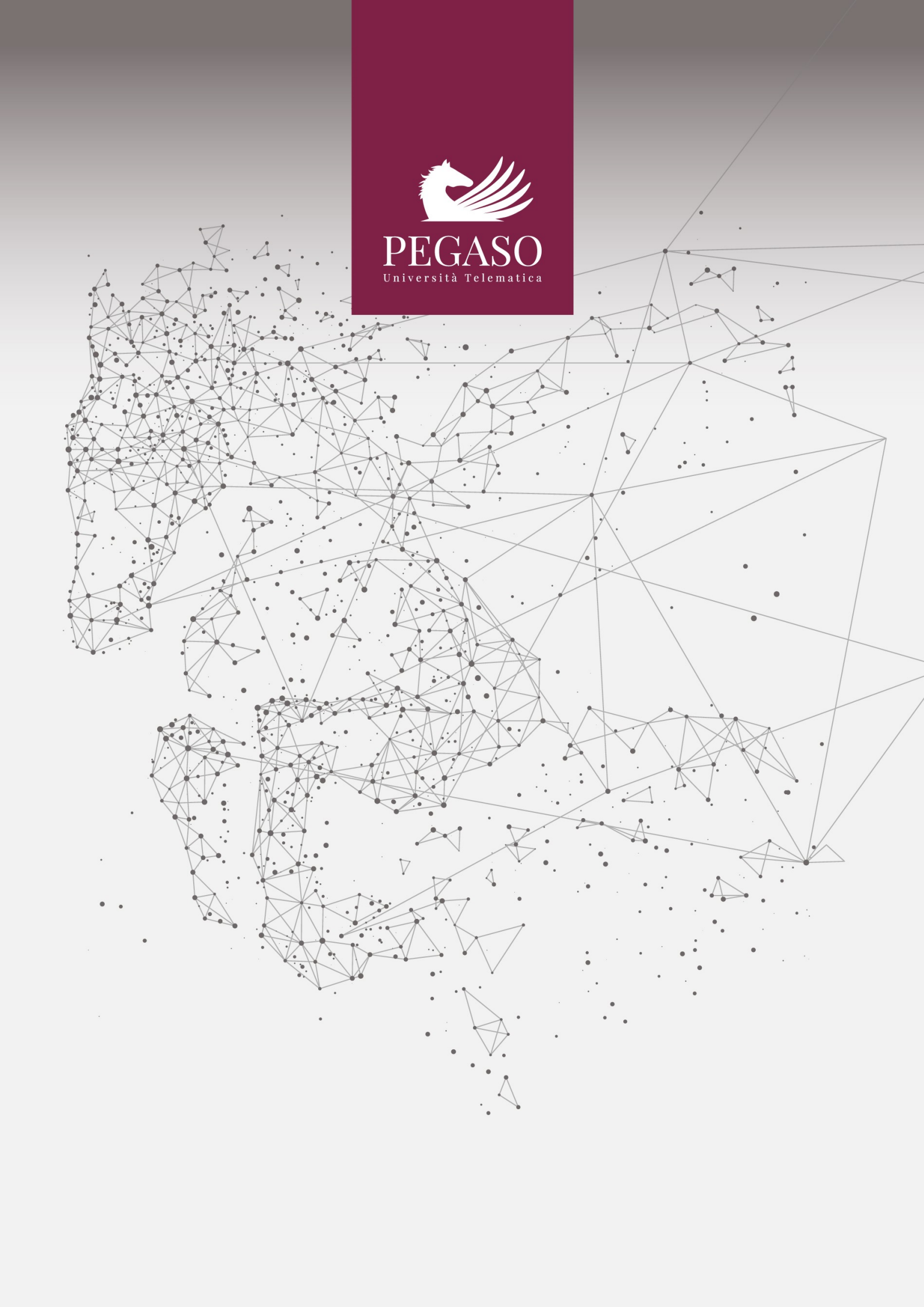
PEGASO
Università Telematica

Indice

1. Notazione asintotica

Per risolvere un problema sono spesso disponibili algoritmi diversi; un criterio usato per misurare la "bontà" di un algoritmo è la quantità di risorse usate per il calcolo dello stesso. Sebbene la risorsa temporale sia di interesse dominante, anche lo spazio è talvolta cruciale, specialmente se si parla di "grande quantità di spazio" da allocare (sia per memorizzare che per manipolare i dati).

Concentriamo la nostra attenzione sul tempo di calcolo: è necessario utilizzare una misura "astratta", dal momento che esprimerlo in "secondi" sarebbe impossibile, essendo questo dipendente dai dati in ingresso, dal linguaggio utilizzato, dal compilatore e dalla potenza della macchina.

Il tempo viene dunque espresso come costo complessivo delle operazioni elementari in funzione della dimensione dei dati in ingresso. Per operazioni elementari consideriamo:

- Operazioni aritmetiche.
- Operazioni logiche.
- Operazioni di confronto.
- Operazioni di assegnazione.

Nella valutazione del tempo di calcolo, ognuna delle operazioni elementari avrà pertanto un costo che è valutato tramite dei valori c_i che in generale sono tutti differenti tra loro.

Si arriva pertanto facilmente ad avere dei tempi di calcolo della forma:

$$T(n) = c_1 + c_2 n + c_3(n-1) + \cdots$$

Cioè funzioni in cui abbiamo diverse costanti c_i e valori n che rappresentano la grandezza dell'input in esame.

In questo tipo di valutazione è molto complesso valutare con esattezza le costanti coinvolte; pertanto, tale problematica viene superata valutando il numero di operazioni in "ordini di grandezza", cioè esprimendolo come limitazione della funzione T(n) al tendere di n all'infinito (cioè al tendere della dimensione dell'input all'infinito) e trascurando inoltre le costanti moltiplicative.

Parliamo di "complessità asintotica" (e dunque di "notazione asintotica" in ordine di grandezza) di una funzione, quando si vuole stimare quanto aumenta il tempo di calcolo al crescere della dimensione n dell'input.

Tale notazione ci consente inoltre di comparare il tasso di crescita (cioè, il comportamento asintotico) di una funzione nei confronti di un'altra.

Esistono tre notazioni asintotiche:

- **Notazione asintotica O** (notazione O grande): limite superiore asintotico.
- **Notazione asintotica Ω** (notazione Omega): limite inferiore asintotico.
- **Notazione asintotica θ** (notazione Theta): limite asintotico stretto.

2. Notazione asintotica O

La notazione asintotica O è il limite superiore asintotico.

Date due funzioni $f(n)$ e $g(n)$, si dice che $f(n) = O(g(n))$ se esiste un valore $c > 0$ tale che $0 \leq f(n) \leq c \cdot g(n)$ per ogni $n \geq n_0$.

Alcune riflessioni:

- se la costante c esiste, allora (da un certo n_0 in poi) $g(n)$ rappresenta una limitazione superiore per $f(n)$.
- La costante è di fatto "assente" dalla notazione (si dice infatti che $f(n) = O(g(n))$ e nella scrittura non inseriamo la costante) perché la sua esistenza mi assicura la limitazione superiore e non sono interessato però a conoscere tale valore.
- Per ogni $n \geq n_0$ implica il fatto che "da un certo punto in poi" viene rispettata la condizione e che prima di n_0 può anche non venir soddisfatta (quindi è possibile che prima di n_0 la funzione $f(n)$ non sia limitata superiormente da $g(n)$.
- Occorre sempre ragionare in termini di "spazio di funzioni", cioè quando parliamo di O grande stiamo di fatto parlando dello spazio di "tutte le funzioni" che limitano superiormente la nostra $f(n)$; sarebbe più corretto scrivere $f(n) \in O(g(n))$.

Un esempio:

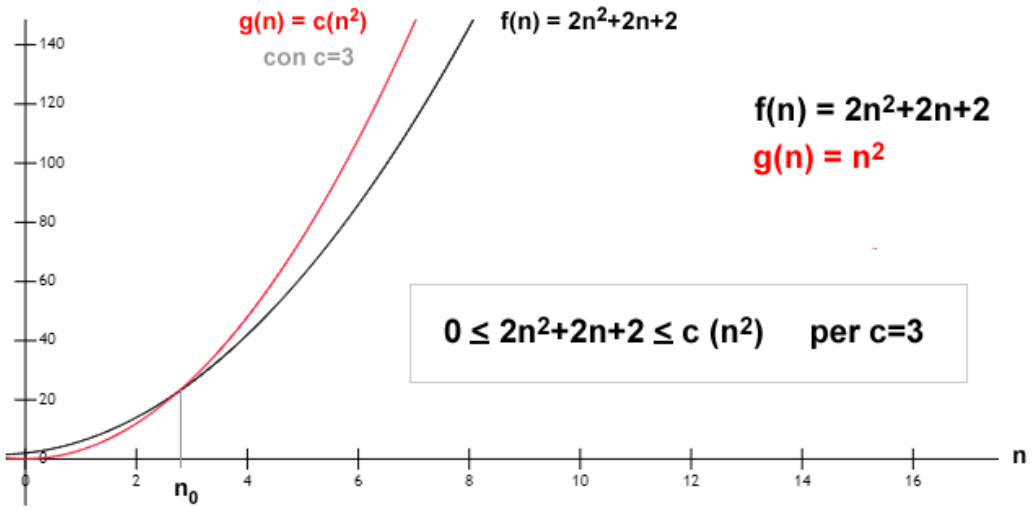

Figura 1 *- Limite Asintotico Superiore*

In tali circostanze la $f(n)$ è una delle funzioni dell'insieme $O(g(n))$.

Nel caso specifico della figura possiamo notare come la funzione:

$$f(n) = 2n^2 + 2n + 2$$

Sia limitata superiormente da:

$$g(n) = 3n^2$$

Cioè la $g(n)$ con costante $c = 3$

Pertanto, si può scrivere che $f(n) = O(n^2)$ avendo eliminato da costante c

In generale una funzione $a_1n^2 + a_2n + a_3$ è sempre $O(n^2)$

3. Notazione asintotica Ω

La notazione asintotica Ω è il limite asintotico inferiore. Date due funzioni $f(n)$ e $g(n)$, si dice che $f(n) = \Omega(g(n))$ se esiste un valore $c > 0$ tale che $0 \leq c \cdot g(n) \leq f(n)$ per ogni $n \geq n_0$.

Alcune riflessioni:

- se la costante c esiste, allora (da un certo n_0 in poi) $g(n)$ rappresenta una limitazione inferiore per $f(n)$.
- La costante è di fatto "assente" dalla notazione (si dice infatti che $f(n) = \Omega(g(n))$ e nella scrittura non inseriamo la costante) perché la sua esistenza mi assicura la limitazione inferiore e non sono interessato però a conoscere tale valore.
- Per ogni $n \geq n_0$ implica il fatto che "da un certo punto in poi" viene rispettata la condizione e che prima di n_0 può anche non venir soddisfatta (quindi è possibile che prima di n_0 la funzione $f(n)$ non sia limitata inferiormente da $g(n)$.
- Occorre sempre ragionare in termini di "spazio di funzioni", cioè quando parliamo di Ω grande stiamo di fatto parlando dello spazio di "tutte le funzioni" che limitano inferiormente la nostra $f(n)$; sarebbe più corretto scrivere $f(n) \in \Omega(g(n))$.

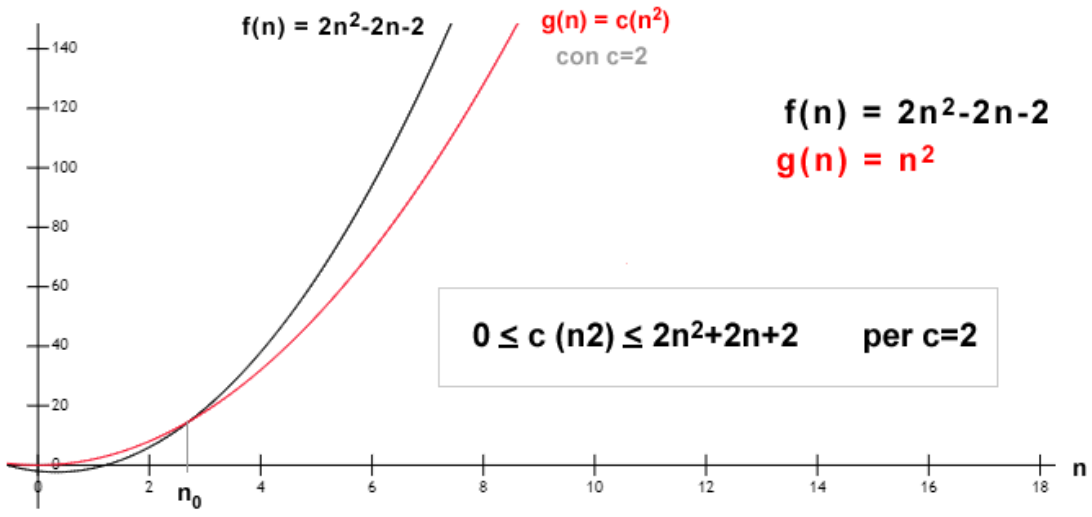

Figura 2 *- Limite asintotico inferiore*

In tali circostanze la $f(n)$ è una delle funzioni dell'insieme $\Omega(g(n))$.

Nel caso specifico della figura possiamo notare come la funzione:

$$f(n) = 2n^2 + 2n + 2$$

Sia limitata inferiormente da:

$$g(n) = 2n^2$$

Cioè la $g(n)$ con costante $c = 2$

Pertanto, si può scrivere che $f(n) = \Omega(n^2)$ avendo eliminato da costante c

4. Analisi O ed Ω

Occorre fare attenzione al fatto che "in generale" O ed Ω non sono la stessa cosa, sebbene negli esempi analizzati in precedenza "sembrerebbe" che coincidano.

Consideriamo il caso della funzione $f(n) = \log_2 n$ e la funzione $g(n) = n$

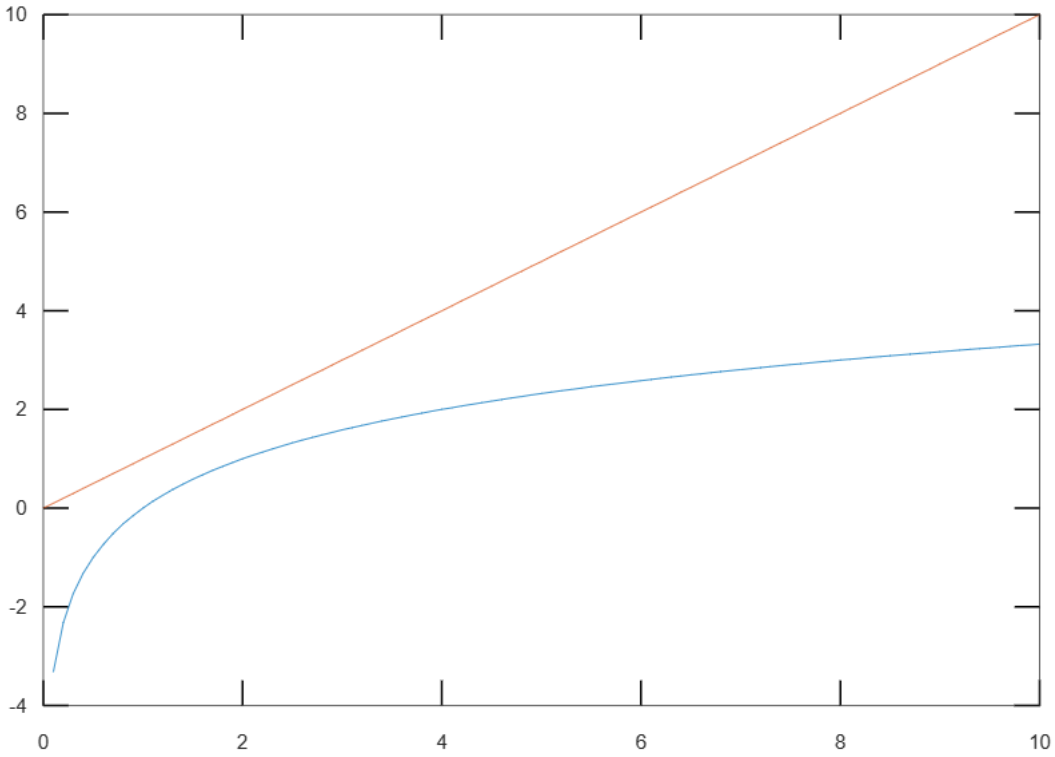

Figura 3 *- Log vs Linear*

In blu è rappresentata la funzione $f(n) = \log_2 n$ mentre in rosso la funzione $g(n) = n$

Quello che possiamo affermare è che $f(n) = \log_2 n = O(n)$

Procediamo per induzione:

se $n = 1$ allora $f(0) = \log_2 0 = 1$ e $g(0) = 1$ per $c = 1$ (ad es.)

Assumendo dunque che $f(n) = \log_2 n \leq n$ possiamo dimostrare che

$f(n+1) = \log_2(n+1) \leq n+1$

Possiamo maggiorare la funzione $\log_2(n+1)$ con $\log_2(2n)$, cioè:

$$\log_2(n+1) \leq \log_2(2n) = \log_2(2) + \log_2(n) \leq n+1$$

Tuttavia, non è vero che $\log_2(n) \geq c * n$ per nessun $n \geq n_0$ e per nessuna scelta di c o n_0

Quindi possiamo concludere che $O(n) \neq \Omega(n)$

La funzione logaritmica è dunque limitata superiormente da una funzione lineare ma tale funzione non limita il logaritmo inferiormente e dunque $O(n) \neq \Omega(n)$

5. Notazione asintotica θ

La notazione asintotica θ è il limite asintotico stretto. Date due funzioni $f(n)$ e $g(n)$, si dice che $f(n) = \theta(g(n))$ se esistono due valori $c1 > 0$ e $c2 > 0$ tali che $0 \leq c1 \cdot g(n) \leq f(n) \leq c2 \cdot g(n)$ per ogni $n \geq n_0$.

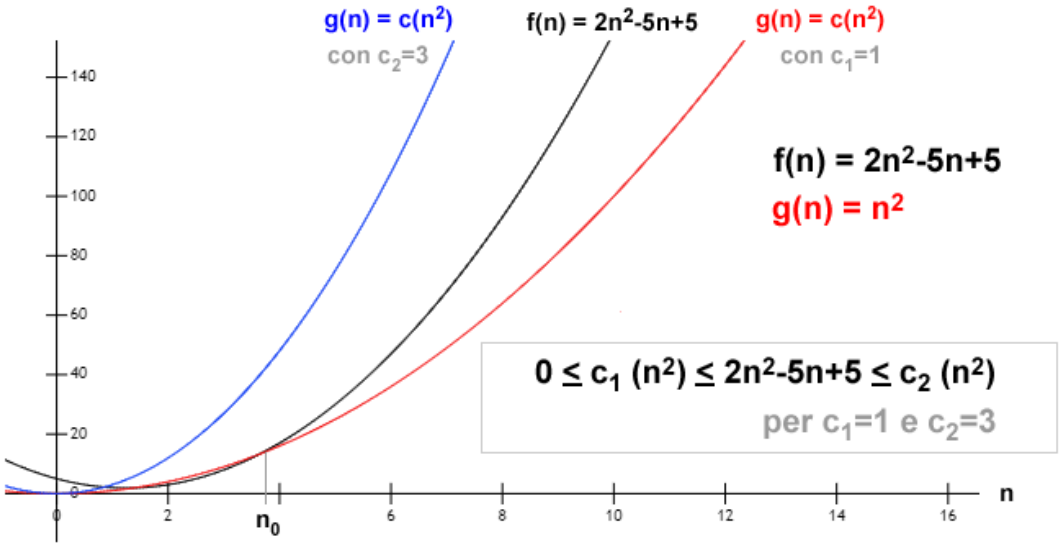

Figura 4 *- Limite asintotico stretto*

Quindi la $f(n)$ è una delle funzioni dell'insieme $\theta(g(n))$.

Se una funzione è $\theta(g(n))$ allora è anche $O(g(n))$ e $\Omega(g(n))$ perché esiste sia un limite asintotico superiore che inferiore. E viceversa, se una funzione è sia $O(g(n))$ che $\Omega(g(n)$ allora è anche $\theta(g(n)$.

Bibliografia

- Alan Bertossi, Alberto Motresor: Algoritmi e strutture di dati, Città Studi Edizioni, terza edizione.
- C. Demetrescu, I. Finocchi, G. F. Italiano: Algoritmi e strutture dati, McGraw-Hill, seconda edizione.
- Crescenzi, Gambosi, Grossi: Strutture di Dati e Algoritmi, Pearson/Addison-Wesley.
- Sedgewick: Algoritmi in C, Pearson, 2015.