

Algoritmi e Strutture Dati

– Teoria della Calcolabilità –

Marina Zanella

`marina.zanella@unibs.it`

(presentazione tratta da una precedente del Prof. Alessandro Saetti)



Università degli Studi di Brescia

L'argomento è trattato nel testo

G. AUSIELLO, F. D'AMORE, G. GAMBOSI, L. LAURA; LINGUAGGI, MODELLI, COMPLESSITÀ;
SECONDA EDIZIONE; FRANCO ANGELI EDITORE; 2014

T-Calcolabilità

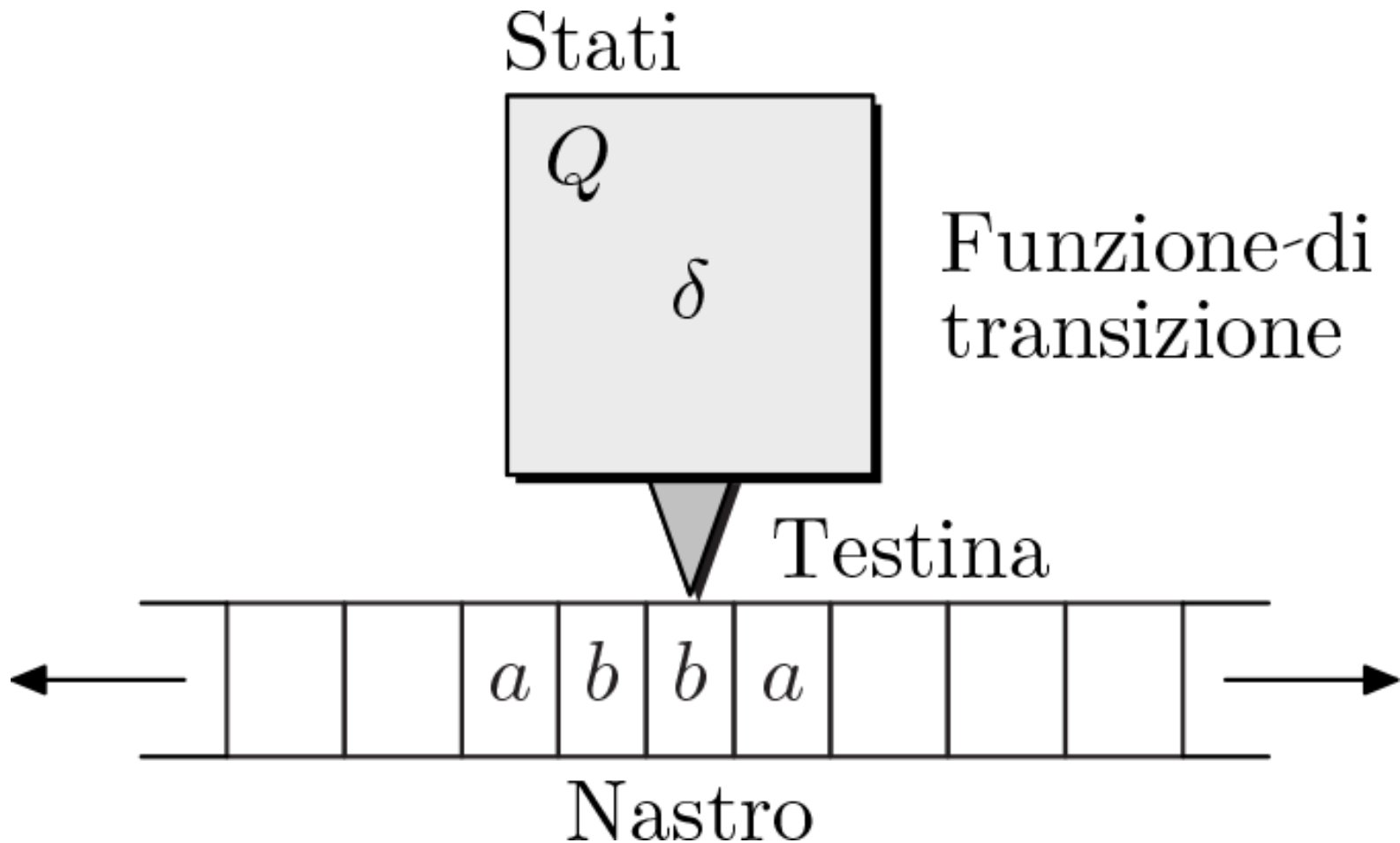
Macchina di Turing = modello astratto di riferimento per la teoria sia della calcolabilità sia della complessità; essa fornisce una definizione formale del concetto di calcolo

Tutti i risultati validi per le macchine di Turing sono applicabili a ogni altro sistema di calcolo o linguaggio di programmazione

Contributi fondamentali di Turing

- Dimostrazione dell'esistenza di una macchina (detta Macchina di Turing universale) capace di simulare qualunque altra macchina di cui sia fornita una descrizione (risultato alla base dei primi calcolatori programmabili negli anni '40 del secolo scorso)
- Creazione di uno strumento formale che associa semplicità di struttura e funzionamento con il potere computazionale massimo di un dispositivo di calcolo automatico
- Scoperta dell'esistenza di problemi *indecidibili*, cioè problemi che non possono essere risolti da macchine di Turing (e quindi non possono essere risolti da alcun algoritmo)

Macchina di Turing deterministica (MTD)



Nastro illimitato, bidirezionale, diviso in celle

MTD a nastro singolo

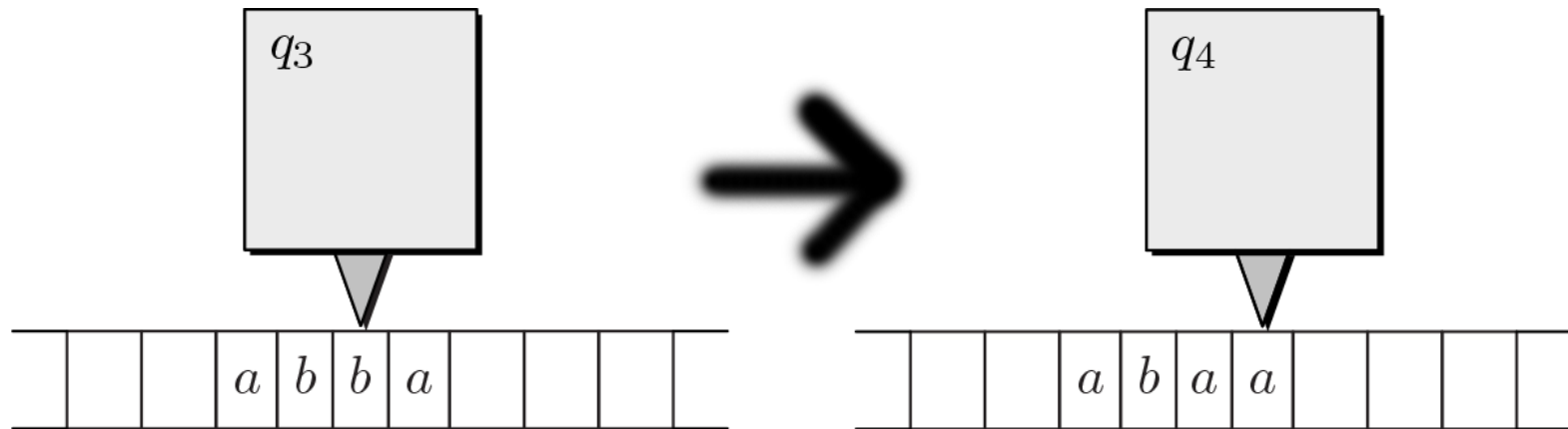
$\mathcal{M} = \langle \Gamma, \mathfrak{b}, Q, q_0, F, \delta \rangle$ dove

- Γ è l'alfabeto dei simboli di nastro
- $\mathfrak{b} \notin \Gamma$ è il carattere blank
- Q è un insieme finito e non vuoto di stati
- $q_0 \in Q$ è lo stato iniziale
- $F \subseteq Q$ è l'insieme degli stati finali
- $\delta : (Q - F) \times (\Gamma \cup \{\mathfrak{b}\}) \rightarrow Q \times (\Gamma \cup \{\mathfrak{b}\}) \times \{d, s, i\}$ è la funzione (parziale) di transizione

d , s e i indicano rispettivamente lo spostamento (di una cella) verso destra, sinistra e l'immobilità della testina

Passo di computazione

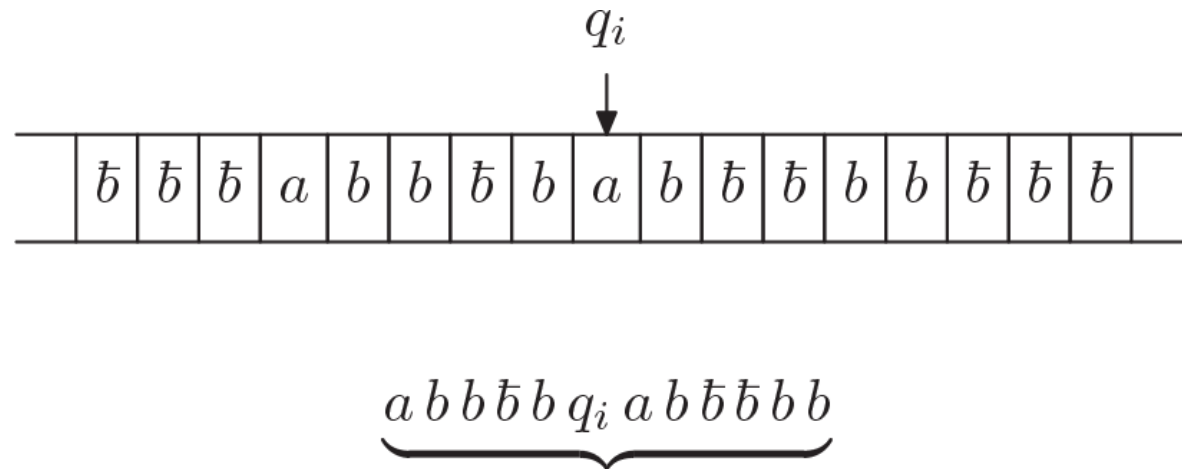
$$\delta(q_3, b) = (q_4, a, d)$$



È l'esecuzione di una transizione, che determina un passaggio da configurazione a configurazione

Configurazione (istantanea)

- Contenuto del nastro (porzione minima - finita - comprendente tutti i caratteri distinti da b) + posizione della testina + stato corrente



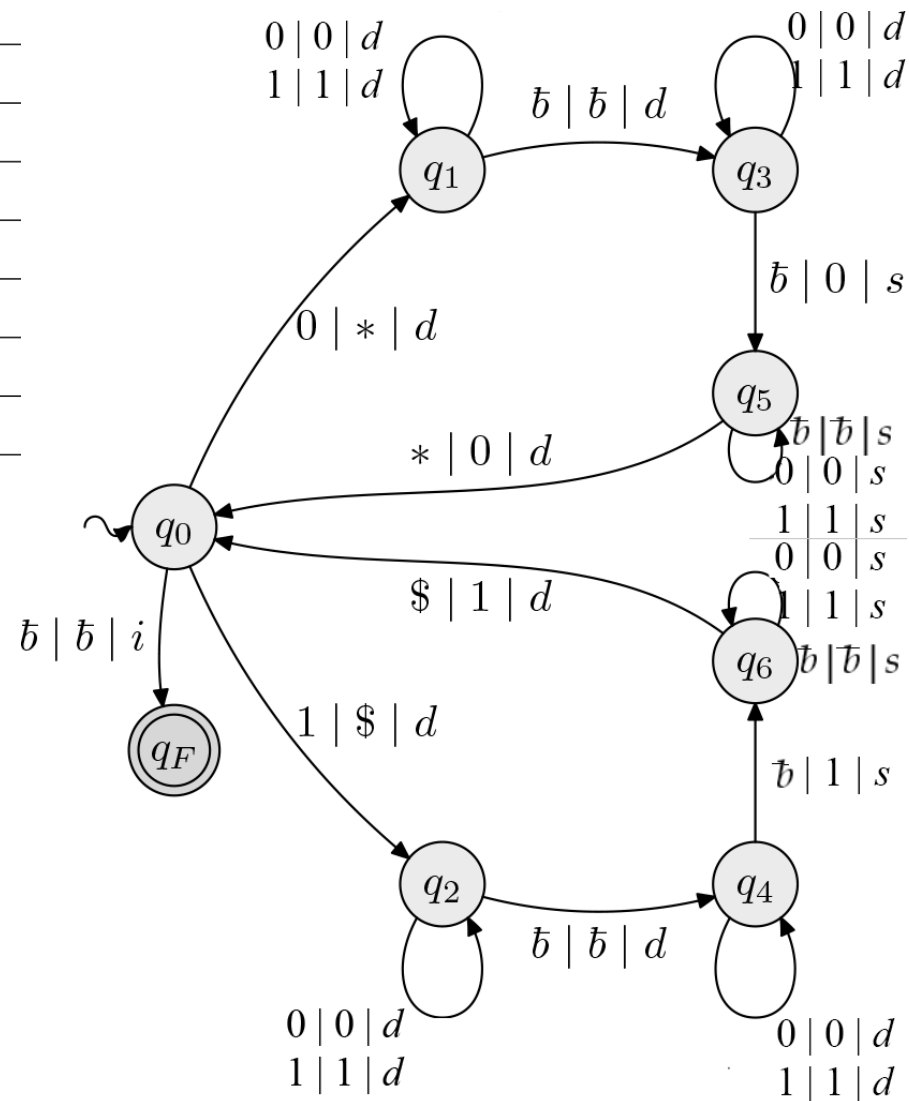
- Configurazione **iniziale**: $c_0 = qx$ (cioè a sinistra della testina compaiono solo istanze di b), $q = q_0$ e $x \in \Gamma^+ \cup \{\text{b}\}$, con $|x|$ finito. Si noti che x rappresenta l'**input della computazione** e, all'inizio del calcolo, la testina è convenzionalmente posizionata sul primo simbolo (non blank) di x

Configurazione

- Configurazione **finale**, detta anche configurazione **di accettazione**: wqy dove $q \in F$
- Una computazione può terminare anche in una configurazione non finale, detta configurazione **di rifiuto**
- **Determinismo**: secondo la funzione di transizione, data una configurazione corrente, esiste al più una configurazione prossima

Matrice e grafo di transizione

	0	1	*	\$	\bar{b}
q_0	$(q_1, *, d)$	$(q_2, \$, d)$	-	-	(q_F, \bar{b}, i)
q_1	$(q_1, 0, d)$	$(q_1, 1, d)$	-	-	(q_3, \bar{b}, d)
q_2	$(q_2, 0, d)$	$(q_2, 1, d)$	-	-	(q_4, \bar{b}, d)
q_3	$(q_3, 0, d)$	$(q_3, 1, d)$	-	-	$(q_5, 0, s)$
q_4	$(q_4, 0, d)$	$(q_4, 1, d)$	-	-	$(q_6, 1, s)$
q_5	$(q_5, 0, s)$	$(q_5, 1, s)$	$(q_0, 0, d)$	-	(q_5, \bar{b}, s)
q_6	$(q_6, 0, s)$	$(q_6, 1, s)$	-	$(q_0, 1, d)$	(q_6, \bar{b}, s)
q_F	-	-	-	-	-



Linguaggi decisi e accettati

Assunzione: alfabeto di input $\Sigma \subseteq \Gamma$

- Una MTD \mathcal{M} **accetta** (**rifiuta**) una stringa x se la computazione di \mathcal{M} con $c_0 = q_0x$ è di accettazione (di rifiuto)
- Una MTD \mathcal{M} **riconosce/decide** un linguaggio $L \subseteq \Sigma^*$ se e solo se, per ogni $x \in \Sigma^*$, esiste una computazione di accettazione da q_0x se $x \in L$, mentre esiste una computazione di rifiuto da q_0x se $x \notin L$. Il linguaggio L è detto *decidibile secondo Turing* (**T-decidibile**). Si noti che, per riconoscere un linguaggio, è necessario che la computazione della MTD termini per ogni input $x \in \Sigma^*$
- Una MTD \mathcal{M} **accetta** il linguaggio $L \subseteq \Sigma^*$ costituito dalla stringhe accettate da \mathcal{M} . Il linguaggio L è detto *semidecidibile secondo Turing* (**T-semidecidibile**)

MTD come riconoscitori

- Se il linguaggio L è deciso da una MTD \mathcal{M} , esso è anche accettato da \mathcal{M} , ma non viceversa. Sia data la configurazione iniziale $c_0 = q_0x$, con $x \in \Sigma^*$, $x \notin L$; se L è deciso, la computazione in corrispondenza di c_0 necessariamente termina (con un rifiuto), mentre, se L è solo accettato, la computazione in corrispondenza di c_0 può non terminare
- Ogni linguaggio T-decidibile è anche T-semidecidibile ma non viceversa

Esercizi

1. Definire una MTD con alfabeto di input $\{0, 1\}$ che riconosca il linguaggio formato da tutte le stringhe di lunghezza dispari (o pari)
2. Definire una MTD con alfabeto di input $\{0, 1\}$ che riconosca il linguaggio formato da tutte le stringhe che contengono una quantità pari (positiva) di 1 (o dispari)
3. Definire una MTD con alfabeto di input $\{0, 1\}$ che riconosca il linguaggio formato da tutte le stringhe del tipo $x\tilde{x}$, con $x \in \{0, 1\}^+$, dove \tilde{x} è la stringa x letta al contrario
4. Definire una MTD con alfabeto di input $\{0, 1\}$ che accetti il linguaggio formato da tutte le stringhe del tipo $x\tilde{x}$, con $x \in \{0, 1\}^+$, ma non riconosca questo linguaggio

(stringhe palindrome di lunghezza pari)

(vedi su Moodle la voce "Note brevi sui linguaggi accettati e decisi")

MTD come trasduttori

Assunzione: alfabeto di input $\Sigma \subseteq \Gamma$

- Una MTD \mathcal{M} **calcola** la funzione parziale $f : \Sigma^* \rightarrow \Sigma^*$ sse, per ogni $x \in \Gamma^*$:
 - se $x \in \Sigma^*$ e $f(x) = y$, allora, a partire dalla configurazione iniziale q_0x , \mathcal{M} raggiunge la configurazione finale $x\mathfrak{b}qy$ ($q \in F$). Si noti che y rappresenta l'**output della computazione** e, alla fine del calcolo, la testina è convenzionalmente posizionata sul primo simbolo (non blank) di y
 - altrimenti (se $x \notin \Sigma^*$ o $f(x)$ non è definita) \mathcal{M} non ha una computazione di accettazione a partire da q_0x

La funzione f è detta **T-calcolabile**

MTD come trasduttori (cont.)

- Funzioni con dominio e codominio arbitrari possono essere calcolate da una MTD codificando gli elementi di ingresso e uscita in stringhe
- Funzioni con più argomenti (con domini e codomini arbitrari) possono essere calcolate da macchine di Turing concatenando le codifiche dei diversi argomenti separate da un opportuno simbolo

Decidibilità e calcolabilità

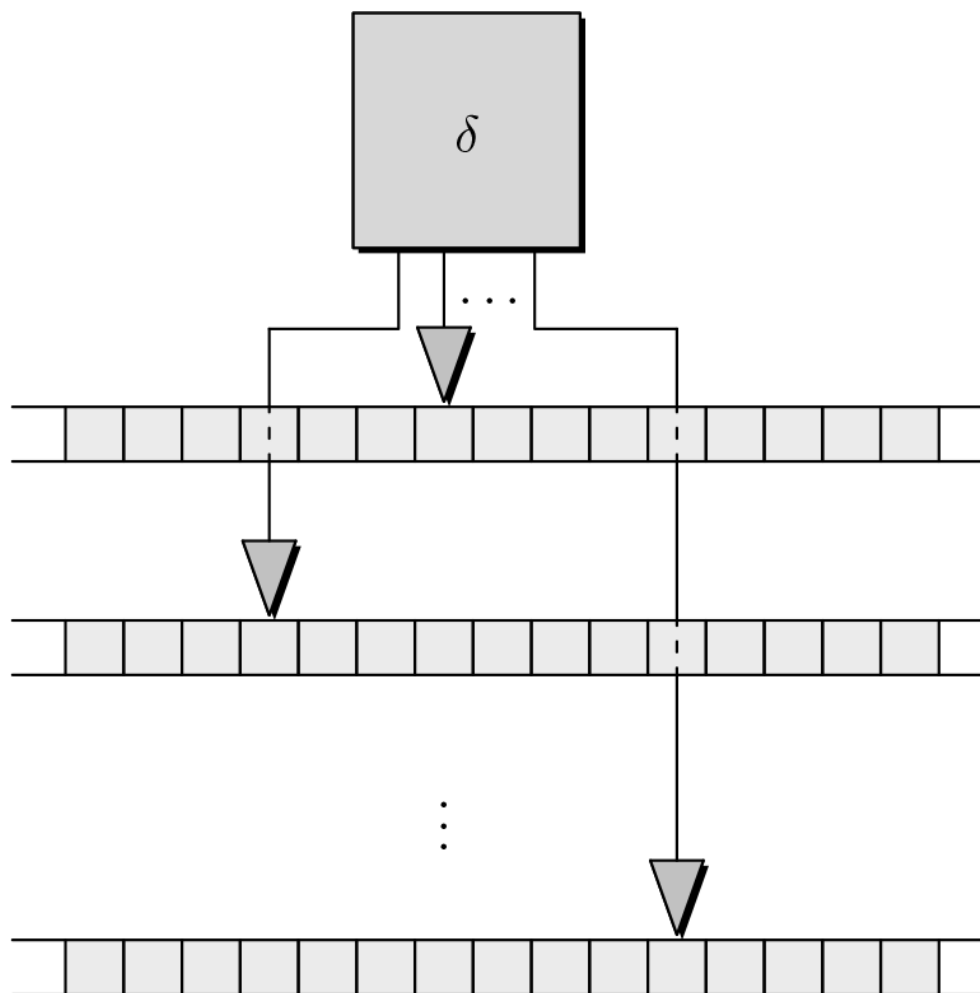
- Tutti i tentativi compiuti nell'ambito della logica matematica e della informatica teorica di definire concetti di calcolo basati su modelli e paradigmi alternativi alla MTD (ad es. macchine a registri, funzioni ricorsive) hanno condotto a classi di linguaggi decidibili e di funzioni calcolabili equivalenti a quelli introdotti da Turing
- **Tesi di Church-Turing** (postulato indimostrabile unanimemente ritenuto valido): ogni algoritmo espresso nel contesto di un qualunque modello di calcolo è realizzabile mediante una MTD

Esercizi

E' la MTD definita a pag. 9, che però ha una computazione di accettazione anche con una stringa d'ingresso vuota.
Modificare tale MTD cosicché non accetti tale stringa.

1. Definire una MTD che calcoli la funzione $f : \mathbb{N} \rightarrow \mathbb{N}$ tale che $f(n) = n$, assumendo che il numero naturale n sia in notazione binaria
2. Definire una MTD che calcoli la funzione $f : \mathbb{N} \rightarrow \mathbb{N}$ tale che $f(n) = 2 \cdot n$, assumendo che il numero naturale n sia in notazione binaria
3. Definire una MTD che calcoli la funzione $f : \mathbb{N} \rightarrow \mathbb{N}$ tale che $f(n) = 2 \cdot n$, assumendo che il numero naturale n sia in notazione decimale
4. Definire una MTD che calcoli la funzione $f : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ tale che $f(n_1, n_2) = n_1 + n_2$, assumendo che i valori naturali n_1 e n_2 siano in notazione binaria, separati da $\#$ e formati dallo stesso numero di cifre
5. Definire una MTD che calcoli la funzione $f : \mathbb{N} \times \mathbb{N} \rightarrow \{T, F\}$ tale che $f(n_1, n_2) = T$ se $n_1 \geq n_2$ e $f(n_1, n_2) = F$ altrimenti, assumendo che n_1 e n_2 siano in notazione binaria e separati da $\#$

Macchina di Turing Multinastro (MTM)



Ogni nastro ha la sua testina

MTM a k nastri ($k \geq 2$)

Introdotta solo per aumentare l'efficienza rispetto alla MTD, non la potenza computazionale

$\mathcal{M}^k = \langle \Gamma, \mathfrak{b}, Q, q_0, Z_0, F, \delta^k \rangle$ dove

- $\Gamma = \bigcup_{i=1}^k \Gamma_i$
- \mathfrak{b}, Q, q_0, F sono come definiti per la MTD (a nastro singolo)
- $Z_0 \notin \Gamma \cup \{\mathfrak{b}\}$ è l'unico simbolo presente (per convenzione) inizialmente su ciascuno dei nastri da 2 a k
- $\delta^k : (Q - F) \times (\Gamma_1 \cup \{\mathfrak{b}\}) \times (\Gamma_2 \cup \{\mathfrak{b}, Z_0\}) \times \cdots \times (\Gamma_k \cup \{\mathfrak{b}, Z_0\}) \rightarrow Q \times (\Gamma_1 \cup \{\mathfrak{b}\}) \times (\Gamma_2 \cup \{\mathfrak{b}, Z_0\}) \times \cdots \times (\Gamma_k \cup \{\mathfrak{b}, Z_0\}) \times \{d, s, i\}^k$

Ad esempio: $\delta^k(q_i, a_{i_1}, \dots, a_{i_k}) = (q_j, a_{j_1}, \dots, a_{j_k}, z_{j_1}, \dots, z_{j_k})$ dove $z_{j_l} \in \{d, s, i\}$ per $l = 1, \dots, k$

Configurazione (istantanea)

- È una stringa del tipo

$$q\#\alpha_1 \uparrow \beta_1\#\alpha_2 \uparrow \beta_2\#\dots\#\alpha_k \uparrow \beta_k$$

dove $\uparrow \notin \Gamma \cup \{\flat, Z_0\}$ è il simbolo che indica la posizione di ciascuna testina e $\# \notin \Gamma \cup \{\flat, Z_0\}$ è un separatore

- È **finale** se $q \in F$
- È **iniziale** se $\alpha_i = \varepsilon$ (cioè a sinistra della testina compaiono solo istanze del simbolo \flat) per $i = 1, \dots, k$, $\beta_1 \in \Gamma_1^*$, $\beta_i = Z_0$ per $i = 2, \dots, k$ e $q = q_0$

MTM: utilizzo convenzionale

(non è l'unico possibile)

- un nastro di input (il primo) a sola lettura, con testina che si muove in un solo verso
- vari nastri di lavoro, in lettura e scrittura, con testina che si muove in entrambi i versi
- un nastro di output a sola scrittura, con testina che si muove in un solo verso

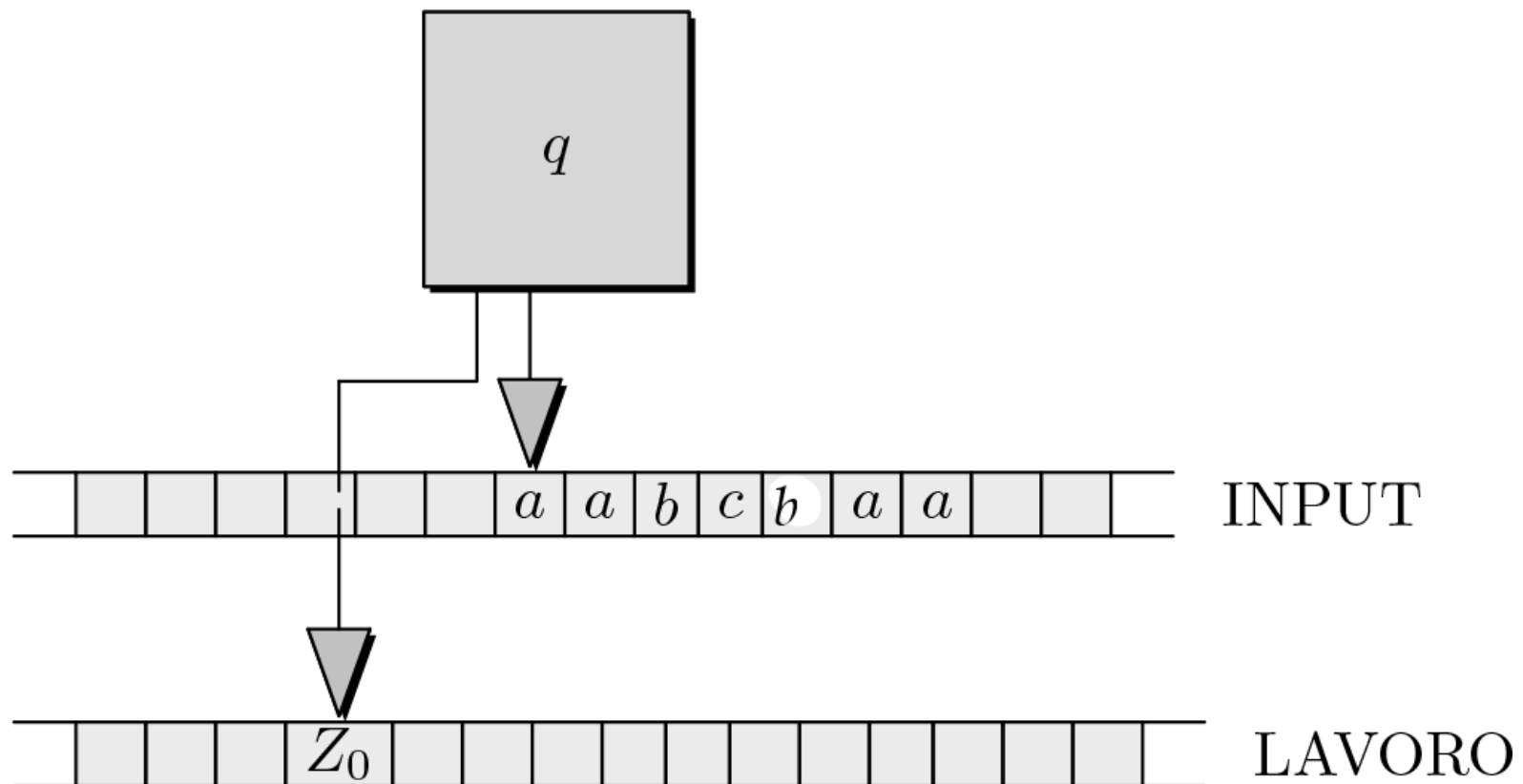
Il nastro di output serve solo se la MTM opera come trasduttore

MTM: decidibilità e calcolabilità

- Valgono le stesse considerazioni su decidibilità e semi-decidibilità secondo Turing fatte per la MTD a singolo nastro
- Una MTM \mathcal{M} **calcola** la funzione $f : \Sigma^* \rightarrow \Sigma^*$ ^{parziale} sse per ogni $x \in \Gamma_1^*$:
 - se $x \in \Sigma^*$ e $f(x) = y$ allora la computazione porta dalla configurazione iniziale $q_0\# \uparrow x\# \uparrow Z_0\# \dots \# \uparrow Z_0$ alla configurazione finale $q\# \uparrow x\# \uparrow y\# \uparrow \flat\# \dots \# \uparrow \flat$ e $q \in F$ (non è l'unica configurazione finale possibile: l'importante è che q sia uno stato finale e che sul nastro di uscita ci sia y)
 - altrimenti (se $x \notin \Sigma^*$ o $f(x)$ è indefinita) \mathcal{M} non ha una computazione di accettazione a partire da $q_0\# \uparrow x\# \uparrow Z_0\# \dots \# \uparrow Z_0$

Esempio

Definire una MTM che riconosca il linguaggio formato da tutte le stringhe del tipo $xc\tilde{x}$, dove $x \in \{a,b\}^+$ e \tilde{x} è la stringa x letta al contrario

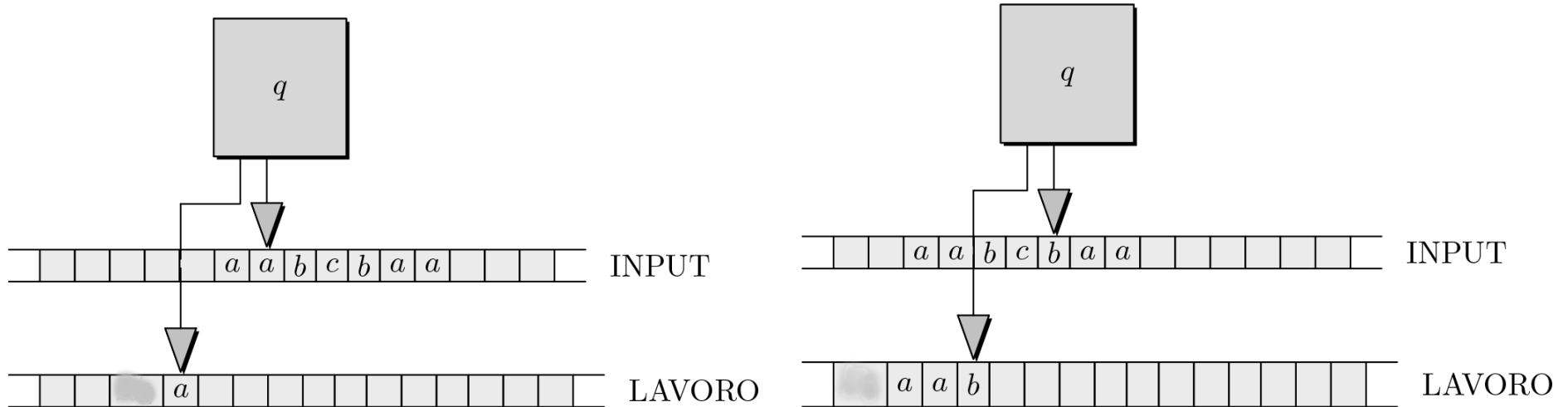


Esempio (cont.)

- L'alfabeto del nastro di input è $\{a, b, c\}$ mentre quello del nastro di lavoro è $\{a, b\}$

Si suppone che il nastro di input sia a sola lettura, con testina monodirezionale

- $Q = \{q_0, q_1, q_2\}$, q_0 è lo stato iniziale e $F = \{q_2\}$



$$\delta(q_0, a, Z_0) = (q_0, a, d)$$

$$\delta(q_0, b, Z_0) = (q_0, b, d)$$

$$\delta(q_0, a, \flat) = (q_0, a, d)$$

$$\delta(q_0, b, \flat) = (q_0, b, d)$$


$$\delta(q_0, c, \flat) = (q_1, \flat, s)$$

$$\delta(q_1, a, a) = (q_1, a, s)$$

$$\delta(q_1, b, b) = (q_1, b, s)$$

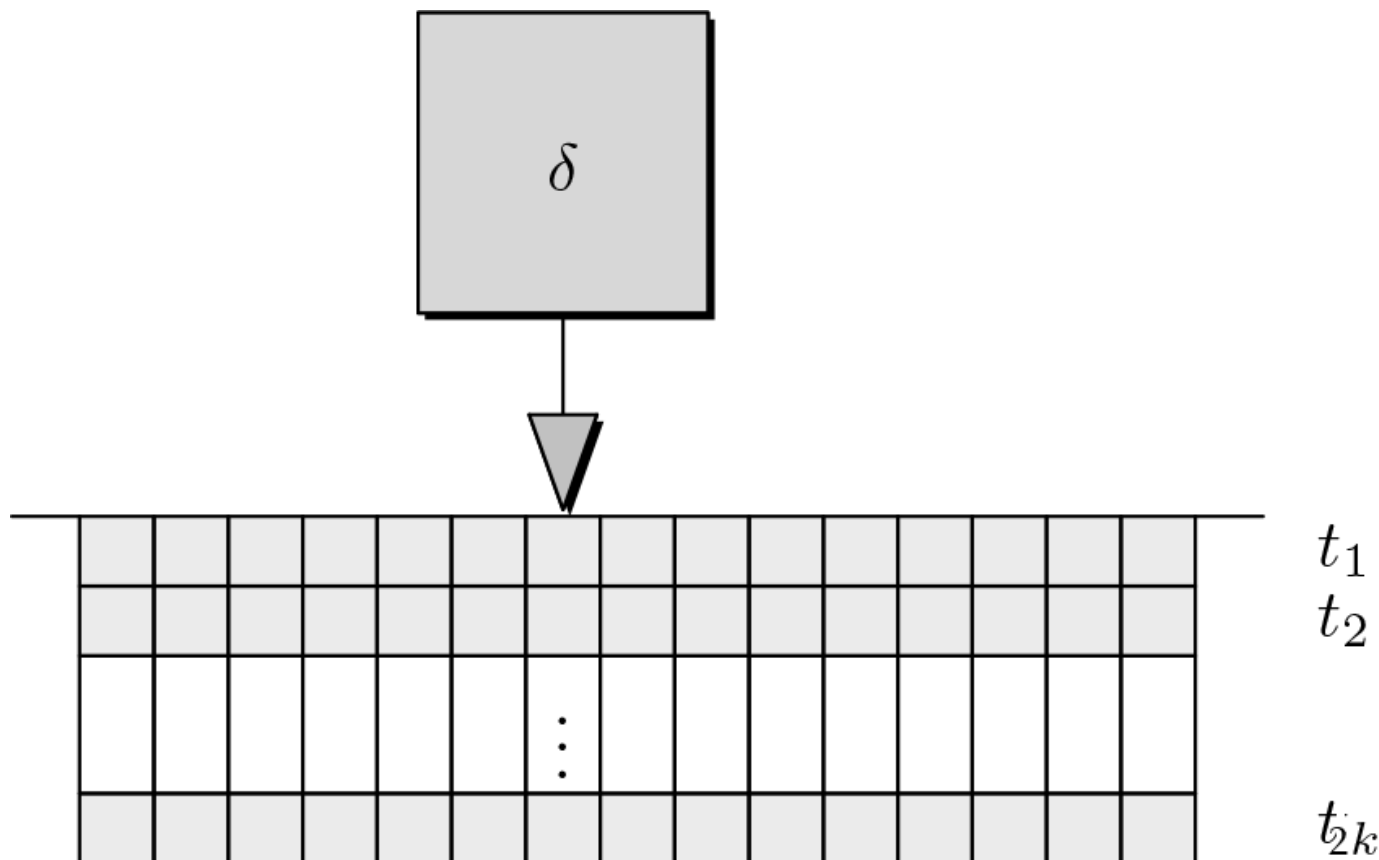
$$\delta(q_1, \flat, \flat) = (q_2, \flat, i)$$

Esercizi

1. Definire una MTM che riconosca il linguaggio formato da tutte le stringhe palindrome con alfabeto $\{a, b\}$

(oppure "accetti ma non decida")
2. Definire una MTM che riconosca il linguaggio formato da tutte le stringhe del tipo $x\tilde{x}$ di alfabeto $\{a, b\}$ dove \tilde{x} è la stringa x letta al contrario (stringhe palindrome di lunghezza pari)

Equivalenza fra MTM e MTD

Una MTM è simulabile attraverso una MTD avente il (singolo) nastro suddiviso in tracce, la cui testina può, con una sola operazione, leggere/scrivere tutti i caratteri disposti sulle tracce in corrispondenza della testina stessa



Equivalenza tra MTM e MTD (cont.)

...	\bar{b}	\bar{b}	\bar{b}	\bar{b}	\downarrow	\bar{b}	\bar{b}	\bar{b}	\bar{b}	...	t_1
...	\bar{b}	\bar{b}	\bar{b}	a	b	b	a	\bar{b}	\bar{b}	...	t_2
...	\bar{b}	\bar{b}	\bar{b}	\bar{b}	\downarrow	\bar{b}	\bar{b}	\bar{b}	\bar{b}	...	t_3
...	\bar{b}	\bar{b}	\bar{b}	b	a	b	c	b	\bar{b}	...	t_4
...	\bar{b}	\bar{b}	\downarrow	\bar{b}	\bar{b}	\bar{b}	\bar{b}	\bar{b}	\bar{b}	...	t_5
...	\bar{b}	d	e	d	d	e	\bar{b}	\bar{b}	\bar{b}	...	t_6
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	
...	\bar{b}	\bar{b}	\bar{b}	\bar{b}	\bar{b}	\downarrow	\bar{b}	\bar{b}	\bar{b}	...	t_{2k-1}
...	f	f	g	h	g	\bar{b}	\bar{b}	\bar{b}	\bar{b}	...	t_{2k}

Equivalenza tra MTM e MTD (cont.)

- A ogni passo della MTM, la MTD equivalente deve
 - percorrere la porzione di nastro compresa tra i due marcatori ↓ più lontani per individuare i caratteri letti dalla MTM
 - ripercorrere la porzione di nastro compresa tra i due marcatori ↓ più lontani (+ eventuali 2 nuove caselle) per scrivere i caratteri scritti dalla MTM e scrivere i nuovi marcatori in corrispondenza degli spostamenti delle testine della MTM
- A ogni passo i marcatori estremi si allontanano di al più due caselle
- Al passo i della MTM, i marcatori della MTD sono lontani al più $2 \cdot i$
- La porzione di nastro di dimensione al più $2 \cdot i$ deve essere percorsa 2 volte, quindi la MTD esegue al più $4 \cdot i$ passi
- Se la MTM esegue t passi, allora la MTD equivalente ne esegue $4 \cdot \sum_{i=1}^t i = 2 \cdot t^2 + 2 \cdot t$

Cardinalità dell'alfabeto della MTD equivalente

- La traccia 1 ha 2 caratteri
- La traccia 2 ha $|\Gamma_1| + 1$ caratteri
- La traccia 3 ha 2 caratteri
- La traccia 4 ha $|\Gamma_2| + 2$ caratteri
- Ecc.

$c_{i_1} \in \{\bar{b}, \downarrow\}$	t_1
$a_{i_1} \in \Gamma_1 \cup \{\bar{b}\}$	t_2
$c_{i_2} \in \{\bar{b}, \downarrow\}$	t_3
$a_{i_2} \in \Gamma_2 \cup \{\bar{b}, Z_0\}$	t_4
$c_{i_3} \in \{\bar{b}, \downarrow\}$	t_5
$a_{i_3} \in \Gamma_3 \cup \{\bar{b}, Z_0\}$	t_6
\vdots	
$c_{i_k} \in \{\bar{b}, \downarrow\}$	t_{2k-1}
$a_{i_k} \in \Gamma_k \cup \{\bar{b}, Z_0\}$	t_{2k}

$$2^k \cdot (|\Gamma_1| + 1) \cdot \prod_{i=2}^k (|\Gamma_i| + 2) - 1 = O((2 \cdot \max_{1 \leq i \leq k} |\Gamma_i|)^k)$$

Macchina di Turing Non Deterministica (MTND)

Introdotta solo per aumentare l'efficienza rispetto alla MTD, non la potenza computazionale

$\mathcal{M} = \langle \Gamma, \mathfrak{b}, Q, q_0, F, \delta_N \rangle$ dove

- $\Gamma, \mathfrak{b}, Q, q_0, F$ sono come definiti per la MTD (a nastro singolo)
- $\delta_N : (Q - F) \times (\Gamma \cup \{\mathfrak{b}\}) \rightarrow \mathcal{P}(Q \times (\Gamma \cup \{\mathfrak{b}\}) \times \{d, s, i\})$, funzione parziale dove \mathcal{P} è l'insieme delle parti (o insieme potenza) del suo argomento

Una MTND \mathcal{M} **accetta** una stringa x se esiste una sua computazione accettante a partire da q_0x

significa "almeno una"

Una MTND \mathcal{M} **rifiuta** una stringa x se tutte le computazioni di \mathcal{M} a partire da q_0x sono rifiutanti

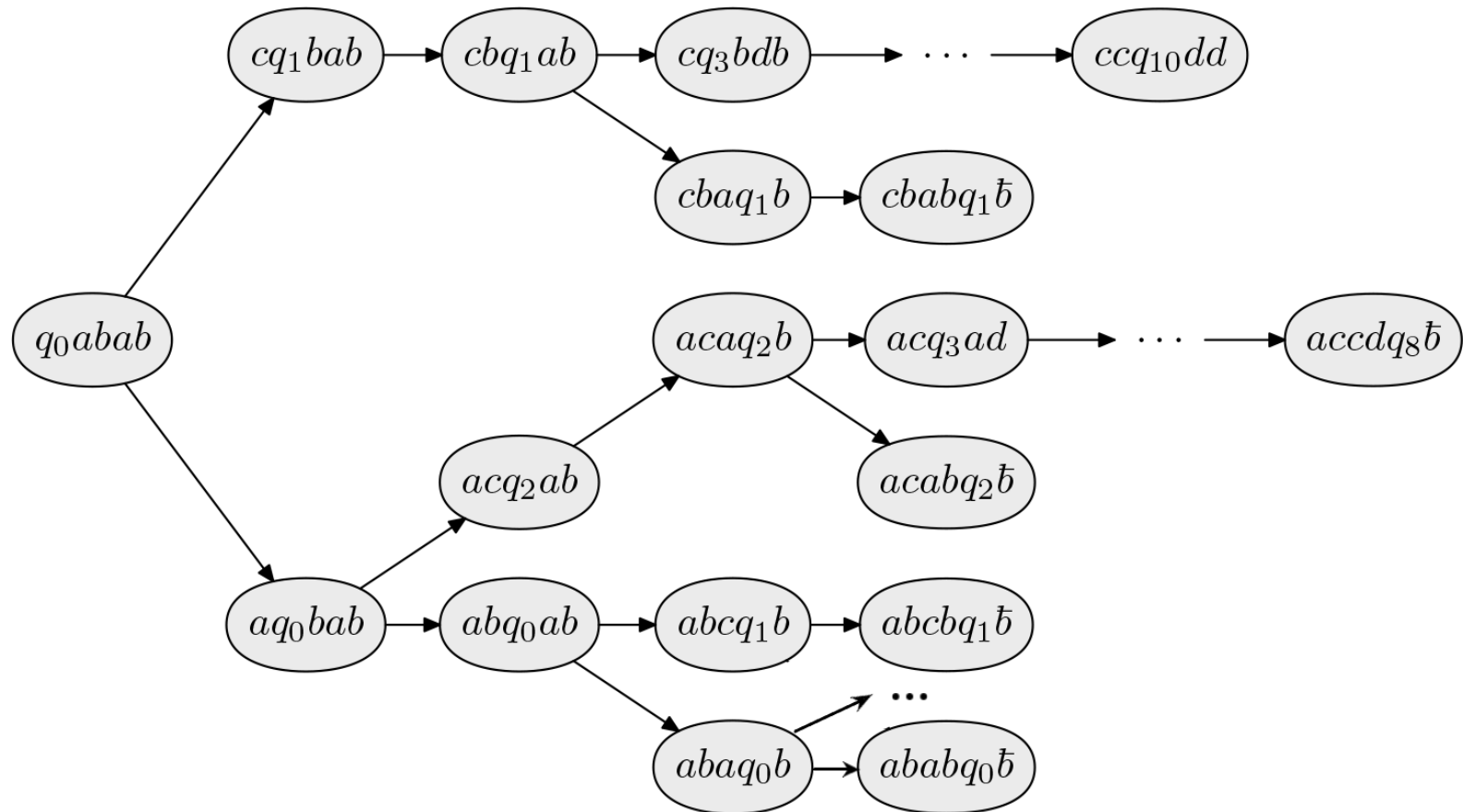
MTND: matrice di transizione

	a	b	c	d	\bar{b}
q_0	$\{(q_0, a, d), (q_1, c, d)\}$	$\{(q_0, b, d), (q_2, c, d)\}$	—	—	—
q_1	$\{(q_1, a, d), (q_3, d, s)\}$	$\{(q_1, b, d)\}$	—	—	—
q_2	$\{(q_2, a, d)\}$	$\{(q_2, b, d), (q_3, d, s)\}$	—	—	—
q_3	$\{(q_3, a, s)\}$	$\{(q_3, b, s)\}$	$\{(q_4, c, d)\}$	—	—
q_4	$\{(q_6, c, d)\}$	$\{(q_7, c, d)\}$	—	—	—
q_5	$\{(q_5, a, d)\}$	$\{(q_5, b, d)\}$	—	$\{(q_7, d, d)\}$	—
q_6	$\{(q_6, a, d)\}$	$\{(q_6, b, d)\}$	—	$\{(q_8, d, d)\}$	—
q_7	—	$\{(q_9, d, s)\}$	—	$\{(q_7, d, d)\}$	—
q_8	$\{(q_9, d, s)\}$	—	—	$\{(q_8, d, d)\}$	—
q_9	$\{(q_3, a, s)\}$	$\{(q_3, b, s)\}$	$\{(q_{10}, x, i)\}$	$\{(q_9, d, s)\}$	—
q_{10}	—	—	—	—	—

Grado di non-determinismo = 2

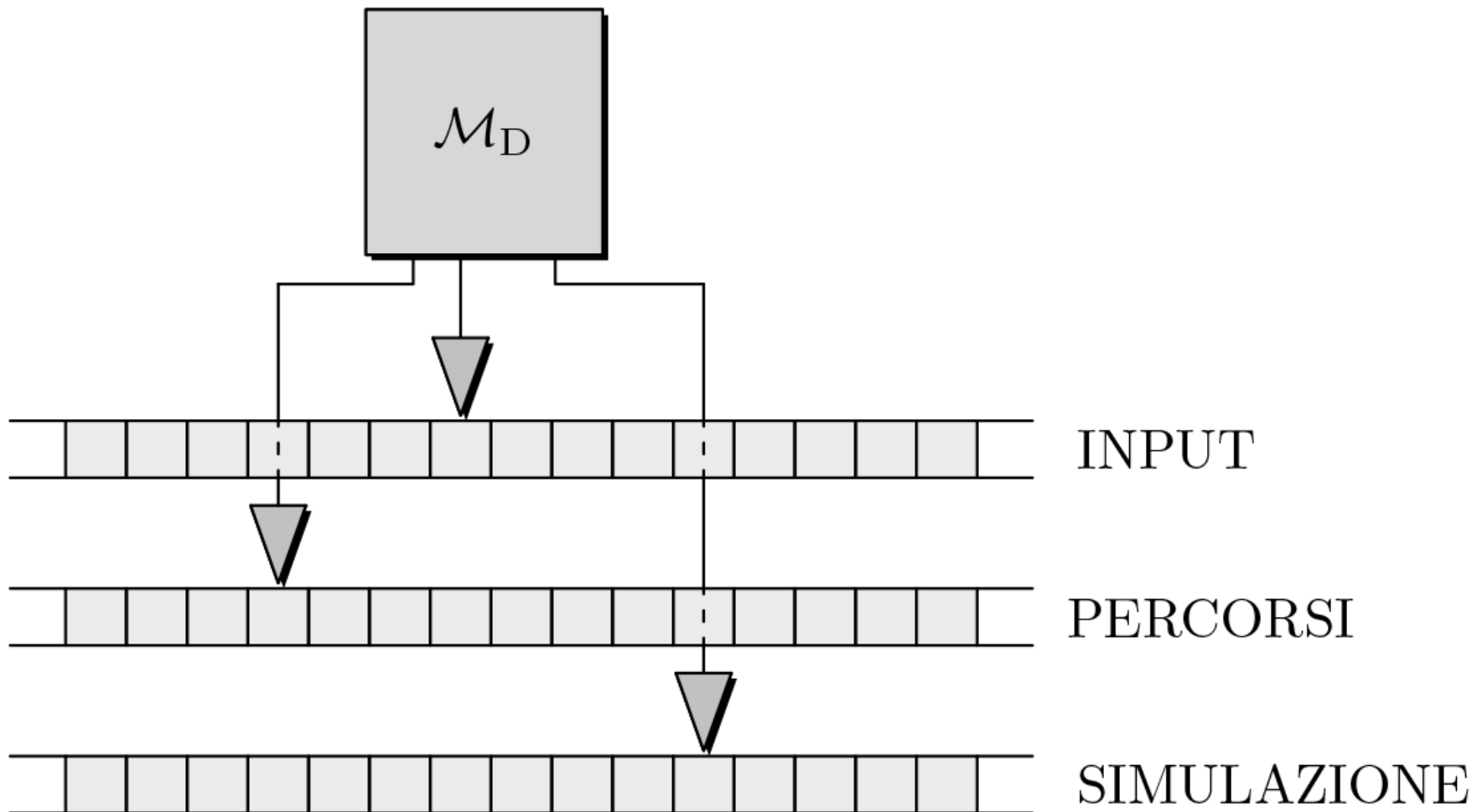
In questo esempio, una stringa $x \in \{a, b\}^*$ è accettata sse $\exists y \in \{a, b\}^*$, con $|y| \geq 2$, tale che $x = uyyv$, con $u, v \in \{a, b\}^*$

MTND: albero delle computazioni



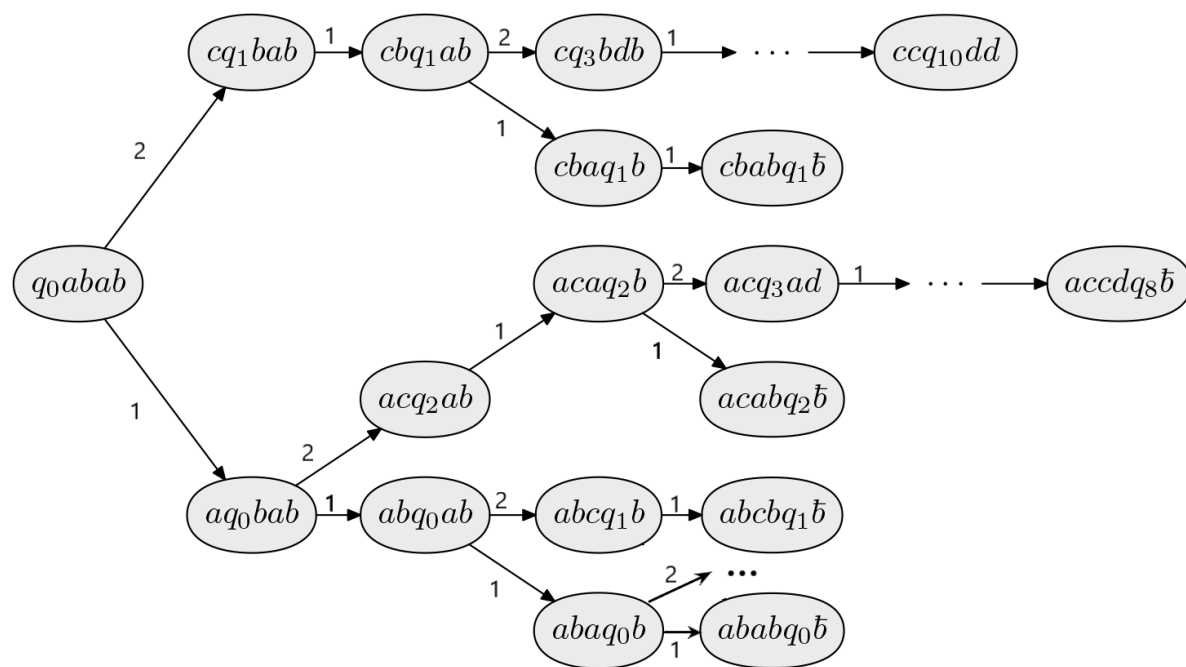
Domanda: questa MTND decide il linguaggio delle stringhe sopra definite o lo accetta soltanto?

Equivalenza tra MTM e MTND



Equivalenza tra MTM e MTND (cont.)

Nel nastro PERCORSI vengono generate in ordine lessicografico tutte le sequenze di lunghezza via via crescente (prima quella di lunghezza 0, poi tutte quelle di lunghezza 1, poi tutte quelle di lunghezza 2, ecc.) composte di interi compresi fra 1 e d ($=$ grado di non determinismo). Per ciascuna sequenza si esegue una fase di simulazione (in tal modo si effettua una visita in ampiezza dell'albero di computazione)



sequenza vuota

1
2
11
12
21
22
111
112
121
122
211
212
221
222
ecc.

Equivalenza tra MTM e MTND (cont.)

Assumiamo che il nastro PERCORSI contenga la sequenza i_1, i_2, \dots, i_k ; la corrispondente (singola) fase di simulazione avviene come segue:

1. Copia l'input x dal nastro INPUT al nastro SIMULAZIONE;
2. $j = 1$;
3. se $|\delta_N(q, a)| < i_j$, termina la fase negativamente (e inizia quella successiva, dal passo 1);
4. altrimenti, applica la i_j -esima terna (q', a', r) di $\delta_N(q, a)$ sul nastro SIMULAZIONE;
5. $j = j + 1$;
6. se $j \leq k$, torna al passo 3;
7. se $j > k$ e la configurazione è finale, termina con esito positivo;
8. altrimenti termina la fase negativamente (e inizia quella successiva, dal passo 1).

Equivalenza tra MTM e MTND (cont.)

Assumendo che la MTND accetti una stringa in k passi, la MTM a 3 nastri accetta la stringa in un numero di passi

$$\sum_{j=0}^k j \cdot d^j = o\left(d^k \sum_{j=0}^k j\right)$$

con un aumento esponenziale del tempo

Equivalenza con certificato

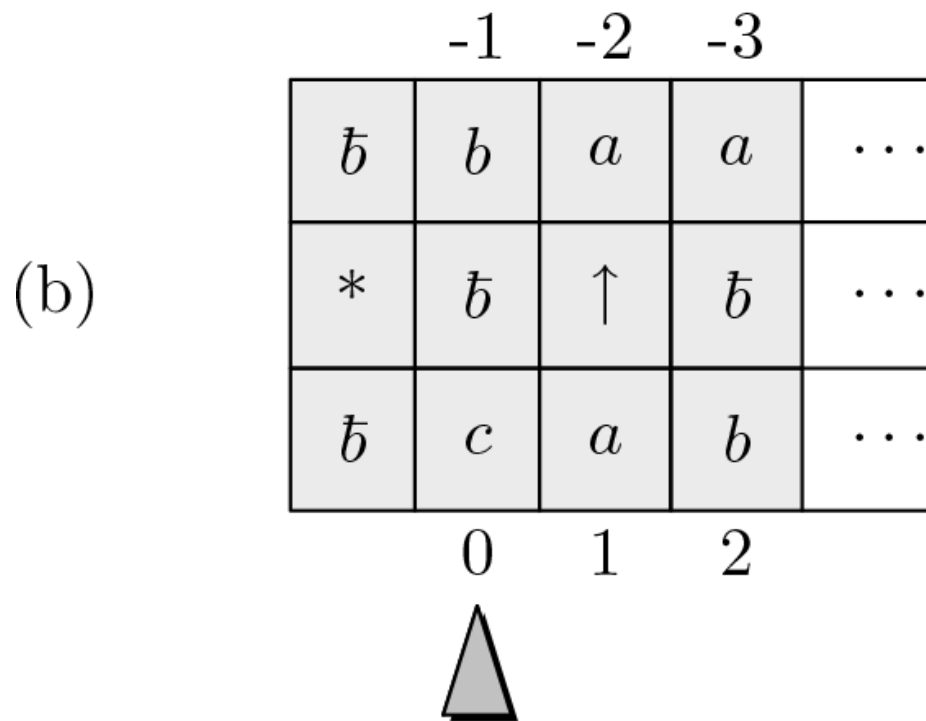
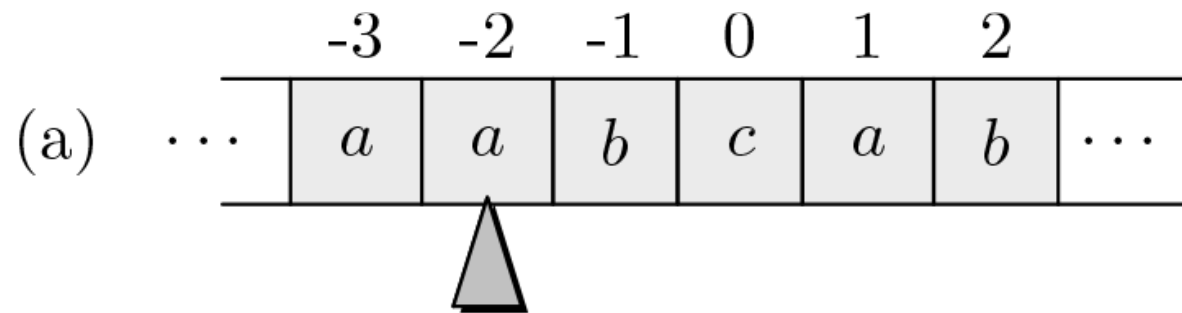
- Sia x una stringa accettata da una MTND. Un **certificato** di x è una stringa i_1, \dots, i_k , con $i_j \in \{1, \dots, d\}$, dove d è il grado di non determinismo, tale che la computazione corrispondente nell'albero delle computazioni è di accettazione
 - Se, per ogni stringa x accettata da una MTND, il nastro PERCORSI della MTM che la simula contiene un certificato di x , allora la simulazione è efficiente
- ⇒ La verifica da parte di una MTM che un certificato y sia la prova che una stringa x soddisfa una determinata condizione (proprietà) può essere effettuata in tempo sensibilmente inferiore rispetto alla ricerca del certificato stesso

Riduzione delle MTD

Ogni MTD può essere resa più semplice in due modi diversi senza che si perda potere computazionale:

- adottando un nastro infinito in una sola direzione (nastro *semi-infinito* a tre tracce)
- diminuendo la cardinalità dell'alfabeto, sino a 1

MT con nastro semi-infinito



MT con nastro semi-infinito: stati

Il simbolo '*' nella traccia centrale segnala l'inizio del nastro (le altre due tracce contengono un blank)

Sia \mathcal{M}' la MT a nastro semi-infinito equivalente alla MTD \mathcal{M} . Allora

$$Q' = \{q_i^\uparrow, q_i^\downarrow\} \cup \{q_{i,j}^\uparrow, q_{i,j}^\downarrow\}, \quad q_i, q_j \in Q$$

$$\text{Quindi } |Q'| = 2|Q| + 2|Q|^2$$

Le frecce \uparrow e \downarrow indicano rispettivamente che il carattere osservato è nella traccia superiore e in quella inferiore

$q'_0 = q_0^\downarrow$ perché nella configurazione iniziale l'input é contenuto nella traccia inferiore a partire all'origine mentre la traccia superiore contiene solo blank

MT con nastro semi-infinito: transizioni

Per ogni $\delta(q_h, a_i) = (q_j, a_k, d)$ - e analogamente per ogni $\delta(q_h, a_i) = (q_j, a_k, s)$ -

$$\delta' \left(q_h^\uparrow, \begin{bmatrix} a_i \\ \uparrow \\ c \end{bmatrix} \right) = \left(q_{h,j}^\uparrow, \begin{bmatrix} a_k \\ \bar{b} \\ c \end{bmatrix}, s \right) \quad \delta' \left(q_h^\downarrow, \begin{bmatrix} c \\ \uparrow \\ a_i \end{bmatrix} \right) = \left(q_{h,j}^\downarrow, \begin{bmatrix} c \\ \bar{b} \\ a_k \end{bmatrix}, d \right)$$

$$\delta' \left(q_{h,j}^\uparrow, \begin{bmatrix} c' \\ \bar{b} \\ c'' \end{bmatrix} \right) = \left(q_j^\uparrow, \begin{bmatrix} c' \\ \uparrow \\ c'' \end{bmatrix}, i \right) \quad \delta' \left(q_{h,j}^\downarrow, \begin{bmatrix} c' \\ \bar{b} \\ c'' \end{bmatrix} \right) = \left(q_j^\downarrow, \begin{bmatrix} c' \\ \uparrow \\ c'' \end{bmatrix}, i \right)$$

$$\delta' \left(q_{h,j}^\uparrow, \begin{bmatrix} \bar{b} \\ * \\ \bar{b} \end{bmatrix} \right) = \left(q_{h,j}^\downarrow, \begin{bmatrix} \bar{b} \\ * \\ \bar{b} \end{bmatrix}, d \right)$$

$$\delta' \left(q_{h,j}^\downarrow, \begin{bmatrix} \bar{b} \\ * \\ \bar{b} \end{bmatrix} \right) = \left(q_{h,j}^\uparrow, \begin{bmatrix} \bar{b} \\ * \\ \bar{b} \end{bmatrix}, d \right)$$

MT con nastro semi-infinito: transizioni (cont.)

Per ogni $\delta(q_h, a_i) = (q_j, a_k, i)$

$$\delta' \left(q_h^{\uparrow}, \begin{bmatrix} a_i \\ \uparrow \\ c \end{bmatrix} \right) = \left(q_j^{\uparrow}, \begin{bmatrix} a_k \\ \uparrow \\ c \end{bmatrix}, i \right)$$

$$\delta' \left(q_h^{\downarrow}, \begin{bmatrix} c \\ \uparrow \\ a_i \end{bmatrix} \right) = \left(q_j^{\downarrow}, \begin{bmatrix} c \\ \uparrow \\ a_k \end{bmatrix}, i \right)$$

MTD con $|\Gamma| = 1$

- I $|\Gamma| + 1$ simboli su cui lavora la MTD originale vengono codificati con sequenze di $k = \lceil \log_2(|\Gamma| + 1) \rceil$ simboli $\in \{b, 1\}$
- Per ogni transizione della MTD originale, la macchina equivalente effettua una serie di transizioni per
 - Determinare il carattere originario osservato
 - Eventualmente modificare tale carattere
 - Eventualmente spostare la testina più a destra o più a sinistra rispetto alla codifica del carattere stesso
- Per ogni transizione da q_i a q_j della MTD originale avremo fino a $3 \cdot (k - 1)$ nuovi stati nella macchina equivalente

MTD con $|\Gamma| = 1$: esempio

Sia \mathcal{M} una MTD operante sui simboli $\{a, b, c, d\} \cup \{\bar{b}\}$. Essa può essere simulata da una MTD \mathcal{M}' operante sui simboli $\{1\} \cup \{\bar{b}\}$, usando, ad es., la seguente codifica

$\bar{b} = \bar{b}\bar{b}\bar{b}$

$a = \bar{b}\bar{b}1$

$b = \bar{b}1\bar{b}$

$c = \bar{b}11$

$d = 1\bar{b}\bar{b}$

e trasformando opportunamente ogni regola di transizione

MTD con $|\Gamma| = 1$: esempio (cont.)

Ad es., la regola $\overset{\text{bb1}}{\delta}(q_i, a) = \overset{\text{b1b}}{(q_k, b, d)}$ di \mathcal{M} viene sostituita dalle seguenti regole in \mathcal{M}'

$$\delta'(q_i, \text{b}) = (q_{i,\text{b}}, \text{b}, d)$$

$$\delta'(q_{i,\text{b}}, \text{b}) = (q_{i,\text{bb}}, \text{b}, d)$$

$$\delta'(q_{i,\text{bb}}, 1) = (q_{i,\text{bb1}}, \text{b}, s)$$

$$\delta'(q_{i,\text{bb1}}, \text{b}) = (q'_{i,\text{bb1}}, 1, s)$$

$$\delta'(q'_{i,\text{bb1}}, \text{b}) = (q''_{i,\text{bb1}}, \text{b}, d)$$

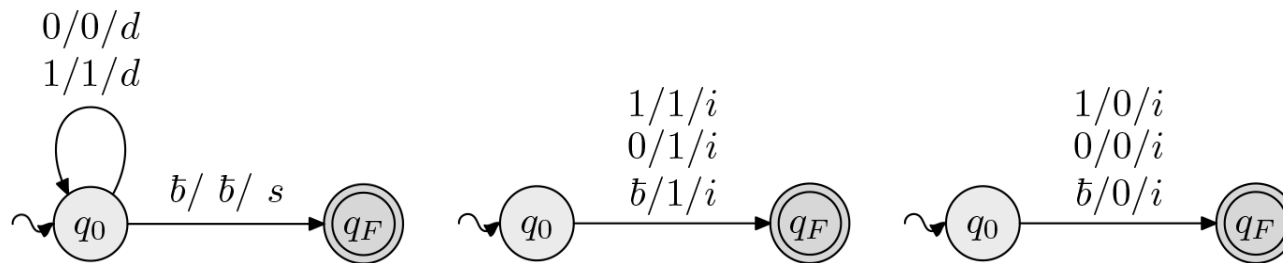
$$\delta'(q''_{i,\text{bb1}}, 1) = (q'''_{i,\text{bb1}}, 1, d)$$

$$\delta'(q'''_{i,\text{bb1}}, \text{b}) = (q_k, \text{b}, d)$$

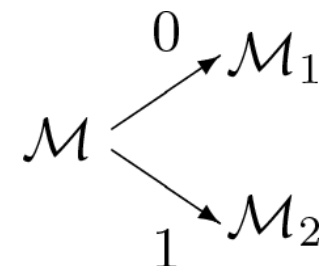
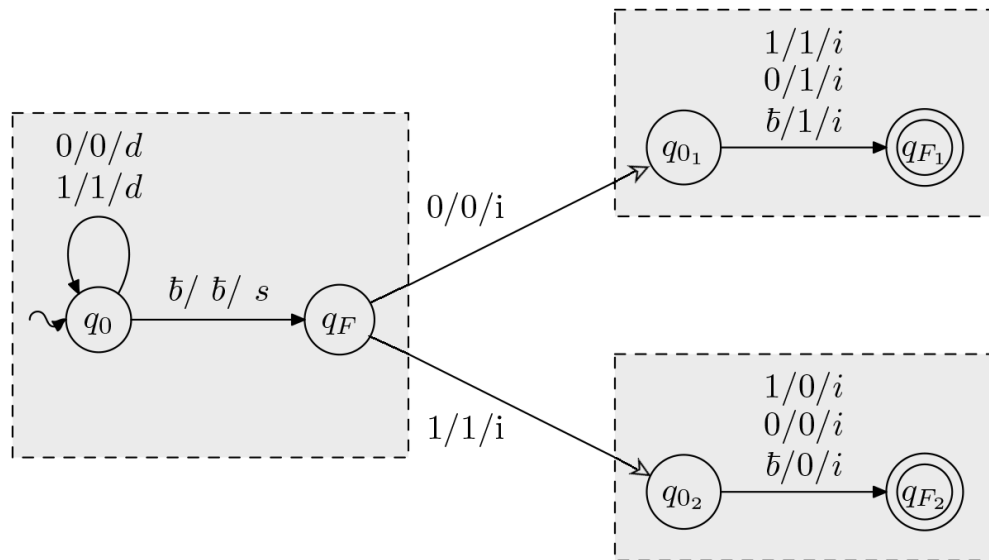
La riduzione del numero dei simboli di nastro comporta un aumento del numero degli stati (regola generale)

Composizione di MTD per diramazione condizionata

Le seguenti tre MTD alfabeto di nastro = $\{0,1\}$



possono essere composte in modo da realizzare una MTD che inverta l'ultimo simbolo di una sequenza di zeri e uni

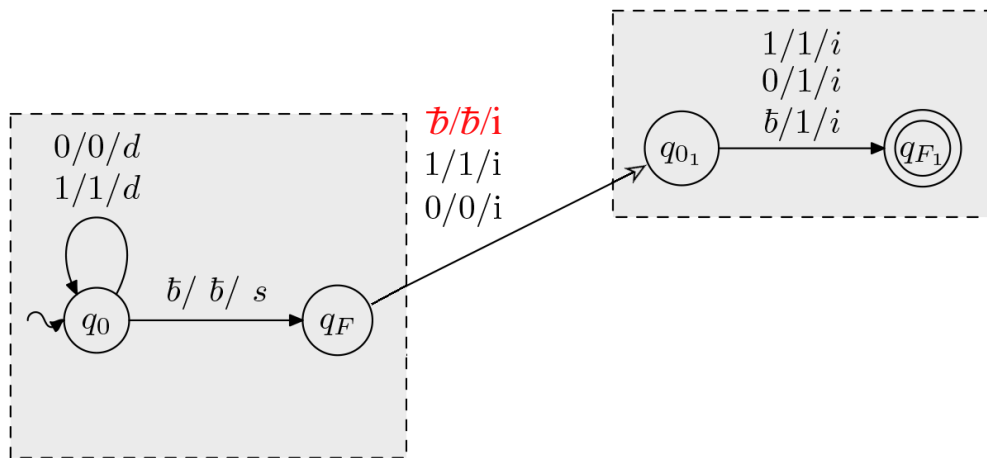


Composizione per diramazione

- Si possono comporre solo MTD aventi lo stesso alfabeto di nastro
- Può accadere che le macchine composte coincidano (tutte o solo alcune) fra loro

Sequenza e ciclo

Una sequenza è una diramazione incondizionata (nel diagramma il ramo di diramazione scompare). **Esempio di sequenza:** $\mathcal{M} \mathcal{M}_1$

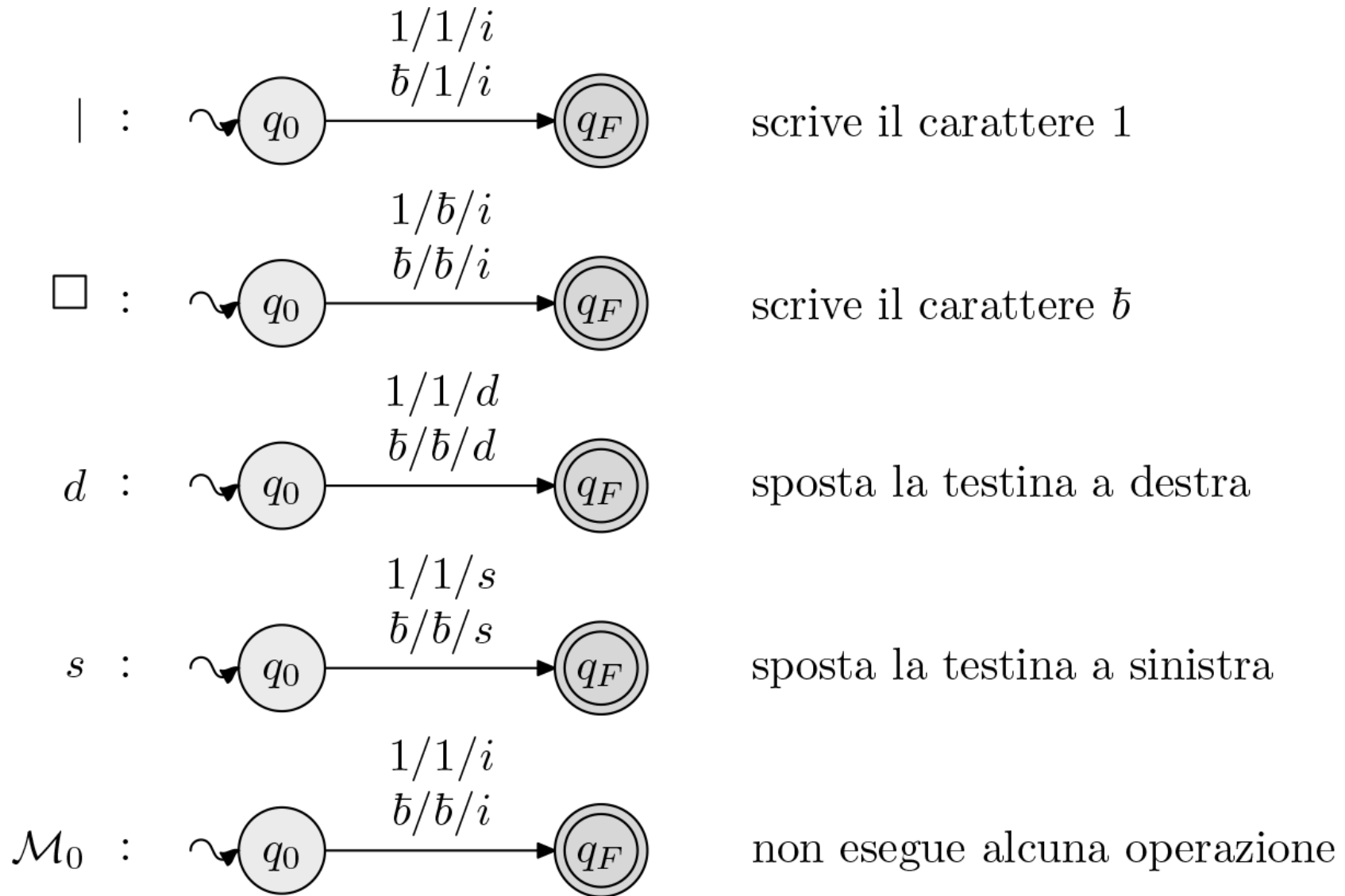


Esempio di ciclo: una delle due macchine verso cui \mathcal{M} si dirama coincide con \mathcal{M} stessa (si noti che la rappresentazione del ramo *else* scompare)

$$\mathcal{M}' = \mathcal{M} \mathcal{M}_1$$

The diagram shows a self-loop on the expression $\mathcal{M} \mathcal{M}_1$, labeled with σ_1 .

MTD elementari (per tutte $\Gamma = \{1\}$)

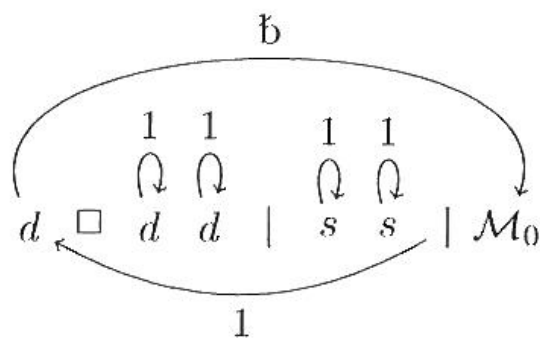


Macchine di Turing elementari

Teorema: Ogni MTD è rappresentabile come composizione per diramazione di macchine elementari e, viceversa, ogni diagramma di composizione di macchine elementari è la rappresentazione di una MTD

La descrizione di una MTD mediante macchine elementari prende il nome di **forma linearizzata**

Esempio: Forma linearizzata di una MTD che, a partire dalla configurazione $q_0 \mathfrak{b} x \mathfrak{b}$, raggiunge la configurazione $\mathfrak{b} x q_F \mathfrak{b} x$, dove $x \in \{1\}^*$:



Sintesi di trasduttori: esercizi

1. Fornire la descrizione linearizzata di una MTD che calcoli la sottrazione tra due interi n e m , con $n \geq m$, espressi in notazione unaria, ovvero una MTD che, a partire dalla configurazione

$$q_0 \mathfrak{b} 1^{n+1} \mathfrak{b} 1^{m+1} \mathfrak{b},$$

raggiunga la configurazione

$$\mathfrak{b} 1^{n+1} \mathfrak{b} 1^{m+1} \mathfrak{b} 1^{n-m+1} q_F \mathfrak{b}$$

2. Fornire la descrizione linearizzata della MTD che calcoli il prodotto di due interi espressi in notazione unaria, ovvero una MTD che, a partire dalla configurazione

$$q_0 \mathfrak{b} 1^{n+1} \mathfrak{b} 1^{m+1} \mathfrak{b},$$

raggiunga la configurazione

$$\mathfrak{b} 1^{n+1} \mathfrak{b} 1^{m+1} \mathfrak{b} 1^{n \cdot m + 1} q_F \mathfrak{b}$$

Sintesi di riconoscitori: esercizio

Sintetizzare mediante composizione di MT elementari una MTD con alfabeto di input $\{1\}$ che riconosca il linguaggio formato da tutte le stringhe di lunghezza pari (o dispari)

(oppure "accetti ma non decida")



Macchina di Turing universale

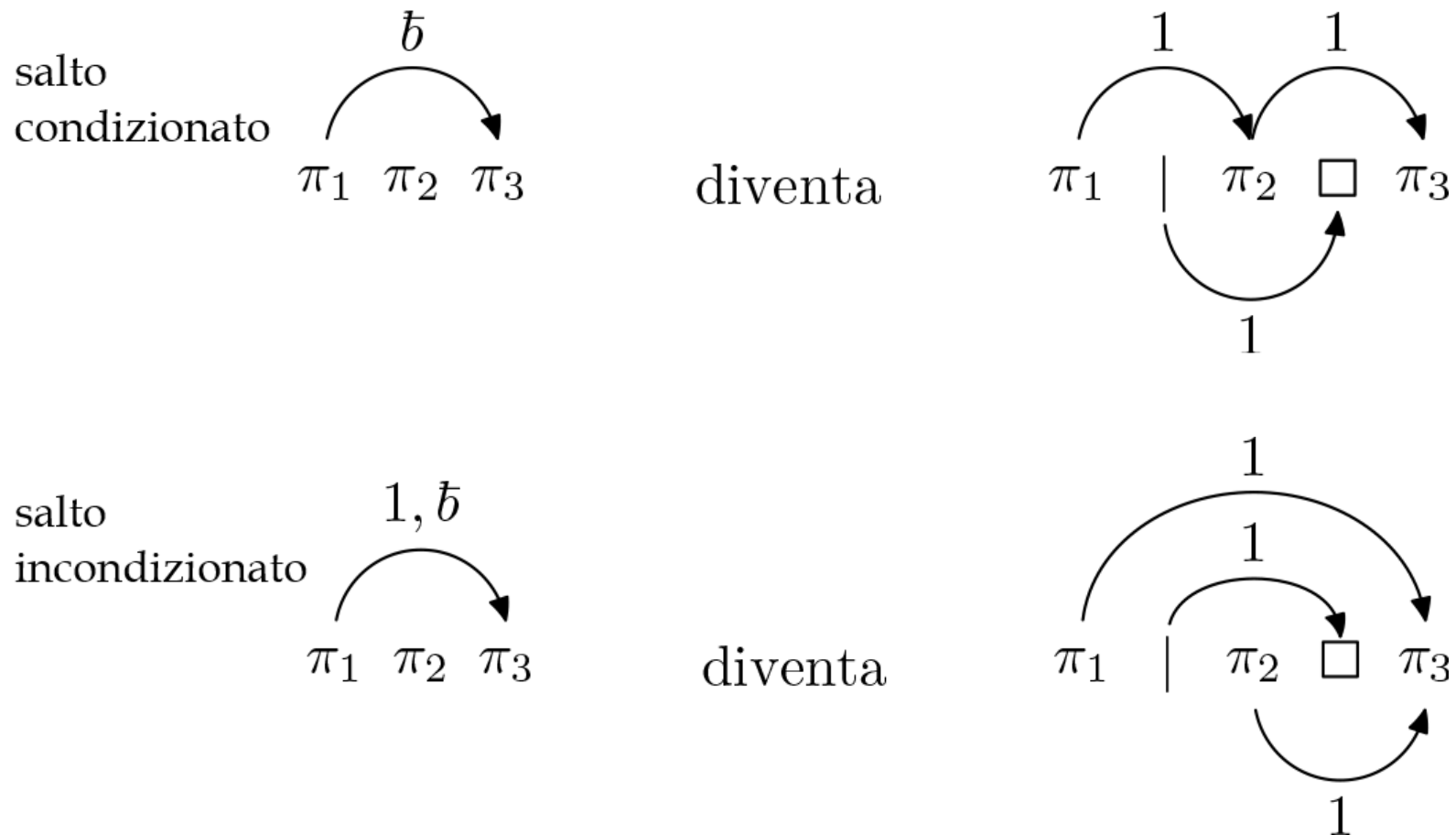
Sia $m : (\Sigma^*)^n \rightarrow \Sigma^*$ una funzione a più argomenti. Diciamo che una MTD calcola m se realizza la computazione che dalla configurazione $q_0x_1\mathbin{\text{`}}x_2\mathbin{\text{`}}\dots\mathbin{\text{`}}x_n$ raggiunge la configurazione $x_1\mathbin{\text{`}}x_2\mathbin{\text{`}}\dots\mathbin{\text{`}}x_n\mathbin{\text{`}}qy$, con q stato finale, sse $m(x_1, \dots, x_n) = y$

Una MTD si dice **universale** se è in grado di simulare ogni altra MTD, cioè se sa calcolare la funzione $u(c_{\mathcal{M}}, x_1, \dots, x_n) = m(x_1, \dots, x_n)$, data una qualunque MTD \mathcal{M} che calcola la funzione m , dove $c_{\mathcal{M}}$ è la codifica di \mathcal{M}

Un teorema ne dimostra l'esistenza

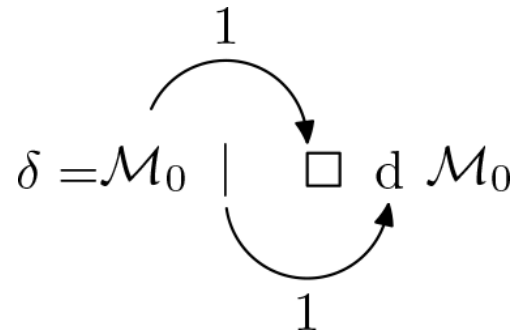
Normalizzazione dei salti

Per convenzione, i salti si possono effettuare con la sola diramazione 1



Macchina δ

Inverte il carattere letto, 1 o \bar{b} , e sposta a destra la testina



D'ora in poi si assume per convenzione che la macchina elementare \mathcal{M}_0 sia sempre presente al termine di una descrizione linearizzata e non sia mai presente in alcuna posizione intermedia

Macchine elementari: insieme minimo

Sfruttando la macchina δ , le descrizioni linearizzate di d , \square e $|$ con diramazioni solo sul simbolo 1 sono le seguenti:

$$d = \delta \ s \ \delta \ \mathcal{M}_0$$

$$\square = \delta \overset{1}{\curvearrowright} s \ \mathcal{M}_0$$

$$| = \delta \overset{1}{\curvearrowright} s \ \delta \ s \ \mathcal{M}_0$$

Per comporre qualsivoglia MTD bastano le macchine elementari δ e s e la diramazione solo sul simbolo 1

Macchina di Turing universale

La descrizione linearizzata di una qualunque MTD può essere codificata come una sequenza su $\{1, \mathfrak{b}\}$, come, ad es., la seguente:

- Le MT s e δ sono codificate con le stringhe 1 e 11
- Il salto condizionato sul simbolo 1 alla MT n -esima è codificato con la stringa 1^{n+3}
- La codifica di una istruzione è separata dalla codifica successiva da \mathfrak{b}
- La stringa 1^3 rappresenta la fine della sequenza di istruzioni

Problema della terminazione

(Halting problem, è semidecidibile)

$$h(c_{\mathcal{M}}, x) = \begin{cases} 1 & \text{se } \mathcal{M} \text{ termina su input } x \\ 0 & \text{se } \mathcal{M} \text{ non termina su input } x \end{cases} \quad \text{non è T-calcolabile}$$

Dimostrazione (per assurdo)

- Supponiamo per assurdo che esista una MTD \mathcal{H} capace di calcolare h
- Allora esiste una MTD $\mathcal{H}' = \mathcal{C}\mathcal{H}$ (\mathcal{C} è una macchina che duplica l'input) capace di calcolare la funzione h' :

$$h'(c_{\mathcal{M}}) = h(c_{\mathcal{M}}, c_{\mathcal{M}}) = \begin{cases} 1 & \text{se } \mathcal{M} \text{ termina su input } c_{\mathcal{M}} \\ 0 & \text{se } \mathcal{M} \text{ non termina su input } c_{\mathcal{M}} \end{cases}$$

Problema della terminazione (cont.)

Dimostrazione (cont.)

- Possiamo allora costruire una MT $\mathcal{H}'' = \mathcal{H}'\mathcal{E}$ (dove \mathcal{E} è una macchina che cicla indefinitamente se la testina è posizionata sul carattere 1 e si arresta immediatamente altrimenti) tale che:
 - \mathcal{H}'' calcola h'' t.c. $h''(c_{\mathcal{M}}) = 0$ sse \mathcal{M} su input $c_{\mathcal{M}}$ non termina
 - \mathcal{H}'' non termina se \mathcal{M} su input $c_{\mathcal{M}}$ termina
- Se l'input di \mathcal{H}'' fosse $c_{\mathcal{H}''}$ avremmo una MTD che
 - termina nel caso in cui la sua computazione non termina
 - non termina nel caso in cui la sua computazione termina
- L'assurdo implica che \mathcal{H}'' non può esistere e quindi \mathcal{H} non può esistere

Indecibilità

- Un programma contenente la dichiarazione di una particolare procedura chiamerà, nel corso della sua esecuzione, la procedura stessa?
- Una variabile definita all'interno di un programma assumerà un particolare valore, durante l'esecuzione del programma stesso?
- Un predicato usato per uscire da un ciclo assumerà il valore VERO?
- Un programma fornirà un certo output in corrispondenza di un particolare input?
- Due programmi sono equivalenti?