macchine a registri

macchine a registri

- RAM (Random Access Machine)
- astrazione ragionevole di un calcolatore
- memoria costituita da un numero arbitrario di celle (registri) indirizzabili direttamente
- una cella può contenere un intero positivo di un numero qualunque di cifre
- CI contatore istruzioni
- accumulatore (ind. 0 in memoria)
- nastri di ingresso e di uscita
- unità centrale in grado di eseguire istruzioni

macchine a registri – repertorio istruzioni

LOAD	[= | *]	operando (intero)
STORE	[*]	operando
ADD	[= | *]	operando
SUB	[= | *]	operando
MULT	[= | *]	operando
DIV	[= | *]	operando
READ	[*]	operando
WRITE	[= | *]	operando
JUMP		etichetta (intero)
JGTZ		etichetta
JZERO		etichetta
HALT		etichetta

LOAD STORE per e dall'accumulatore

*   indirizzamento indiretto   = operando immediato

JGTZ JZERO funzione del contenuto dell' accumulatore

macchine a registri – programmi

il programma non è contenuto in memoria ma è cablato

esempio: programma per il calcolo di x^y

```
    READ     1      lettura di x dal nastro di ingresso
    READ     2      lettura di y dal nastro di ingresso
    LOAD     =1
    STORE    3      inizializzazione del risultato
5   LOAD     2
    JZERO    14     fine ciclo while?
    LOAD     1
    MULT     3
    STORE    3
    LOAD     2
    SUB      =1
    STORE    2
    JUMP     5
14  WRITE    3      stampa del risultato
    HALT
```

modelli di costo per RAM

modello a costi uniformi

costo unitario per l'esecuzione di ogni istruzione

esempio: eseguire il programma per calcolare x^y costa O(y); infatti le 9 istruzioni del ciclo vengono eseguite y volte

critica: calcolare 10 x 10 costa quanto calcolare 1.000.000 x 1.000.000

critica: il costo per accedere ad una cella di memoria è indipendente dal numero di celle

modello a costi logaritmici

l(n) = 1	se n=0
l(n) = int(log n)+1	se n>0

l'elaborazione di un intero n ha costo l(n)

l'accesso al registro i ha costo l(i)

modelli di costo per RAM

esempio: valutazione di costo con modello a costi logaritmici

	READ	1	$l(1)+l(x)$	$O(\log x)$
	READ	2	$l(2)+l(y)$	$O(\log y)$
	LOAD	=1	$l(1)$	$O(1)$
	STORE	3	$l(3)+l(1)$	$O(1)$
5	LOAD	2	$l(2)+l(y)$	$O(\log y)$
	JZERO	14	$l(y)$	$O(\log y)$
	LOAD	1	$l(1)+l(x)$	$O(\log x)$
	MULT	3	$l(3)+l(x)+l(x^y)$	$O(\log x + y \log x)$
	STORE	3	$l(3)+l(x^y)$	$O(y \log x)$
	LOAD	2	$l(2)+l(y)$	$O(\log y)$
	SUB	=1	$l(1)+l(y)$	$O(\log y)$
	STORE	2	$l(2)+l(y)$	$O(\log y)$
	JUMP	5	1	$O(1)$
14	WRITE	3	$l(3)+l(x^y)$	$O(y \log x)$
	HALT		1	$O(1)$

la porzione di codice dentro la cornice viene eseguita O(y) volte

costo di esecuzione: $O(y^2 \log x)$

modelli di costo per RAM

- osservazione: in pratica in situazioni in cui non si debbano fare operazioni aritmetiche ed in cui le dimensioni della memoria sono prefissate i due modelli coincidono
- osservazione: la presenza di operazioni aritmetiche con precisione infinita impone l'uso di modelli di calcolo analoghi al modello a costi logaritmici
- critica: nel modello a costi logaritmici la moltiplicazione dovrebbe costare di più della addizione; allo stato attuale non esiste un algoritmo che calcoli la moltiplicazione tra 2 interi ad n bit in meno di n log n

RAM e MT

una funzione f: $N^n \rightarrow N$ è calcolabile da una RAM se esiste un programma P tale che:

se $f(x_1,...,x_n)=y$, allora P, attivato con $x_1,...,x_n$ sul nastro di input, termina con y sul nastro di output

se $f(x_1,...,x_n)$ non e' definita, allora P non termina

RAM e MT

teorema: sia $M=<\{0,1\},\underline{b},K,q_0,F,\delta>$ una MT con nastro seminfinito; esistono una RAM ed un programma P tali che se M computa $q_0x \vdash^* q_Fy$ e la RAM ha la stringa x nelle celle 2,....,|x|+1, al termine della computazione la RAM ha la stringa y nelle celle 2,....,|y|+1; inoltre la RAM simula T passi di M in tempo O(T log T) nel modello a costi logaritmici

dimostrazione:

stabiliamo una corrispondenza tra la cella i del nastro di M e la cella i+1 della memoria della RAM

usiamo la cella 1 della RAM per rappresentare la posizione della testina di M

all'inizio la cella 1 contiene il valore 2 (posizione iniziale della testina di M)

P contiene una sequenza di istruzioni per ogni stato di M (MT elementari)

dimostrazione (MT→RAM)

esempio:

se $\delta(q_1,0)=<q_2,1,d>$ e $\delta(q_1,1)=<q_3,0,s>$
allora P contiene

```
....................
q1  LOAD   *1     acquisizione del contenuto del nastro
    JGTZ   q1'    lettura di un 1 o di uno 0?
    LOAD   =1
    STORE  *1     scrittura di un 1 sul nastro
    LOAD   1
    ADD    =1
    STORE  1      movimento a destra della testina
    JUMP   q2     aggiornamento dello stato
q1' LOAD   =0
    STORE  *1     scrittura di uno 0 sul nastro
    LOAD   1
    SUB    =1
    STORE  1      movimento a sinistra della testina
    JUMP   q3     aggiornamento dello stato
....................
```

dimostrazione (MT$\rightarrow$RAM)

ogni passo di M e' simulato da al piu' 8 istruzioni ognuna con costo

O(log maxmem)

maxmem e' il max numero di celle usate da M

se M esegue T passi allora maxmem ≤ T+1

costo complessivo O(T log T)

RAM e MT

teorema: data una RAM con programma P che calcola la funzione f esiste una MT M tale che se $f(x_1,\ldots,x_n)=y$ e sul nastro di input sono memorizzati in binario gli interi $x_1,\ldots,x_n$ la macchina M termina con la rappresentazione binaria di y sul nastro di output e se $f(x_1,\ldots,x_n)$ non è definita M non termina; inoltre se la computazione della RAM ha costo T nel modello a costi logaritmici allora M la simula in O(p(T)) passi; p(T) è un polinomio in T

dimostrazione: M ha tre nastri di lavoro, un nastro di input e un nastro di output

sul nastro 1 sono memorizzati in binario i dati memorizzati nella memoria della RAM preceduti dal proprio indirizzo, rappresentato in binario

sul nastro 2 e' rappresentato il contenuto dell'accumulatore

il nastro 3 e' usato per il funzionamento di M

dimostrazione (RAM→MT)

memoria della RAM

i	mem[i]
1	11
10	
11	
100	1001
101	10
110	
111	
1000	1
1001	
1010	
….	

nastro 1 della MT

$	$	1	$	1	1	$	$	1	0	0	$	1	0	0	1	$	$	1	0	1	$	1	0	$	$	1	0	0	0	$	1	$	$

dimostrazione (RAM→MT)

per ogni istruzione del programma della RAM, M ha un insieme di stati per eseguire le operazioni

- istruzioni di salto JUMP, JGTZ, JZERO, HALT:

  si controlla se il contenuto dell'accumulatore verifica le condizioni di salto e si gestiscono i corrispondenti cambiamenti di stato
- istruzioni che non modificano il contenuto della memoria LOAD, ADD, SUB, MULT, DIV, WRITE:

  si cerca sul nastro 1 la cella di memoria su cui si deve operare

  si esegue l'operazione prevista

dimostrazione (RAM→MT)

- istruzioni che modificano il contenuto della memoria STORE,READ

  si trascrive sul nastro 3 il contenuto del nastro 1 per tutto cio' che e' a destra della cella di memoria da modificare

  si esegue l'operazione

  si ricopia sul nastro 1 il contenuto del nastro 3 a destra della cella modificata

analisi di complessità

ogni sequenza di transizioni relativa ad un’ operazione di RAM costa ad M, nel caso peggiore, un tempo polinomiale nell'ordine della massima lunghezza lungmax del nastro 1

quindi se la RAM opera con costo totale T la MT M opera in tempo O(T p(lungmax))

dimostrazione (RAM→MT)

memoria della RAM

i	mem[i]
1	11
10	
11	
100	1001
101	10
110	
111	
1000	1
1001	
1010	
….	

nastro 1 della MT

$	$	1	$	1	1	$	$	1	0	0	$	1	0	0	1	$	$	1	0	1	$	1	0	$	$	1	0	0	0	$	1	$	$

ogni sequenza di transizioni relativa ad un’ operazione di RAM costa ad M, nel caso peggiore, un tempo polinomiale nell'ordine della massima lunghezza lungmax del nastro 1

dimostrazione (RAM→MT)

memoria della RAM

i	mem[i]
1	11
10	
11	
100	1001
101	10
110	
111	
1000	1
1001	
1010	
….	

nastro 1 della MT

$	$	1	$	1	1	$	$	1	0	0	$	1	0	0	1	$	$	1	0	1	$	1	0	$	$	1	0	0	0	$	1	$	$

ogni sequenza di transizioni relativa ad un' operazione di RAM costa ad M, nel caso peggiore, un tempo polinomiale nell'ordine della massima lunghezza lungmax del nastro 1

quindi se la RAM opera con costo totale T la MT M opera in tempo O(T p(lungmax))
infatti con costo T la RAM ha eseguito al più T istruzioni

dimostrazione (RAM→MT)

nel modello a costi logaritmici

se sul nastro 1 e' presente l' indirizzo i vuol dire che la RAM ha acceduto almeno una volta alla cella di indirizzo i e quindi ha pagato almeno una volta log i

d'altro canto l'indirizzo i occupa su nastro 1 proprio log i celle

se sul nastro 1 compare il contenuto della cella i allora la RAM ha pagato almeno una volta log(mem[i])

d'altro canto mem[i] occupa su nastro 1 proprio log(mem[i]) celle

quindi lungmax≤T

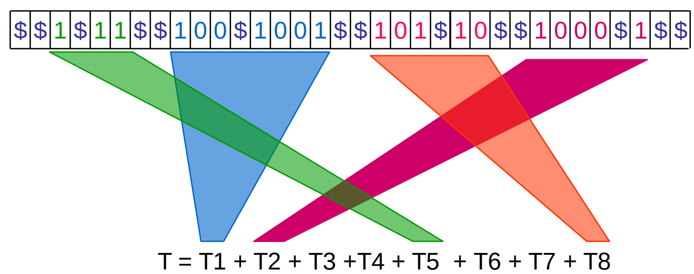
quindi il costo totale e' O(Tp(T))

dimostrazione (RAM→MT)

memoria della RAM

i	mem[i]
1	11
10	
11	
100	1001
101	10
110	
111	
1000	1
1001	
1010	
….	

nastro 1 della MT



RAM e MT

- osservazione: tutto ciò che è calcolabile con una MT è calcolabile anche con una RAM e viceversa
- tesi di Church-Turing: i vari modelli di calcolo introdotti per definire la computabilità hanno al piu' lo stesso potere computazionale delle MT
- osservazione: una MT ed una RAM con modello a costi logaritmici possono simularsi a vicenda in tempo polinomiale
- osservazione: per stabilire se un problema ammette una soluzione calcolabile in tempo polinomiale possiamo indifferentemente far riferimento ad una MT o ad una RAM a costi logaritmici