

# Algoritmi e Strutture Dati

## Capitolo 2

Modelli di calcolo e  
notazione asintotica

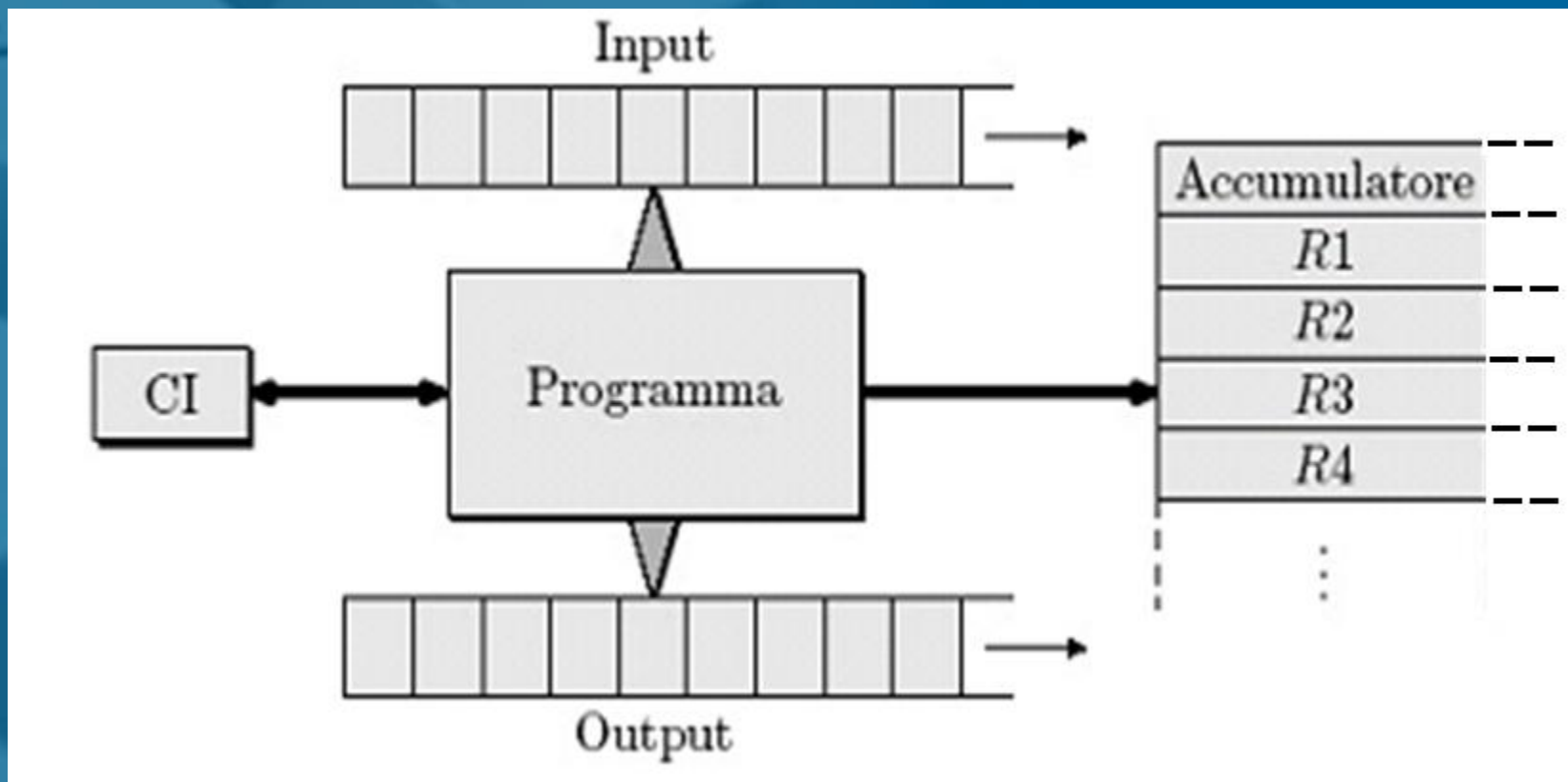
# Modello di calcolo

- Per valutare la **complessità temporale** dei vari algoritmi Fibonacci, abbiamo pedissequamente contato le **linee di pseudocodice** di volta in volta mandate in esecuzione
- Tuttavia, contare il numero di linee dello pseudocodice di un algoritmo non è sufficiente a comprenderne la complessità temporale: quali **operazioni reali** si celano dietro lo pseudocodice???
- Anche l'analisi dello **spazio utilizzato** è stata piuttosto arbitraria: ho ipotizzato che le celle di memoria utilizzate avessero capacità infinita!
- Per valutare la **complessità** di un algoritmo in modo oggettivo, bisogna quindi stabilire un **modello di calcolo** di riferimento su cui esso viene eseguito, ovvero una **macchina astratta** sulla quale si definisce **a priori** l'insieme delle **operazioni ammissibili ed eseguibili** durante una computazione, specificandone i relativi **costi** (in termini di **tempo** e **spazio** utilizzato)
- Alcuni modelli di calcolo famosi: **Macchina di Turing, Automi a Stati Finiti, Funzioni Ricorsive, Macchina a Registri**, etc... (sono tutti modelli di calcolo **equivalenti**, cioè in grado di calcolare le stesse funzioni)

# Il nostro modello di calcolo: la RAM

- La RAM (*Random Access Machine*) è un tipo particolare di macchina a registri (locazioni di memoria ad accesso diretto)
- La RAM è definita da:
  - un nastro di ingresso e uno di uscita, ove saranno scritti l'input e l'output, rispettivamente
  - una memoria ad accesso diretto strutturata come un array (di dimensione infinita) in cui ogni cella può contenere un valore intero arbitrariamente grande
  - un programma finito di istruzioni elementari
  - un registro detto accumulatore (contiene gli operandi dell'istruzione corrente)
  - un registro detto contatore delle istruzioni (contiene l'indirizzo dell'istruzione successiva)
- La RAM è un'astrazione dell'architettura di von Neumann

# La RAM



# Criterio di costo uniforme

- Nel modello **RAM a costi uniformi**, le seguenti **istruzioni elementari** hanno un costo **unitario**
  - istruzione ingresso/uscita (lettura /scrittura del nastro di input/output)
  - operazione aritmetico (+,-,×,÷,√, elevamento a potenza, etc.) /logica (**test**)
  - accesso/modifica del contenuto della memoria
- **Complessità temporale** **tempo(I)** misurata come **numero di istruzioni elementari** eseguite sull'istanza di input **I**
- **Complessità spaziale** **spazio(I)** misurata come **numero massimo di celle di memoria** della RAM occupate durante l'esecuzione sull'istanza di input **I**

# Dimensione dell'input

- Misureremo le risorse di calcolo usate da un algoritmo (tempo di esecuzione / occupazione di memoria ) in funzione della **dimensione** dell'istanza **I** in input
- Esistono due modalità di **caratterizzazione** della dimensione dell'input:
  - **Quantità di memoria effettiva** utilizzata per codificare l'input (ad esempio, **numero di bit** necessari per rappresentare un valore in input, come nel problema di Fibonacci)
  - **Parametrizzazione** della dimensione dell'input (ad esempio, **numero di elementi** di una sequenza da ordinare)
- In tutti i problemi che incontreremo in futuro, la dimensione dell'input sarà **parametrizzata** attraverso il numero **n** di elementi dell'istanza

# Notazione asintotica

# Notazione asintotica e operazioni dominanti

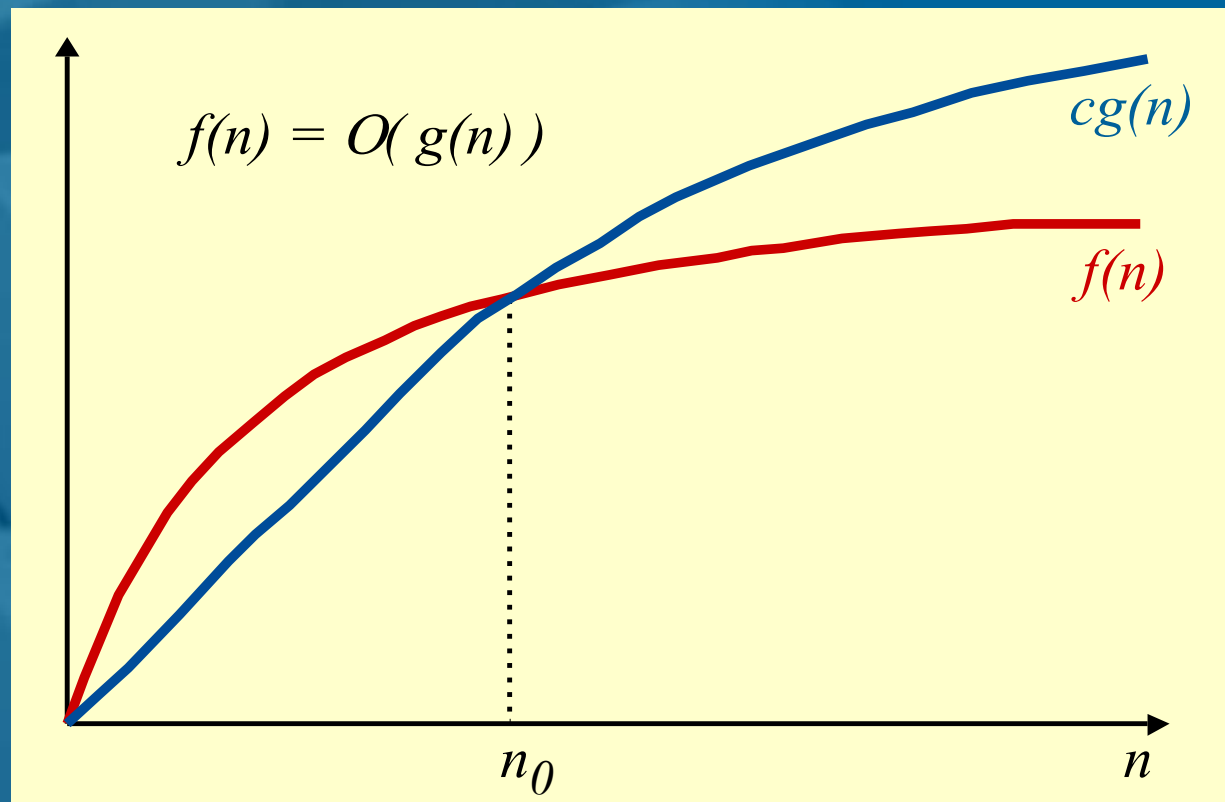
- Complessità temporale e spaziale saranno espresse in **notazione asintotica rispetto alla dimensione dell'input**
- La notazione asintotica è un' **astrazione** utile per descrivere l'ordine di grandezza di una funzione ignorando i dettagli non influenti, come **costanti moltiplicative** e **termini di ordine inferiore**
- **Semplificazione:** Utilizzando l'analisi asintotica, non dovremo andare a conteggiare tutte le operazioni eseguite, ma sarà sufficiente considerare le cosiddette **operazioni dominanti**, ovvero quelle che nel **caso peggiore** vengono eseguite più spesso; queste si trovano **annidate** nei cicli più interni dello pseudocodice che descrive l'algoritmo
- **NOTA:** Nel prosieguo, ci concentreremo su funzioni di variabile intera non negativa a valori reali non negativi

$$f : n \in \mathbb{N} \rightarrow \mathbb{R}_{\geq 0}$$



# Notazione asintotica **O** ('o' grande)

$f(n) = O(g(n))$  se  $\exists$  due costanti  $c > 0$  e  $n_0 \geq 0$  tali che  $f(n) \leq c g(n)$  per ogni  $n \geq n_0$



# Legame con il concetto di limite

- Se  $g(n)$  è definitivamente diversa da 0 per  $n \rightarrow \infty$  (praticamente, tutti i casi di nostro interesse), avremo che

$$f(n) = O(g(n)) \iff \limsup_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty$$

ovvero  $f(n) = O(g(n))$  se e solo se  $f(n)$  è un infinito di ordine non superiore a  $g(n)$

- **Ulteriore semplificazione:** tutte le funzioni che studieremo avranno un andamento ‘regolare’ al crescere di  $n$ , e quindi potremo stabilire che  $f(n) = O(g(n))$  semplicemente verificando che  $\lim_{n \rightarrow \infty} f(n)/g(n) = k < \infty$

# Un caso notevole: i polinomi

Sia  $f(n) = a_d n^d + a_{d-1} n^{d-1} + \dots + a_0$  un polinomio di grado  $d$  (con  $a_d > 0$ ); dimostriamo che  $f(n) = O(n^d)$

Se scegliamo  $c = a_d + |a_{d-1}| + \dots + |a_0| \Rightarrow$

$$c n^d = a_d n^d + |a_{d-1}| n^d + \dots + |a_0| n^d \geq a_d n^d + |a_{d-1}| n^{d-1} + \dots + |a_0| \geq a_d n^d + a_{d-1} n^{d-1} + \dots + a_0 = f(n)$$

$\Rightarrow f(n) \leq c n^d \quad \forall n \geq 0$ , e quindi posso scegliere  $n_0 = 0$

Alternativamente (e più semplicemente):

$$\limsup_{n \rightarrow \infty} \frac{a_d n^d + \dots + a_0}{n^d} = \lim_{n \rightarrow \infty} \frac{a_d n^d + \dots + a_0}{n^d} = a_d < \infty \Rightarrow f(n) = O(n^d)$$

# Esempi

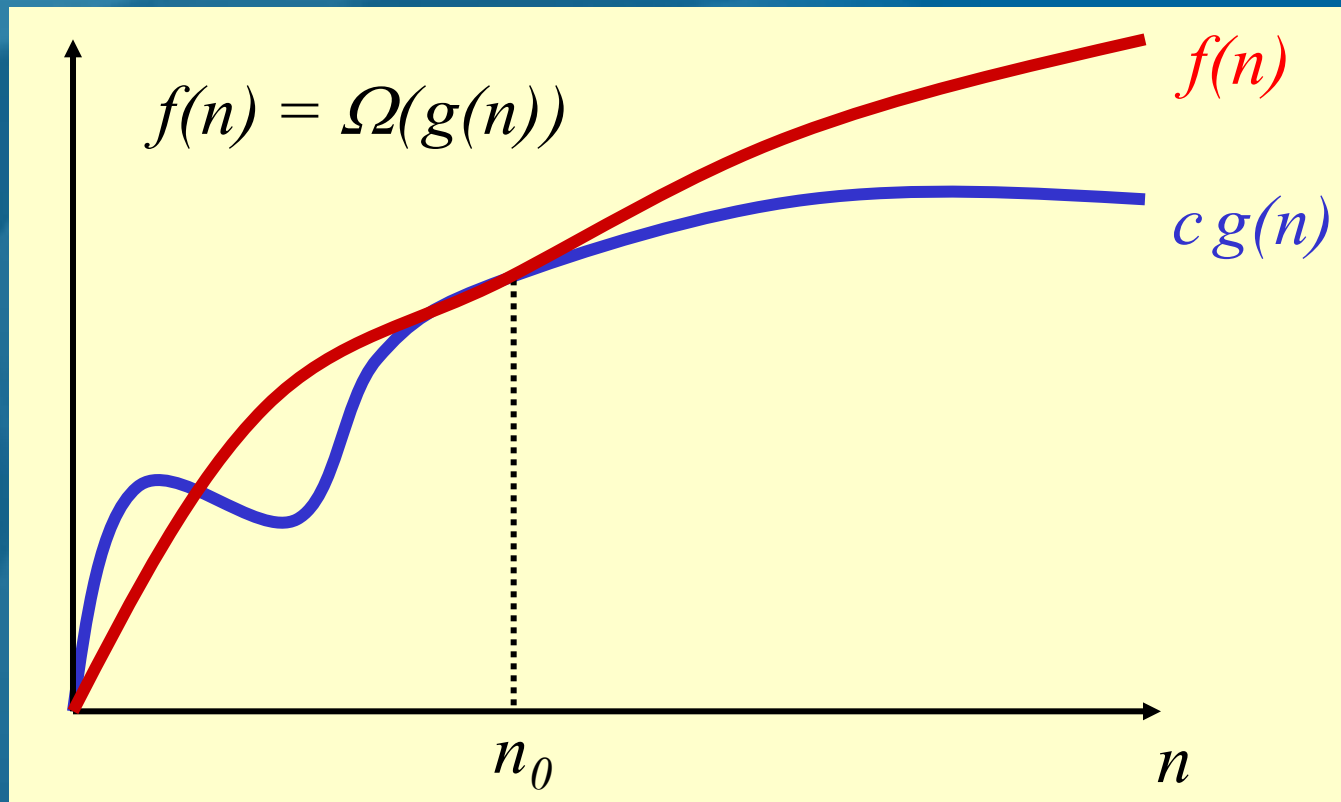
- Sia  $f(n) = 2n^2 - 3n$ ; applichiamo la dimostrazione appena fatta per mostrare che  $f(n) = O(n^2)$ :
  - scegliendo  $c = a_2 + |a_1| + |a_0| = 2 + 3 + 0 = 5$
  - avremo che  $2n^2 - 3n \leq 5n^2$  per ogni  $n \geq n_0 = 0$ .
- Alternativamente, in modo più semplice:
  - $\lim_{n \rightarrow \infty} (2n^2 - 3n)/n^2 = 2 < \infty \Rightarrow 2n^2 - 3n = O(n^2)$
- Più in generale,  $2n^2 - 3n = O(n^d) \forall d \geq 2$ , in quanto in tal caso
  - $\lim_{n \rightarrow \infty} (2n^2 - 3n)/n^d < \infty$
- $2n^2 - 3n = O(a^n) \forall a > 1$ , in quanto  $\lim_{n \rightarrow \infty} (2n^2 - 3n)/a^n = 0 < \infty$
- Invece,  $2n^2 - 3n \neq O(n)$ , poiché  $\lim_{n \rightarrow \infty} (2n^2 - 3n)/n = \infty$

# Osservazioni sulla notazione $O$

- Si noti che  $O(g(n))$  è un insieme di funzioni, ovvero è (informalmente) l'insieme di funzioni che crescono al più come  $g(n)$
- Ad esempio,  $O(n^2)$  contiene tutti i polinomi di primo e secondo grado, nonché tutte le funzioni che crescono al più come  $n^2$ , come ad esempio  $f(n) = 51$ , oppure  $f(n) = n \log^5 n$ , oppure ancora  $f(n) = n\sqrt{n}$
- Sarebbe quindi più opportuno scrivere che, ad esempio,  $2n^2+3n \in O(n^2)$ , ma con un piccolo abuso di notazione scriveremo sempre  $2n^2+3n = O(n^2)$
- Una particolare classe asintotica è quella che denoteremo con  $O(1)$ : questa contiene tutte le funzioni che crescono al più come una costante, quindi ad esempio  $f(n) = 51$ , oppure  $f(n) = 10^{11234}$ , ma anche, ad esempio,  $f(n) = 1/n$

# Notazione asintotica $\Omega$ (omega grande)

$f(n) = \Omega(g(n))$  se  $\exists$  due costanti  $c > 0$  e  $n_0 \geq 0$  tali che  $f(n) \geq c g(n)$  per ogni  $n \geq n_0$



# Legame con il concetto di limite

- Se  $g(n)$  è definitivamente diversa da 0 per  $n \rightarrow \infty$  (praticamente, tutti i casi di nostro interesse), avremo che

$$f(n) = \Omega(g(n)) \iff \liminf_{n \rightarrow \infty} \frac{f(n)}{g(n)} > 0$$

ovvero  $f(n) = \Omega(g(n))$  se e solo se  $f(n)$  è un infinito di ordine non inferiore a  $g(n)$

- **Ulteriore semplificazione:** tutte le funzioni che studieremo avranno un andamento ‘regolare’ al crescere di  $n$ , e quindi potremo stabilire che  $f(n) = \Omega(g(n))$  semplicemente verificando che  $\lim_{n \rightarrow \infty} f(n)/g(n) = k > 0$

# Un caso notevole: i polinomi

Sia  $f(n) = a_d n^d + a_{d-1} n^{d-1} + \dots + a_0$  un polinomio di grado  $d$  (con  $a_d > 0$ ); dimostriamo che  $f(n) = \Omega(n^d)$

Infatti:  $f(n)/n^d = a_d + a_{d-1} n^{-1} + \dots + a_0 n^{-d} \Rightarrow$

$$\exists n_0: \forall n \geq n_0 \quad a_d - |a_{d-1}| n^{-1} - \dots - |a_0| n^{-d} > 0$$

Se scegliamo  $c = a_d - |a_{d-1}| n_0^{-1} - \dots - |a_0| n_0^{-d} \Rightarrow$

$$c n^d = a_d n^d - |a_{d-1}| n^d n_0^{-1} - \dots - |a_0| n^d n_0^{-d}$$

e poiché per  $n \geq n_0$  si ha  $n^d n_0^{-k} \geq n^d n^{-k} = n^{d-k}$

$$c n^d \leq a_d n^d - |a_{d-1}| n^{d-1} - \dots - |a_0| \leq a_d n^d + a_{d-1} n^{d-1} + \dots + a_0 = f(n)$$

$$\Rightarrow \forall n \geq n_0 \quad c n^d \leq f(n)$$

Alternativamente (e più semplicemente):

$$\liminf_{n \rightarrow \infty} \frac{a_d n^d + \dots + a_0}{n^d} = \lim_{n \rightarrow \infty} \frac{a_d n^d + \dots + a_0}{n^d} = a_d > 0 \Rightarrow f(n) = \Omega(g(n))$$



# Esempi

- Sia  $f(n) = 2n^2 - 5n$ ; applichiamo la dimostrazione appena fatta per mostrare che  $f(n) = \Omega(n^2)$ :

$$f(n)/n^2 = (2n^2 - 5n)/n^2 = 2 - 5/n$$

ma  $2 - 5/n > 0$  per  $n \geq 3$  (quindi  $n_0=3$ );

Scegliendo  $c = a_2 - |a_1|/n_0 - |a_0|/n_0^2 = 2 - 5/3 = 1/3$

avremo che  $2n^2 - 5n \geq (1/3)n^2$  per  $n \geq n_0=3$ .

- Alternativamente, in modo più semplice:

$$\lim_{n \rightarrow \infty} (2n^2 - 5n)/n^2 = 2 > 0 \implies 2n^2 - 5n = \Omega(n^2)$$

- Più in generale,  $2n^2 - 5n = \Omega(n^d) \forall d \leq 2$ , in quanto in tal caso

$$\lim_{n \rightarrow \infty} (2n^2 - 5n)/n^d > 0$$

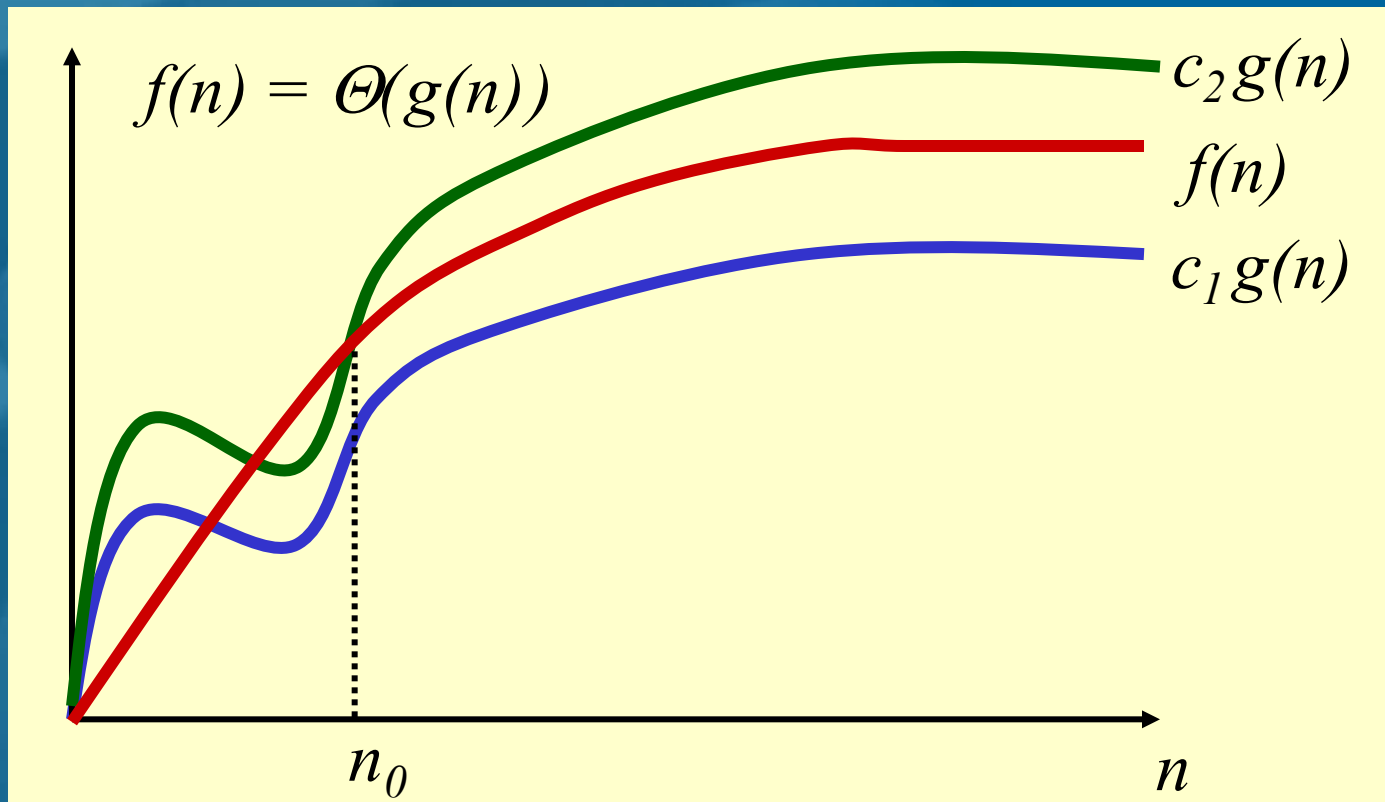
- $2n^2 - 5n = \Omega(\log_a n) \forall a > 1$ , in quanto  $\lim_{n \rightarrow \infty} (2n^2 - 5n)/\log_a n = \infty$
- Invece,  $2n^2 - 5n \neq \Omega(n^d) \forall d \geq 3$ , poiché  $\lim_{n \rightarrow \infty} (2n^2 - 5n)/n^d = 0$

# Osservazioni sulla notazione $\Omega$

- Al pari di  $O(g(n))$ , anche  $\Omega(g(n))$  è un insieme di funzioni, ovvero è (informalmente) l'insieme di funzioni che crescono almeno come  $g(n)$
- Ad esempio,  $\Omega(n^2)$  contiene tutti i polinomi di grado almeno pari a 2, nonché tutte le funzioni che crescono almeno come  $n^2$ , come ad esempio  $f(n) = 3n^3$ , oppure  $f(n) = n^2 \log^5 n$
- Sarebbe quindi più opportuno scrivere che, ad esempio,  $2n^2 - 5n \in \Omega(n^2)$ , ma con un piccolo abuso di notazione scriveremo sempre  $2n^2 - 5n = \Omega(n^2)$
- Una particolare classe asintotica è quella che denoteremo con  $\Omega(1)$ : questa contiene tutte le funzioni che crescono almeno come una costante, quindi ad esempio  $f(n) = 51$ , oppure  $f(n) = 10^{11234}$ , ma anche, ad esempio,  $f(n) = \log n$

# Notazione asintotica $\Theta$ (theta grande)

$f(n) = \Theta(g(n))$  se  $\exists$  tre costanti  $c_1, c_2 > 0$  e  $n_0 \geq 0$  tali che  $c_1 g(n) \leq f(n) \leq c_2 g(n)$  per ogni  $n \geq n_0$



# Legame con il concetto di limite

- Se  $g(n)$  è definitivamente diversa da 0 per  $n \rightarrow \infty$  (praticamente, tutti i casi di nostro interesse), avremo che

$$f(n) = \Theta(g(n)) \Leftrightarrow 0 < \liminf_{n \rightarrow \infty} \frac{f(n)}{g(n)} \leq \limsup_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty$$

ovvero  $f(n) = \Theta(g(n))$  se e solo se  $f(n)$  è un infinito dello stesso ordine di  $g(n)$

- **Ulteriore semplificazione:** tutte le funzioni che studieremo avranno un andamento ‘regolare’ al crescere di  $n$ , e quindi potremo stabilire che  $f(n) = \Theta(g(n))$  semplicemente verificando che  $\lim_{n \rightarrow \infty} f(n)/g(n) = k$ , con  $0 < k < \infty$
- Ad esempio,  $2n^2 + 3n = \Theta(n^2)$ , in quanto  $\lim_{n \rightarrow \infty} (2n^2 + 3n)/n^2 = 2$ , che è  $> 0$  e  $< \infty$

# Osservazioni sulla notazione $\Theta$

- Al pari di  $O(g(n))$  e di  $\Omega(g(n))$ , anche  $\Theta(g(n))$  è un insieme di funzioni, ovvero è (informalmente) l'insieme di funzioni che crescono come  $g(n)$
- Ad esempio,  $\Theta(n^2)$  contiene tutti i polinomi di grado pari a 2
- Sarebbe quindi più opportuno scrivere che, ad esempio,  $2n^2 - 5n \in \Theta(n^2)$ , ma con un piccolo abuso di notazione scriveremo sempre  $2n^2 - 5n = \Theta(n^2)$
- Una particolare classe asintotica è quella che denoteremo con  $\Theta(1)$ : questa contiene tutte le funzioni che crescono esattamente come una costante, quindi ad esempio  $f(n) = 51$ , oppure  $f(n) = 10^{11234}$

# Rivisitazione degli algoritmi per Fibonacci

	Numero di linee di codice	Occupazione di memoria
<b>fibonacci1</b>	$1 = \Theta(1)$	$1 = \Theta(1)$
fibonacci2	$3F_n - 2 \approx \Phi^n = \Theta(\Phi^n)$	$\approx \Phi^n = \Theta(\Phi^n)^{(*)}$
fibonacci3	$2n = \Theta(n)$	$n+1 = \Theta(n)$
fibonacci4	$4n-5 = \Theta(n)$	$4 = \Theta(1)$
fibonacci5	$2n+1 = \Theta(n)$	$5 = \Theta(1)$
fibonacci6	$\log n \leq T(n) \leq 4(1+\log n)$ $\Rightarrow T(n) = \Theta(\log n)$	$\approx \log n = \Theta(\log n)^{(*)}$

\* per le variabili di lavoro delle chiamate ricorsive

# Relazioni tra $O$ , $\Omega$ e $\Theta$

$$f(n) = \Theta(g(n)) \Rightarrow f(n) = O(g(n))$$

$$f(n) = O(g(n)) \not\Rightarrow f(n) = \Theta(g(n))$$

$$f(n) = \Theta(g(n)) \Rightarrow f(n) = \Omega(g(n))$$

$$f(n) = \Omega(g(n)) \not\Rightarrow f(n) = \Theta(g(n))$$

$$f(n) = \Theta(g(n)) \Leftrightarrow f(n) = \Omega(g(n)) \text{ e } f(n) = O(g(n))$$

# Notazione asintotica **$o$** (' **$o$** ' piccolo)

$f(n) = o(g(n))$  se  $\forall c > 0, \exists n_0 \geq 0$  tale che

$$f(n) \leq c g(n) \text{ per ogni } n \geq n_0$$

- Se  $g(n)$  è definitivamente diversa da 0 per  $n \rightarrow \infty$  (praticamente, tutti i casi di nostro interesse), avremo che:

$$f(n) = o(g(n)) \iff \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0$$

- Notare che  $o(g(n)) \subset O(g(n))$
- Ad esempio,  $2n^2 + 3n = o(n^d) \forall d \geq 3$ , in quanto  $\lim_{n \rightarrow \infty} (2n^2 + 3n)/n^d = 0$



## Notazione asintotica $\omega$ (omega piccolo)

$f(n) = \omega(g(n))$  se  $\forall c > 0, \exists n_0 \geq 0$  tale che

$$f(n) \geq c g(n) \text{ per ogni } n \geq n_0$$

- Se  $g(n)$  è definitivamente diversa da 0 per  $n \rightarrow \infty$  (praticamente, tutti i casi di nostro interesse), avremo che:

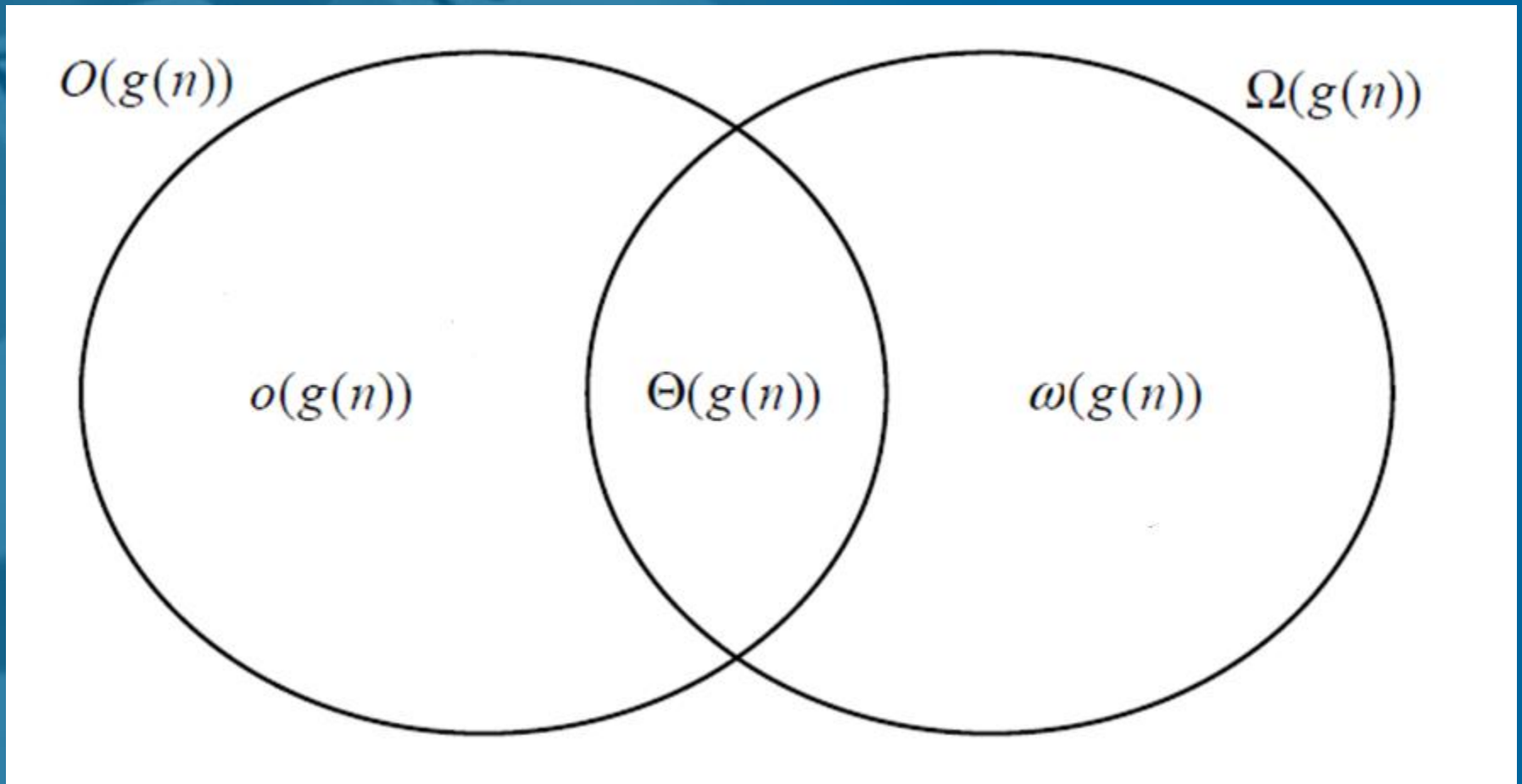
$$f(n) = \omega(g(n)) \iff \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \infty$$

- Notare che  $\omega(g(n)) \subset \Omega(g(n))$
- Ad esempio,  $2n^5 + 4n^2 = \omega(n^d) \forall d \leq 4$ , in quanto  $\lim_{n \rightarrow \infty} (2n^5 + 4n^2)/n^d = \infty$

# Analogie

 $O$  $\Omega$  $\Theta$  $o$  $\omega$  $\leq$  $\geq$  $=$  $<$  $>$

# Graficamente



# Proprietà della notazione asintotica

## Transitività

$$f(n) = \Theta(g(n)) \text{ e } g(n) = \Theta(h(n)) \Rightarrow f(n) = \Theta(h(n))$$

$$f(n) = O(g(n)) \text{ e } g(n) = O(h(n)) \Rightarrow f(n) = O(h(n))$$

$$f(n) = \Omega(g(n)) \text{ e } g(n) = \Omega(h(n)) \Rightarrow f(n) = \Omega(h(n))$$

$$f(n) = o(g(n)) \text{ e } g(n) = o(h(n)) \Rightarrow f(n) = o(h(n))$$

$$f(n) = \omega(g(n)) \text{ e } g(n) = \omega(h(n)) \Rightarrow f(n) = \omega(h(n))$$

## Riflessività

$$f(n) = \Theta(f(n))$$

$$f(n) = O(f(n))$$

$$f(n) = \Omega(f(n))$$

## Simmetria

$$f(n) = \Theta(g(n)) \Leftrightarrow g(n) = \Theta(f(n))$$

## Simmetria trasposta

$$f(n) = O(g(n)) \Leftrightarrow g(n) = \Omega(f(n))$$

$$f(n) = o(g(n)) \Leftrightarrow g(n) = \omega(f(n))$$

# Relazioni asintotiche notevoli

## Polinomi

$$P(n) = a_d n^d + a_{d-1} n^{d-1} + \dots + a_0$$

$$a_d > 0$$

$$P(n) = O(n^d), P(n) = \Omega(n^d) \Rightarrow P(n) = \Theta(n^d)$$

$$P(n) = O(n^c) \forall c \geq d, P(n) \neq O(n^c) \forall c < d$$

$$P(n) = \Omega(n^c) \forall c \leq d, P(n) \neq \Omega(n^c) \forall c > d$$

$$P(n) = o(n^c) \forall c > d, P(n) = \omega(n^c) \forall c < d$$

## Esponenziali

$$f(n) = a^n$$

$$a > 1$$

$$\lim_{n \rightarrow \infty} \frac{a^n}{n^d} = \infty$$

$$a^n = \omega(n^d) \forall d > 0$$

$$\Rightarrow a^n = \Omega(n^d) \forall d > 0$$

## Logaritmi

$$f(n) = \log_b n \quad b > 1$$

$$\lim_{n \rightarrow \infty} \frac{(\log_b n)^c}{n^d} = 0, \forall c, d \geq 1$$

$$(\log_b n)^c = o(n^d) \forall c, d \geq 1$$

$$\Rightarrow (\log_b n)^c = O(n^d) \forall c, d \geq 1$$

## Fattoriali

$$f(n) = n! = n \cdot (n-1) \cdot (n-2) \cdot \dots \cdot 2 \cdot 1$$

$$n! = o(n^n) \Rightarrow n! = O(n^n)$$

$$n! = \omega(a^n) \Rightarrow n! = \Omega(a^n) \forall a > 0$$

# Domanda di approfondimento

- Qual è la complessità temporale degli algoritmi `Fibonacci6`, `Fibonacci4` e `Fibonacci2` in funzione della rappresentazione dell'input?