Complessità computazionale

→ uno dei modi per catalogare gli algoritmi in base alla loro efficienza ne misura la velocità rispetto all'input.

→ un problema può avere dei suoi limiti computazionali, ossia può "richiedere un numero minimo di operazioni".

Un problema è della forma:

$$\pi : \{0,1\}^* \longrightarrow \{0,1\}^*$$

Def. π ha un limite superiore $O(f(n))$ se $\exists$ algoritmo che lo risolve in $O(f(n))$.
π ha un limite inferiore $O(f(n))$ se ogni algoritmo che lo risolve è $O(f(n))$.

→ $f(n) = \Omega(g(n)) \overset{def}{\Longleftrightarrow} \exists k>0 \; \exists n_0>0 \; \forall n \geq n_0, \; g(n) \leq k f(n)$.

In altre parole essere $\Omega(f(n))$ vuol dire avere caso ottimo $f(n)$.

$\rightarrow \quad \underbrace{f(n) = \Theta(g(n)) \overset{\text{def}}{\Longleftrightarrow} f(n) = O(g(n)) \text{ e } g(n) = O(f(n)).}_{\text{CASO MEDIO}}$

$\rightarrow \quad f(n) = O(g(n)) \text{ e } f(n) = \Omega(g(n)) \Longrightarrow$
$\Longrightarrow f(n) = \Theta(g(n))$ [infatti
$f(n) = \Omega(g(n)) \Longleftrightarrow g(n) = O(f(n))$].

Se un problema ha sia limite sup. che inf. coincid., si dice che ha complessità $\Theta(f(n))$.

Alcuni problemi richiedono almeno la lettura dell'input, e spesso questo permette già di dare un limite inferiore banale (fooling argument)

esempio: un ordinam. di un array richiede sicuram. la lettura degli elementi, quindi il prob. ha sicuram. limite inf. $\Omega(n)$.

In realtà possiamo dare una stima migliore per l'ordinam. per confronto, mostrando che è $\Omega(n \cdot \log n)$ (vedremo poi che è esattam. $\Theta(n \cdot \log n)$ mostrando il merge sort)

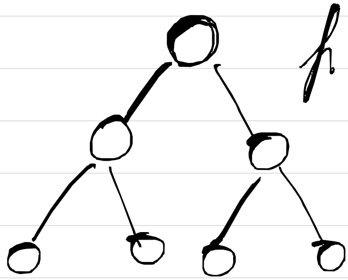
Considerati due elementi $A[i], A[J]$, l'esito del loro confronto può essere $<, >$ o $=$.

Il problema prende in input un array di n elementi e ne restituisce la permut. t.c. $A[\sigma(1)] \leq A[\sigma(2)] \leq \ldots$

Ogni confronto permette di distinguere tra tre casi, e ogni altro confronto al più triplica queste distinzioni. L'albero decisionale ha almeno $n!$ foglie, quindi — poiché t, che è il numero di confr. eff., è l'altezza dell'albero — deve valere:

$$3^t \geq n! \Longrightarrow t = \Omega(\log(n!)) = \Omega(n \log(n))$$

→ si è usato che in un K-albero da n foglie e altezza h, $K^h \geq n$.



$f(h) \geq 2 f(h-1)$

$f(\cdot)$ — foglie ad altezza.