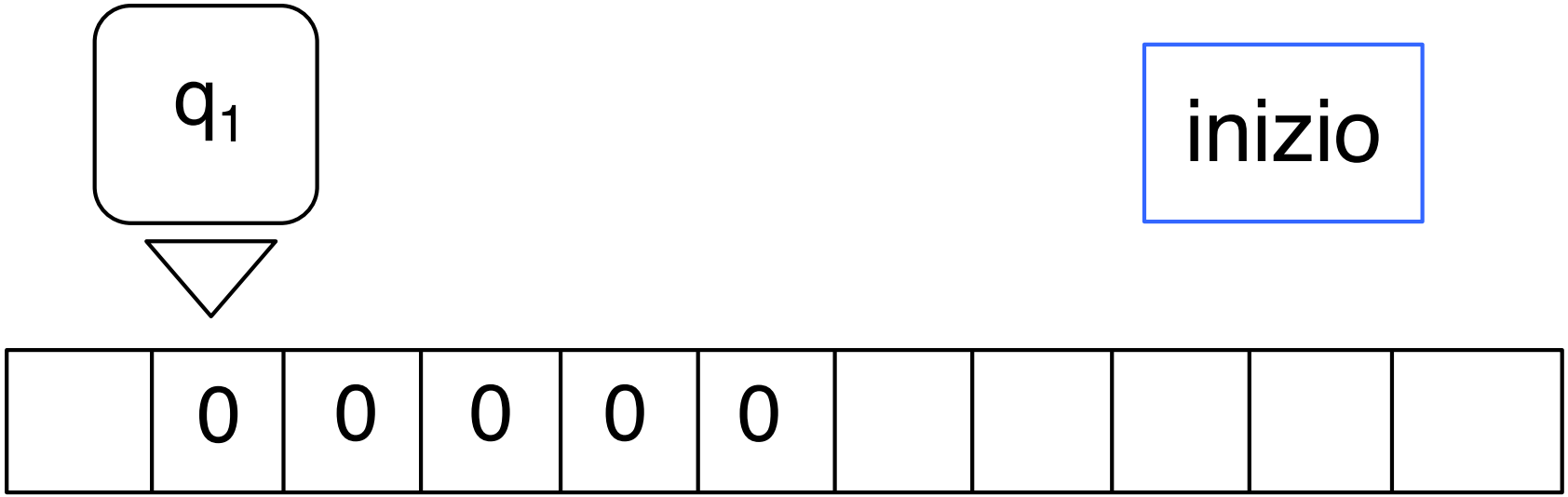
  0^i, con i intero pari


q_1
inizio
0 0 0 0
q_0
fine
0

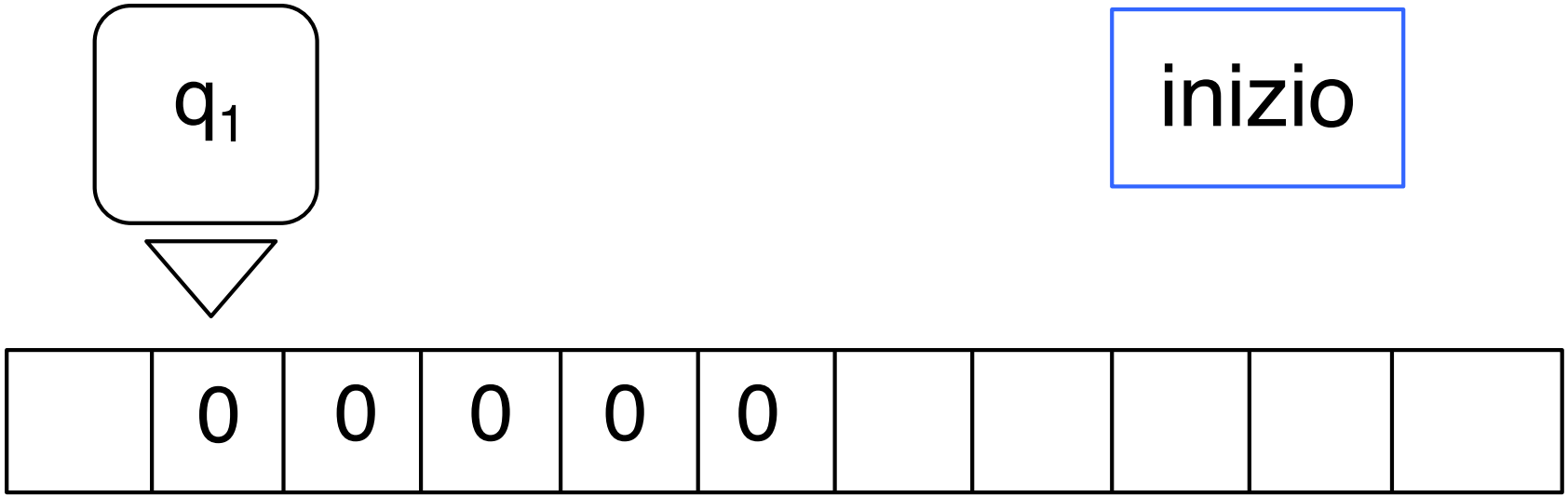

q_1
inizio
0 0 0 0
Posizione iniziale standard: la testina della MT si trova sul primo simbolo dell'input a sinistra


Non esiste una posizione finale standard. La testina può trovarsi ovunque. L'importante è l'output lasciato sul nastro.
q_0
fine
0

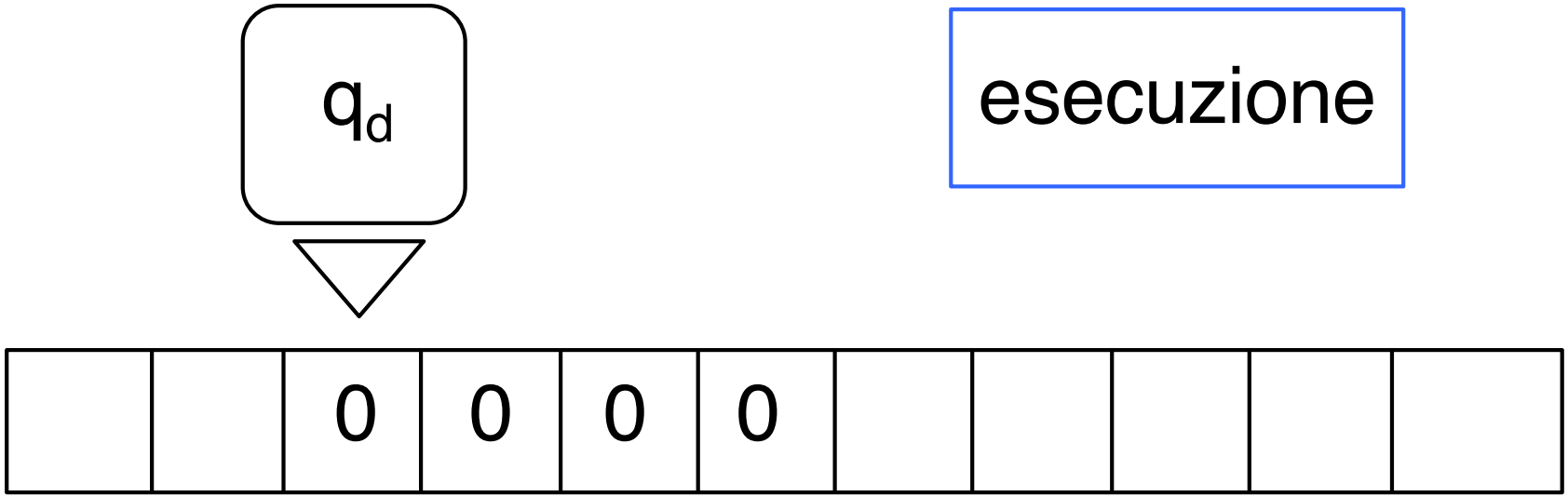

q_1
inizio
0
1
$
^_^
Quali input deve aspettarsi la MT?



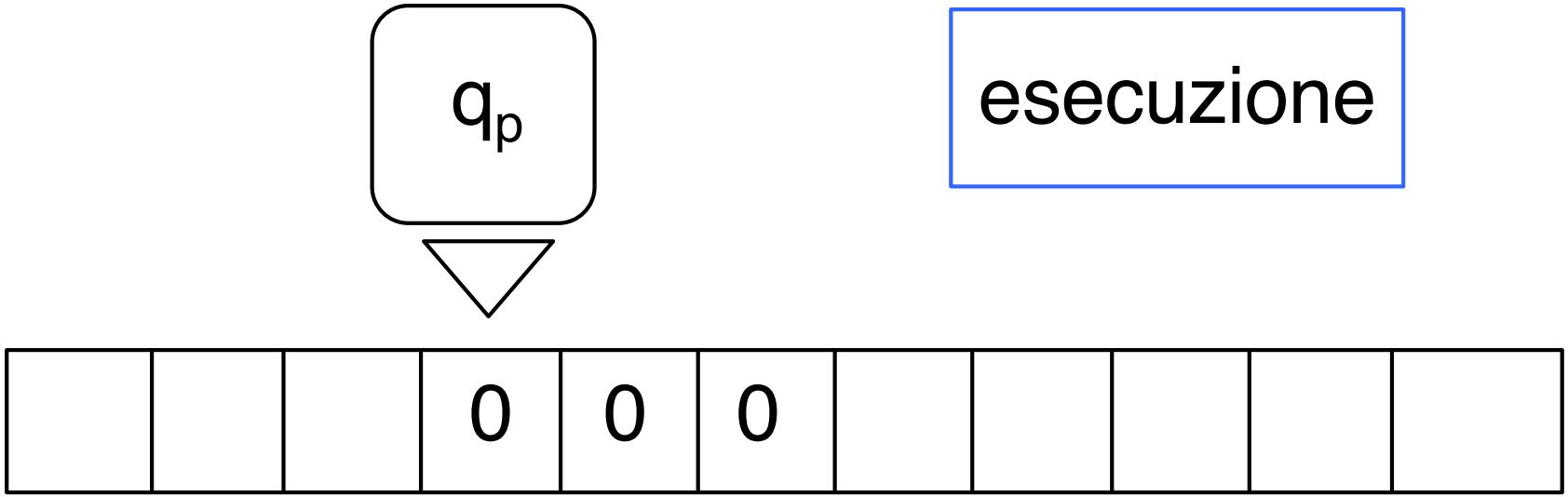
Si tratta di un esercizio teorico, non di testing di software nel mondo reale, quindi va bene ipotizzare che ci siano solo i simboli dell'alfabeto specificato nell'esercizio (e naturalmente celle vuote: [ ])



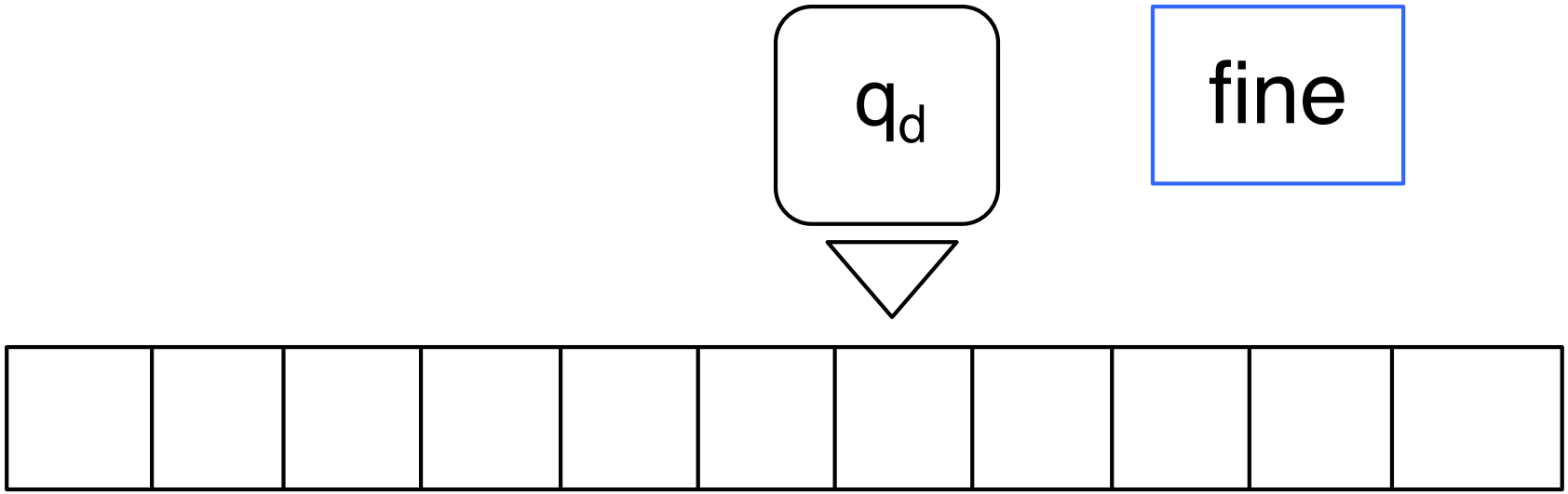
- la MT deve capire se gli 0 sul nastro sono pari o dispari
- visto che l'input deve essere cancellato, ogni volta che la MT incontra uno 0, lo cancella e va a destra



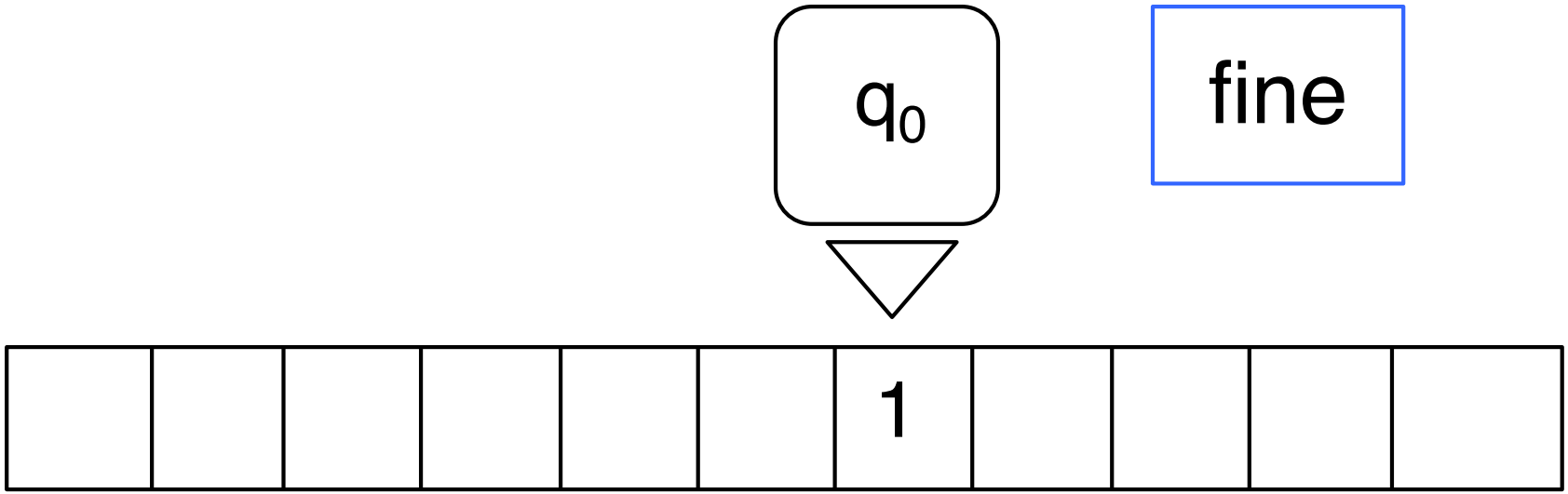
- come fa la MT a distinguere pari e dispari?
- all'inizio, cancellato il primo 0, la MT ha contato un numero dispari (uno) di 0, quindi entra in uno stato che esprime il fatto che finora la MT ha contato un numero dispari di 0
- allo 0 successivo, la MT avrà contato un numero pari di 0, quindi entrerà in uno stato che esprime la parità degli 0
- man mano che avanza, a ogni 0 che cancella, la MT oscillerà tra questi due stati, quello pari e quello dispari



- la MT oscillerà tra lo stato pari e lo stato dispari a ogni 0 che incontra (e che cancella)
- il risultato finale dipenderà da in quale stato la MT si troverà dopo aver cancellato l'ultimo 0



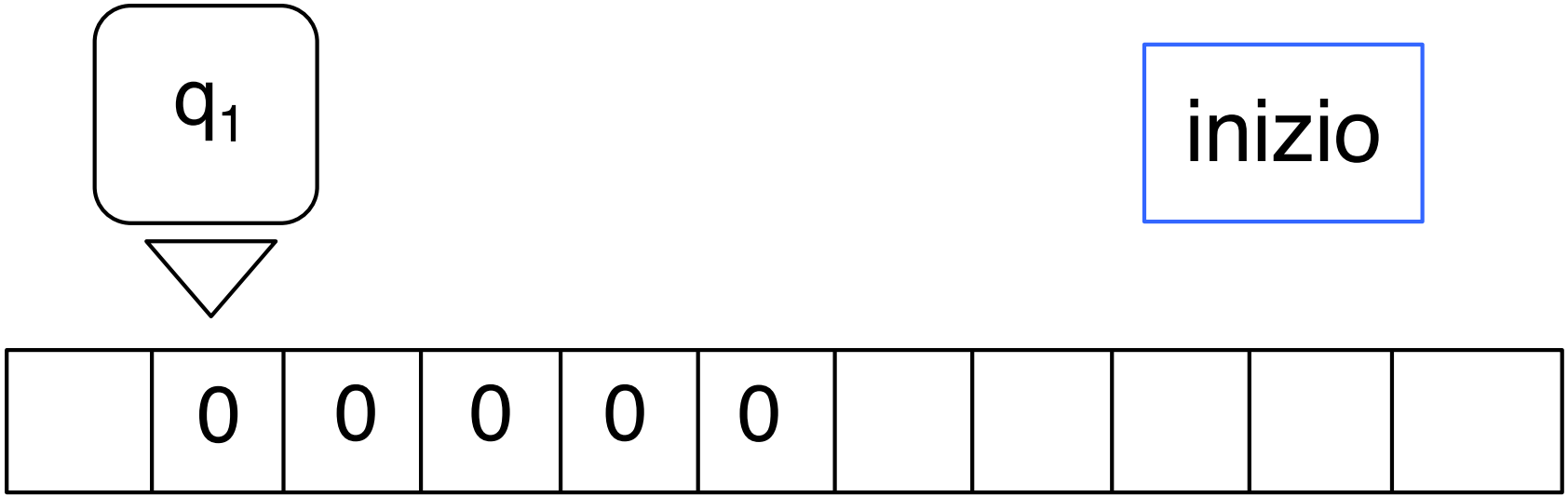
- come fa la MT ad accorgersi di aver contato tutti gli 0?
- quando la MT, per la prima volta dall'inizio dell'esecuzione, si ritroverà una cella vuota anziché uno 0 sotto la testina
- a quel punto il conteggio (e cancellazione) degli 0 è finito e lo stato della MT indica se ce n'è stato un numero pari oppure dispari



- a questo punto, se lo stato della MT è quello dispari, la MT scriverà un 1 (“no”) e entrerà nello stato finale
- se lo stato è quello pari, la MT scriverà uno 0 (“sì”) e entrerà nello stato finale

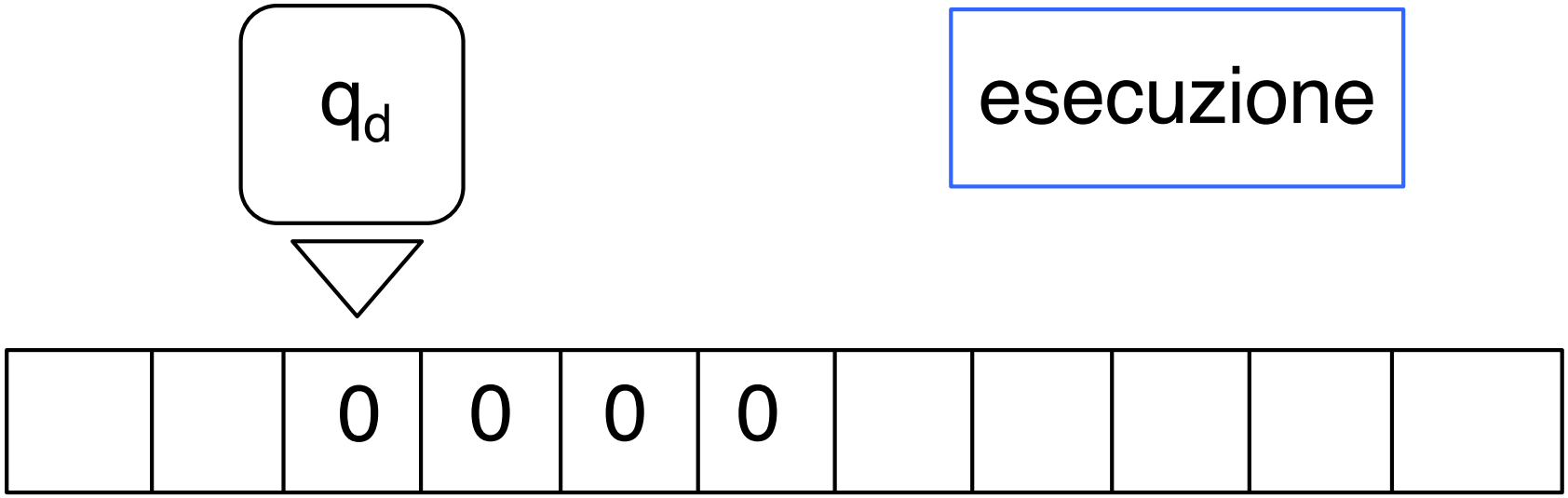
Scrittura del programma

- Ora esprimiamo queste diverse configurazioni e azioni della MT in termini di istruzioni del suo programma
- Le istruzioni hanno la struttura:
  - stato attuale
  - simbolo letto
  - simbolo scritto
  - movimento
  - nuovo stato



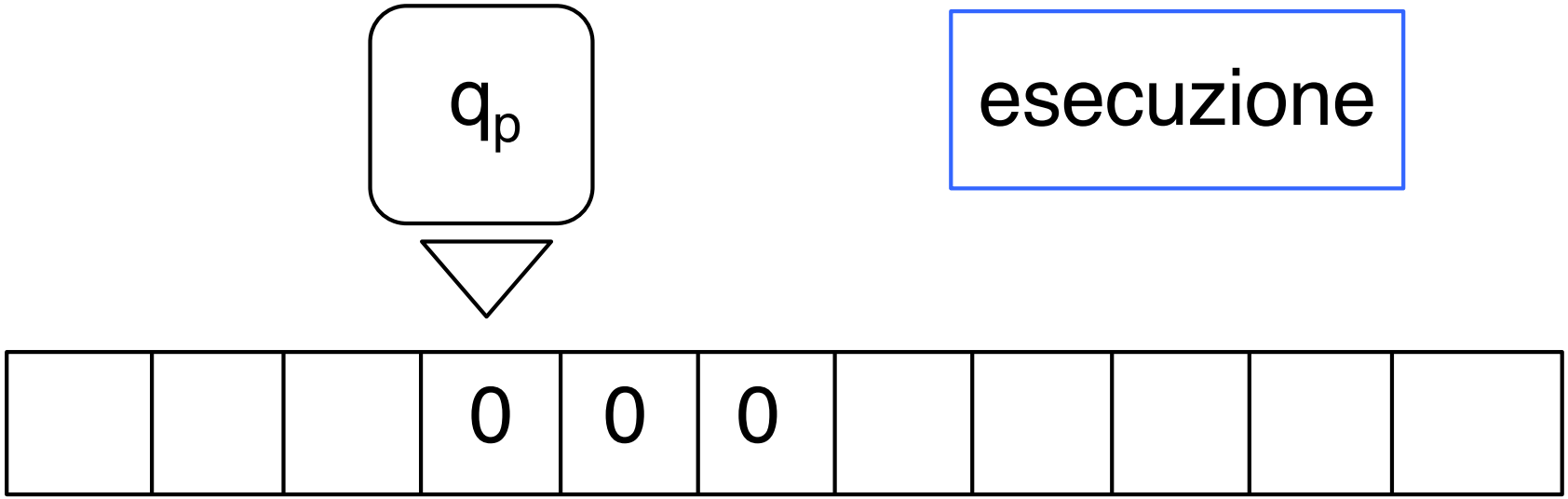
q_1 0 [ ] R q_d

- nello stato iniziale, la MT legge uno 0,, cancella lo 0, va a destra e va nello stato dispari



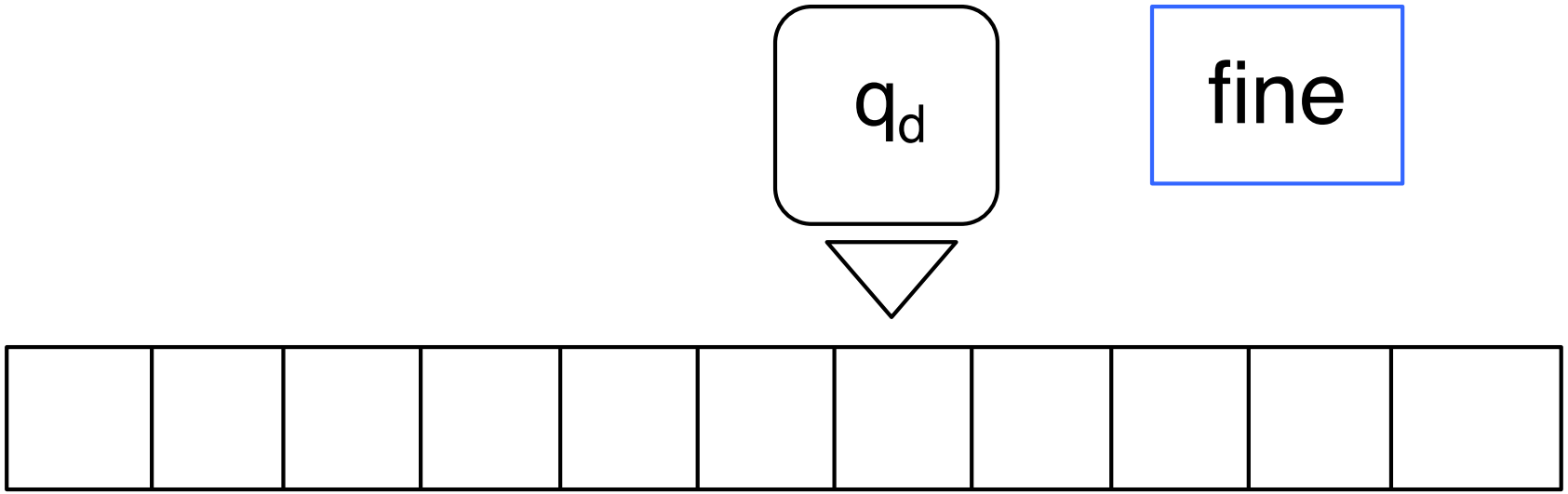
$$q_d\ 0\ [\ ]\ R\ q_p$$

- se è nello stato dispari e incontra uno 0, cancella lo 0, va a destra e la MT passa allo stato pari



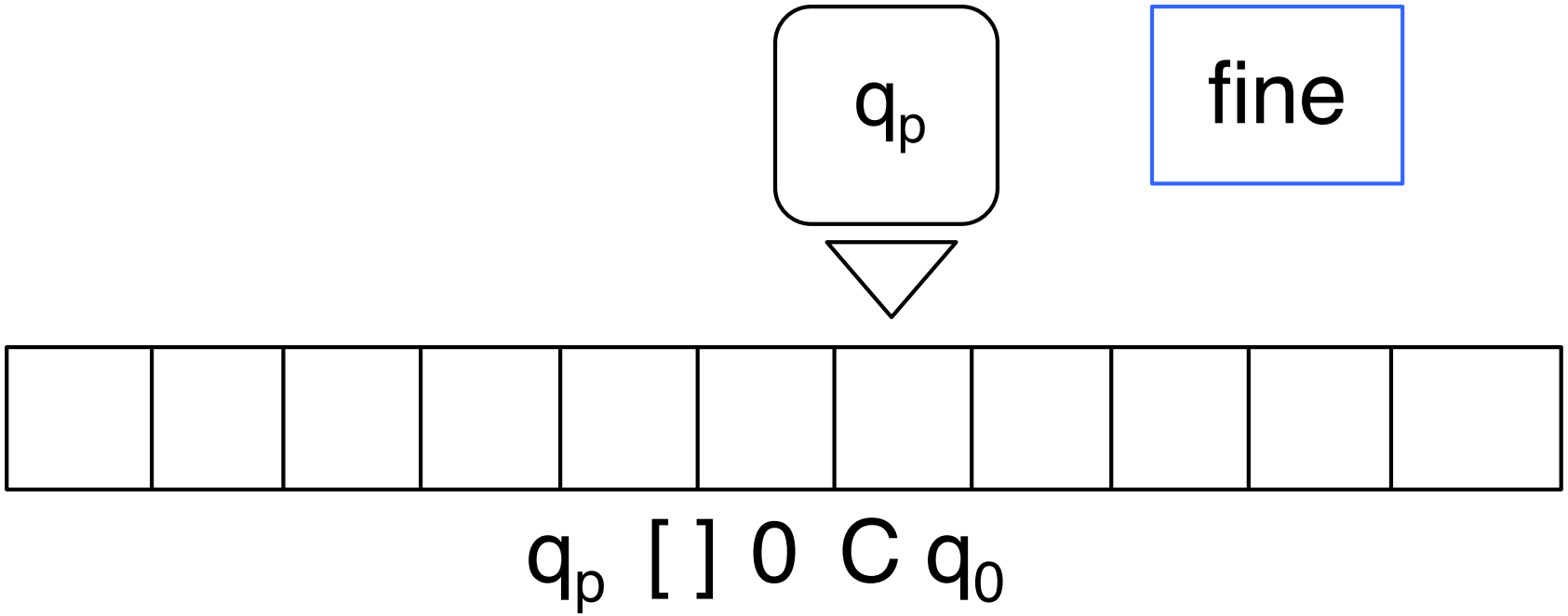
q_p 0 [ ] R q_d

- se è nello stato pari e incontra uno 0, cancella lo 0, va a destra e passa allo stato dispari
- e così via, l'oscillazione continua finché ci sono 0 sul nastro



q_d [ ] 1 C q_0

- se gli 0 finiscono e la MT è nello stato dispari, scrive un 1 e entra nello stato finale, ossia termina
- a questo punto il movimento non è importante, ma scegliamo di non farle cambiare posizione (C)



- se gli 0 finiscono e la MT è nello stato pari, scrive uno 0 per accettare l'input come appartenente al linguaggio, e termina

Il programma della MT

q_0 0 [ ] R q_d

q_d 0 [ ] R q_p

q_p 0 [ ] R q_d

q_d [ ] 1 C q_1

q_p [ ] 0 C q_1

Condizioni al contorno

- È sempre utile farsi domande sulle condizioni al contorno, ovvero sia i casi estremi
- Come si deve comportare la MT se fin da subito non trova alcuno 0?
- Se considerate “zero” un numero pari, allora se non ci sono zeri, cioè c’è la stringa vuota sul nastro, la MT dovrà scrivere uno 0 di accettazione

Aggiunta di istruzione

q_0 0 [ ] R q_d

q_d 0 [ ] R q_p

q_p 0 [ ] R q_d

q_d [ ] 1 C q_1

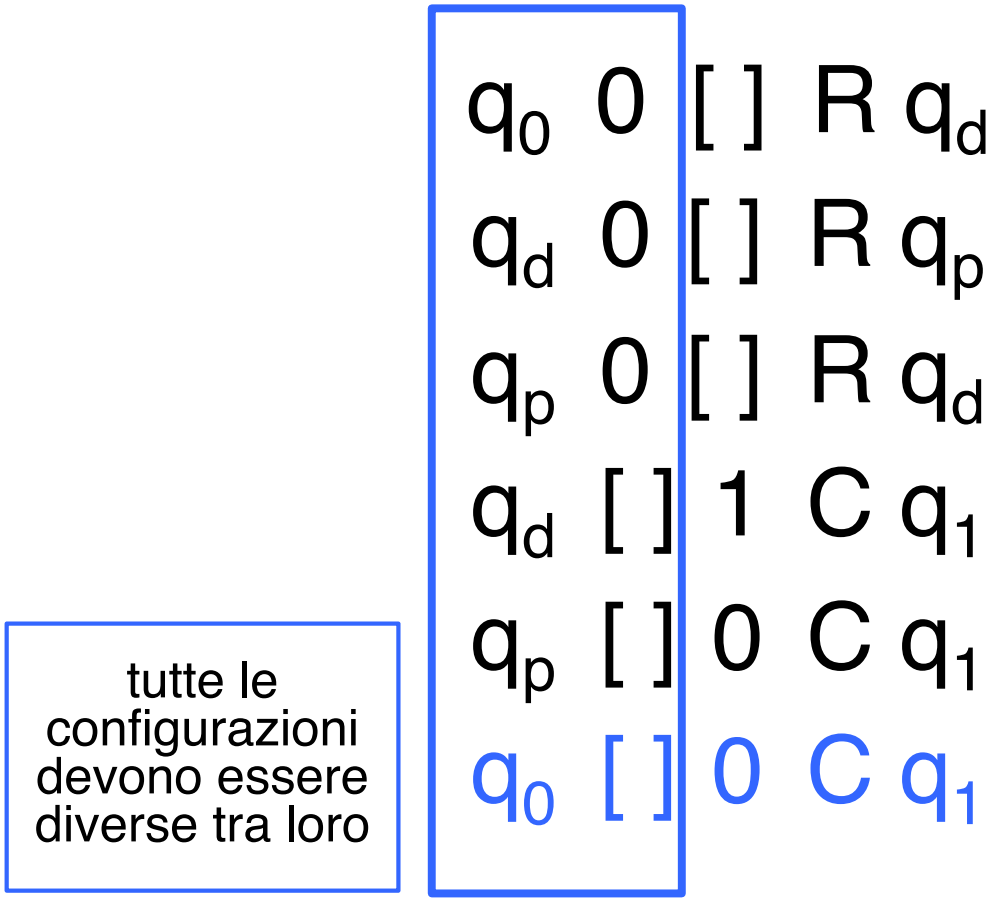
q_p [ ] 0 C q_1

q_0 [ ] 0 C q_1

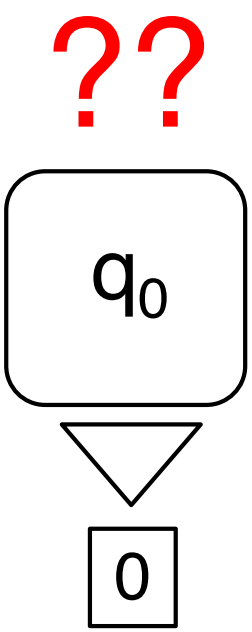
Caratteristiche di un programma di una MT

- La nuova istruzione può essere aggiunta in qualunque posizione
- L'ordine con cui sono scritte le istruzioni non conta
- Viene comunque eseguita la sola istruzione attivata dall'attuale configurazione della MT
- L'istruzione attivata deve essere unica, altrimenti si perde il determinismo della MT

Determinismo di una MT



Determinismo di una MT



q_0 0 [ ] R q_d

q_0 0 [ ] R q_p

q_p 0 [ ] R q_d

q_d [ ] 1 C q_1

q_p [ ] 0 C q_1

q_0 [ ] 0 C q_1

Esercizio 2

- Creare una macchina di Turing (MT) che accetti il seguente linguaggio:

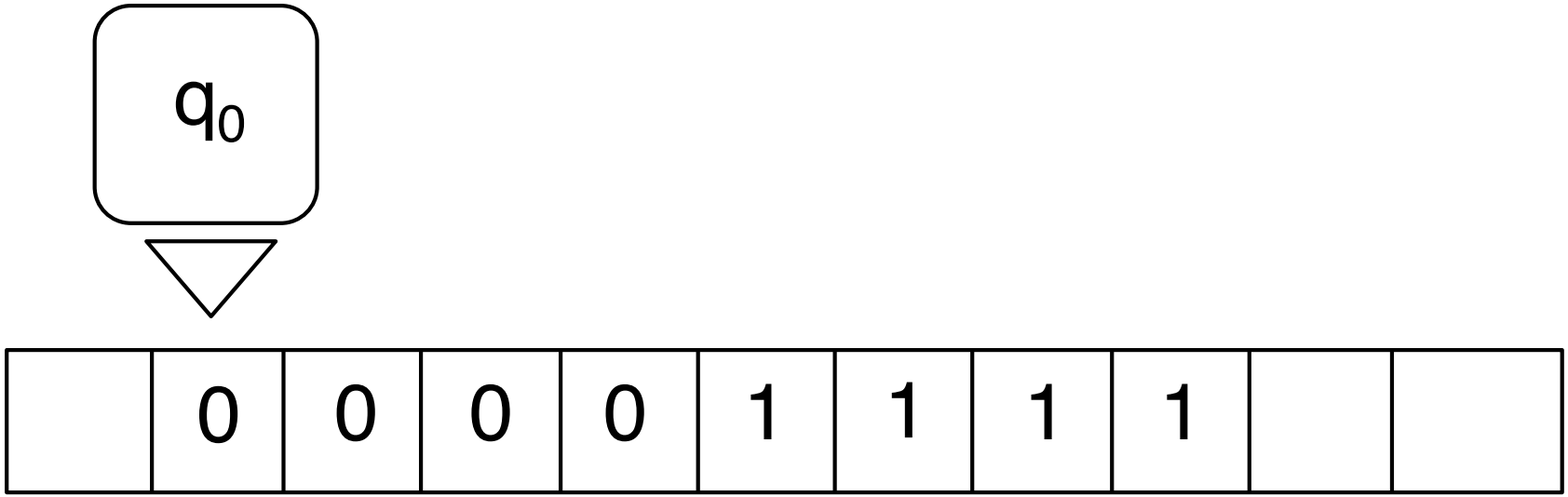
  0^i1^i, con $i \geq 0$

Il linguaggio

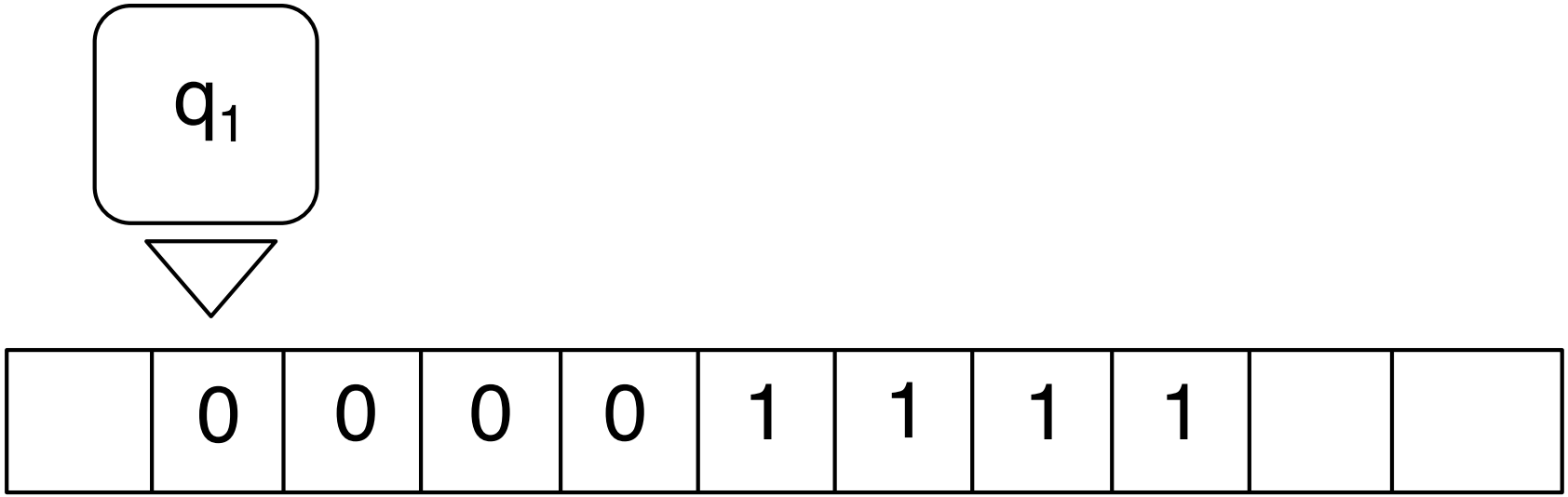
- Il linguaggio:

  0^i1^i, con $i \geq 0$

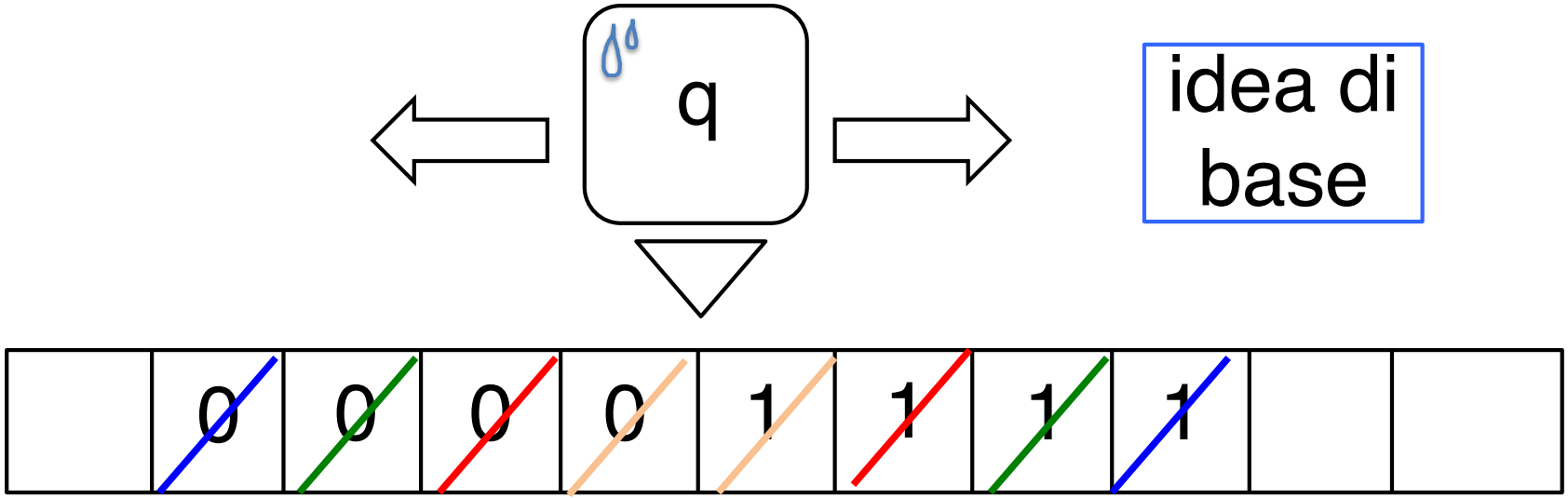
  è costituito da stringhe che hanno 0 a sinistra e 1 a desta nella stessa quantità
- 00001111 fa parte del linguaggio
- 0001111 no
- per semplicità, ipotizziamo che in input arrivino stringhe 0^i1^j, e programmiamo la nostra MT per controllare se i=j o meno
- ciò vuol dire che non dobbiamo preoccuparci di input come 0000101111



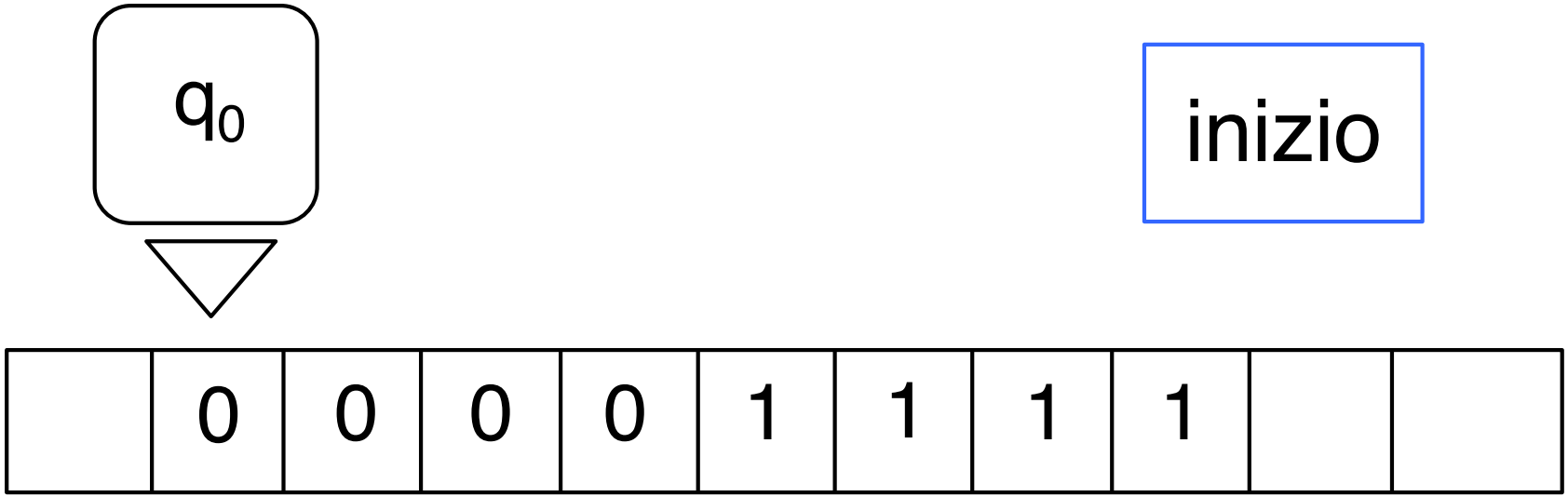
- un’idea che sorge spontanea è contare quanti 0 ci sono, contare quanti 1 ci sono e vedere se le due quantità coincidono
- purtroppo le MT NON sono in grado di contare perché non hanno una memoria interna (se non quella contenente il loro programma)



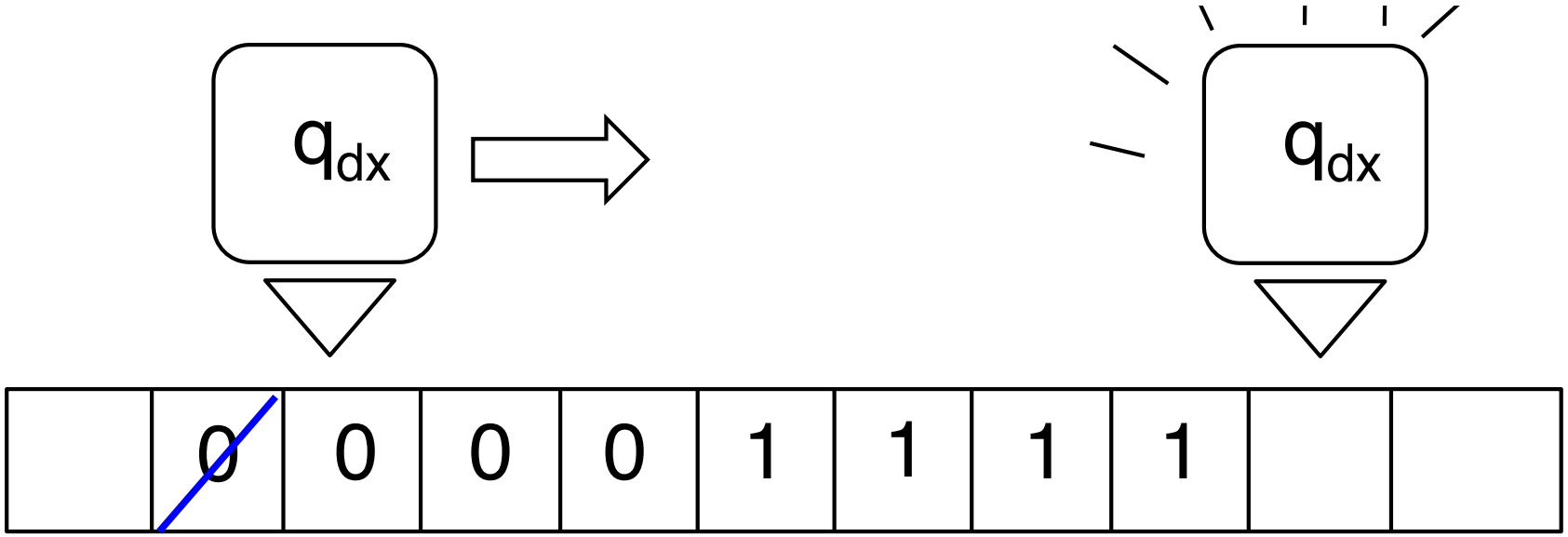
- non possiamo, come abbiamo fatto nell'esercizio precedente, usare gli stati interni della macchina per tenere traccia della situazione
- prima, qualunque fosse la lunghezza della stringa, avevamo bisogno solo di due stati (pari, dispari)
- ora, se volessimo contare gli 0, avremmo bisogno di tanti stati quanti 0 arrivano in input
- questa soluzione non è possibile perché il programma è fisso, e non può adattarsi alla dimensione dell'input in arrivo



- “sbucciamo” la stringa da sinistra (eliminando uno 0) e da destra (eliminando un 1) facendo andare la MT da un capo all’altro della stringa
- a un certo punto, se gli 0 e gli 1 sono in pari quantità, rimarremo con il nastro vuoto: la stringa appartiene al linguaggio accettato dalla MT
- se gli 0 e gli 1 non sono in pari quantità, allora o avanzeranno 0 senza 1, o viceversa, e la stringa non appartiene al linguaggio



- eliminiamo il primo 0
- facciamo scorrere la MT verso destra fino alla fine della stringa
- la MT si accorge della fine della stringa quando incontra una cella vuota



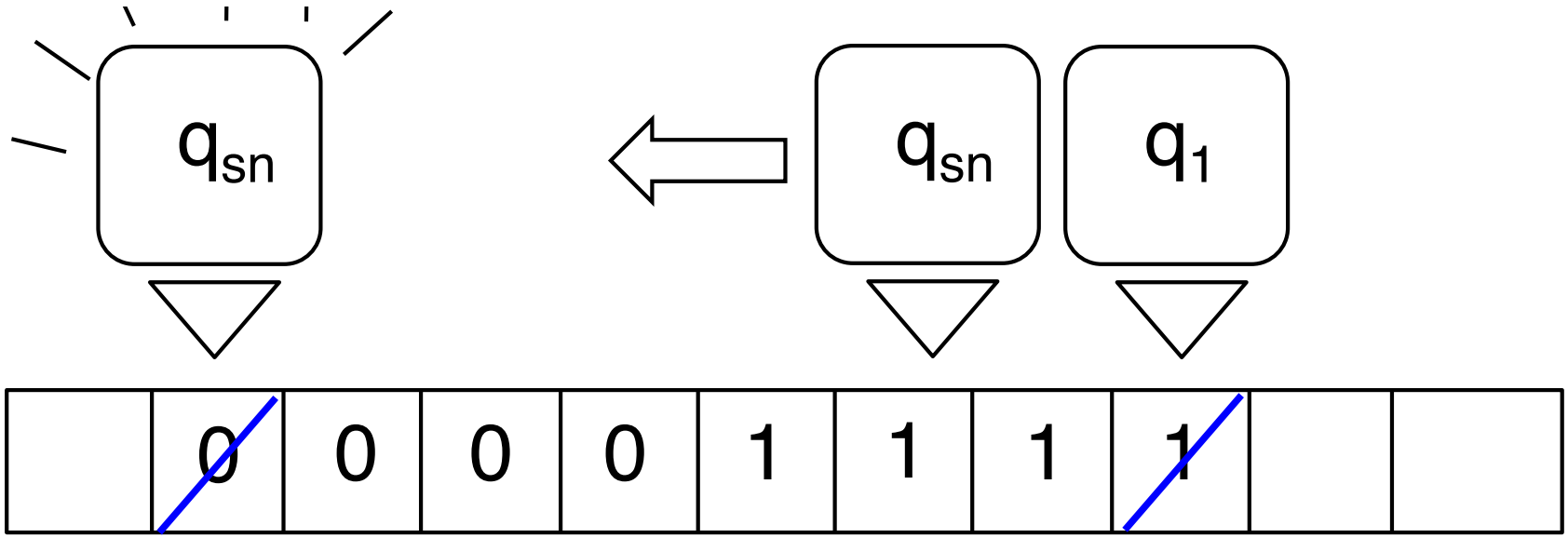
q_0 0 [ ] R q_{dx}

q_{dx} 0 0 R q_{dx}

q_{dx} 1 1 R q_{dx}

q_{dx} [ ] [ ] L q_1

q_{dx} è lo stato in cui la MT scorre a destra



q_1 1 [ ] L q_{sn}

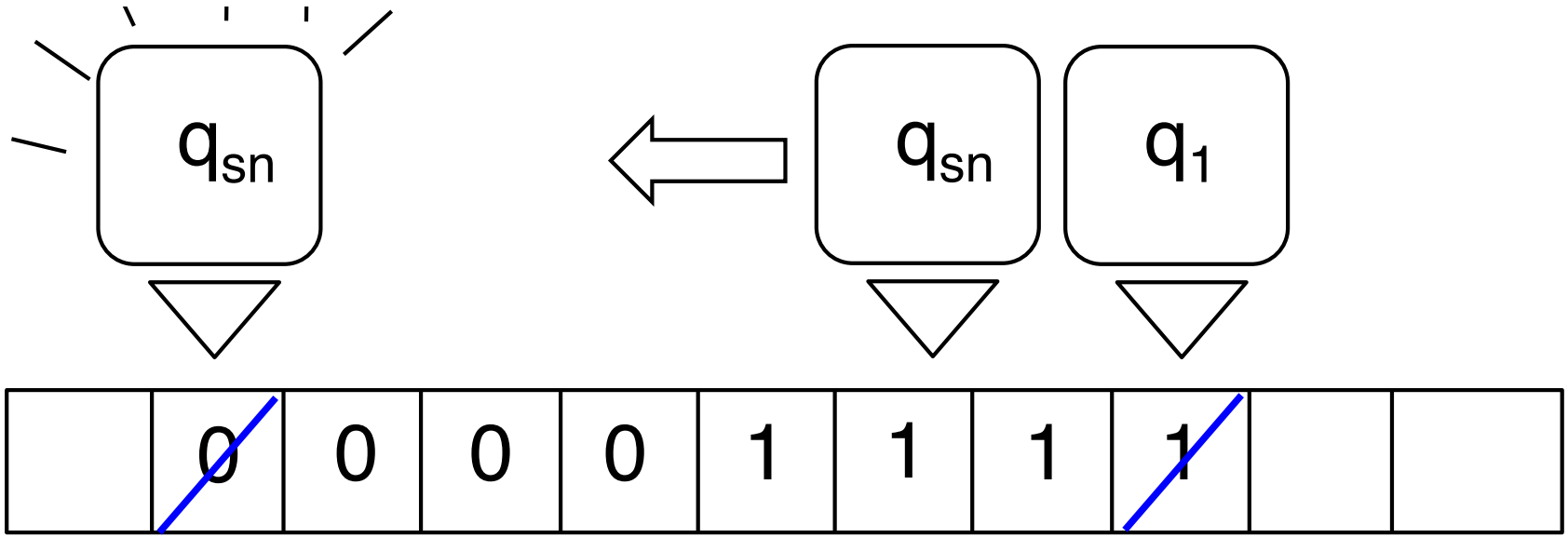
q_{sn} 1 1 L q_{sn}

q_{sn} 0 0 L q_{sn}

q_{sn} [ ] [ ] R q_0

q_1 è lo stato in cui la MT cancella un 1

q_{sn} è lo stato in cui la MT scorre a sinistra



quando siamo arrivati all'estrema sinistra…

q_1 1 [ ] L q_{sn}

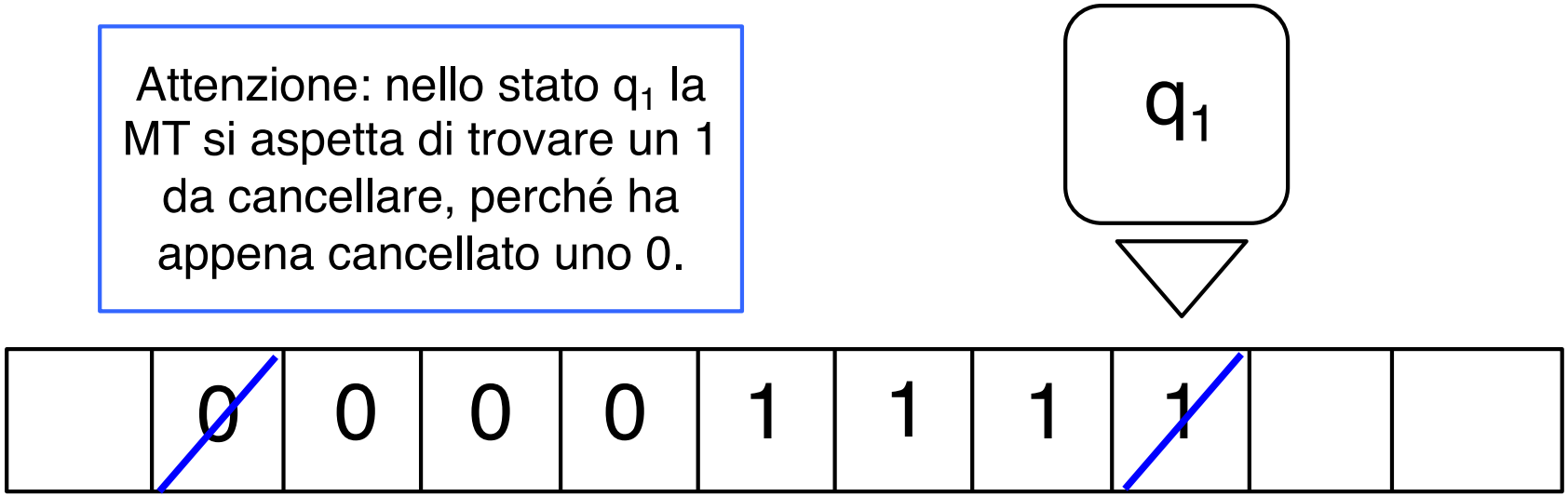
q_{sn} 1 1 L q_{sn}

q_{sn} 0 0 L q_{sn}

q_{sn} [ ] [ ] R q_0

…torniamo allo stato iniziale q_0 perché torniamo a voler cancellare uno 0

Attenzione: nello stato q_1 la MT si aspetta di trovare un 1 da cancellare, perché ha appena cancellato uno 0.

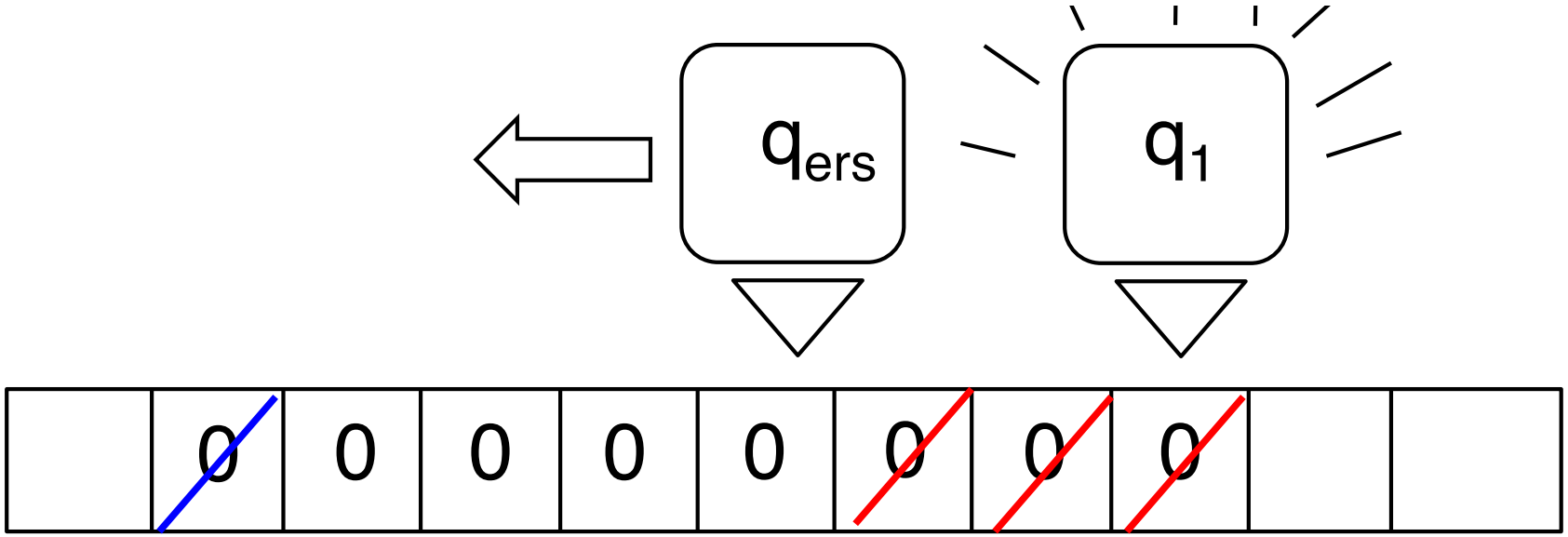


q_1 1 [ ] L q_{sn}

q_{sn} 1 1 L q_{sn}

q_{sn} 0 0 L q_{sn}

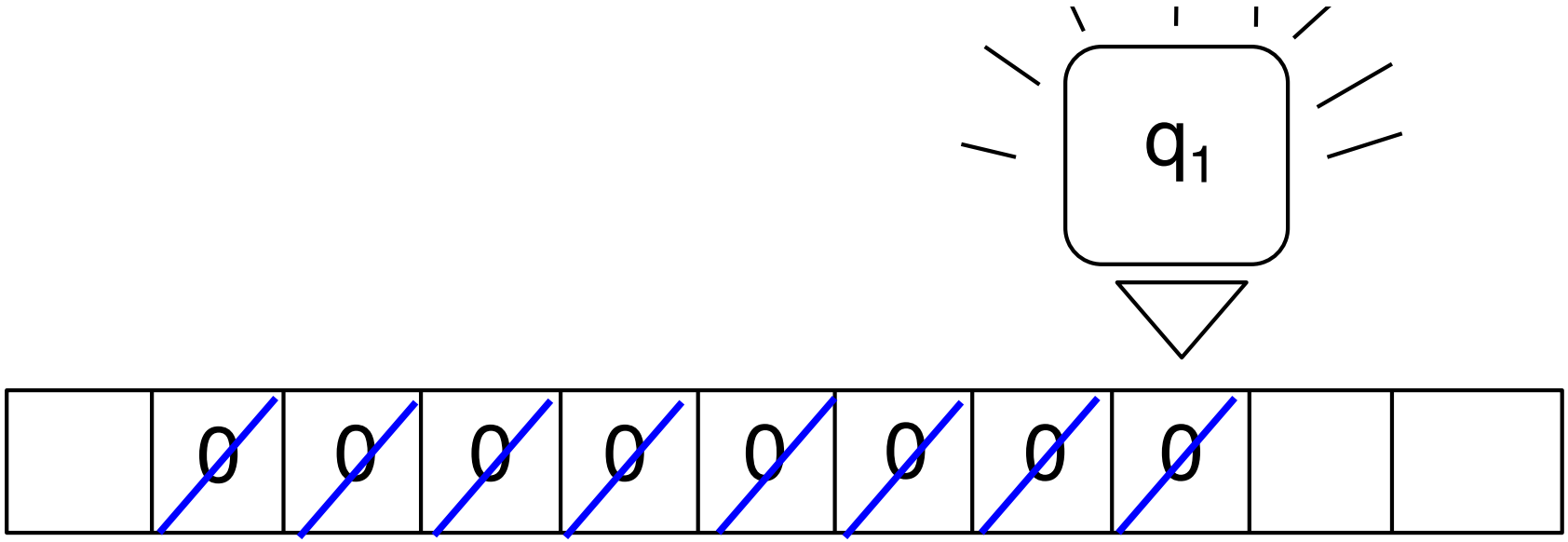
q_{sn} [ ] [ ] R q_0



…ma se ci sono più 0 che 1, la MT potrebbe aver già cancellato tutti gli 1 e trovarsi nello stato q1 con uno 0 sotto la testina.

q_1 0 [ ] L q_{ers}
q_{ers} 0 [ ] L q_{ers}
q_{ers} [ ] 1 C q_f

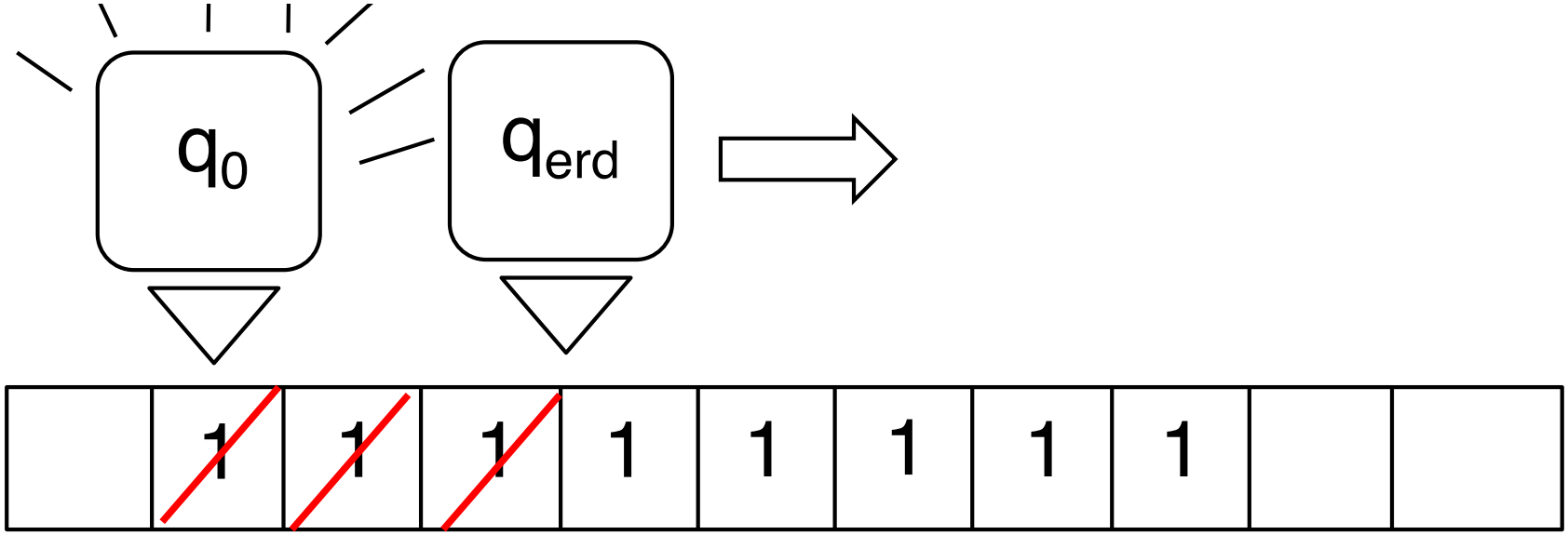
Lo stato q_{ers} è quello in cui la MT cancella tutti gli 0 rimasti, scrive 1 in output e poi termina.



Oppure, può darsi che sia stato cancellato l'ultimo 0 ma non ci fosse l'1 da cancellare, allora la MT nello stato $q_{1...}$

q_1 [ ] 1 C q_f

…si ritroverebbe con [ ] sotto la testina. La stringa era sbilanciata e quindi va rifiutata.



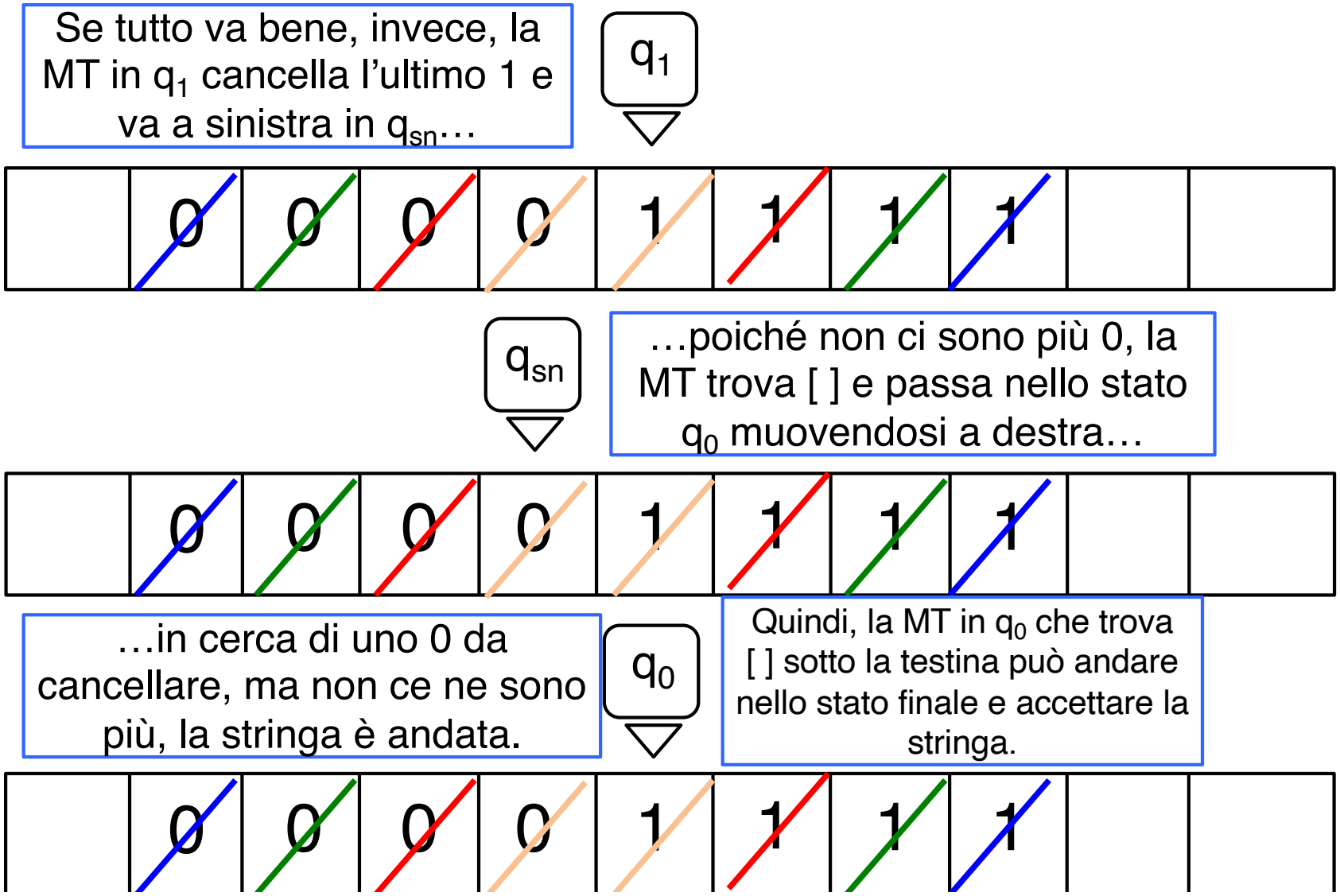
Analogamente, se ci sono più 1 che 0, quando la MT riprende a cercare 0 nello stato q_0, se si trova un 1 sotto la testina…

q_0 1 [ ] R q_{erd}

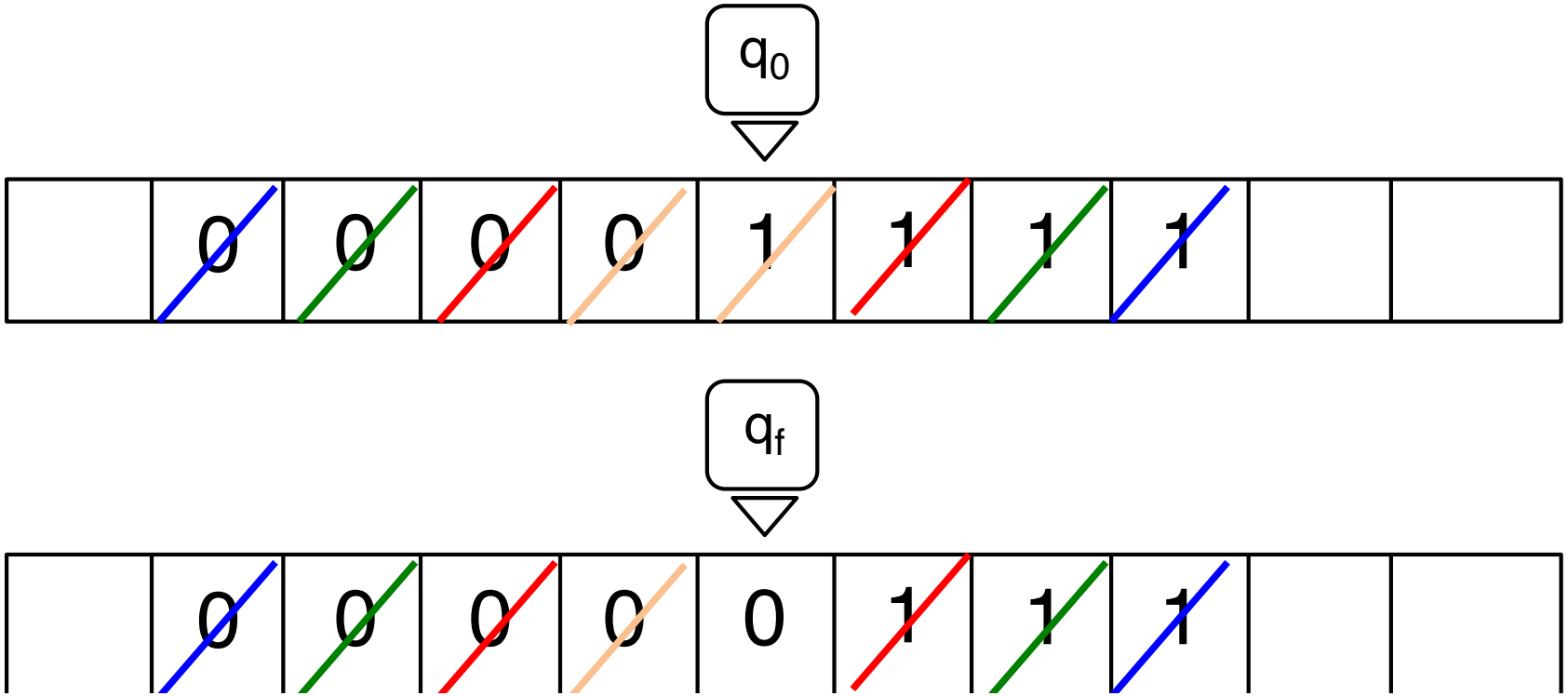
q_{erd} 1 [ ] R q_{erd}

q_{erd} [ ] 1 C q_f

…deve entrare in uno stato q_{erd} in cui la MT cancella tutti gli 1 rimasti, scrive 1 in output e poi termina.


Se tutto va bene, invece, la MT in q_1 cancella l'ultimo 1 e va a sinistra in q_{sn}…
q_1
0 0 0 0 1 1 1 1
q_{sn}
…poiché non ci sono più 0, la MT trova [ ] e passa nello stato q_0 muovendosi a destra…
0 0 0 0 1 1 1 1
…in cerca di uno 0 da cancellare, ma non ce ne sono più, la stringa è andata.
q_0
Quindi, la MT in q_0 che trova [ ] sotto la testina può andare nello stato finale e accettare la stringa.
0 0 0 0 1 1 1 1

q_0 [ ] 0 C q_f



Aggreghiamo le istruzioni

q_0 0 [ ] R q_{dx}
q_{dx} 0 0 R q_{dx}
q_{dx} 1 1 R q_{dx}
q_{dx} [ ] [ ] L q_1
q_1 1 [ ] L q_{sn}
q_{sn} 1 1 L q_{sn}
q_{sn} 0 0 L q_{sn}
q_{sn} [ ] [ ] R q_0
q_1 0 [ ] L q_{ers}
q_{ers} 0 [ ] L q_{ers}
q_{ers} [ ] 1 C q_f

q_1 [ ] 1 C q_f
q_0 1 [ ] R q_{erd}
q_{erd} 1 [ ] R q_{erd}
q_{erd} [ ] 1 C q_f
q_0 [ ] 0 C q_f

Elimina lo 0 e scorri a destra

q_0 0 [ ] R q_{dx}
q_{dx} 0 0 R q_{dx}
q_{dx} 1 1 R q_{dx}
q_{dx} [ ] [ ] L q_1
q_1 1 [ ] L q_{sn}
q_{sn} 1 1 L q_{sn}
q_{sn} 0 0 L q_{sn}
q_{sn} [ ] [ ] R q_0
q_1 0 [ ] L q_{ers}
q_{ers} 0 [ ] L q_{ers}
q_{ers} [ ] 1 C q_f

q_1 [ ] 1 C q_f
q_0 1 [ ] R q_{erd}
q_{erd} 1 [ ] R q_{erd}
q_{erd} [ ] 1 C q_f
q_0 [ ] 0 C q_f

Arrivato all'estrema destra, cancella l'1 e vai a sinistra

q_0 0 [ ] R q_{dx}
q_{dx} 0 0 R q_{dx}
q_{dx} 1 1 R q_{dx}
q_{dx} [ ] [ ] L q_1
q_1 1 [ ] L q_{sn}
q_{sn} 1 1 L q_{sn}
q_{sn} 0 0 L q_{sn}
q_{sn} [ ] [ ] R q_0
q_1 0 [ ] L q_{ers}
q_{ers} 0 [ ] L q_{ers}
q_{ers} [ ] 1 C q_f

q_1 [ ] 1 C q_f
q_0 1 [ ] R q_{erd}
q_{erd} 1 [ ] R q_{erd}
q_{erd} [ ] 1 C q_f
q_0 [ ] 0 C q_f

Arrivato all'estrema sinistra, torna a destra e riparti dall'inizio

q_0 0 [ ] R q_{dx}
q_{dx} 0 0 R q_{dx}
q_{dx} 1 1 R q_{dx}
q_{dx} [ ] [ ] L q_1
q_1 1 [ ] L q_{sn}
q_{sn} 1 1 L q_{sn}
q_{sn} 0 0 L q_{sn}
q_{sn} [ ] [ ] R q_0
q_1 0 [ ] L q_{ers}
q_{ers} 0 [ ] L q_{ers}
q_{ers} [ ] 1 C q_f

q_1 [ ] 1 C q_f
q_0 1 [ ] R q_{erd}
q_{erd} 1 [ ] R q_{erd}
q_{erd} [ ] 1 C q_f
q_0 [ ] 0 C q_f

Ci sono più 0 che 1, non va bene

q_0 0 [ ] R q_{dx}
q_{dx} 0 0 R q_{dx}
q_{dx} 1 1 R q_{dx}
q_{dx} [ ] [ ] L q_1
q_1 1 [ ] L q_{sn}
q_{sn} 1 1 L q_{sn}
q_{sn} 0 0 L q_{sn}
q_{sn} [ ] [ ] R q_0
q_1 0 [ ] L q_{ers}
q_{ers} 0 [ ] L q_{ers}
q_{ers} [ ] 1 C q_f

q_1 [ ] 1 C q_f
q_0 1 [ ] R q_{erd}
q_{erd} 1 [ ] R q_{erd}
q_{erd} [ ] 1 C q_f
q_0 [ ] 0 C q_f

Ci sono più 1 che 0, non va bene

q_0 0 [ ] R q_{dx}
q_{dx} 0 0 R q_{dx}
q_{dx} 1 1 R q_{dx}
q_{dx} [ ] [ ] L q_1
q_1 1 [ ] L q_{sn}
q_{sn} 1 1 L q_{sn}
q_{sn} 0 0 L q_{sn}
q_{sn} [ ] [ ] R q_0
q_1 0 [ ] L q_{ers}
q_{ers} 0 [ ] L q_{ers}
q_{ers} [ ] 1 C q_f

q_1 [ ] 1 C q_f
q_0 1 [ ] R q_{erd}
q_{erd} 1 [ ] R q_{erd}
q_{erd} [ ] 1 C q_f
q_0 [ ] 0 C q_f

Ci sono tanti 0 quanti 1, va bene

q_0 0 [ ] R q_{dx}
q_{dx} 0 0 R q_{dx}
q_{dx} 1 1 R q_{dx}
q_{dx} [ ] [ ] L q_1
q_1 1 [ ] L q_{sn}
q_{sn} 1 1 L q_{sn}
q_{sn} 0 0 L q_{sn}
q_{sn} [ ] [ ] R q_0
q_1 0 [ ] L q_{ers}
q_{ers} 0 [ ] L q_{ers}
q_{ers} [ ] 1 C q_f

q_1 [ ] 1 C q_f
q_0 1 [ ] R q_{erd}
q_{erd} 1 [ ] R q_{erd}
q_{erd} [ ] 1 C q_f
q_0 [ ] 0 C q_f

Esercizio 3

- Creare una macchina di Turing (MT) che accetti il seguente linguaggio:

  0^i1^i, con $i \geq 0$

- Abbandoniamo l'ipotesi che in input arrivino stringhe 0^i1^j
- Modificare la MT dell'esercizio 2 in modo che possa rifiutare anche input come 0000101111, in cui gli 0 e gli 1 sono mischiati