Equivalenza tra MT e MTND

Le MTND sono più “efficienti” ma non più potenti computazionalmente delle MT

Teorema. Data una macchina non deterministica M con grado di nondeterminismo $d=\nu(M)$ esiste una MT M_D equivalente che simula k passi di M in $O(kd^k)$ passi

Dim.

L'albero di computazione di M viene visitato in ampiezza da M_D (perché non in profondità?)

M_D ha 3 nastri

nastro 1: contiene l'input
nastro 2: viene usato per generare, in ordine lessicografico, tutte le sequenze finite composte da cifre comprese tra 1 e d
nastro 3: nastro di lavoro

Per ogni sequenza generata sul nastro 2, M_D copia l'input sul nastro 3.

Le transizioni di ogni insieme $\delta_N(q,\sigma)$ sono numerate da 1 a d.

Ogni sequenza di lunghezza s sul nastro 2 è in corrispondenza con una computazione di M di s passi

Gli s numeri di ogni sequenza (compresi tra 1 e d) sono usati per scegliere ad ogni passo una transizione tra le d possibili

Esempio: se s=4 e d=2 e la sequenza è 2122 M_D sceglie per la prima mossa la seconda transizione disponibile, per la seconda mossa la prima, ecc.

Se su qualche foglia dell'albero di computazione di M c'è uno stato finale, allora M_D lo raggiunge in tempo finito altrimenti M_D non raggiunge mai uno stato finale.

Se M termina in *k* passi M_D ha bisogno di

$$O\left(\sum_{j=0}^{k} jd^j\right) = O(kd^k) \text{ passi}$$

Per dimostare che

$$O\left(\sum_{j=0}^{k} jd^j\right) = O(kd^k) \text{ passi}$$

si può procedere come segue. Tenendo conto del fatto che

$$\sum_{j=0}^{k} d^j = (d^{k+1}-1)/(d-1)$$

e derivando si ottiene:

$$\sum_{j=1}^{k} jd^{j-1} = (kd^{k+1}-(k+1)d^k+1)/(d-1)^2$$

e quindi

$$\sum_{j=1}^{k} jd^j = O(kd^k)$$

3.9 IL PROBLEMA DELLA TERMINAZIONE (HALTING PROBLEM)

Data una MT $M=<\Sigma,ƀ,K,q_0,F,\delta>$ sia c_M la codifica di M in Σ. Per $x \in \Sigma^*$ definiamo il predicato della terminazione

$$h(D_M,x) \quad \begin{array}{l} =1 \text{ se M con input x termina} \\ =0 \text{ se M con input x non termina} \end{array}$$

Teorema. Il predicato della terminazione delle macchine di Turing non è T-calcolabile.

NOTA BENE. Invece è T-calcolabile il predicato:

$$h(c_M,x) \quad \begin{array}{l} =1 \text{ se M con input x termina} \\ = \text{indefinito, altrimenti} \end{array}$$

Dim.

Supponiamo che il predicato sia calcolabile, cioè che esista una macchina di Turing H che calcola h. Costruiamo la macchina H' che calcola il predicato

$$h'(c_M) \quad \begin{array}{l} =1 \text{ se M con input } c_M \text{ termina} \\ =0 \text{ se M con input } c_M \text{ non termina} \end{array}$$

H' non è altro che la composizione di due macchine: la prima con input c_M fornisce $c_M ƀ c_M$, la seconda è la macchina H che prende in input e calcola il predicato della terminazione. In altre parole H' è la macchina che verifica se una MT termina quando le viene fornito in input il proprio codice.

Possiamo ora costruire una nuova macchina H'' che prende in input c_M e calcola la funzione:

$$h''(c_M) \begin{array}{l} =0 \text{ se } h'(c_M) = 0 \\ =\text{indefinito altrimenti} \end{array}$$

H'', cioè, termina con 0 se H' si è fermata con 0 e si mette a ciclare, se H' si è fermata con 1. Cosa accade ora se calcoliamo $h''(c_{H''})$:

$$h''(c_{H''}) \begin{cases} = \text{indefinita se } h''(c_{H''}) \text{ è definita} \\ = 0 \text{ se } h''(c_{H''}) \text{ è indefinita} \end{cases}$$

In ogni caso abbiamo una contraddizione. Quindi non può esistere la macchina H.