lezione 8 +

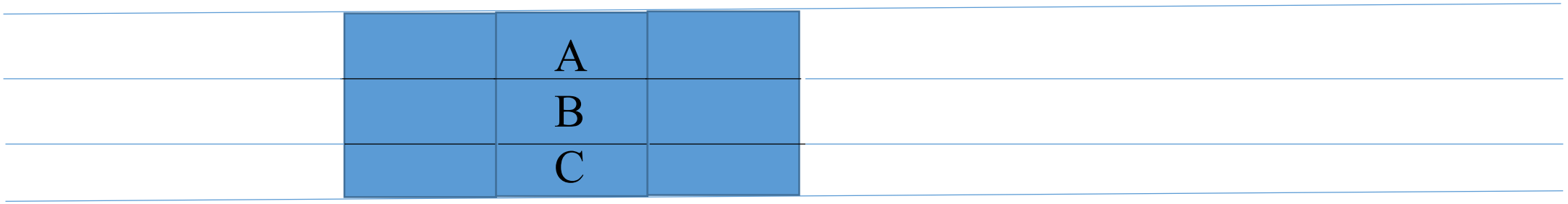
Tecniche di programmazione per le TM

1) memoria (finita) nello stato: $[q, \alpha]$ con α in Γ^+
esempio: riconoscere 01* + 10*

([q0,B],0) =([q0,0],0,R)
([q0,0],1)=([q0,0],1,R)
([q0,0],B)=(qf, B,R)

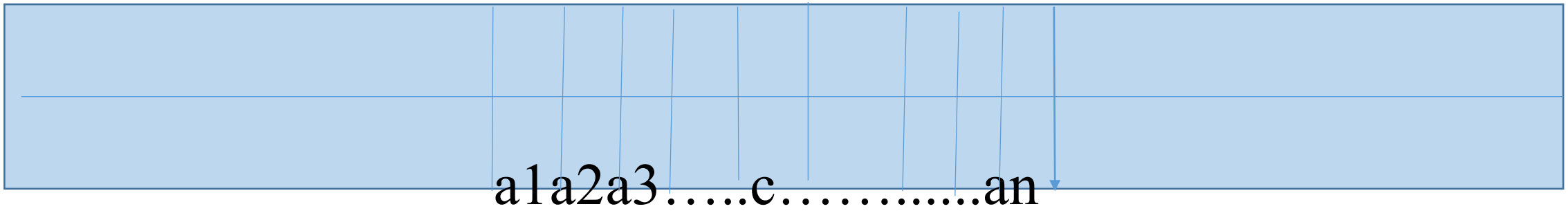
in modo simile si tratta il caso in cui il primo simbolo dell'input è 1

2) tracce multiple



è una sola cella con 3 simboli che possiamo trattare in modo indipendente

riconoscere L={wcw | w in {0,1}+}



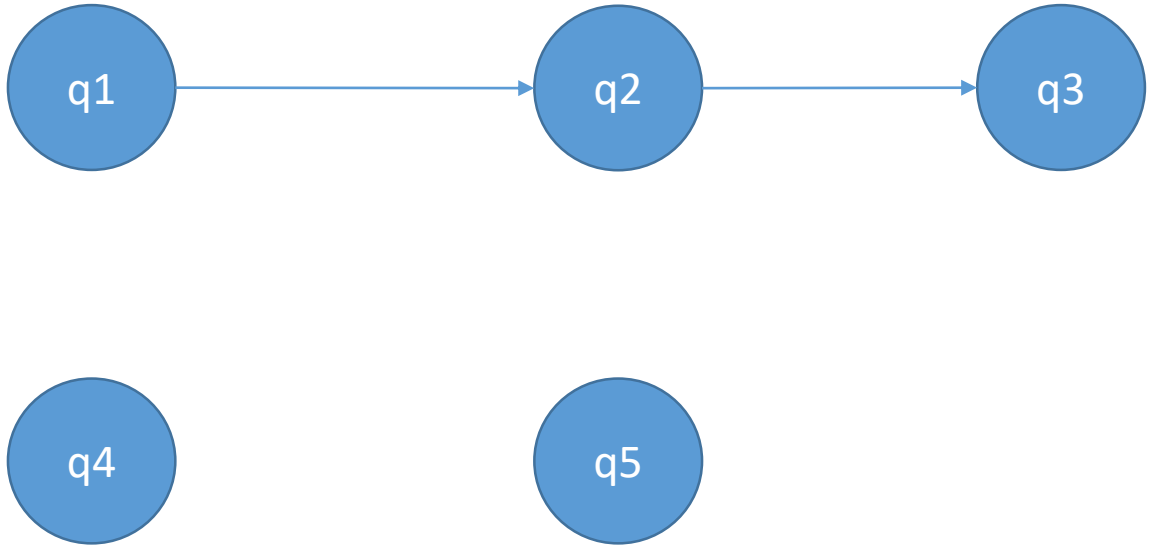
subroutines

moltiplicare interi 0^m10^n1 ➔ 0^{mn}

il nastro occupa $0^{m-k}10^n10^{kn}$

l'idea è di consumare uno 0 a sinistra e di copiare n 0 a destra

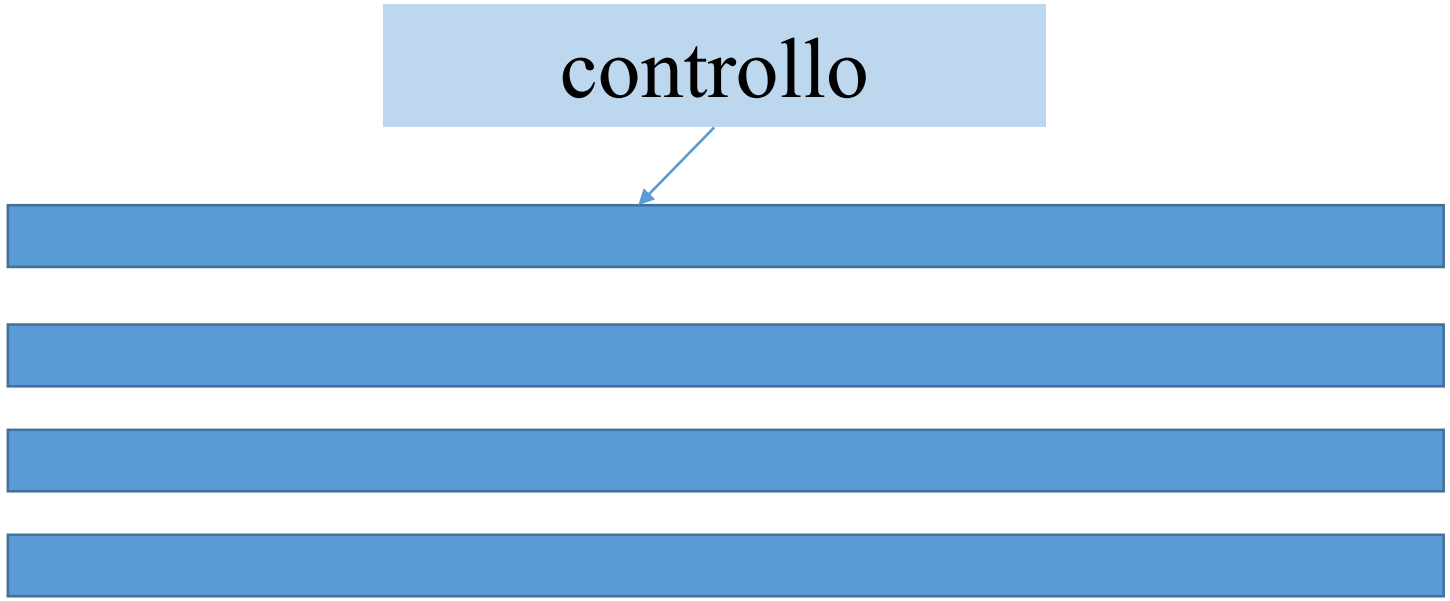
questa operazione la possiamo fare con una subroutine


q1
q2
q3
q4
q5

estensioni del modello base

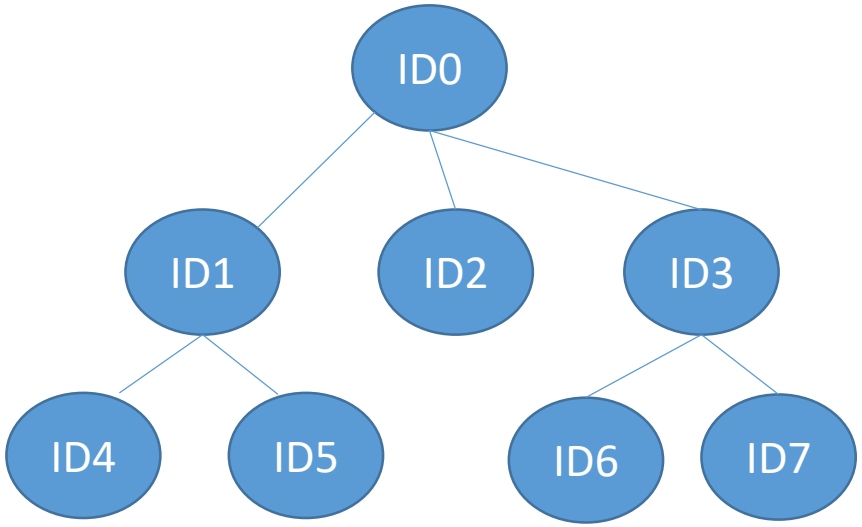
multi nastro => simulabile con multi track

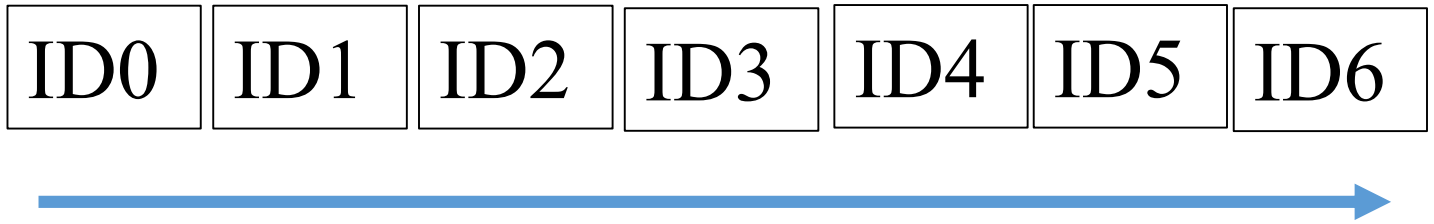
aumento quadratico del tempo di esecuzione !!



nondeterminismo => simulabile da TM deterministica

$\delta(q,X)=\{(q1,Y1,D1), (q2,Y2,D2),\ldots,\}$





si percorre l'albero in larghezza

$1+m+m^2+m^3+\ldots.m^n < n\ m^n$ aumento esponenziale

es. 8.2.4*

calcolo di funzioni <--> linguaggi

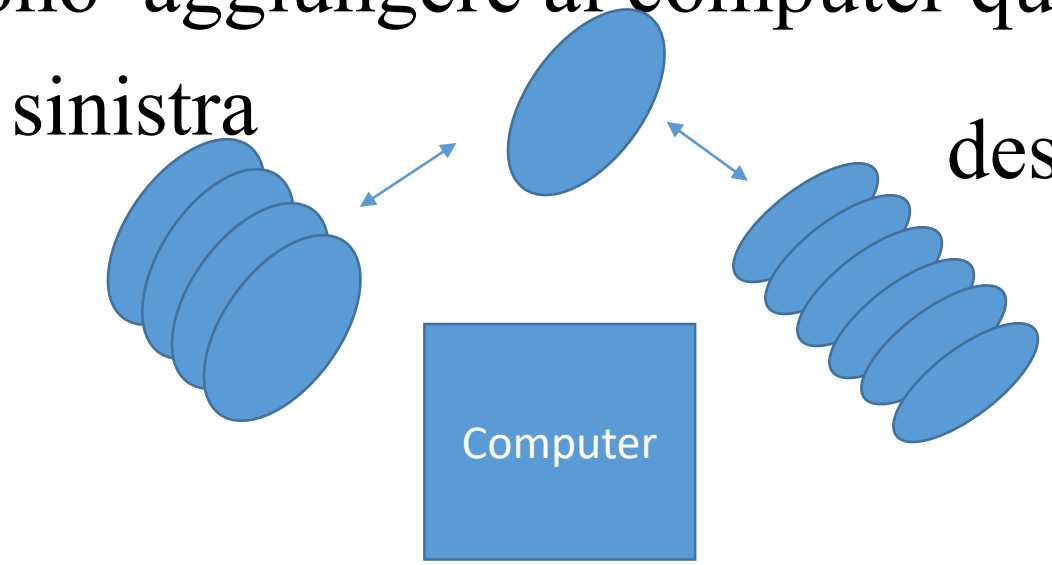
--funzione f (interi positivi) = grafo =[x,f(x)] che è un linguaggio,
TM riconosce il grafo

--una TM calcola f se, dato x sul nastro, si ferma con f(x) sul nastro

TM e computer reali

un computer reale può simulare una TM, unico problema è il nastro potenzialmente illimitato

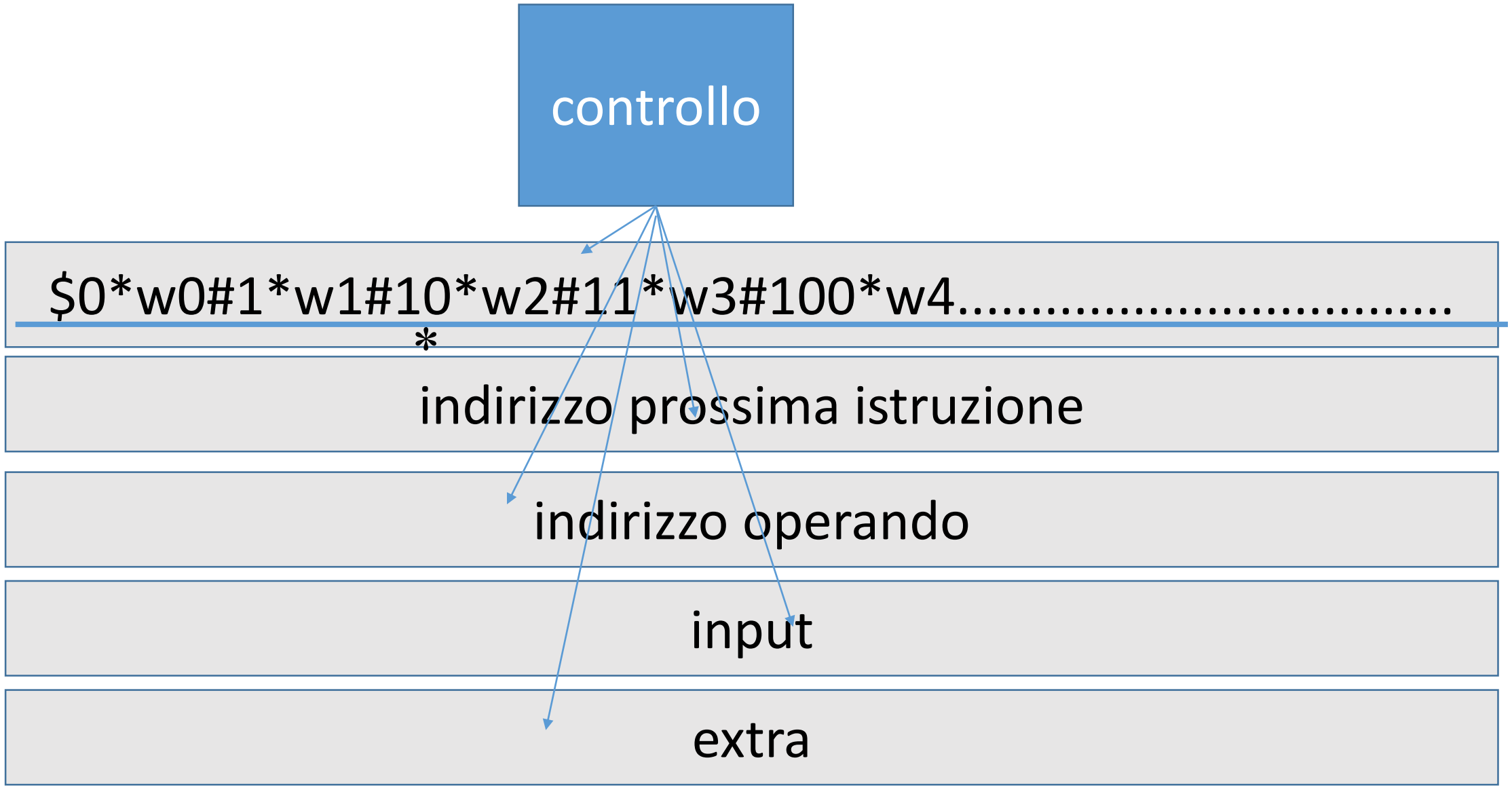
si può simulare con dispositivi di memoria di massa che si possono aggiungere al computer quando serve



modello di computer

a) memoria = sequenza indefinitivamente lunga di parole (di lunghezza variabile) ciascuna con un indirizzo 0,1,2,…
b) il programma è memorizzato in alcune parole della memoria. Ogni istruzione è semplice, tipo assembler, esempi: mettere un valore in una parola della memoria, sommare 2 valori, ecc. Supponiamo che ci sia l'indirizzamento indiretto (puntatori)
c) ciascuna istruzione tocca un n. finito di parole e cambia il valore di al più una parola
d) le operazioni vengono effettuate su qualsiasi parola di memoria (niente registri)

la TM multinastro che simula un Computer



TM simula il ciclo-istruzione del computer:

1.cerca in memoria (primo nastro) l'operazione corrente

2.se l'istruzione contiene un indirizzo, questo viene caricato sul nastro 3 e l'istruzione è marcata con *

3.copiamo il valore sul nastro 3

4. eseguiamo l'istruzione

    -copiare il valore in un altro indirizzo (che si trova nell'istruzione)

    -somma

    -salto nastro 3 -> nastro 2

5. se non si salta, si aggiunge 1 all'indirizzo del nastro 2

visto che i valori aumentano di dimensioni inserire nuovi valori costa tempo proporzionale alla dimensione della memoria

confronto dei tempi d'esecuzione della TM e del computer

--* è un problema: ogni operazione allunga la parola di 1 bit al massimo

--complessità della singola operazione: ogni operazione applicata a parole di dimensione k prende al massimo $O(k^2)$ passi della TM

con queste ipotesi: memoria dopo n passi = n parole e ogni parola occupa n posizioni del nastro
memoria= $O(n^2)$ posizioni del nastro
simulare 1 operazione del computer può richiedere $O(n^2)$

se il computer ci mette n passi la TM ce ne mette $O(n^3)$

relazione polinomiale

per simulare più nastri con 1 solo, si aumenta di un quadrato e quindi $O(n^6)$