Algoritmica 07/08
2. CRESCITA DELLE FUNZIONI

In informatica si usa la seguente notazione asintotica per indicare il possibile ordine di grandezza di una funzione. La variabile indipendente n è in genere un intero positivo e indica la "dimensione" dei dati di un problema (per es. il numero di bit necessario a descrivere i dati). La funzione di cui si studia l'ordine di grandezza è in genere proporzionale al tempo di calcolo (complessità in tempo), oppure alla memoria impiegata oltre a quella necessaria ai dati d'ingresso (complessità in spazio).

Notazione Θ

$\Theta(g(n)) = \{f(n)$: esistono tre costanti positive c_1, c_2, n_0, tali che $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ per ogni $n \ge n_0\}$

Cioè le funzioni f(n) e g(n) hanno lo stesso andamento al crescere di n a meno di costanti moltiplicative, e contano solo i termini di ordine massimo. Per esempio $f(n) = n^2-3n+1$ è di ordine $\Theta(n^2)$.

La notazione Θ si impiega per esempio per indicare il tempo di un algoritmo di cui si conosce compiutamente il comportamento e che, a pari valore di n, si comporta allo stesso modo per tutti gli insiemi di dati di dimensione n che gli si presentano.

Notazione O

$\mathbf{O}(g(n)) = \{f(n)$: esistono due costanti positive c, n_0, tali che $0 \le f(n) \le cg(n)$ per ogni $n \ge n_0\}$

Cioè la funzione f(n) ha un andamento che non sale al di sopra di g(n) al crescere di n, a meno di una costante moltiplicativa. Anche qui contano solo i termini di ordine massimo. Per esempio $f(n) = n^2-3n+1$ è di ordine $\mathbf{O}(n^2)$, ma anche $\mathbf{O}(n^3)$ ecc..

La notazione O si impiega per esempio per indicare il tempo di un algoritmo di cui **non** si conosce compiutamente il comportamento, ma che si sa che non può superare g(n); oppure che **non** si comporta allo stesso modo per tutti gli insiemi di dati di dimensione n che gli si presentano, ma per alcuni richiede tempo $\Theta(g(n))$, per altri meno.

Notazione Ω

$\Omega(g(n))$ = {f(n): esistono due costanti positive c, n_0, tali che $0 \leq cg(n) \leq f(n)$ per ogni $n \geq n_0$}

Cioè la funzione f(n) ha un andamento che non scende al di sotto di g(n) al crescere di n, a meno di una costante moltiplicativa. Anche qui contano solo i termini di ordine massimo. Per esempio $f(n) = n^2-3n+1$ è di ordine $\Omega(n^2)$, ma anche $\Omega(n \log n)$ ecc..

La notazione Ω si impiega per esempio per indicare il limite inferiore al tempo di soluzione di un problema, che si applica quindi a *tutti* i suoi algoritmi di soluzione.

Altre notazioni

Ne ricordiamo solo una, che indica un limite non stretto. Abbiamo detto per esempio che $f(n) = n^2-3n+1$ è di ordine $O(n^2)$, ma anche di ordine $O(n^3)$. Il primo ordine è *stretto*, il secondo non lo è. Naturalmente se la f(n) è nota non ha senso usare $O(n^3)$, ma può essere utile indicare che f(n) è di ordine inferiore a questo. Usiamo a questo proposito l''ordine **o**:

Notazione o

$\mathbf{o}(g(n))$ = {f(n): per qualsiasi costante positiva c, esiste una costante $n_0>0$ tale che $0 \leq f(n) < cg(n)$ per ogni $n \geq n_0$}

Le definizioni di **O** e **o** sono simili, ma la seconda vale *per qualsiasi* costante c. Per esempio $f(n) = n^2-3n+1$ è di ordine $\mathbf{o}(n^3)$.

Vedere esempi nel testo **B.**