Teorema. Sia $M=<\{0,1\},ƀ,K,q_0,F,\delta>$ una MT con nastro seminfinito; esistono una RAM ed un programma P tali che se M esegue la computazione $q_0x|\!\!—^*q_Fy$ 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.

Dim.

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).

per ogni stato q_i di M, P contiene due sequenze di istruzioni per ogni stato di M, corrispondenti alle due transizioni $\delta(q_i,1)$ e $\delta(q_i,0)$.

<u>**Esempio**</u>: Se $\delta(q_1,0)=<q_2,1,d>$ e $(q_1,1)=<q_3,0,s>$ allora P contiene il seguente frammento:

```
....................
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
....................
```

Ogni passo di M è simulato da al più 8 istruzioni ognuna con costo $O(\log i_{max})$ dove i_{max} è il massimo numero di celle usate da M

Se M esegue T passi allora $i_{max} \leq T+1$

Quindi, il costo complessivo è $O(T \log T)$

Un analogo teorema può essere dimostrato per una macchina di Turing multinastro M. Se M opera in tempo T la macchina a registri che la simula opera al più in tempo $O(T \log T)$.

Teorema. Data una RAM con programma P che calcola la funzione f esiste una MT M tale che se $f(x_1,...,x_n)=y$ e sul nastro di input sono memorizzati in binario gli interi $x_1,...,x_n$ la macchina M termina con la rappresentazione binaria di y sul nastro di output e se $f(x_1,...,x_n)$ non e' definita M non termina; inoltre se la computazione della RAM ha costo T nel modello a costi logaritmici allora M la simula in $O(T^2)$ passi.

Dim. 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, cioè:

$\#i_1\#R[i_1]\#i_2\#R[i_2]\#i_3\#R[i_3] \ldots \#i_m\#R[i_m]$.

Sul nastro 2 è rappresentato il contenuto dell'accumulatore. Il nastro 3 è usato per il funzionamento di M.

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 il registro su cui si deve operare

  si esegue sul nastro contenente la rappresentazione dell'accumulatore l'operazione prevista

- Istruzioni che modificano il contenuto della memoria STORE, READ

  si trascrive sul nastro 3 il contenuto del nastro 1 a destra del registro da modificare

  si esegue l'operazione modificando sul nastro 1 la rappresentazione del contenuto del registro da modificare

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

Analisi di complessità.

i) Ogni sequenza di transizioni relativa ad una operazione di RAM costa ad M, nel caso peggiore, un tempo dell'ordine della massima lunghezza t_{max} del nastro 1 e se la RAM opera con costo totale T allora $t_{max} \leq T$.

Infatti nel modello a costi logaritmici

- se sul nastro 1 è presente l'indirizzo *i* vuol dire che la RAM ha acceduto almeno una volta al registro R[i] e quindi ha pagato almeno una volta un costo pari a *l(i)*,

- se sul nastro 1 compare il contenuto del registro *i* allora la RAM ha pagato almeno una volta un costo di *l(R[i])*

In definitiva, se la macchina a registri opera con un costo T allora la macchina di Turing opera in tempo $O(T \cdot t_{max})$ e poiché $t_{max} \leq T$ il costo totale è $O(T^2)$