

3 Analisi delle funzioni di costo

Abbiamo concluso la prima parte delle lezioni che ci introduceva alle equazioni di ricorrenza: ora andremo a capire i fondamenti matematici che ci permetteranno di analizzarne una famiglia più grande.

3.1 Proprietà della notazione asintotica

3.1.1 Regola generale

Teorema 1. $f(n) = a_k n^k + a_{k-1} n^{k-1} + \dots + a_1 n + a_0, a_k > 0 \Rightarrow f(n) = \Theta(n^k)$

Dimostrazione. Limite superiore: $\exists c > 0, \exists m \geq 0 : f(n) \leq cn^k, \forall n \geq m$

$$\begin{aligned}
 f(n) &= a_k n^k + a_{k-1} n^{k-1} + \dots + a_1 n + a_0 \\
 &\leq a_k n^k + |a_{k-1}| n^{k-1} + \dots + |a_1| n + |a_0| \\
 &\leq a_k n^k + |a_{k-1}| n^k + \dots + |a_1| n^k + |a_0| n^k \\
 &= (a_k + |a_{k-1}| + \dots + |a_1| + |a_0|) n^k \\
 &\stackrel{?}{\leq} cn^k
 \end{aligned}$$

↓ *a_k > 0 per def., rendo gli altri positivi*
 ↓ *∀n ≥ 1, elevo tutte le potenze a k*
 ↓ *raccolgo n^k*
 ↓ *esiste una costante c*
 ↓ *che rende la disequazione vera?*

che è vera per $c \geq (|a_k| + |a_{k-1}| + \dots + |a_1| + |a_0|)$ (il coefficiente) e per $m = 1$. □

Dimostrazione. Limite inferiore: $\exists d > 0, \exists m \geq 0 : f(n) \geq dn^k, \forall n \geq m$

$$\begin{aligned}
 f(n) &= a_k n^k + a_{k-1} n^{k-1} + \dots + a_1 n + a_0 \\
 &\geq a_k n^k - |a_{k-1}| n^{k-1} - \dots - |a_1| n - |a_0| \\
 &\geq a_k n^k - |a_{k-1}| n^{k-1} - \dots - |a_1| n^{k-1} - |a_0| n^{k-1} \\
 &\stackrel{?}{\geq} dn^k
 \end{aligned}$$

↓ *normalizzo aggiungendo un segno negativo*
 ↓ *∀n ≥ 1, elevo alla k - 1*

L'ultima equazione è vera se:

$$d \leq a_k - \frac{|a_{k-1}|}{n} - \frac{|a_{k-2}|}{n} - \dots - \frac{|a_1|}{n} - \frac{|a_0|}{n} > 0 \iff n > \frac{|a_{k-1}| + \dots + |a_0|}{a_k} = m$$

□

Abbiamo dimostrato sia il limite superiore che quello inferiore, possiamo affermare che è un Θ .

Ad esempio $17n^3 - 47n^2 + 123n + 17 = \Theta(n^3)$. Oppure $2n^3 + 7 = \Theta(n^2)$.

3.1.2 Funzioni di costo particolari

La complessità di $f(n) = 5$ è pari a $\Theta(1)$, questa classe di complessità rappresenta quegli algoritmi che sono così ben congegnati che indipendentemente dalla dimensione dell'input impiegano un tempo costante a risolvere il problema. Ad esempio richiedere il minimo in un vettore ordinato.

Dimostriamolo.

$$\begin{aligned}
 f(n) &= 5 \geq c_1 n^0 \Rightarrow c_1 \leq 5 \\
 f(n) &= 5 \leq c_2 n^0 \Rightarrow c_2 \geq 5 \\
 f(n) &= \Theta(n^0) = \Theta(1)
 \end{aligned}$$

La complessità di $f(n) = 5 + \sin(n)$ è pari a $\Theta(1)$, in quanto $\sin(n)$ oscilla fra 1 e -1.

3.1.3 Proprietà delle notazioni

Teorema 2 (Dualità). Se $f(n)$ è limitata superiormente da $g(n)$, allora $g(n)$ è limitata inferiormente da $f(n)$.

$$f(n) = \mathcal{O}(g(n)) \Leftrightarrow g(n) = \Omega(f(n))$$

La dimostrazione avviene tramite passaggi algebrici.

Dimostrazione.

$$\begin{aligned} f(n) = \mathcal{O}(g(n)) &\Leftrightarrow f(n) \leq c g(n), \forall n \geq m \\ &\Leftrightarrow g(n) \geq \frac{1}{c} f(n), \forall n \geq m \\ &\Leftrightarrow g(n) \geq c' f(n), \forall n \geq m, c' = \frac{1}{c} \\ &\Leftrightarrow g(n) = \Omega(f(n)) \end{aligned}$$

ribalto la disequazione, $c > 0$
rinomino $\frac{1}{c}$ a c'
passo alla notazione dei quantificatori

□

Teorema 3 (eliminazione delle costanti).

$$\begin{aligned} f(n) = \mathcal{O}(g(n)) &\Leftrightarrow af(n) = \mathcal{O}(g(n)), \forall a > 0 \\ f(n) = \Omega(g(n)) &\Leftrightarrow af(n) = \Omega(g(n)), \forall a > 0 \end{aligned}$$

La dimostrazione avviene sempre tramite semplici passaggi algebrici.

Dimostrazione.

$$\begin{aligned} f(n) = \mathcal{O}(g(n)) &\Leftrightarrow f(n) \leq cg(n), \forall n \geq m \\ &\Leftrightarrow f(n) \leq acg(n), \forall n \geq m, \forall a > 0 \\ &\Leftrightarrow f(n) \leq c'g(n), \forall n \geq m, c' = ac > 0 \\ &\Leftrightarrow af(n) = \mathcal{O}(g(n)) \end{aligned}$$

Introduco una costante a
raccolgo ac sotto un'unica costante c'
per definizione

□

Ad esempio $2 \log n = \Theta(\log n)$, ignoriamo quindi le costanti numeriche.

Teorema 4 (sommatoria - sequenza di algoritmi).

$$\begin{cases} f_1(n) = \mathcal{O}(g_1(n)) \\ f_2(n) = \mathcal{O}(g_2(n)) \end{cases} \Rightarrow f_1(n) + f_2(n) = \mathcal{O}(\max(g_1(n), g_2(n)))$$

$$\begin{cases} f_1(n) = \Omega(g_1(n)) \\ f_2(n) = \Omega(g_2(n)) \end{cases} \Rightarrow f_1(n) + f_2(n) = \Omega(\max(g_1(n), g_2(n)))$$

La dimostrazione.

$$\begin{aligned} f_1(n) = \mathcal{O}(g_1(n)) \wedge f_2(n) = \mathcal{O}(g_2(n)) &\Rightarrow \\ f_1(n) \leq c_1 g_1(n) \wedge f_2(n) \leq c_2 g_2(n) &\Rightarrow \quad \text{per definizione} \\ f_1(n) + f_2(n) \leq c_1 g_1(n) + c_2 g_2(n) &\Rightarrow \quad \text{raccolgo} \\ f_1(n) + f_2(n) \leq \max\{c_1, c_2\} (2 \cdot \max(g_1(n), g_2(n))) &\Rightarrow \quad \text{per definizione} \\ f_1(n) + f_2(n) = \mathcal{O}(g_1(n) + g_2(n)) \end{aligned}$$

Ad esempio se ripeto due volte un algoritmo lineare, l'algoritmo risultante sarà comunque lineare. Mentre se ho un algoritmo lineare ed un algoritmo quadratico, la complessità risultante è una combinazione delle due, il limite superiore della complessità totale sarà quindi quadratica.

Teorema 5 (cicli annidati). Se $f_1(n)$ viene ripetuto $f_2(n)$ volte, allora $f_1(n) \cdot f_2(n)$ è limitato superiormente dal prodotto delle complessità e limitato inferiormente dal prodotto delle complessità.

$$\begin{cases} f_1(n) = \mathcal{O}(g_1(n)) \\ f_2(n) = \mathcal{O}(g_2(n)) \end{cases} \Rightarrow f_1(n) \cdot f_2(n) = \mathcal{O}(g_1(n) \cdot g_2(n))$$

$$\begin{cases} f_1(n) = \Omega(g_1(n)) \\ f_2(n) = \Omega(g_2(n)) \end{cases} \Rightarrow f_1(n) \cdot f_2(n) = \Omega(g_1(n) \cdot g_2(n))$$

La dimostrazione è molto semplice.

$$f_1(n) = \mathcal{O}(g_1(n)) \wedge f_2(n) = \mathcal{O}(g_2(n)) \Rightarrow$$

$$f_1(n) \leq c_1 g_1(n) \wedge f_2(n) \leq c_2 g_2(n) \Rightarrow$$

$$f_1(n) \cdot f_2(n) \leq c_1 c_2 g_1(n) g_2(n)$$

\downarrow per definizione
 \downarrow raccolgo, $c_1 c_2 > 0$

Ad esempio, se ripeto n volte un algoritmo di costo $\log n$, l'algoritmo complessivo avrà un costo di $n \log n$.

Teorema 6 (simmetria). Se $f(n)$ è limitata superiormente ed inferiormente da $g(n)$, allora anche $g(n)$ è limitata inferiormente e superiormente da $f(n)$.

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

Vuol dire semplicemente che $2n^2 + 7 = \Theta(n^2) \Leftrightarrow n^2 = \Theta(2n^2 + 7)$.

La dimostrazione avviene tramite la proprietà di dualità:

Dimostrazione.

$$\begin{aligned} f(n) = \Theta(g(n)) &\Rightarrow f(n) = \mathcal{O}(g(n)) \Rightarrow g(n) = \Omega(f(n)) \\ f(n) = \Theta(g(n)) &\Rightarrow f(n) = \Omega(g(n)) \Rightarrow g(n) = \mathcal{O}(f(n)) \end{aligned}$$

□

Teorema 7 (transitività). Se $f(n)$ è limitata superiormente da $g(n)$, e $g(n)$ è limitata superiormente da $h(n)$, allora $f(n)$ è limitata superiormente da $h(n)$.

$$\begin{cases} f(n) = \mathcal{O}(g(n)) \\ g(n) = \mathcal{O}(h(n)) \end{cases} \Rightarrow f(n) = \mathcal{O}(h(n))$$

La dimostrazione è banale:

Dimostrazione.

$$\begin{aligned} f(n) = \mathcal{O}(g(n)) \wedge g(n) = \mathcal{O}(h(n)) &\Rightarrow \\ f(n) \leq c_1 g(n) \wedge g(n) \leq c_2 h(n) &\Rightarrow \\ f(n) \leq c_1 c_2 h(n) &\Rightarrow \\ f(n) = \mathcal{O}(h(n)) &\end{aligned}$$

\downarrow applico la definizione
 \downarrow sostituisco $g(n)$ con $c_2 h(n)$
 \downarrow $c_1 c_2 > 0$, elimino la costante

□

Altre funzioni di costo

Vogliamo provare che $\boxed{\log n = \mathcal{O}(n)}$

Dimostriamo per induzione che $\exists c > 0, \exists m \geq 0 : \log n \leq cn, \forall n \geq m$.

- **caso base** ($n = 1$):

$$\begin{aligned} \log n &\leq cn \\ \log 1 &\leq c \cdot 1 \\ 0 &\leq c \end{aligned}$$

↓ *n = 1*
↓ *semplifico*

- **ipotesi induttiva:** $\log k \leq ck, \forall k \leq n$

- **passo induttivo** dimostriamo la proprietà per $n + 1$:

$$\begin{aligned} \log(n+1) &\leq \log(n+n) = \log 2n \quad \forall n \geq 1 \\ &= \log 2 + \log n \\ &= 1 + \log n \\ &\leq 1 + cn \\ &\stackrel{?}{\leq} c(n+1) \\ 1 + cn &\leq c(n+1) \\ 1 + cn &\leq cn + c \\ 1 &\leq c \end{aligned}$$

↓ *log ab = log a + log b*
↓ *log₂ 2 = 1*
↓ *per ipotesi induttiva*
↓ *obiettivo*
↓ *metto a confronto*
↓ *moltiplico*
↓ *semplifico*

Classificazione delle funzioni

È possibile trarre un ordinamento dalle principali espressioni, estendendo le relazioni che abbiamo dimostrato fino ad ora. Per ogni $r < s, h < k, a < b$:

$$\mathcal{O}(1) \subset \mathcal{O}(\log^r n) \subset \mathcal{O}(\log^s n) \subset \mathcal{O}(n^h) \subset \mathcal{O}(n^h \log^r n) \subset \mathcal{O}(n^h \log^s n) \subset \mathcal{O}(n^k) \subset \mathcal{O}(a^n) \subset \mathcal{O}(b^n)$$

Ricordati che $\log^r(n) = (\log n)^r$. Non è detto che gli esponenti siano interi. $\sqrt[1000]{n}$ cresce più velocemente di $\log^2 n$. n cresce meno velocemente di $n \log n$, il quale a sua volta cresce meno velocemente di $n \log^2 n$.

3.2 Analisi per livelli

Nell'analisi per livelli (o dell'albero di ricorsione) andiamo sostanzialmente a "srotolare" la ricorrenza in un albero, i cui nodi rappresentano i costi dei vari livelli della ricorsione.

3.2.1 Esempi di analisi per livelli

Analisi per livelli dell'algoritmo della ricerca binaria

Equazione di ricorrenza della ricerca binaria:

$$T(n) = \begin{cases} T(n/2) + b & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Assumiamo per semplicità che $n = 2^k$, ovvero che $k = \log n$. È possibile quindi risolvere questa ricorrenza nel seguente modo:

$$\begin{aligned} T(n) &= b + T\left(\frac{n}{2}\right) \\ &= b + b + T\left(\frac{n}{4}\right) \\ &= b + b + b + T\left(\frac{n}{8}\right) \\ &= \dots \\ &= \underbrace{b + b + \dots + b}_{\log n} + T(1) \\ &= b \log n + T(1) \\ &= \Theta(\log n) \end{aligned}$$

↓ *T(n/2) + b*
↓ *T(n/4) + b*
↓ *svolgo log n operazioni*
↓ *semplifico*
↓ *eliminazione delle costanti*

Analisi per livelli del primo tentativo della moltiplicazione di Karatsuba

Equazione di ricorrenza:

$$T(n) = \begin{cases} 4T(n/2) + n & n > 1 \\ 1 & n \leq 1 \end{cases}$$

È possibile risolvere questa ricorrenza nel modo seguente:

$$\begin{aligned}
 T(n) &= n + 4T(n/2) \\
 &= n + 4 \left(4T\left(\frac{n}{2} \cdot \frac{1}{2}\right) + \frac{n}{2} \right) \\
 &= n + 4^{n/2} + 16T(n/4) \\
 &= n + 2n + 16 \left(4T\left(\frac{n}{4} \cdot \frac{1}{2}\right) + \frac{n}{4} \right) \\
 &= n + 2n + 16^{n/4} + 64T(n/8) \\
 &= \dots \\
 &= \underbrace{n + 2n + 4n + 8n + \dots + 2^{\log n - 1} n}_{\text{sommatoria}} + 4^{\log n} T(1) \\
 &= n \sum_{j=0}^{\log n - 1} 2^j + 4^{\log n}
 \end{aligned}$$

↓ 4T($\frac{n}{2} \cdot \frac{1}{2}$) + $\frac{n}{2}$
 ↓ semplifico
 ↓ 4T($\frac{n}{4} \cdot \frac{1}{2}$) + $\frac{n}{4}$, $4^{n/2} = 2n$
 ↓ semplifico
 ↓ svolgo $\log n$ operazioni
 ↓ raccolgo

Ciò che abbiamo ottenuto è una forma chiusa, non più un'equazione di ricorrenza: non è ancora nella sua forma definitiva, dobbiamo ancora trattarla.

$$\begin{aligned}
 T(n) &= n \sum_{j=0}^{\log n - 1} 2^j + 4^{\log n} \\
 &= n \cdot \frac{2^{\log n} - 1}{2 - 1} + 4^{\log n} \\
 &= n(n - 1) + 4^{\log n} \\
 &= n^2 - n + n^2 \\
 &= 2n^2 - n \\
 &= \Theta(n^2)
 \end{aligned}$$

↓ Applico $\forall x \neq 1 : \sum_{j=0}^k x^j = \frac{x^{k+1} - 1}{x - 1}$,
 dove $k = \log n - 1$
 ↓ $2^{\log n} = n^{\log_2 2} = n^1$
 ↓ moltiplico: $n(n - 1) = n^2 - n$
 ↓ semplifico: $4^{\log n} = n^{\log 4} = n^2$
 ↓ raccolgo n^2
 ↓ regola generale ↑

Possiamo concludere che $T(n) = n + 4T(n/2) = \Theta(n^2)$.

Modifichiamo l'esempio precedente

Esaminiamo la seguente equazione di ricorrenza:

$$T(n) = \begin{cases} 4T(n/2) + n^3 & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Per semplicità consideriamo $n = 2^\ell$.

livello	dim.	costo chiam.	no. chiamate	costo livello
0	n	n^3	1	n^3
1	$n/2$	$(n/2)^3$	4	$4(n/2)^3$
2	$n/4$	$(n/4)^3$	16	$16(n/4)^3$
3	$n/8$	$(n/8)^3$	64	$64(n/8)^3$
...
i	$n/2^i$	$(n/2^i)^3$	4^i	$4^i(n/2^i)^3$
...
$\ell - 1$	$n/2^{\ell-1}$	$(n/2^{\ell-1})^3$	$4^{\ell-1}$	$4^{\ell-1}(n/2^{\ell-1})^3$
$\ell = \log n$	1	$T(1)$	$4^{\log n}$	$4^{\log n}$

Sommando il costo di tutti i $\ell - 1$ livelli ed il livello ℓ -esimo otteniamo:

$$\begin{aligned}
T(n) &= \sum_{i=0}^{\log n-1} 4^i \cdot \frac{n^3}{2^{3i}} + 4^{\log n} \\
&= n^3 \sum_{i=0}^{\log n-1} \frac{2^{2i}}{2^{3i}} + 4^{\log n} \\
&= n^3 \sum_{i=0}^{\log n-1} \left(\frac{1}{2}\right)^i + 4^{\log n} \\
&= n^3 \sum_{i=0}^{\log n-1} \left(\frac{1}{2}\right)^i + n^2 \\
&\leq n^3 \sum_{i=0}^{\infty} \left(\frac{1}{2}\right)^i + n^2 \\
&= n^3 + \frac{1}{1-\frac{1}{2}} + n^2 \\
&= 2n^3 + n^2
\end{aligned}$$

semplifico: $4^i = 2^{2i}$,
porto fuori: n^3
semplifico: $\frac{2^{2i}}{2^{3i}} = \left(\frac{1}{2}\right)^i$
cambio di base, $4^{\log n} = n^{\log 4} = n^2$
estensione della sommatoria ad ∞
Serie geometrica infinita decrescente:
 $\forall x, |x| < 1 : \sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$, dove $x = \frac{1}{2}$
semplifico: $\frac{1}{1-\frac{1}{2}} = 2$

Abbiamo dimostrato che $T(n) \leq 2n^3 + n^2$, possiamo quindi affermare che $T(n) = \mathcal{O}(n^3)$, ma non possiamo affermare che $T(n) = \Theta(n^3)$ poiché abbiamo dimostrato solo un limite superiore.

Suggerimento. Tutte le volte che notiamo in un'equazione di ricorrenza una parte polinomiale, come ad esempio n^3 , possiamo dire con certezza che l'equazione è $\Omega(n^3)$.

Ora possiamo affermare che $T(n) = \Theta(n^3)$.

Modifichiamo l'esempio precedente ulteriormente

Esaminiamo la seguente equazione di ricorrenza:

$$T(n) = \begin{cases} 4T(n/2) + n^2 & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Cambia il costo della chiamata, che passa dall'essere n^3 all'essere n^2 . Di conseguenza cambia anche il costo del livello.

Livello	Dim.	Costo chiam.	no. chiamate	Costo livello
0	n	n^2	1	n^2
1	$n/2$	$(n/2)^2$	4	$4(n/2)^2$
2	$n/4$	$(n/4)^2$	16	$16(n/4)^2$
3	$n/8$	$(n/8)^2$	64	$64(n/8)^2$
...
i	$n/2^i$	$(n/2^i)^2$	4^i	$4^i(n/2^i)^2$
...
$\ell - 1$	$n/2^{\ell-1}$	$(n/2^{\ell-1})^2$	$4^{\ell-1}$	$4^{\ell-1}(n/2^{\ell-1})^2$
$\ell = \log n$	1	$T(1)$	$4^{\log n}$	$4^{\log n}$

Sommando il costo di tutti i $\ell - 1$ livelli ed il livello ℓ -esimo otteniamo:

$$\begin{aligned}
T(n) &= \sum_{i=0}^{\log n-1} \frac{n^2}{2^{2i}} \cdot 4i + 4^{\log_2 n} \\
&= n^2 \sum_{i=0}^{\log n-1} \frac{2^{2i}}{2^{2i}} + n^2 \\
&= n^2 \sum_{i=0}^{\log n-1} 1 + n^2 \\
&= n^2 \log n + n^2 \\
&= \Theta(n^2 \log n)
\end{aligned}$$

semplifico: $4^i = 2^{2i}$, $4^{\log n} = n^{\log_2 4} = n^2$
 porto fuori: n^2
 semplifico: $\frac{2^{2i}}{2^{2i}} = 1$
 svolgo la sommatoria: $\sum_{i=0}^{\log n-1} 1 = \log n$
 regola generale

Conclusioni

Per riassumere, se consideriamo i termini non ricorrenti, n ha prodotto n^2 , n^2 ha prodotto $n^2 \log n$ ed n^3 ha prodotto n^3 . Quando avremo a disposizione lo strumento “master theorem” riusciremo semplicemente guardando l’equazione di ricorrenza a capire quale complessità essa produce.

3.3 Metodo di sostituzione

Vediamo ora un ulteriore meccanismo per risolvere le equazioni di ricorrenza in quanto il metodo precedente in alcuni casi può non esserci di aiuto. Il metodo di sostituzione è un metodo in cui si cerca di “indovinare” (*guess*) una soluzione in base alla propria esperienza ed in seguito si dimostra che questa intuizione è corretta tramite dimostrazione per induzione.

Primo esercizio

$$T(n) = \begin{cases} T(\lfloor n/2 \rfloor) + n & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Risolvendolo tramite il metodo precedente otteniamo:

$$\begin{aligned}
T(n) &= n \sum_{i=0}^{\log n} \left(\frac{1}{2}\right)^i \\
&\leq n \sum_{i=0}^{\infty} \left(\frac{1}{2}\right)^i \\
&\leq n \frac{1}{1 - \frac{1}{2}} \\
&= 2n
\end{aligned}$$

estensione della sommatoria ad ∞
 Serie geometrica decrescente infinita:
 $\forall x, |x| < 1 : \sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$, dove $x = \frac{1}{2}$
 $\frac{1}{1-\frac{1}{2}} = 2$

Sapendo già il risultato proviamo – per tentativi – a dimostrare che $T(n) = \mathcal{O}(n)$

Limite superiore: Dobbiamo dimostrare che $\exists c > 0, \exists m \geq 0 : T(n) \leq cn, \forall n \geq m$

- **caso base** dimostriamo $T(1)$:

$$T(1) = 1 \stackrel{?}{\leq} c \cdot 1 \iff c \geq 1$$

- **ipotesi induttiva** $\forall k < n : T(k) \leq ck$, ossia assumiamo che per tutti i valori più piccoli di n la dimostrazione sia già stata fatta.

- **ipotesi induttiva** dimostriamo la disequazione per $T(n)$:

$$\begin{aligned} T(n) &= T(\lfloor \frac{n}{2} \rfloor) + n \\ &\leq c \lfloor \frac{n}{2} \rfloor + n \\ &\leq c \frac{n}{2} + n \\ &= \left(\frac{c}{2} + 1\right)n \\ &= \left(\frac{c}{2} + 1\right)n \stackrel{?}{\leq} cn \\ &\Rightarrow \frac{c}{2} + 1 \leq c \\ &\Rightarrow c \geq 2 \end{aligned}$$

sost. ip. ind. con $k = \lfloor \frac{n}{2} \rfloor$
semplifico l'intero inferiore
raccolgo n
obiettivo
semplifico

Abbiamo quindi provato che $T(n) \leq cn$, con due diversi valori della nostra costante c : nel caso base è risultata $c \geq 1$, mentre nel passo induttivo è risultata pari a $c \geq 2$. Quindi la nostra ipotesi è valida per qualsiasi valore di c t.c. $c \geq a$ e $c \geq 2$, ovvero $\forall c \geq 2$.

Quanto abbiamo dimostrato vale per $n = 1$ e per tutti i valori di n successivi, di conseguenza $m = 1$.

Al solo scopo didattico (in quanto lo potremmo dedurlo dal termine non ricorsivo) proviamo passo passo anche il limite inferiore, ossia che $T(n) = \Omega(n)$

Limite inferiore: Dobbiamo dimostrare che $\exists d > 0, \exists m \geq 0 : T(n) \geq dn, \forall n \geq m$.

Nota. Usiamo la costante d al solo scopo di non confonderla la costante c usata nella dimostrazione precedente.

- **caso base** $T(1)$:

$$T(1) = 1 \stackrel{?}{\geq} d \cdot 1 \iff d \leq 1$$

- **ipotesi induttiva** $\forall k < n : T(k) \geq dk$, ossia assumiamo che per tutti i valori più piccoli di n la mia dimostrazione sia già stata fatta.

- **ipotesi induttiva** dimostriamo la disequazione per $T(n)$

$$\begin{aligned} T(n) &= T(\lfloor \frac{n}{2} \rfloor) + n \\ &\geq d \lfloor \frac{n}{2} \rfloor + n \\ &\geq d \frac{n}{2} - 1 + n \\ &= \left(\frac{d}{2} - \frac{1}{n} + 1\right)n \\ &= \left(\frac{d}{2} - \frac{1}{n} + 1\right)n \stackrel{?}{\geq} dn \\ &\Rightarrow \frac{d}{2} - \frac{1}{n} + 1 \geq n \\ &\Rightarrow d \leq 2 - \frac{2}{n} \end{aligned}$$

sost. ip. ind. con $d = \frac{n}{2}$
semplifico l'intero inferiore
raccolgo n
obiettivo
semplifico

Abbiamo quindi provato che $T(n) \geq dn$, con due diversi valori della nostra costante d : nel caso base è risultata $d \leq 1$, mentre nel passo induttivo è risultata pari a $d \leq 2 - \frac{2}{n}$. $d = 1$ è valore che soddisfa entrambe le disequazioni, dimostra quindi la nostra tesi.

Quanto abbiamo dimostrato vale per $n = 1$ e per tutti i valori di n successivi, di conseguenza $m = 1$.

Per concludere abbiamo provato che $T(n) = T(\lfloor \frac{n}{2} \rfloor) + n$ è limitata sia superiormente $T(n) = \mathcal{O}(n)$, sia inferiormente $T(n) = \Omega(n)$, possiamo affermare quindi con assoluta certezza che $T(n) = \Theta(n)$.

Nota che avremmo potuto evitare la dimostrazione del limite inferiore osservando che

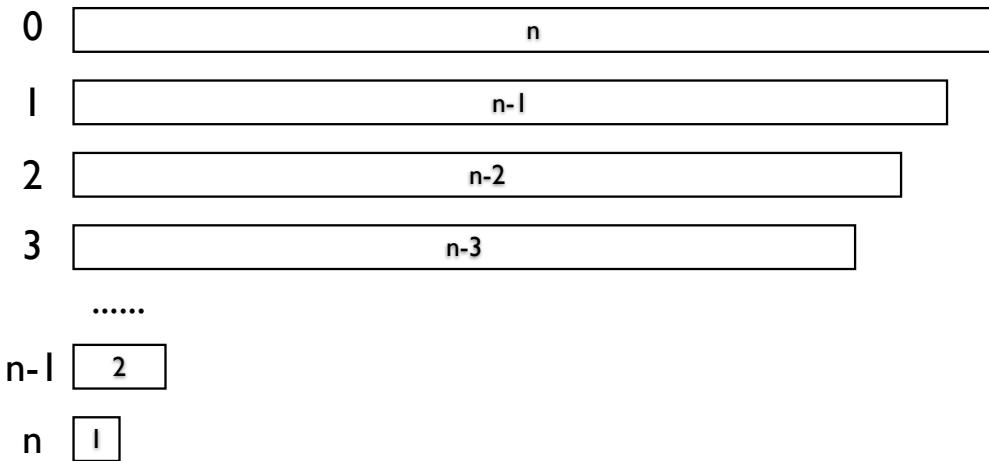
$$T(n) = T(\lfloor \frac{n}{2} \rfloor) + n \stackrel{?}{\geq} dn$$

l'ultima disequazione risulta vera per $d \leq 1$, la quale è una condizione identica a quella del caso base. Nota che non abbiamo fatto nemmeno ricordo all'ipotesi induttiva.

Cosa succede se si sbaglia l'intuizione

$$T(n) = \begin{cases} T(n-1) + n & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Nota. L'equazione di ricorrenza rappresenta il caso in cui selection sort sia espresso in forma ricorsiva.



Possiamo rappresentare la funzione nel seguente modo:

$$T(n) = \sum_{i=1}^n i = \frac{n(n+1)}{2} = \Theta(n^2)$$

Effettuiamo un tentativo e proviamo a dimostrare che $\boxed{T(n) = \mathcal{O}(n)}$

Limite superiore: Dobbiamo dimostrare che $\exists c > 0, \exists m \geq 0 : T(n) \leq cn, \forall n \geq m$.

- **caso base** lo saltiamo perché vedremmo subito che è sbagliato.
- **ipotesi induttiva** $\forall k < n : T(k) \leq ck$.
- **passo induttivo** dimostriamo la disequazione per $T(n)$:

$$\begin{aligned}
 T(n) &= T(n-1) + n \\
 &= c(n-1) + n \\
 &= cn - c + n \\
 &= (c+1)n - c \\
 &\leq (c+1)n \\
 &- g \text{ di } (c+1)n \stackrel{?}{\leq} cn \\
 &\Rightarrow c+1 \leq c
 \end{aligned}$$

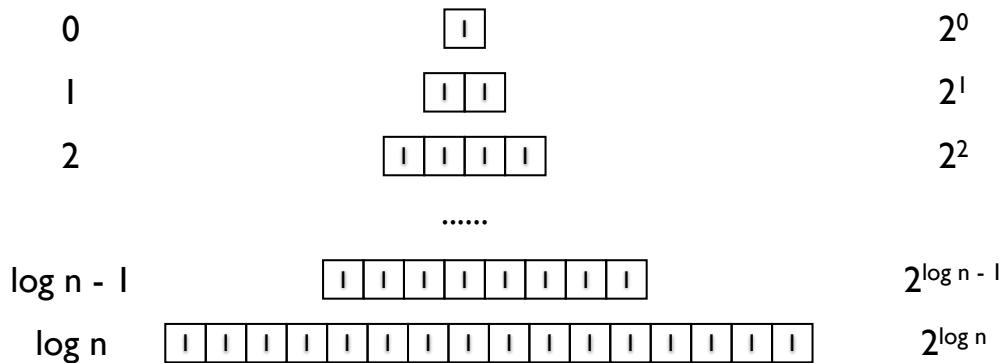
sost. ip. ind. con $k = n-1$
 multiplico
 raccolgo n
 rimuovo l'elemento negativo
 obiettivo
 semplifico

Possiamo notare che l'ultima disequazione risulta impossibile. Dunque quando proviamo a dimostrare qualcosa di sbagliato non riusciremo a dimostrarlo.

Difficoltà matematica

$$T(n) = \begin{cases} T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + 1 & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Nota. È possibile ottenere questa equazione di ricorrenza dall'algoritmo che calcola il minimo di un vettore non ordinato in maniera ricorsiva.



$$T(n) = \sum_{i=0}^{\log n} 2^i = n + \frac{n}{2} + \frac{n}{4} + \dots + 1 = \mathcal{O}(n)$$

Effettuiamo un tentativo per $T(n) = \mathcal{O}(n)$

- **ipotesi induttiva:** $\forall k < n : T(k) \leq ck$.
 - **passo induttivo:** dimostriamo la disequazione per $T(n)$:

$$\begin{aligned}
 T(n) &= T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + 1 \\
 &= c(\lfloor \frac{n}{2} \rfloor) + c(\lceil \frac{n}{2} \rceil) + 1 \\
 &= cn + 1 \\
 &= cn + 1 \stackrel{?}{\leqslant} cn \\
 \Rightarrow 1 &\leqslant 0
 \end{aligned}$$

sost. ip. ind.
semplifichiamo, $\lfloor \frac{n}{2} \rfloor + \lceil \frac{n}{2} \rceil = n$
obiettivo
semplifico

Anche in questo caso notiamo che l'ultima disequazione risulta impossibile, ma – a differenza del caso precedente – non riusciamo a dimostrare il passo induttivo per un termine di ordine inferiore. Il tentativo risulta quindi errato.

Proviamo quindi ad utilizzare un'ipotesi induttiva *più stretta*.

- ipotesi induttiva più stretta: $\exists c > 0, \exists m \geq 0 : T(n) \leq cn - b, \forall n \geq m, b > 0.$

Abbiamo introdotto una costante $b > 0$ nella nostra tesi, questa modifica ci permetterà di dimostrare correttamente il passo induttivo.

- ipotesi induttiva $\exists b > 0, \forall k < n : T(k) \leq ck - b$.

- **passo induttivo** dimostriamo la disequazione per $T(n)$:

$$\begin{aligned}
 T(n) &= T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + 1 \\
 &= c(\lfloor \frac{n}{2} \rfloor) - b + c(\lceil \frac{n}{2} \rceil) - b + 1 \\
 &= cn - 2b + 1 \\
 &= cn - 2b + 1 \stackrel{?}{\leq} cn - b \\
 \Leftrightarrow &-2b + 1 \leq -b \\
 \Leftrightarrow &b \geq 1
 \end{aligned}$$

↓ sost. ip. ind.
 ↓ semplifichiamo
 ↓ obiettivo
 ↓ semplifico

- **caso base**

$$T(1) = 1 \stackrel{?}{\leq} c \cdot 1 - b \Leftrightarrow c \geq b + 1$$

Per concludere abbiamo provato che $T(n) \leq cn - b \leq cn$ con diversi valori delle costanti c e b , nel passo induttivo $\forall b \geq 1, \forall c$, nel caso base $\forall c \geq b + 1$. Una coppia di valori di b e c che rispettano queste disequazioni sono $b = 1, c = 2$.

Questo vale per $n = 1$, e per tutti i valori di n successivi, quindi per $m = 1$.

Abbiamo quindi provato che $T(n) = \mathcal{O}(n)$.

Dimostriamo il limite inferiore facendo un tentativo per $\boxed{T(n) = \Omega(n)}$

Dobbiamo dimostrare che $\exists d > 0, \exists m \geq 0 : T(n) \geq dn, \forall n \geq m$.

- **passo induttivo**

$$\begin{aligned}
 T(n) &= T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + 1 \\
 &\geq d(\lfloor \frac{n}{2} \rfloor) + d(\lceil \frac{n}{2} \rceil) + 1 \\
 &= dn + 1 \\
 &= dn + 1 \stackrel{?}{\geq} dn
 \end{aligned}$$

↓ sost. ip. ind.
 ↓ semplifichiamo
 ↓ obiettivo

L'ultima disequazione risulta vera $\forall d$.

- **caso base**

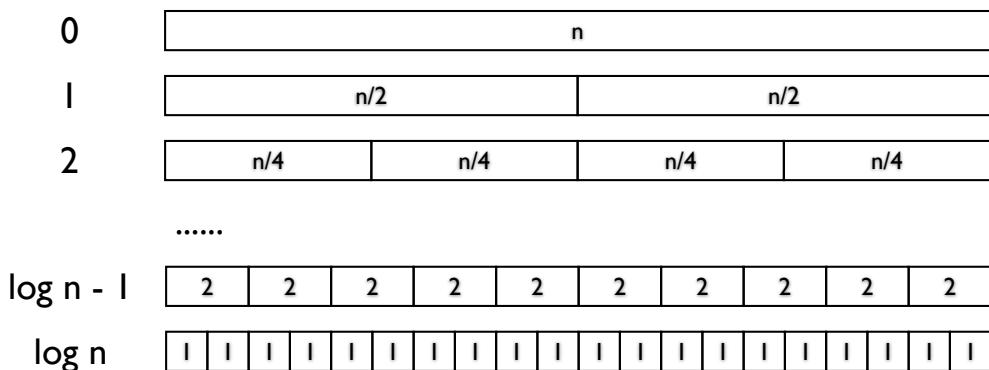
$$T(n) = 1 \geq d \cdot 1 \iff d \leq 1$$

Abbiamo quindi provato che $T(n) = \Omega(n)$

Problemi con i casi base

$$T(n) = \begin{cases} 2T(\lfloor \frac{n}{2} \rfloor) + n & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Proviamo a visualizzarlo graficamente così:



Nota. È molto simile all'equazione dell'algoritmo mergeSort che sappiamo avere una complessità di $\mathcal{O}(n \log n)$.

Effettuiamo un tentativo per $T(n) = \mathcal{O}(n \log n)$

Dobbiamo dimostrare che $\exists c > 0, \exists m \geq 0 : T(n) \leq cn \log n, \forall n \geq m$.

- **Ipotesi induttiva:** $\exists c > 0, \forall k < n : T(k) \leq ck \log k$
- **Passo di induzione:** Dimostriamo la disequazione per $T(n)$:

$$\begin{aligned}
 T(n) &= 2T(\lfloor \frac{n}{2} \rfloor) + n \\
 &\leq 2c\lfloor \frac{n}{2} \rfloor \log \lfloor \frac{n}{2} \rfloor + n \\
 &\leq 2c\frac{n}{2} \log \frac{n}{2} + n \\
 &= cn \log \frac{n}{2} + n \\
 &= cn(\log n - 1) + n \\
 &= cn \log n - cn + n \\
 &\leq cn \log n - cn + n \stackrel{?}{\leq} cn \log n \\
 &\Leftrightarrow -cn + n \leq 0 \\
 &\Leftrightarrow c \geq 1
 \end{aligned}$$

↓ sost. ip. ind. con $k = \lfloor \frac{n}{2} \rfloor$
 ↓ rimuovo l'intero inferiore
 ↓ semplifico
 ↓ $\log \frac{n}{2} = \log n - \log_2 2 = \log n - 1$
 ↓ moltiplico
 ↓ obiettivo
 ↓ semplifico

- **caso base:** dimostriamo la disequazione per $T(1)$

$$T(1) = 1 \stackrel{?}{\leq} 1 \cdot c \log 1 = 0 \Rightarrow 1 \not\leq 0$$

È falso, ma non è un problema, non a caso si chiama notazione asintotica: il valore di m lo possiamo scegliere noi.

- **caso base:** dimostriamo la disequazione per $T(2), T(3)$

$$\begin{aligned}
 T(2) &= 2T(\lfloor \frac{2}{2} \rfloor) + 2 = 4 \leq 1 \cdot c \cdot 2 \log 2 \Leftrightarrow c \geq 2 \\
 T(3) &= 2T(\lfloor \frac{3}{2} \rfloor) + 2 = 5 \leq 1 \cdot c \cdot 3 \log 3 \Leftrightarrow c \geq \frac{5}{3 \log 3} \\
 T(4) &= 2T(\lfloor \frac{4}{2} \rfloor) + 2 = 2T(\lfloor 2 \rfloor) + 4
 \end{aligned}$$

Non è necessario provare la terza disequazione, in quanto viene espressa in base ai casi base diversi da $T(1)$ che sono già stati dimostrati e quindi possono costituire la base per la nostra induzione.

Riassumendo:

Abbiamo provato che $T(n) \leq cn \log n$

- Nel passo induttivo: $\forall c \geq 1$
- Nel caso base: $\forall c \geq 2, c \geq \frac{5}{3 \log 3}$

Visto che sono tutte disequazioni con il segno \geq , è sufficiente utilizzare $c \geq \max \left\{ 1, 2, \frac{5}{3 \log 3} \right\}$

Questo vale per $n = 2, n = 3$, e per tutti i valori di n successivi, quindi $m = 2$.

Ultimo esercizio

$$T(n) = \begin{cases} 9T(\lfloor n/3 \rfloor) + n & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Effettuiamo un tentativo $\boxed{T(n) = \mathcal{O}(n^2)}$

Dobbiamo dimostrare che $\exists c > 0, \exists m \geq 0 : T(n) \leq cn^2, \forall n \geq m$

- **ipotesi induttiva:** $\exists c > 0 : T(k) \leq ck^2, \forall k < n$
- **passo induttivo** dimostriamo la disequazione per $T(n)$:

$$\begin{aligned} T(n) &= 9T\left(\lfloor \frac{n}{3} \rfloor\right) + n \\ &\leq 9c\left(\lfloor \frac{n}{3} \rfloor\right)^2 + n \\ &\leq 9c\left(\frac{n^2}{9}\right) + n \\ &= cn^2 + n \\ &= cn^2 + n \leq cn^2 \end{aligned}$$

↓ sost. ip. ind. con $k = \lfloor \frac{n}{3} \rfloor$
 ↓ rimuovo l'intero inferiore
 ↓ semplifico il 9
 ↓ obiettivo

L'ultima disequazione risulta falsa per un termine di ordine inferiore: proviamo quindi a modificare l'ipotesi induttiva e a ripetere il passo induttivo.

- **ipotesi induttiva più stretta** $\exists c > 0 : T(k) \leq c(k^2 - k), \forall k < n$
- **passo induttivo** dimostriamo la disequazione per $T(n)$:

$$\begin{aligned} T(n) &= 9T\left(\lfloor \frac{n}{3} \rfloor\right) + n \\ &\leq 9c\left(\lfloor \frac{n}{3} \rfloor^2 - \lfloor \frac{n}{3} \rfloor\right) + n \\ &\leq 9c\left(\left(\frac{n}{3}\right)^2 - \frac{n}{3}\right) + n \\ &\leq 9c\left(\frac{n^2}{9} - \frac{n}{3}\right) + n \\ &\leq cn^2 - 3cn + n \\ &\leq cn^2 - 3cn + n \stackrel{?}{\leq} cn^2 - cn \\ &\Leftrightarrow 2cn \geq cn \\ &\Leftrightarrow c \geq \frac{1}{2} \end{aligned}$$

↓ sost. ip. ind. con $k = \lfloor \frac{n}{3} \rfloor$
 ↓ rimuovo l'intero inferiore
 ↓ svolgo la potenza
 ↓ moltiplico
 ↓ obiettivo
 ↓ semplifico

- **caso base**

$$\begin{aligned} T(1) &= 1 \leq c(1^2 - 1) = 0, \text{ falso} \\ T(2) &= 9T(0) + 2 = 11 \leq c(2^2 - 2) \Leftrightarrow c \geq 11/2 \\ T(3) &= 9T(1) + 3 = 12 \leq c(3^2 - 3) \Leftrightarrow c \geq 12/6 \\ T(4) &= 9T(2) + 4 = 13 \leq c(4^2 - 4) \Leftrightarrow c \geq 13/12 \\ T(5) &= 9T(3) + 5 = 14 \leq c(5^2 - 5) \Leftrightarrow c \geq 14/20 \\ T(6) &= 9T(4) + 6 \end{aligned}$$

Non è necessario andare oltre poiché $T(6)$ dipende da $T(2)$ che è già stato dimostrato.

Riassumendo i parametri scelti sono:

- $c \geq \max\left\{\frac{1}{2}, \frac{11}{2}, \frac{12}{6}, \frac{13}{12}, \frac{14}{20}\right\}$
- $m = 1$

Nota che l'esempio combina le due difficoltà insieme, ma è stato creato artificiosamente: infatti se avessimo scelto come ipotesi più stretta $T(n) \leq cn^2 - bn$, il problema sui casi base non si sarebbe posto. Abbiamo quindi dimostrato che $T(n) \leq c(n^2 - n) \leq cn^2, \forall n \geq 1, \forall c \geq \frac{14}{20}$, ossia che $T(n) = \mathcal{O}(n)$

Riassumendo

Il metodo di sostituzione è composto da tre parti:

1. si *indovina* una possibile soluzione e si formula un'ipotesi induttiva;
2. si *sostituisce* nella ricorrenza le espressioni $T(\cdot)$, utilizzando l'ipotesi induttiva;
3. si *dimostra* che la soluzione è valida anche per il caso base.

Bisogna fare attenzione:

- ad ipotizzare soluzioni troppo “strette”
- ad alcuni casi particolari che richiedono astuzie matematiche
- ai casi base in cui compare il logaritmo in quanto potrebbe complicare le cose

3.4 Metodo dell'esperto (o delle ricorrenze comuni)

Esiste un'ampia classe di ricorrenze che possono essere risolte facilmente facendo ricorso ad alcuni teoremi, ognuno dei quali si occupa di una classe particolare di equazioni di ricorrenza.

Teorema (Ricorrenze lineari con partizione bilanciata). *Siano a e b costanti intere tale che $a \geq 1$ e $b \geq 2$, e c, β costanti reali tali che $c > 0$ e $\beta \geq 0$. Sia $T(n)$ data dalla relazione di ricorrenza:*

$$T(n) = \begin{cases} aT\left(\frac{n}{b}\right) + cn^\beta & n > 1 \\ d & n \leq 1 \end{cases}$$

Posto $\alpha = \frac{\log a}{\log b} = \log_b a$, allora:

$$T(n) = \begin{cases} \Theta(n^\alpha) & \alpha > \beta \\ \Theta(n^\alpha \log n) & \alpha = \beta \\ \Theta(n^\beta) & \alpha < \beta \end{cases}$$

Commento Affrontiamo le equazioni di ricorrenza in cui la dimensione viene divisa in b parti, dove b dev'essere almeno pari a 2; l'algoritmo ricorsivo dev'essere richiamato a volte, dove a è almeno 1. Nella versione estesa, che vedremo fra poco, vedremo che i parametri a e b verranno “rilassati”.

Prendiamo ad esempio il primo esercizio $T(n) = 4T\left(\frac{n}{2}\right)$ e vediamo che si può risolvere semplicemente calcolando $\alpha = \log_2 a = \log_2 4 = 2 > \beta = 1$, possiamo quindi concludere che $T(n) = \Theta(n^\alpha) = \Theta(n^2)$.

Dimostrazione del teorema delle ricorrenze lineari con partizione bilanciata. Assumiamo che n sia una potenza intera di b , ossia che $n = b^k$, $k = \log_b n$ poiché ci permetterà di semplificare i calcoli successivi ed è ininfluente sul risultato. Ad esempio supponiamo che l'input abbia dimensione $b^k + 1$ ($2^8 + 1 = 257$ bit), se estendiamo l'input fino ad una dimensione b^{k+1} ($2^8 + 1 = 512$ bit, facendo del *padding*, l'input sarebbe stato esteso al massimo di un fattore costante b (2 nel nostro caso), il che è ininfluente al fine della complessità computazionale.

Calcoliamo l'albero delle ricorrenze per la seguente equazione di ricorrenza:

$$T(n) = \begin{cases} aT\left(\frac{n}{b}\right) + cn^\beta & n > 1 \\ d & n \leq 1 \end{cases}$$

livello	dim.	costo chiam.	no. chiamate	costo livello
0	b^k	$cb^{k\beta}$	1	$cb^{k\beta}$
1	b^{k-1}	$cb^{(k-1)\beta}$	a	$a \cdot cb^{(k-1)\beta}$
2	b^{k-2}	$cb^{(k-2)\beta}$	a^2	$a^2 \cdot cb^{(k-2)\beta}$
\vdots	\vdots	\vdots	\vdots	\vdots
i	b^{k-i}	$cb^{(k-i)\beta}$	a^i	$a^i \cdot cb^{(k-i)\beta}$
\vdots	\vdots	\vdots	\vdots	\vdots
$k-1$	b	$cb^{(k-(k-1))\beta} = cb^\beta$	a^{k-1}	$a^{k-1} \cdot cb^\beta$
k	1	d	a^k	$a^k \cdot d$

Sommendo i costi totali del k -esimo livello e dei livelli fino al $k-1$, si ottiene:

$$\begin{aligned}
 T(n) &= da^k + \sum_{i=0}^{k-1} a^i \cdot cb^{(k-i)\beta} \\
 &= da^k + \sum_{i=0}^{k-1} a^i \cdot cb^{k\beta} \cdot b^{-i\beta} \\
 &= da^k + cb^{k\beta} \sum_{i=0}^{k-1} \frac{a^i}{b^{i\beta}} \\
 &= da^k + cb^{k\beta} \sum_{i=0}^{k-1} \left(\frac{a}{b^\beta}\right)^i
 \end{aligned}$$

moltiplico
porto fuori i termini non dipendenti da i
raccolgo i

A questo punto ho una formula chiusa ma non ancora nella sua forma definitiva.

Facciamo alcune osservazioni.

- $a^k = a^{\log_b n} = a^{\frac{\log n}{\log b}} = 2^{\log_2 a \frac{\log n}{\log b}} = 2^{\log_2 n \frac{\log a}{\log b}} = n^{\frac{\log a}{\log b}} = n^\alpha$
- $\alpha = \frac{\log a}{\log b} \Leftrightarrow \alpha \log b = \log a \Leftrightarrow \log b^\alpha = \log a \Leftrightarrow a = b^\alpha$
- poniamo $q = \frac{a}{b^\beta} = \frac{b^\alpha}{b^\beta} = b^{\alpha-\beta}$

Grazie alle osservazioni appena fatto possiamo sostituire i parametri nell'equazione finale:

$$\begin{aligned}
 T(n) &= da^k + cb^{k\beta} \sum_{i=0}^{k-1} \left(\frac{a}{b^\beta}\right)^i \\
 &= dn^\alpha + cb^{k\beta} \sum_{i=0}^{k-1} q^i
 \end{aligned}$$

sostituisco: $\alpha^k \rightarrow n^\alpha$
sostituisco: $a \rightarrow b^\alpha$, $q = \frac{b^\alpha}{b^\beta}$

Da qui si aprono tre possibilità, ossia che 1. $\alpha > \beta$ 2. $\alpha = \beta$ 3. $\alpha < \beta$ Studiamole una per una.

1. Caso $\alpha > \beta$, ne segue che $q = b^{\alpha-\beta} > 1$

$$\begin{aligned}
T(n) &= dn^\alpha + cb^{k\beta} \sum_{i=0}^{k-1} q^i \\
&= n^\alpha d + cb^{k\beta} \left[\frac{q^k - 1}{q - 1} \right] \\
&\leq n^\alpha d + cb^{k\beta} \frac{q^k}{q - 1} \\
&= n^\alpha d + \frac{cb^{k\beta} a^k}{b^{k\beta}} \frac{1}{q - 1} \\
&= n^\alpha d + \frac{ca^k}{q - 1} \\
&= n^\alpha d + \frac{cn^\alpha}{q - 1} \\
&= n^\alpha \left[d + \frac{c}{q - 1} \right]
\end{aligned}$$

serie geometrica finita
introduco la disequazione
sostituisco: $q = \frac{a}{b^\beta} \Rightarrow q^k = \frac{a^k}{b^{k\beta}}$
semplifico: $b^{k\beta}$
sostituisco: $a^k = n^\alpha$
raccolgo per n^α

Visto che d, c e q sono tutti termini positivi e costanti possiamo concludere che n^α limita superiormente l'espressione e che quindi $T(n) = \mathcal{O}(n^\alpha)$. Infine per via della componente non ricorsiva dn^α , $T(n)$ è anche $\Omega(n^\alpha)$, possiamo concludere che $T(n) = \Theta(n^\alpha)$.

2. Caso $\alpha = \beta$, ne segue che $q = b^{\alpha-\beta} = 1$

$$\begin{aligned}
T(n) &= dn^\alpha + cb^{k\beta} \sum_{i=0}^{k-1} q^i \\
&= n^\alpha d + cn^\beta k \\
&= n^\alpha d + cn^\alpha k \\
&= n^\alpha(d + ck) \\
&= n^\alpha(d + c \frac{\log n}{\log b})
\end{aligned}$$

$q^i = 1^i = 1$
 $1 + 2 + \dots + k - 1 = k$
sostituisco: $\beta = \alpha$
raccolgo per n^α
sostituisco: $k = \log_b n$

Visto che d, c e $\log b$ sono tutti termini positivi e costanti e che non abbiamo introdotto disequazioni, possiamo affermare che $T(n) = \Theta(n^\alpha \log n)$.

3. Caso $\alpha < \beta$, ne segue che $q = b^{\alpha-\beta} < 1$

$$\begin{aligned}
T(n) &= dn^\alpha + cb^k \sum_{i=0}^{k-1} q^i \\
&= dn^\alpha + cb^k \left[\frac{q^k - 1}{q - 1} \right] \\
&= dn^\alpha + cb^k \left[\frac{1 - q^k}{1 - q} \right] \\
&\leq dn^\alpha + cb^k \left[\frac{1}{1 - q} \right] \\
&= n^\alpha d + \frac{cn^\beta}{1 - q}
\end{aligned}$$

serie geometrica finita
inversione, $1 - q > 0$
introduco la disequazione
sostituisco: $b^k = n$

$n^\alpha < n^\beta$ quindi considero il polinomio di grado maggiore, di conseguenza $T(n)$ è $\mathcal{O}(n^\beta)$. Poiché $T(n) = \Omega(n^\beta)$ per via del termine non ricorsivo, possiamo affermare che $T(n) = \Theta(n^\beta)$.

Fine dimostrazione. □

Teorema (Ricorrenze lineari con partizione bilanciata estesa). *Sia $a \geq 1$, $b > 1$, $f(n)$ asintoticamente positiva, e sia*

$$T(n) = \begin{cases} aT\left(\frac{n}{b}\right) + f(n) & n > 1 \\ d & n \leq 1 \end{cases}$$

Sono dati tre casi:

1. $\exists \varepsilon > 0 : f(n) = \mathcal{O}(n^{\alpha-\varepsilon})$ allora $T(n) = \Theta(n^\alpha)$;
2. $f(n) = \Theta(n^\alpha)$ allora $T(n) = \Theta(f(n) \log n)$;
3. $\exists \varepsilon > 0 : f(n) = \mathcal{O}(n^{\alpha+\varepsilon}) \wedge \exists c : 0 < c < 1, \exists m > 0 : af\left(\frac{n}{b}\right) \leq cf(n), \forall n \geq m$ allora $T(n) = \Theta(f(n))$.

Non vedremo la dimostrazione poiché è troppo complessa.

Commento Il teorema precedente funzionava con n^β , aveva una serie di condizioni semplificative, ora non ci sono più. Nel secondo caso se $f(n) = n^\beta$ ritorniamo esattamente al secondo caso del teorema precedente, ma qui possiamo prendere in considerazione funzioni più complesse.

Esercizi

Le soluzioni sono indicate fra parentesi quadre.

- $T(n) = 9T\left(\frac{n}{3}\right) + n$ [$\mathcal{O}(n^{2-\varepsilon})$, con $\varepsilon < 1$]
- $T(n) = T\left(\frac{2}{3}n\right) + 1$ [$\Theta(n^0)$]
- $T(n) = 3T\left(\frac{n}{4}\right) + n \log n$ [$c = 3/4, m = 1$]
- $T(n) = 2T\left(\frac{n}{2}\right) + n \log n$ [nessun caso applicabile]

Teorema (Ricorrenze lineari di ordine costante). *Siano $\{a_1, a_2, \dots, a_h\}$ costanti intere non negative, con h costante positiva, c e β costanti reali tali che $c > 0$ e $\beta \geq 0$, e sia $T(n)$ definita dalla relazione di ricorrenza:*

$$T(n) = \begin{cases} \sum_{1 \leq i \leq h} a_i T(n-1) + cn^\beta & n > m \\ \Theta(1) & n \leq m \leq h \end{cases}$$

Posto $a = \sum_{1 \leq i \leq h} a_i$, allora:

1. $T(n)$ è $\Theta(n^{\beta+1})$, se $a = 1$;
2. $T(n)$ è $\Theta(a^n n^\beta)$, se $a \geq 2$.

Commento Questo teorema tratta ricorrenze lineari *di ordine costante* perché tutte le volte rimuoviamo dalla dimensione di input n una quantità costante.

Esercizi

Le soluzioni sono indicate fra parentesi.

- $T(n) = T(n-10) + n^2$ [costo polinomiale]
- $T(n) = T(n-2) - T(n-1) + 1$ [costo esponenziale]