Efficienza di un Algoritmo

Wed, 9 Mar

Tempo

Per definire **l'efficienza** di un *algoritmo* per quanto riguarda il **tempo di esecuzione**, si definiscono dei *limiti*: un *valore massimo*, il **caso peggiore**, il tempo *più lungo* che l'algoritmo può impiegare, e un *valore minimo*, il **caso migliore**, il tempo *più breve* che l'algoritmo può impiegare.

Il **tempo medio** *non* è la *media* tra questi due valori, ma il tempo che l'algoritmo impiega in una **situazione di input standard**.

Per **pesare** questi valori, prendiamo in considerazione n, il **numero di input**.
È possibile ricavare una **funzione del tempo di esecuzione** su n per *descrivere* il tempo di esecuzione di un algoritmo.

Esempio di valutazione dei tempi di esecuzione

Calcolare i tempi di esecuzione di un algoritmo di ricerca sequenziale.

- **Algoritmo di ricerca sequenziale**
  - `int v[], int k, int n`
    v[ ] è un vettore di n elementi, k è l'elemento da cercare

  `int ric_seq(int v[], int k)`

  `begin`

  - `p = 1` — $c1 \cdot 1$
    la posizione iniziale è convenzionalmente 1, mentre quella finale è n

  `while(v[p] ≠ k) AND (p ≤ length(v))` — $c2 \cdot (t_w + 1)$

  - `p++` — $c3 \cdot t_w$
    *t_w sta per **true(while)**, ovvero per quando la condizione dell while è veriticata*

  `if(p ≤ length(v))` — $c4 \cdot 1$

  - `return(p)` — $c5 \cdot t_i$
    *t_i sta per **true(if)**, ovvero per quando la condizione dell'if è verificata*

  `else`

  - `return(-1)` — $c6 \cdot f_i$
    *f_i sta per **false(if)**, ovvero per quando la condizione dell'if non è verificata*

  `end`

- Il **caso migliore**, quando viene eseguito il *numero minore* di *istruzioni*, si verifica quando k si trova in prima posizione, ovvero eseguendo **4** istruzioni.
- Il **caso peggiore**, quando viene eseguito il *numero massimo* di *istruzioni*, si verifica quando k non è presente nel vettore, ovvero eseguendo **3+(2n)+1 ≈ n** istruzioni.
- il **tempo medio**, con un *input standard*, si verifica quando k si trova al *centro* del vettore, *in posizione* $n/2$, ovvero eseguendo **3+n** istruzioni.

> La probabilità di trovare k in una qualunque delle n posizioni del vettore è la stessa, il caso medio si verifica quando k si trova in posizione centrale, dato che è equiprobabile che si trovi all'inizio o alla fine.

Ad ogni istruzione, assegnamo un **tempo di esecuzione** *costante* (c1, c2...) *moltiplicato* per il *numero di volte* che tale istruzione viene *eseguita*.

Possiamo ottenere un **tempo generico di esecuzione** *sommando* le costanti assegnate:

$$Trs(n) = c1 + c2(t_w + 1) + c3t_w + c4 + c5t_i + c6f_i$$

Il **caso migliore** si ottiene per $t_w = 0$ → $t_i = 1, f_i = 0$

$$Tm(n) = c1 + c2 + c4 + c5$$

Il **caso peggiore** si ottiene per $t_w = n$ → $t_i = 0, f_i = 1$

- $Tp(n) = c1 + c2(n+1) + c3n + c4 + c6$

  $(c2 + c3)n + (c1 + c2 + c4 + c6)$

  [con $c = 1$] $2n + 4 \approx 2n \approx n$