



# Macchine di Turing multinastro: equivalenza

---

***21 aprile 2023***

## Varianti della MdT

Esistono diverse varianti della definizione di macchina di Turing deterministica.

Vedremo: • lo **Stayer**

- la Macchina di Turing **multinastro**
- la Macchina di Turing **non deterministica**

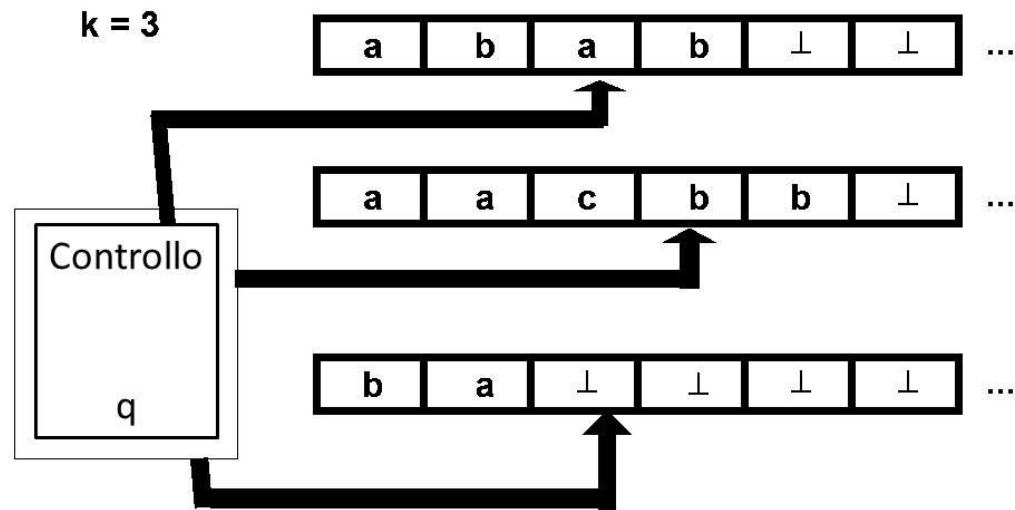
Tali macchine di Turing hanno lo **stesso potere computazionale** (o espressivo) delle MdT deterministiche, cioè riconoscono la stessa classe di linguaggi.

Sono stati proposti molti altri modelli di computazione (+ o – simili alle MdT). Sorprendentemente:

Tutti i modelli «**ragionevoli**» con un accesso **non restrittivo** ad una **memoria illimitata** risultano essere **equivalenti**.

# Macchina di Turing multinastro

Una macchina di Turing multinastro (abbreviata MdTM) è una macchina di Turing in cui si hanno più nastri contemporaneamente accessibili in scrittura e lettura che vengono aggiornati tramite più testine (una per nastro).



# Macchina di Turing multinastro

## Definizione (MdT a $k$ nastri)

Dato un numero intero positivo  $k$ , **una macchina di Turing con  $k$  nastri** è una settupla

$$(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$$

dove  $Q, \Sigma, \Gamma, q_0, q_{\text{accept}}, q_{\text{reject}}$  sono definiti come in una MdT deterministica e *la funzione di transizione*  $\delta$  è definita al modo seguente:

$$\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R, \text{Stay}\}^k$$

**Nota:**  $\Gamma^k$  è il prodotto cartesiano di  $k$  copie di  $\Gamma$  e  $\{L, R, \text{Stay}\}^k$  è il prodotto cartesiano di  $k$  copie di  $\{L, R, \text{Stay}\}$

## Definizione (MdT multinastro)

Una **macchina di Turing multinastro** è una macchina di Turing a  $k$  nastri, dove  $k$  è un numero intero positivo.

# Funzione di transizione di una MdTM

Espressione

$$\delta(q_i, a_1, \dots, a_k) = (q_j, b_1, \dots, b_k, D_1, \dots, D_k)$$

indica che se la MdT a  $k$  nastri  $M$

- ▶ si trova nello stato  $q_i$
- ▶ la testina  $t$  legge  $a_t$  per  $t = 1 \dots, k$

allora

- ▶  $M$  va nello stato  $q_j$
- ▶ la testina  $t$  scrive  $b_t$  e si muove nella direzione  $D_t \in \{L, R, Stay\}$  per  $t = 1 \dots, k$

# Computazione di una MdTM

Se  $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$  è una macchina di Turing con  $k$  nastri,

$(q_0 w, q_0, \dots, q_0)$  è una **configurazione iniziale** con input  $w$

$(u_1 q_{accept} v_1, \dots, u_k q_{accept} v_k)$  è una **configurazione di accettazione**.

Il **linguaggio  $L(M)$  riconosciuto** da  $M$  è

$$L(M) = \{w \in \Sigma^* \mid \exists u_t, v_t \in \Gamma^*, t \in \{1, \dots, k\} : \\ (q_0 w, \dots, q_0) \rightarrow^* (u_1 q_{accept} v_1, \dots, u_k q_{accept} v_k)\}$$

Il modello di MdT «**potenziato**» con la possibilità di avere più di un nastro, permette di riconoscere più linguaggi?

### Teorema

I due modelli di Mdt e MdTM hanno stesso potere computazionale.

### Dimostrazione

In un verso è ovvio: ogni MdT è una MdTM con  $k=1$  nastri.

Viceversa, dimostriamo che per ogni MdT **multinastro** **M** esiste una MdT (a **singolo** nastro) **S** equivalente ad **M**, cioè  $L(\mathbf{S}) = L(\mathbf{M})$ .

Ci riferiremo al **contenuto (significativo) del nastro**.



Supponiamo che **M** sia una MdT a **k** nastri.

Costruiamo una MdT (a singolo nastro)

$$\mathbf{S} = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$$

che simuli **M**.

**M** = Multi  
**S** = Singolo

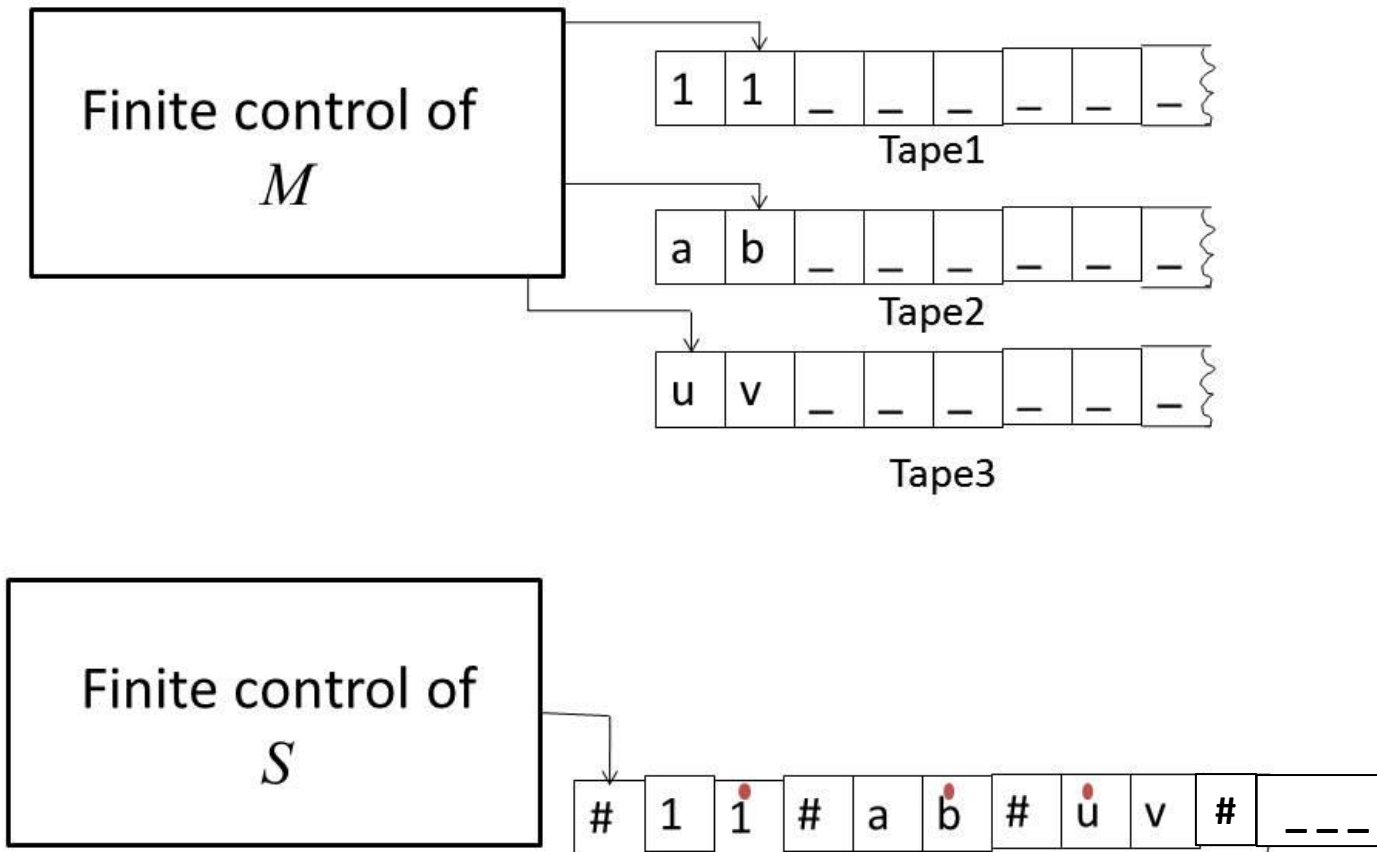
La macchina **S** deve codificare su un solo nastro

- il **contenuto** significativo di ogni nastro
- la **posizione** della testina su ogni nastro

Vediamo come fare dapprima su un **esempio**.



## Esempio: $k=3$



## Più in generale

- Il contenuto del nastro di  $S$  sarà la concatenazione di  $k$  blocchi separati da  $\#$  (seguita da  $\sqcup$ ).
- Ogni blocco corrisponderà a un nastro di  $M$  e avrà una lunghezza variabile che dipende dal contenuto del nastro corrispondente.
- Un elemento  $\dot{\gamma}$ , con  $\gamma \in \Gamma$ , nel blocco  $t$ -esimo indica la posizione della testina del nastro  $t$ -esimo,  $t \in \{1, \dots, k\}$ .

Quindi a una configurazione di  $M$

$$(u_1 q a_1 v_1, \dots, u_k q a_k v_k)$$

corrisponderà la configurazione di  $S$

$$q' \# u_1 \dot{a}_1 v_1 \# \dots \# u_k \dot{a}_k v_k \#$$

(per qualche stato  $q'$ )

**Nota:** Se  $\Gamma'$  è l'alfabeto dei simboli di nastro di  $M$ , l'alfabeto dei simboli di nastro di  $S$  sarà  $\Gamma' \cup \{\dot{\gamma} \mid \gamma \in \Gamma'\} \cup \{\#, \dot{\#}\}$ .

Sia  $w = w_1 \dots w_n$  una stringa di input per  $S$ .

$S$  comincia nella sua **configurazione iniziale** per  $w$ , cioè

$$q_0 w_1 \dots w_n$$

Poi, con una serie di passi, dalla sua configurazione iniziale passa alla configurazione corrispondente alla **configurazione iniziale di  $M$** , cioè quella con  $w$  sul primo nastro, gli altri nastri vuoti e ogni testina sulla prima cella del suo nastro (per qualche stato  $q'$ ).

$$q' \# w_1 \dots w_n \# \sqcup \# \dots \# \sqcup \#$$

## Equivalenza fra MdT e MdTM

Per ogni mossa del tipo  $\delta(q, a_1, \dots, a_k) = (s, b_1, \dots, b_k, D_1, \dots, D_k)$

- ▶ S scorre il nastro verso destra, fino al primo  $\perp$  memorizzando nello stato i simboli marcati sui singoli nastri;  
Memorizzazione: stato aggiuntivo per ogni possibile sequenza di  $\leq k$  simboli di  $\Gamma$
- ▶ la testina si riposiziona all'inizio del nastro,

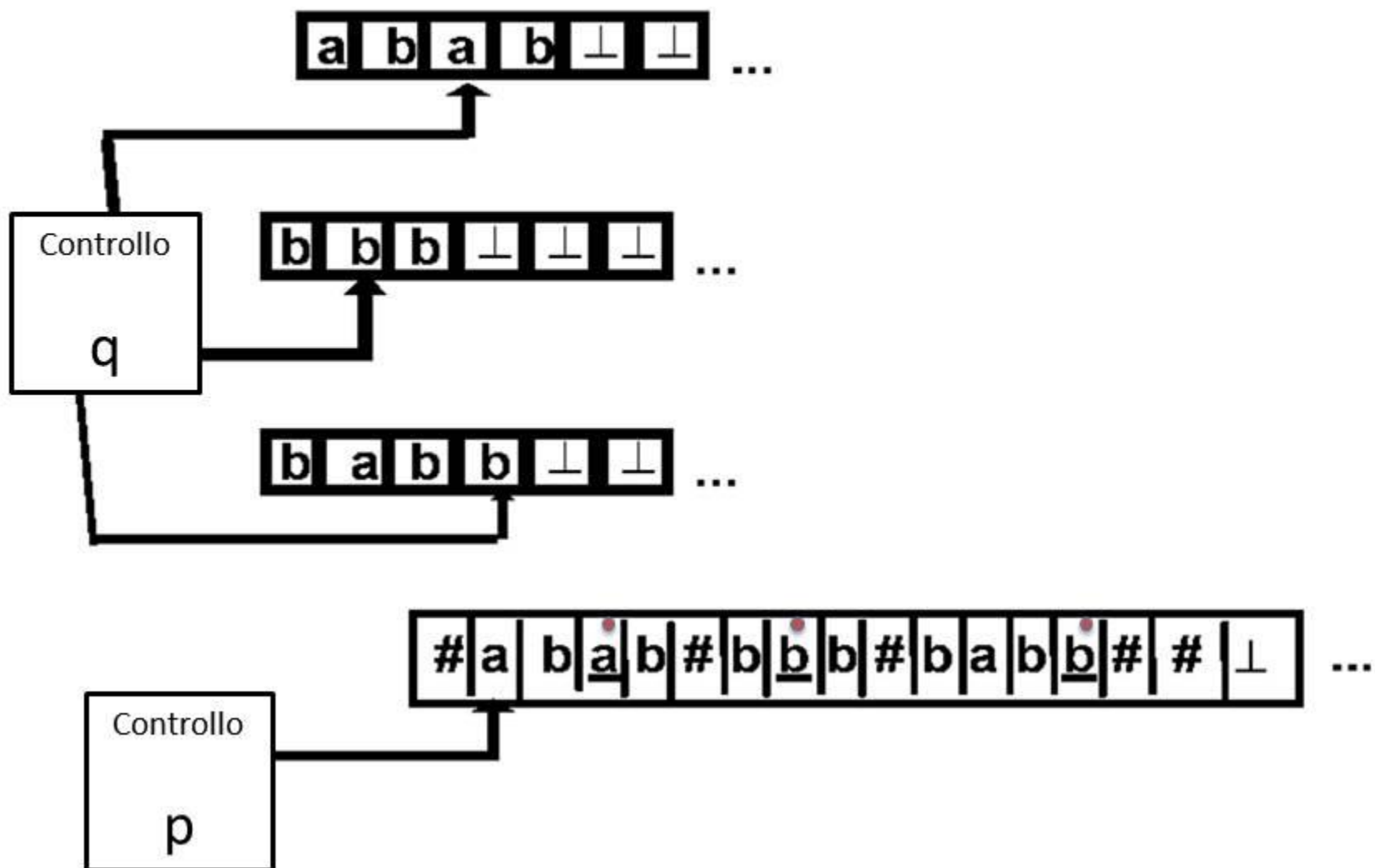
Primo  
passaggio

- ▶ poi il nastro viene scorso di nuovo eseguendo su ogni blocco (cioé un nastro di M) le azioni che simulano quelle delle testine di M: scrittura e spostamento.

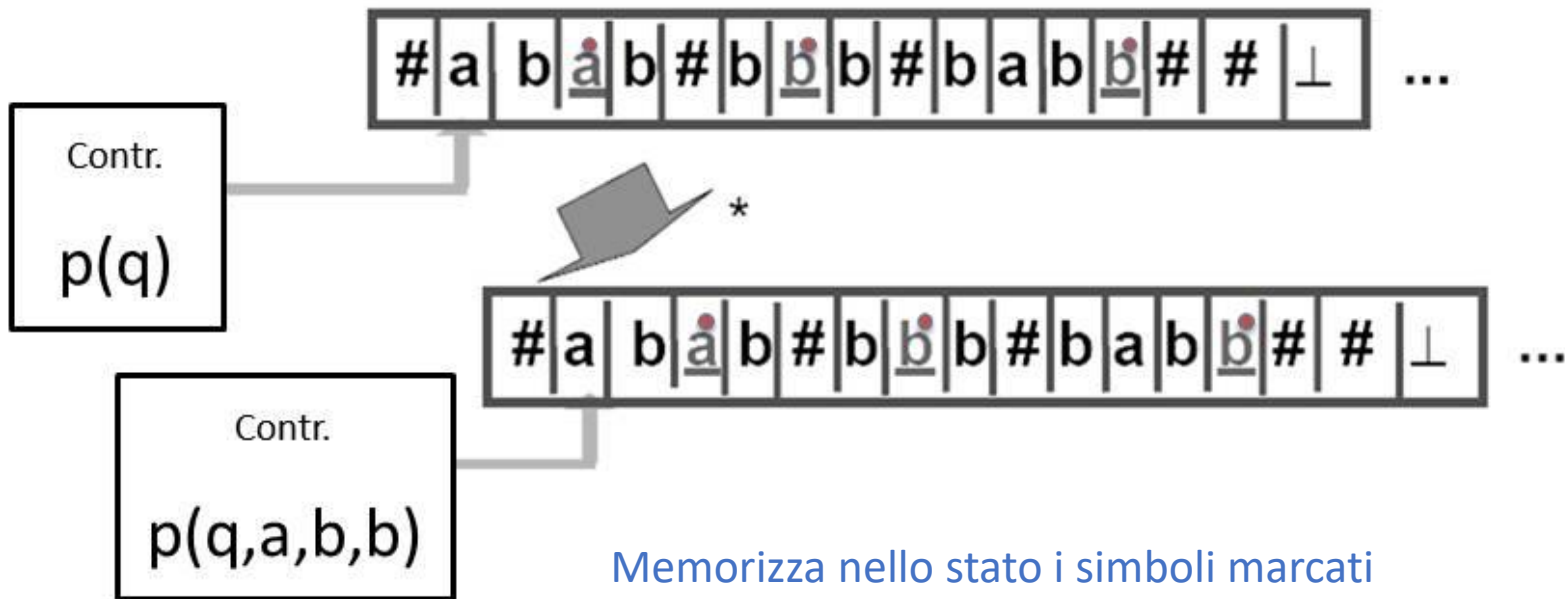
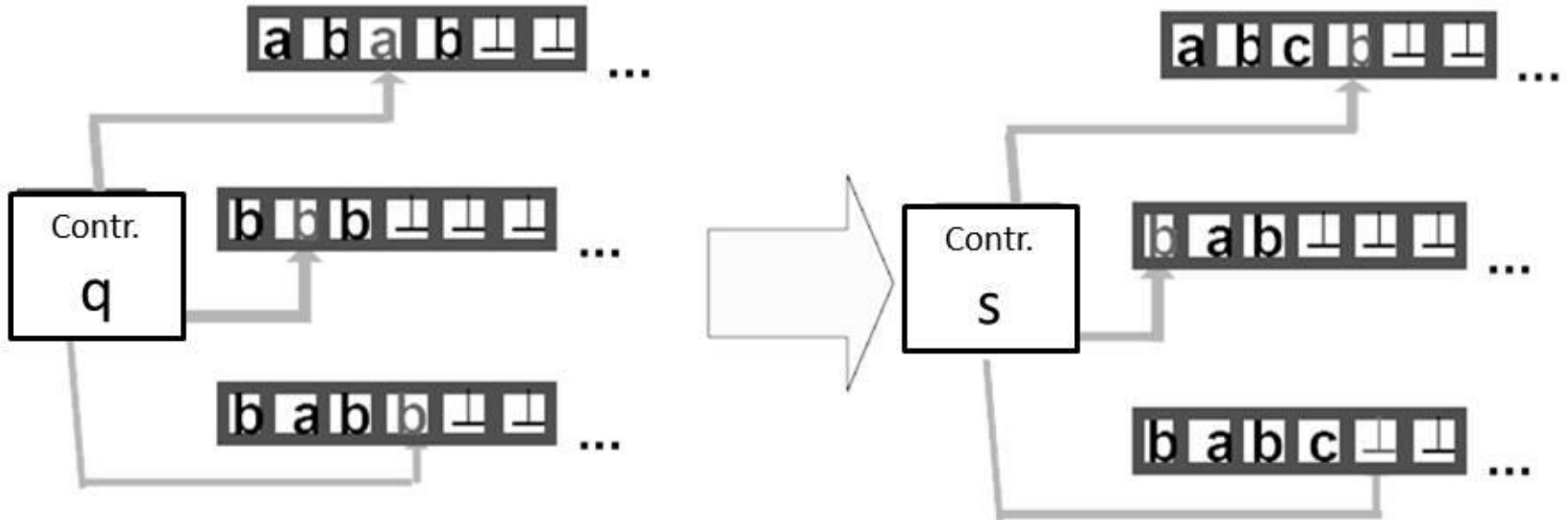
Secondo  
passaggio

- ▶ Durante il secondo passaggio S scrive su tutti i simboli "marcati" e
- ▶ sposta il marcatore alla nuova posizione della testina corrispondente di M

# Esempio



$$\delta(q, a, b, b) = (s, c, a, c, R, L, R)$$

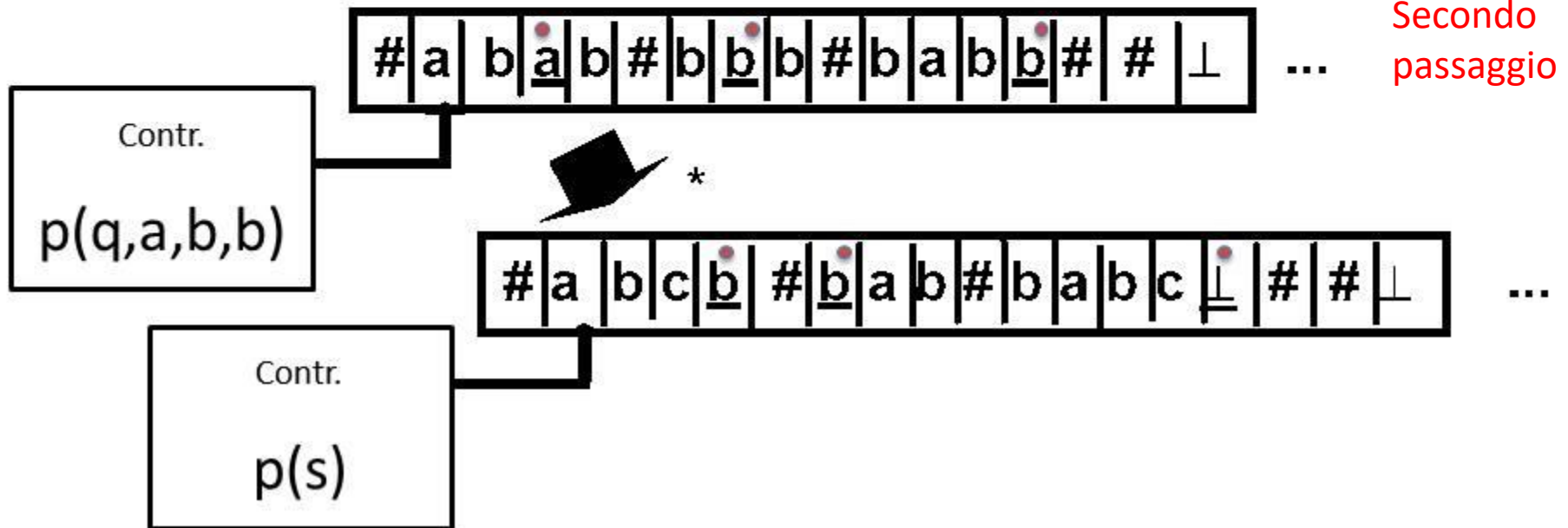
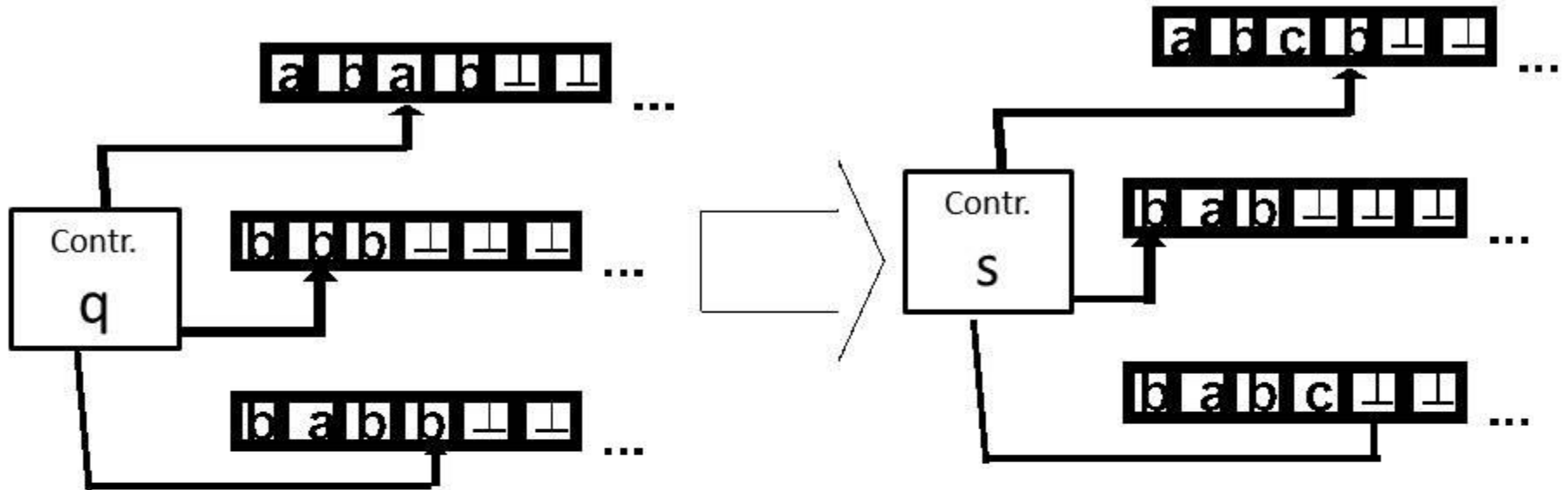


Primo  
passaggio

Memorizza nello stato i simboli marcati



$$\delta(q, a, b, b) = (s, c, a, c, R, L, R)$$



- Nota. Nel corso della simulazione  $S$  potrebbe spostare una delle testine virtuali (puntino) su un delimitatore  $\#$ . Questo significa che  $M$  ha spostato la testina del nastro (corrispondente al segmento del nastro di  $S$ ) sulla parte di nastro contenente solo  $\sqcup$  (sulla cella a destra del carattere diverso da  $\sqcup$  più a destra del suo nastro). In questo caso,  $S$  cambia  $\dot{\#}$  con  $\dot{\sqcup}$  e deve spostare di una posizione tutto il suffisso del nastro a partire da  $\dot{\#}$  (cambiato con  $\#$ ) fino all'ultimo  $\#$  a destra.
- Poi la MdT entra nello stato che ricorda il nuovo stato di  $M$  e riposiziona la testina all'inizio del nastro.
- Infine, se  $M$  si ferma, anche  $S$  si ferma

La MdT  $S$  avrà **molte più stati** di  $M$  e la computazione su uno stesso input necessiterà di **molte più passi**, ma ...  **$S$  simula  $M$** , cioè  $L(S) = L(M)$  !

## Teorema

*Per ogni MdT multinastro  $M$  esiste una MdT (a singolo nastro)  $S$  equivalente ad  $M$ , cioè  $L(S) = L(M)$ .*

## Teorema

*Un linguaggio  $L$  è Turing riconoscibile se e solo se esiste una macchina di Turing multinastro  $M$  che lo riconosce, cioè tale che  $L(M) = L$ .*

- Nota. Nel corso della simulazione  $S$  potrebbe spostare una delle testine virtuali (puntino) su un delimitatore  $\#$ . Questo significa che  $M$  ha spostato la testina del nastro (corrispondente al segmento del nastro di  $S$ ) sulla parte di nastro contenente solo  $\sqcup$  (sulla cella a destra del carattere diverso da  $\sqcup$  più a destra del suo nastro). In questo caso,  $S$  cambia  $\#$  con  $\sqcup$  e deve spostare di una posizione tutto il suffisso del nastro a partire da  $\#$  (cambiato con  $\#$ ) fino all'ultimo  $\#$  a destra.

1. Progettare una macchina di Turing che **sposta** l'input a destra di una casella.
2. Progettare una macchina di Turing che calcola il **successore** in binario.

Questi esercizi ci mostrano cosa può **fare** una MdT: **copiare**, **spostare**, calcolare **successore**, **somma**, **differenza** (in unario o binario), ... **contare**, ovvero riconoscere occorrenze di **ugual numero**, lunghezza pari ad una **potenza di 2**, .....

## Esercizio

Progettare una MdT che **calcoli** la funzione **f( x,y)**, differenza intera di due interi positivi x e y

$$\begin{aligned} f(x, y) &= x - y && \text{se } x \geq y \\ f(x, y) &= 0 && \text{altrimenti.} \end{aligned}$$

Si supponga che l'input sia  $\langle x \rangle 0 \langle y \rangle$ , dove  $\langle n \rangle = 1^n$ , è la rappresentazione unaria dell'intero positivo n.

3.8 Give implementation-level descriptions of Turing machines that decide the following languages over the alphabet  $\{0,1\}$ .

- <sup>A</sup>a.  $\{w \mid w \text{ contains an equal number of 0s and 1s}\}$
- b.  $\{w \mid w \text{ contains twice as many 0s as 1s}\}$
- c.  $\{w \mid w \text{ does not contain twice as many 0s as 1s}\}$

E con una MdT a 2 (o più) nastri, avremmo «vantaggi»?  
Provate.



## MdT per il successore in ordine radix

Descrivere una MdT che data una stringa  $w$  sull'alfabeto  $\{1, 2\}$  calcoli la successiva di  $w$  nell'ordine radix.

**Ordine radix** = ordine per lunghezza e a parità di lunghezza ordine numerico secondo  $1 < 2$ .

### *Esempio*

$w = 1221$  successore di  $w = 1222$

$w' = 222$  successore di  $w' = 1111$