(ANCORA) MACCHINE DI TURING

Corso di Informatica Teorica - modulo 2

Prof. **Settimo Termini**

Ancora sulle MdT

Riprendiamo adesso alcune delle cose iniziate nella lezione scorsa.
Il simulatore proposto, pur efficacissimo, sottolinea un aspetto del funzionamento che è quello più simile a quello degli automi finiti, viene messo un po' in ombra, proprio il funzionamento "quasi manuale" sul nastro dettato dalle "quadruple".

Perdiamo allora un po' di tempo a sviluppare passo passo un esempio di computo della MdT che fa la sottrazione.

Sottrazione

Riportiamo nuovamente le quadruple già viste la lezione scorsa:

$$\left.\begin{array}{l} q_1 \mid Bq_1 \\ q_1 BRq_2 \end{array}\right\} \quad \text{cancella una barretta sulla sinistra}$$

$$\left.\begin{array}{l} q_2 \mid Rq_2 \\ q_2 BRq_3 \end{array}\right\} \quad \text{localizza il B di separazione}$$

$$\left.\begin{array}{l} q_3 \mid Rq_3 \\ q_3 BLq_4 \end{array}\right\} \quad \text{localizza il termine destro}$$

$$\left.\begin{array}{l} q_4 \mid Bq_4 \\ q_4 BLq_5 \end{array}\right\} \quad \text{cancella una barretta sulla destra}$$

Sottrazione

$q_5 \mid Lq_6\}$ se il secondo numero è stato cancellato si FERMA altrimenti va a q_6 ed a sinistra

$\left.\begin{array}{l} q_6 \mid Lq_6 \\ q_6BLq_7 \end{array}\right\}$ localizza il B di separazione

$\left.\begin{array}{l} q_7 \mid Lq_8 \\ q_7BRq_9 \end{array}\right\}$ invia alla (α) se il primo numero non è stato tutto cancellato altrimenti alla (β)

Sottrazione

$$(\alpha) \qquad \left.\begin{array}{l} q_8 \mid Lq_8 \\ q_8 BRq_1 \end{array}\right\} \quad \text{localizza il termine sinistro e ritorna a } q_1$$

$$(\beta) \qquad \left.\begin{array}{l} q_9 BRq_9 \\ q_9 \mid Lq_9 \end{array}\right\} \quad \text{entra in un ciclo infinito}$$

Esempio di calcolo

Esempio di calcolo di "$2 - 1$"

$$\ldots B \underset{\uparrow}{|} \underset{q_1}{} || B || B \ldots \qquad q_1 \mid B q_1$$

$$\ldots B \underset{\uparrow}{B} \underset{q_1}{} || B || B \ldots \qquad q_1 B R q_2$$

$$\ldots B B \underset{\uparrow}{|} \underset{q_2}{} | B || B \ldots \qquad q_2 \mid R q_2$$

$$\ldots B B | \underset{\uparrow}{|} \underset{q_2}{} B || B \ldots \qquad q_2 \mid R q_2$$

$$\ldots B B || \underset{\uparrow}{B} \underset{q_2}{} || B \ldots \qquad q_2 B R q_3$$

$$\ldots B B || B \underset{\uparrow}{|} \underset{q_3}{} | B \ldots \qquad q_3 \mid R q_3$$

MACCHINE DI TURING

$$\ldots BB \,||\, B \,|\, \underset{\uparrow}{|}\, B \ldots \qquad q_3 \,|\, Rq_3$$
$$\qquad\qquad\quad q_3$$

$$\ldots BB \,||\, B \,||\, \underset{\uparrow}{B} \ldots \qquad q_3BLq_4$$
$$\qquad\qquad\quad\;\; q_3$$

$$\ldots BB \,||\, B \,|\, \underset{\uparrow}{|}\, B \ldots \qquad q_4 \,|\, Bq_4$$
$$\qquad\qquad\quad q_4$$

$$\ldots BB \,||\, B \,|\, \underset{\uparrow}{B}\, B \ldots \qquad q_4BLq_5$$
$$\qquad\qquad\quad\;\; q_4$$

$$\ldots BB \,||\, B \,|\, \underset{\uparrow}{|}\, BB \ldots \qquad q_5 \,|\, Lq_6$$
$$\qquad\qquad\quad q_5$$

Ricordando che non c'è nessuna quadrupla che inizia con $q5\ B$ osserviamo che se la seconda stringa fosse stata già tutta cancellata, la MdT si sarebbe fermata.

MACCHINE DI TURING

$$
\begin{array}{ll}
\ldots BB \,||\, \underset{\uparrow}{B} \,|\, BB \ldots & q_6 B L q_7 \\
\qquad\qquad\quad q_6 & \\
\ldots BB \,|\, \underset{\uparrow}{|} \, B \,|\, BB \ldots & q_7 \,|\, L q_8 \\
\qquad\qquad\quad q_7 & \\
\ldots BB \,\underset{\uparrow}{|}\, |\, B \,|\, BB \ldots & q_8 1 L q_8 \\
\qquad\qquad\quad q_8 & \\
\ldots B\, B \,||\, B \,|\, BB \ldots & q_8 B R q_1 \\
\qquad\qquad\quad q_8 & \\
\ldots B \,\underset{\uparrow}{|}\, |\, B \,|\, B \ldots & q_1 \,|\, B q_1 \\
\qquad\qquad\quad q_1 &
\end{array}
$$

Esempio di calcolo

? DOMANDA **Cosa ha fatto?**

Ha cancellato una barretta dalla stringa che rappresenta il primo numero ed una dalla seconda ed è ritornato alla posizione iniziale. Quindi se seguissimo nuovamente il funzionamento passo passo, vedremmo che sotto l'effetto delle quadruple la MdT cancellerebbe di nuovo una barretta dalla prima stringa e una dalla seconda e così via.

Nel corso del computo abbiamo anche osservato che se la stringa che rappresenta il secondo numero fosse stata tutta cancellata, la macchina si sarebbe fermata perché non esiste una quadrupla che inizia con q_5B.

Esempio di calcolo

? **DOMANDA** **Cosa succede se il primo numero viene cancellato prima del secondo?** Cioè se in $n_1 - n_2$, $n_1 < n_2$

Porteremo adesso l'esempio di un calcolo molto semplice, quello di "0 − 1"

Rappresentando 0 con | e 1 con || si ha:

... B q1 | B | | B ...	q1 | B q1
... B q1 B B | | B ...	q1 B R q2
... B B q2 B | | B ...	q2 B R q3
... B B B q3 | | B ...	q3 | R q3
... B B B | q3 | B ...	q3 | R q3
... B B B | | q3 B ...	q3 B L q4
... B B B | q4 | B ...	q4 | B q4
... B B B | q4 B B ...	q4 B L q5
... B B B q5 |B B ...	q5 | L q6

Osserviamo che se la seconda stringa fosse stata già tutta cancellata, non essendoci una quadrupla che inizia con q5 B, la MdT si sarebbe fermata.

Vediamo adesso che succede:

... B B B q5 |B B ... q5 | L q6

... B B q6 B | B B ... q6 B L q7

... B q7 B B |B B ... q7 B R q9

La MdT avrebbe dovuto incontrare ancora un pezzo della prima stringa che invece è stata già tutta cancellata. Allora a questo punto si innesca il meccanismo di non terminazione.

... B B q9 B |B B ... q9 B R q9

... B B B q9 |B B ... q9 | L q9

... B B q9 B | B B ... q9 B R q9

Si vede che - per effetto delle ultime due quadruple - nella situazione data, la MdT non si fermerà più.

Qualche osservazione

La MdT fa bene ed esattamente quello che noi gli abbiamo chiesto di fare, dare il risultato corretto nel caso in cui il minuendo è maggiore o uguale del sottraendo, non fermarsi, se il risultato avrebbe dovuto essere un numero negativo e noi stiamo considerando SOLO funzioni dai numeri naturali ai naturali.

E tutto viene fatto bene relativamente alle convenzioni fatte sulla rappresentazione dei numeri, diverse in ingresso e in uscita.

Macchine di Turing

Abbiamo già visto che possiamo avere rappresentazioni diverse dalla rappresentazione standard della MdT mediante quadruple tra cui c'è quella mediante il "**diagramma di flusso**".

Riprendiamo l'esempio già visto:

Una MdT che scrive tre simboli su un nastro inizialmente bianco e che si ferma esaminando il quadrato che contiene il simbolo più a sinistra.

e ricordiamo il cambio di simboli:
indicheremo **B** ***con*** $\mathbf{S_0}$ ***e la barretta*** | ***con*** $\mathbf{S_1}$

Macchine di Turing

Ricordiamo che la MdT non ha bisogno di un'istruzione specifica di fermata; si ferma quando non ha istruzioni da eseguire. Possiamo eseguire queste azioni mediante le quadruple seguenti:

$q_1\ S_0\ S_1\ q_1$ $\qquad$ $q_1\ S_1\ \mathrm{L}\ q_2$,

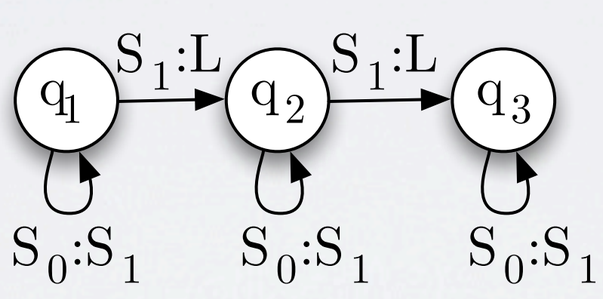
$q_2\ S_0\ S_1\ q_2$ $\qquad$ $q_2\ S_1\ \mathrm{L}\ q_3$,

$q_3\ S_0\ S_1\ q_3$

La rappresentazione mediante tabella di queste istruzioni si ottiene ponendo lungo un asse i simboli che la macchina può scrivere e lungo l'altro asse gli stati interni della macchina. L'elemento che si trova nella casella (i, j) ci dice cosa deve fare la macchina quando sta leggendo il simbolo i-esimo trovandosi nello stato j-esimo.

Macchine di Turing

Il diagramma di flusso della MdT precedente è quindi dato da:



Macchine di Turing

Una MdT che fa operazioni di scrittura di simboli sul nastro senza interpretare questi "numericamente", è stato introdotto prima come ausilio per introdurre in modo didatticamente semplice diversi modi nei quali possiamo rappresentare le MdT.

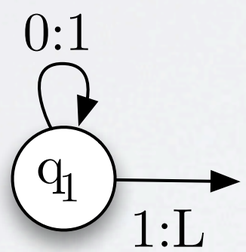
Ma adesso, senza un apparentemente rilevanza per problemi di "calcolabilità numerica", chiediamoci quanto facilmente possiamo progettare MdT che fanno "giochini" vari.

Macchine di Turing

Vogliamo costruire una MdT che scriva una stringa di n simboli 1.
Abbiamo già visto che possiamo costruire una MdT che scrive la stringa 111 (tre simboli 1 concatenati) con tre stati.

? DOMANDA È possibile, in generale, estendere questo procedimento e ottenere una MdT con n stati che scriva una stringa di n simboli 1?

Il diagramma di flusso di una tale macchina si può ottenere concatenando n copie del pezzo elementare:



ed eliminando la freccia che porta allo stato successivo nel pezzo elementare finale.

Macchine di Turing

Ma questo forse non è sempre il modo più economico e conveniente di ottenere il risultato cercato. Vediamo se è possibile costruire con meno di $2n$ stati una MdT che scrive una stringa di $2n$ 1.

Per prima cosa costruiamo una MdT che duplica il numero di 1 presenti sul nastro.

OSSERVAZIONE In base alle convenzioni usate la MdT deve partire trovandosi nello stato q1 ed esaminando il simbolo più a sinistra di una stringa di n 1 su un nastro che non contiene altro.

OSSERVAZIONE Sempre per le convenzioni fatte, la MdT si deve fermare esaminando il simbolo più a sinistra di una stringa di 1 lunga il doppio di quella iniziale; il nastro non contiene altro.

MdT che raddoppia una stringa data

Azioni principali da compiere:

- Scrivere a sinistra della stringa iniziale due 1.
- Cancellare in corrispondenza un 1 dalla stringa iniziale.
- Iterare questo processo fino a quando la stringa iniziale non è tutta cancellata.
- Localizzarsi in posizione standard di uscita.
- Fermarsi.

q_1 1 L q_2
q_2 B L q_3
q_2 1 L q_3
q_3 B 1 q_3
q_3 1 L q_4
q_4 B 1 q_4
q_4 1 R q_5

Sotto l'azione di queste quadruple la MdT:

- Si sposta a sinistra.
- Lascia un blank (a sinistra).
- Scrive due 1 (sempre a sinistra).
- Si sposta (a destra) di un quadrato.

q_5 B R q_6

q_5 1 R q_5

Sotto l'azione di queste quadruple la MdT si sposta sempre a destra rimanendo nello stesso stato fino a quando legge 1, cambiando stato quando incontra il primo blank.

q_6 B L q_7

q_6 1 R q_6

La MdT ha di nuovo incontrato la stringa iniziale, si sposta fino a quando non l'ha percorsa tutta, alla fine torna indietro di un quadrato cambiando stato.

q_7 1 B q_7

q_7 B L q_8

Cancella una barretta e si sposta a sinistra.

Adesso c'è una biforcazione:

Se la stringa iniziale è stata tutta cancellata allora si sposta a sinistra sulla nuova stringa scritta fino a occupare la posizione standard.

$q_8\ B\ \mathrm{L}\ q_{11}$

$q_{11}\ 1\ \mathrm{L}\ q_{11}$

$q_{11}\ B\ \mathrm{R}\ q_{12}$

Se la stringa iniziale non è stata tutta cancellata allora si deve ripetere il processo di aggiungere due barrette alla nuova stringa e cancellare una barretta della stringa iniziale.

Macchine di Turing

È conveniente riutilizzare le quadruple iniziali, ma per far ciò dobbiamo riportare la MdT nelle condizioni iniziali:

$q_8 1 L q_9$

$q_9 1 L q_9$

$q_9 B L q_{10}$ blank di separazione tra le due stringhe

$q_{10} 1 L q_{10}$ ripercorre tutta la nuova stringa

si trova sul B immediatamente a sinistra della nuova stringa

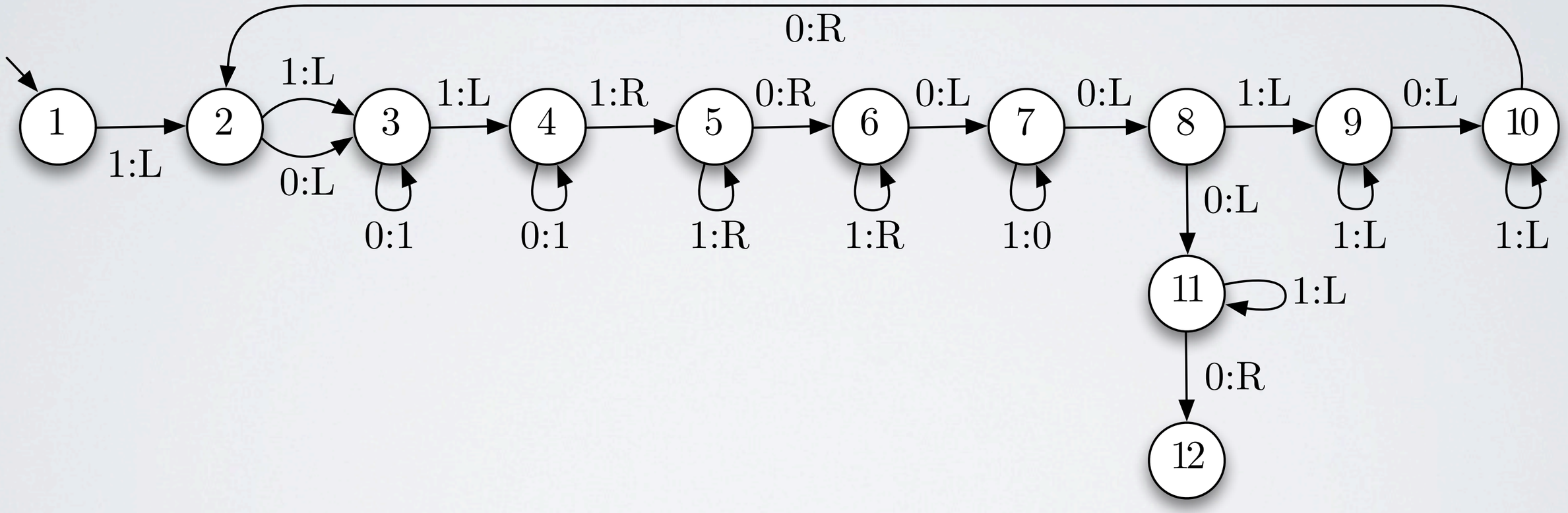
$q_{10} B R q_2$ scritta, si sposta a destra andando quindi sul primo simbolo.

della nuova stringa e ritornando nello stato q_2

Adesso dovrà stampare di nuovo due 1 a sinistra.

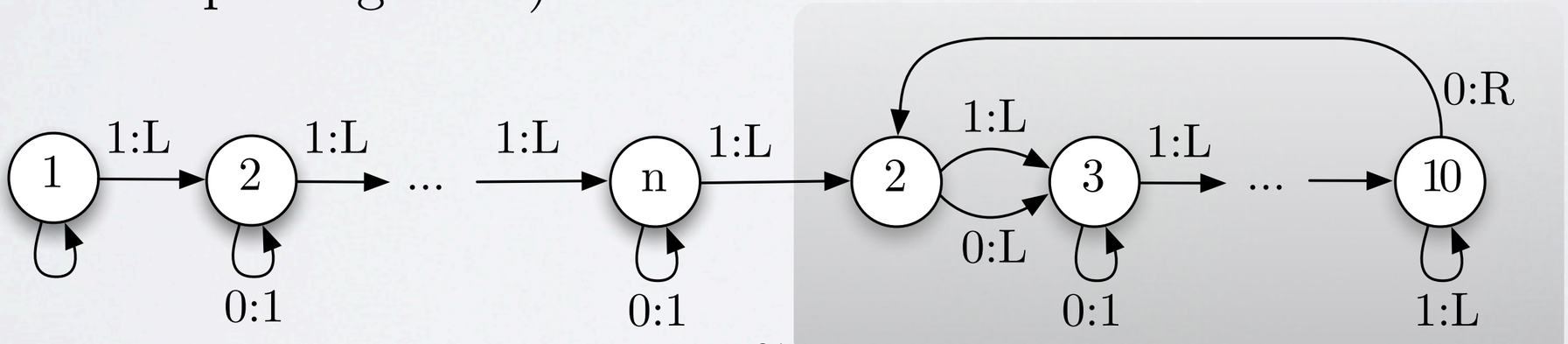
Macchine di Turing

Una MdT di questo tipo può essere rappresentata mediante il seguente diagramma di flusso.



Macchine di Turing

Torniamo adesso al problema lasciato in sospeso:
"progettare una macchina (con meno di $2n$ stati) che scriva una stringa di $2n$ simboli 1 su un nastro bianco"
Basta **mettere assieme** una MdT che stampa una stringa di n 1 e la MdT precedente.
Poiché possiamo identificare l'ultimo nodo della prima MdT col primo della seconda, avremo una MdT che esegue il lavoro richiesto con $n + 11$ stati (un risparmio notevole per n grande).



Iterando il “trucco” di scrivere una stringa più corta di quella richiesta e poi raddoppiarla potremmo risparmiare ancora sul numero di stati necessari.

Consideriamo il caso di una stringa di 100 simboli 1.

Metodo **diretto**:	MdT con **100 stati**
Metodo duplicazione usato **una volta**:	MdT con **61 stati**
Metodo duplicazione usato **due volte**:	MdT con **47 stati**
Metodo duplicazione usato **tre** volte (scrivere 12, 3 raddoppi aggiungere 4):	MdT con 48 stati (non conviene)

E se consideriamo una stringa di 1000 simboli?

Macchine di Turing

Abbiamo portato avanti un po' la tecnica di usare le MdT per giocare un poco con le stringhe. Abbiamo costruito "aggeggi" con un numero diverso di stati che lasciano, al momento di fermarsi, stringhe di lunghezza diversa sul nastro.

Anzi possiamo vedere ancora altri modi di "comporre" le due MdT considerate prima.

Macchine di Turing

Potremmo adesso porre varie altre domande, ad esempio:

Esistono sicuramente MdT che sono più "produttive" di altre nel senso che prima di fermarsi lasciano scritti più simboli sul nastro.

In che modo questo è legato al numero di stati della MdT?

Produttività delle macchine di Turing

Data una MdT μ definiamo produttività della MdT μ il numero di 1 che si trovano sul nastro quando la MdT si è fermata nella configurazione standard, a partire dal nastro bianco.
Se non si ferma o si ferma in un'altra configurazione, diciamo che la produttività di μ è 0.
Definiamo adesso la funzione p come segue:

D **DEFINIZIONE** ***La funzione produttività p, per ogni n, assume il valore p(n) uguale al valore della produttività delle MdT con n stati più produttive.***

Produttività delle macchine di Turing

PROPRIETÀ 1 Si ha che, per ogni n $p(n+1) > p(n)$

Per verificare questa proprietà, basta prendere una qualunque delle MdT a n stati più produttive e aggiungere ad essa un nuovo stato in maniera opportuna; ad esempio:

Produttività delle macchine di Turing

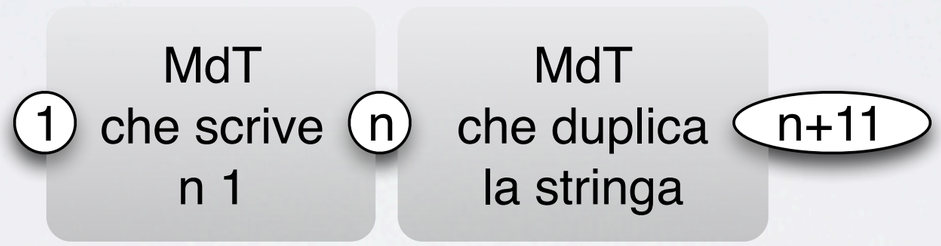
Questa è una MdT con $n + 1$ stati e che ha produttività strettamente maggiore della produttività della MdT più produttiva a n stati: $p(n) + 1$.
Infatti produce una stringa che ha un 1 in più della stringa prodotta dalla MdT con n stati di cui è una estensione.

OSSERVAZIONE Ovviamente, può darsi che esistano MdT con $n + 1$ stati con produttività molto più grande di quella che abbiamo trovato.

Produttività delle macchine di Turing

P PROPRIETÀ 2 Si ha che, per ogni n, $\quad p(n+11) > 2n$

Questa proprietà discende immediatamente osservando che prima abbiamo trovato una MdT con $n+11$ stati e che ha produttività $2n$. Tale MdT può essere schematizzata come segue:



O OSSERVAZIONE Anche in questo caso, come nel precedente, niente ci autorizza a concludere che la produttività $p(n+11)$ sia proprio eguale a $2n$ perché abbiamo esaminato solo ***una*** MdT con $n+11$ stati.

Produttività delle macchine di Turing

La definizione della funzione $p(n)$ è abbastanza intuitiva e sembra che sia sufficientemente "delineata" una strategia per trovare il valore $p(n)$ assunto dalla funzione p una volta dato n.
Ma poiché è abbastanza noioso calcolare $p(n)$ a mano potremmo far fare questo lavoro a una macchina.
Visto che le MdT sono proprio macchine che computano vorremmo progettare una MdT che calcola la funzione p.

? **DOMANDA** **Quale strategia possiamo seguire?**

Facciamo un po' di prove sul simulatore, per raccogliere dei "dati empirici" e poi torniamo al problema.

Produttività delle macchine di Turing

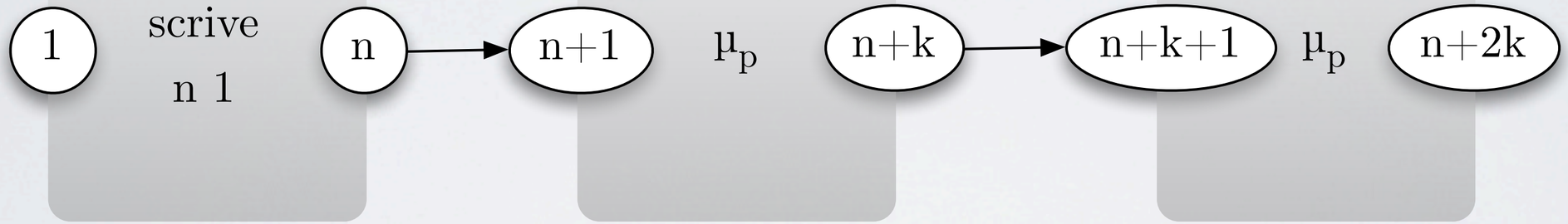
Prima di imbarcarci in una avventura che non sembra semplice poniamoci la domanda:

? **DOMANDA** **Ma è proprio sicuro che una tale macchina esista?**

P **PROPOSIZIONE** ***Non esiste nessuna MdT che calcola la funzione p.***

D **DIMOSTRAZIONE** Ammettiamo che tale macchina esista e chiamiamola μ_p. Sia k il numero degli stati della MdT μ_p. Sotto questa ipotesi esisterà sicuramente un'altra ${\mu^*}_p$ che ha produttività $p(p(n))$ ed è quella schematizzata come segue:

Produttività delle macchine di Turing



Produttività delle macchine di Turing

Possiamo dunque dire che $p(n + 2k) \geq p(p(n))$

Sappiamo già che

$$p(n + 1) > \mathrm{p}(n) \text{ per ogni } n$$

per cui possiamo ancora dire che

$$\text{se } i > j \text{ allora } p(i) > p(j)$$

e, in modo equivalente:

$$\text{se } p(j) \geq p(i) \text{ allora j} \geq i.$$

Ricordando che avevamo appena trovato che $p(n + 2k) \geq p(p(n))$, ponendo $j = n + 2k$ e $i = p(n)$ possiamo concludere che

$$n + 2k \geq p(n) \text{ per ogni } n$$

Produttività delle macchine di Turing

Poniamo adesso $m + 11 = n$.

Si ha $m + 11 + 2k \geq p(m + 11)$
poiché abbiamo già mostrato che $p(m + 11) \geq 2m$ per ogni m,
abbiamo che $m + 11 + 2k \geq 2m$ per ogni m.

Poiché k è fisso, per m sufficientemente grande arriviamo a un assurdo.

Quindi non esiste alcuna MdT che calcola p.