# COMPLESSITÀ DEI PROBLEMI

Un algoritmo corretto per un problema computazionale deve terminare sempre e produrre un output che corrisponde all'istanza dell'input. Ci sono infiniti algoritmi corretti e di ognuno si possono calcolare la complessità asintotica. L'obiettivo è di classificare i problemi in base alla loro difficoltà, cioè la quantità di risorse necessario per risolverli. Ad ogni problema bisogna associare l'algoritmo più efficiente che lo risolve, senza però considerare tutti gli infiniti algoritmi.

## Complessità $O(f(n))$ di un problema

Se un algoritmo risolve un problema con complessità temporale $O(f(n))$, spendendo $O(f(n))$ o meno risorse (meno perché potrebbe esistere un algoritmo con costo minore che non conosciamo).

$f(n)$ è limitato superiormente.

## Complessità $\Omega(f(n))$ di un problema

Se ogni algoritmo che risolve un problema ha complessità temporale $\Omega(f(n))$.

È la spesa minima che si può avere per risolvere un problema, anche se non è detto che un problema sia risolvibile $O(f(n))$. $f(n)$ è un limite inferiore. Non possiamo considerare tutti gli algoritmi che lo risolvono, ragionando sulle istanze più difficili.

## Complessità $\Theta(f(n))$ di un problema

Se un algoritmo ha contemporaneamente complessità temporale $O(f(n))$ e $\Omega(f(n))$; esiste almeno un algoritmo che lo risolve e non è possibile risolvere un problema spendendo meno di $O(f(n))$. Il limite inferiore e superiore coincidono poiché $f(n)$ è la complessità intrinseca del problema, anche se può capitare che sia ignota e quindi non si può risolvere.

Se un problema è $\Theta(f(n))$, anche l'algoritmo è $\Theta(f(n))$, ma non viceversa.

## Problemi della complessità ignota

**Problema del commesso viaggiatore**: trovare il circuito più breve che trova n città.

**Upper – bound**: esiste un algoritmo che ha complessità $O(n^2 2^n)$.

**Lower – bound**: il problema è $\Omega(f(n))$; non è mai stato dimostrato che il problema non possa essere risolto in tempo polinomiale e lineare. Se esistesse un algoritmo che risolve il problema in tempo polinomiale, esisterebbero algoritmi per risolvere tutti i problemi appartenenti ad una classe (NP) in tempo polinomiale.