# MACCHINE DI TURING

La macchina di Turing (mdT) è un MODELLO DI CALCOLO computazionale basato su

MEMORIA:
Una TESTINA di lettura/scrittura, che si posa su una CELLA DI UN NASTRO INFINITO, contenente il codice di simboli. All'inizio la testina è posizionata sulla prima cella

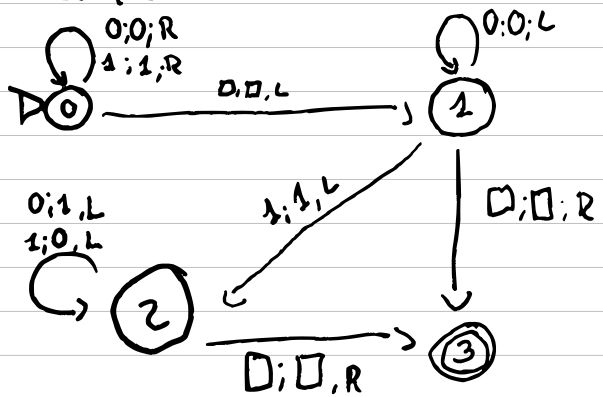
PARTE di CONTROLLO:
Esistono n stati FINITI, che la testina modifica a seconda del programma

È costituita da un alfabeto di lavoro $\Sigma$, contenente il simbolo □ e da un grafo delle transizioni, $G = \langle V, E \rangle$ tale che:

$V = \{q_0\} \cup F \cup Q$ è l'insieme degli stati: ($q_0$ è lo stato iniziale, F è l'insieme degli STATI FINALI e Q l'insieme degli altri stati)

E è l'insieme delle transizioni, in cui ad ognuno è associata una tripla $(\sigma, \tau, m)$ in cui $\sigma$ e $\tau$ sono simboli appartenenti a $\Sigma$ e $m \in \{R, L, S\}$ indica il MOVIMENTO

Es. complemento bit a bit




tabella

| IN | σ | τ | OUT | m |
|---|---|---|---|---|
| $q_0$ | 0 | 0 | $q_0$ | R |
| $q_0$ | 1 | 1 | $q_0$ | R |
| $q_0$ | □ | □ | $q_1$ | L |
| $q_1$ | 0 | 0 | $q_1$ | L |
| $q_1$ | 1 | 1 | $q_2$ | L |
| $q_1$ | □ | □ | $q_3$ | R |
| $q_2$ | 0 | 1 | $q_2$ | L |
| $q_2$ | 1 | 0 | $q_2$ | L |
| $q_2$ | □ | □ | $q_3$ | R |

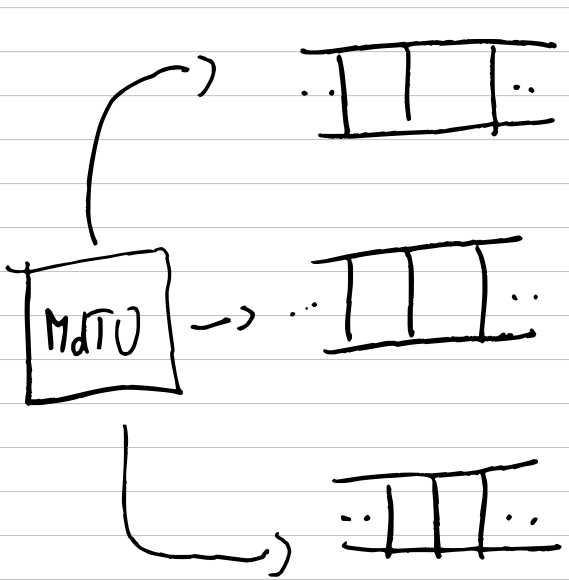
È possibile costruire anche una macchine di turing con k nastri, anche se è dimostrato che ha la STESSA POTENZA DI CALCOLO di una MdT a nastro singolo

PROBLEMA delle MdT:
Sono "hardwired", eseguono un solo programma.
I computer sono invece programmabili!

SOLUZIONE: MdT UNIVERSALE:

programmabile e può simulare se stessa!

E' possibile utilizzando 3 NASTRI




program. Descrizione e input

execution Descrizione

vuoto

stato attuale

vuoto

Nastro di M (emulazione di M)

# Nastro

Si codifica il nastro di M come stringa di simboli:

0 con Z  1 con U  □ con B  dato $\Sigma \in \{0,1,\square\}$

# Nastro 2 - STATI

Si codificano gli stati attraverso la loro numerazione BINARIA

$q_0 = 0$  $q_1 = 1$  $q_2 = 10$  $q_3$ 11 ...

# Nastro 1 - DESCRIZIONE di M.

- i movimenti sono codificati con
  Sinistra L  Destra D  Nulla S
- gli stati sono codificati in binario
- i caratteri del nastro sono codificati con
  Z U B

quindi

| | | | | | | |
|---|---|---|---|---|---|---|
| $q_0$ | 0 | $q_0$ | 1 | R | => | 0Z0ZR |
| $q_0$ | 1 | $q_0$ | 0 | R | => | 0U0ZR |
| $q_0$ | □ | $q_2$ | □ | R | | 0B10BR |
| $q_2$ | 0 | $q_2$ | 0 | L | | 10Z10ZR |
| $q_2$ | 1 | $q_2$ | 1 | L | | 10U10UL |
| $q_2$ | □ | $q_1$ | □ | R | | 10B1BR |

=> 0Z0ZR0U0ZRBB10BR10Z10ZR10U10UL10B1BR

Viene salvata su nastro 1

PASSI :

1: Copia sul terzo nastro l'input x codificato mediante i simboli U e Z (programmazione dell' input)
2. Inizializza il contenuto del 2° nastro con lo stato iniziale di M.
3. In base allo stato (2° nastro) e al simbolo letto (3° nastro), trova la transizione applicabile nel 1° nastro. Se non trova RICETTA
4. Applica la transizione, modificando il 3° nastro e aggiornando il 2° col nuovo STATO
5. Se il nuovo STATO è finale allora termina nell' UNICO STATO FINALE di U
altrimenti, torna nel passo 3

CON SOLO 28 STATI è possibile simulare QUALSIASI MdT !!!

↳ Interpreti: Nonostante la dimensione finita di essi sono in grado di eseguire un programma di lunghezza qualsiasi

Problema di fermata e calcolabilità

# Decidibilità

Un linguaggio $L$ è decidibile se la funzione CARATTERISTICA $X_L$ è decidibile.

Ergo, dato un input $y$, è possibile dire, matematicamente, se APPARTIENE o meno al linguaggio

$X_L(y) = 1$ appartiene

$X_L(y) = 0$ altrimenti

il problema della fermata è INDECIDIBILE.

Un linguaggio $L$ è semi-decidibile se esiste una MdT tale che, dato input $y$, se $y \in L$ allora MdT($y$) termina in una conf finale, altrimenti NON è detto

L STOP è semidecidibile:
possiamo sapere se un programma specifico si ferma, ma NON se qualsiasi programma lo farà