# Esercizi Capitolo 2 - Analisi di Algoritmi

Alberto Montresor

19 Agosto, 2014

Alcuni degli esercizi che seguono sono associati alle rispettive soluzioni. Se il vostro lettore PDF lo consente, è possibile saltare alle rispettive soluzioni tramite collegamenti ipertestuali. Altrimenti, fate riferimento ai titoli degli esercizi. Ovviamente, si consiglia di provare a risolvere gli esercizi personalmente, prima di guardare la soluzione.
Per molti di questi esercizi l'ispirazione è stata presa dal web. In alcuni casi non è possibile risalire alla fonte originale. Gli autori originali possono richiedere la rimozione di un esercizio o l'aggiunta di una nota di riconoscimento scrivendo ad `alberto.montresor@unitn.it`.

# 1 Problemi

## 1.1 Esercizio 2.4 del libro

Si dimostri che tutti i polinomi crescono meno di tutti gli esponenziali, cioè che $n^k$ è $O(b^n)$, per qualsiasi $k > 0$ e $b > 1$.

**Soluzione:** Sezione 2.1

## 1.2 Esercizio 2.8 del libro

Si considerino le funzioni $f(n) = n$ e $g(n) = n^{1+\sin n}$. Si dimostri che le due funzioni non sono confrontabili in ordine di grandezza, cioè che non vale né che $f(n)$ è $O(g(n))$, né che $f(n)$ è $\Omega(g(n))$.

**Soluzione:** Sezione 2.2

## 1.3 Esercizio 2.9 del libro

Dato un vettore di $n$ interi, si vuole decidere se esiste un elemento che compare due volte. Determinare una limitazione inferiore e una superiore alla complessità di questo problema.

**Soluzione:** Sezione 2.3

## 1.4 Esercizio 2.13 del libro

Scrivere un algoritmo ricorsivo che riceve come parametro un vettore di $n$ elementi tale che il suo tempo di esecuzione $T(n)$ verifichi la relazione

$$T(n) = \begin{cases} d & \text{per } n \leq k \\ 5T(n/2) + 2T(n-1) + cn^2 & \text{per } n > k \end{cases}$$

con $c$, $d$ e $k$ costanti opportune. L'ordine di grandezza di $T(n)$ è polinomiale oppure no?

**Soluzione:** Sezione 2.4

## 1.5 Ricorrenze

Trovare la soluzione delle seguenti relazioni di ricorrenza, utilizzando il master theorem quando possibile, oppure il metodo di sostituzione o il metodo iterativo.

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
T(n) &= 4T(n/2) + n^2 \\
T(n) &= 4T(n/2) + n^3 \\
T(n) &= T(n/4) + T(3n/4) + n \\
T(n) &= T(n-d) + T(d) + n^2 \quad d \geq 1 \text{ costante} \\
T(n) &= T(n-1) + \log n \\
T(n) &= 2T(\sqrt{n}) + \log n \\
T(n) &= \sqrt{n}T(\sqrt{n}) + n \log \sqrt{n} \\
T(n) &= \sqrt{n}T(\sqrt{n}) + O(n)
\end{aligned}
$$

**Soluzione:** Sezione 2.5

## 1.6 Limite superiore per le sommatorie

Si dimostri, per induzione, che $\sum_{i=1}^{n} i^h$ è $O(n^{h+1})$.

**Soluzione:** Sezione 2.6

## 1.7 Stima di fattoriale

Dimostrare che $\log n! = \Theta(n \log n)$.

**Soluzione:** Sezione 2.7

## 1.8 Valutazione complessità

Dato il seguente algoritmo:

```
mistery(integer[] A, integer n)
  integer k ← 0
  for i ← 1 to n do
    k ← k + A[i]/n
  if n > 44 then
    return 3 · mistero(A, n − 2) + k
  else
    return k
```

se ne descriva la relazione di ricorrenza e se ne calcoli la complessità.

**Soluzione:** Sezione 2.8

## 1.9 Ordinamento funzioni

Ordinare le seguenti funzioni in accordo alla loro complessità asintotica. Si scriva $f(n) < g(n)$ se $O(f(n)) \subset O(g(n))$. Si scriva $f(n) = g(n)$ se $O(f(n)) = O(g(n))$.
Le funzioni da ordinare:

$$\begin{aligned}
f_1(n) &= 2^{n+2} \\
f_2(n) &= \log^2 n \\
f_3(n) &= (\log_n(\sqrt{n})^2 n) + 1/n^2 \\
f_4(n) &= 3n^{0.5} \\
f_5(n) &= 16^{n/4} \\
f_6(n) &= 2\sqrt{n} + 4n^{1/4} + 8n^{1/8} + 16n^{1/16} \\
f_7(n) &= \sqrt{(\log n)(\log n)} \\
f_8(n) &= \frac{n^3}{(n+1)(n+3)} \\
f_9(n) &= 2^n
\end{aligned}$$

**Soluzione:** Sezione 2.9

## 1.10 Limite superiore-inferiore

Data la sequente equazione di ricorrenza,

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

fornire un limite inferiore e un limite superiore alla complessità asintotica della funzione $T(n)$.

**Soluzione:** Sezione 2.10

## 1.11 Esercizio

Trovare un limite asintotico superiore e un limite asintotico inferiore alla seguente ricorrenza, facendo uso del metodo di sostituzione:

$$T(n) = 2T(n/8) + 2T(n/4) + n \log n$$

**Soluzione:** Sezione 2.11

## 1.12 Esercizio

Trovare un limite asintotico superiore e un limite asintotico inferiore alla seguente ricorrenza, facendo uso del metodo di sostituzione:

$$T(n) = 2T(n/8) + 2T(n/4) + n$$

**Soluzione:** Sezione 2.12

## 1.13 Limite sommatoria

Dimostrare che:

$$\sum_{i=1}^{n} \log i = \Theta(n \log n)$$

**Soluzione:** Sezione 2.13

## 1.14 Trova la costante

Per ognuna delle seguenti coppie di funzioni $f(n)$ e $g(n)$, proporre un costante $c$ appriata tale per cui $f(n) \leq c \cdot g(n)$ per tutti gli $n > 1$.

1. $f(n) = n^2 + n + 1, g(n) = 2n^3$
2. $f(n) = n\sqrt{n} + n^2, g(n) = n^2$
3. $f(n) = n^2 - n + 1, g(n) = n^2/2$

**Soluzione:** Sezione 2.14

## 1.15 Scegli la funzione

Per la soluzione di un problema, vi è stato proposto di scegliere fra un algoritmo iterativo che ha complessità $O(n^2)$ e un algoritmo ricorsivo che ha la seguente complessità:

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

Quale algoritmo preferite? Motivate la risposta.

**Soluzione:** Sezione 2.15

## 1.16 Dimostrazione

Dimostrate che per ogni costante $a > 0$, ogni costante $b > 0$, $(n + a)^b = \Theta(n^b)$.

**Soluzione:** Sezione 2.16

## 1.17 Funzione misteriosa

Determinare l'ordine di complessità della funzione della seguente funzione function():

```
integer function(integer n)
    if n = 0 then
        return 0
    else if n è dispari then
        return 1
    else
        return function(n/2) + function(n/2) + 1
```

**Soluzione:** Sezione 2.17

## 1.18 Ordinamento funzioni

Ordinare le seguenti funzioni in accordo alla notazione $O()$. Descrivete l'ordine in questo modo: $f(n) < g(n)$ se $f(n) = o(g(n))$, scrivete invece $f(n) = g(n)$ se $f(n) = \Theta(g(n))$. Esempio: $10n < 2n^2 = 3n^2 + n$.
Funzioni da ordinare: $1/n$, $2^{\log n}$, $2^{100}$, $2^{2^n}$, $2^n$, $2n \log^2 n$, $3n^{0.5}$, $4^{\log n}$, $4n^{3/2}$, $4^n$, $5n$, $6n \log n$, $\log \log n$, $\log^2 n$, $n \log_4 n$, $n^{0.01}$, $n^2 \log n$, $n^3$, $\sqrt{\log n}$, $\sqrt{n}$.

**Soluzione:** Sezione 2.18

## 1.19 Esercizio sostituzione

Risolvere la seguente equazione di ricorrenza:

$$T(n) = \begin{cases} 1 & n = 1 \\ T(\sqrt{n}) + 1 & n > 0 \end{cases}$$

- Suggerimento 1: utilizzare una sostituzione di variabile opportuna.
- Suggerimento 2: riscrivere la ricorrenza in ragione della variabile sostituita.

**Soluzione:** Sezione 2.19

# 2 Soluzioni

## 2.1 Esercizio 2.4 del libro

Vogliamo provare che $n^k = O(a^n), \forall k > 0, \forall a > 1$. La seguente proprietà è facilmente dimostrabile:

$$\lim_{n\to\infty} \frac{n^b}{a^n} = 0$$

per tutte le costanti $a, b$ con $a > 1$.
In base all'Esercizio 3.1, questo prova la nostra affermazione.

## 2.2 Esercizio 2.8 del libro

La funzione $\sin n$ oscilla fra $-1$ e $+1$; il che significa che la funzione "oscilla" fra il comportamento di $n^{1-1}$ e $n^{1+1}$ (si veda Figura 1).
Per dimostrare che $f(n)$ è $O(g(n))$, dobbiamo trovare valori $n_0 > 0$, $c > 0$ tali che $f(n) \leq cg(n)$ per qualsiasi $n > n_0$. Ma $g(n)$ è periodicamente uguale a 1, quindi tali valori non possono esistere.
Per dimostrare che $f(n)$ è $\Omega(g(n))$, dobbiamo trovare valori $n_0 > 0$, $c > 0$ tali che $f(n) \geq cg(n)$ per qualsiasi $n > n_0$. Ma $g(n)$ è periodicamente uguale a $n^2$, il che significa che tali valori non possono esistere.
Quindi le due funzioni non sono confrontabili.

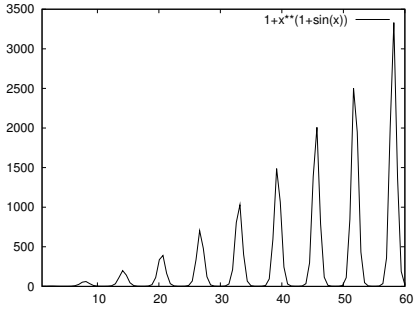


Figura 1: Funzione $n^{1+\sin n}$

## 2.3 Esercizio 2.9 del libro

Supponiamo che esistano due interi uguali nel vettore. Poiché essi possono essere collocati in qualunque posizione nel vettore, qualunque algoritmo di ricerca deve necessariamente ispezionare ogni casella. Quindi un limite inferiore alla complessità è $\Omega(n)$.
Un possibile algoritmo consiste nell'ordinare i valori e cercare poi se esistono elementi ripetuti nel vettore ordinati. Questo algoritmo costituisce un limite superiore di complessità $O(n \log n)$.

## 2.4 Esercizio 2.13 del libro

Un'assurda funzione che dà origine a quell'equazione di ricorrenza è la seguente:

```
integer crazy(integer[] A, integer i, j)
  integer c ← 0
  if (j − i > 10) then
    for k ← 1 to 5 do
      c ← c + crazy(A, i + k, ⌊(i + j)/2⌋)
    c ← c + crazy(A, i, j − 1)
    c ← c + crazy(A, i + 1, j)
    for k1 ← i to j do
      for k2 ← k1 + 1 to j do
        if A[k1] = A[k2] then
          c ← c + 1
  return c
```

Il costo della procedura è esponenziale.

## 2.5 Ricorrenze

**Ricorrenza:** $T(n) = 4T(n/2) + n$

Utilizzando il Master Theorem, si ricade nel caso 1. Infatti, $n^{\log_b a} = n^{\log_2 4} = n^2$; si ha anche che $f(n) = n = n^{2-\epsilon}$, con $\epsilon = 1 > 0$. Quindi $T(n) = \Theta(n^2)$.

**Ricorrenza:** $T(n) = 4T(n/2) + n^2$

Utilizzando il Master Theorem, si ricade nel caso 2. Infatti, $n^{\log_b a} = n^{\log_2 4} = n^2 = f(n)$. Quindi $T(n) = \Theta(n^2 \log n)$.

**Ricorrenza:** $T(n) = 4T(n/2) + n^3$

Utilizzando il Master Theorem, si ricade nel caso 3. Infatti, $n^{\log_b a} = n^{\log_2 4} = n^2 = n^{3-\epsilon}$, dove $f(n) = n^3$. Ricordiamo però che dobbiamo anche dimostrare che $a \cdot f(n/b) \leq c \cdot f(n)$ per qualche $c < 1$ ed $n$ sufficientemente grande. Infatti, $4(n/2)^3 = n^3/2 \leq cn^3$ se $c \geq 1/2$. Quindi, $T(n) = \Theta(n^3)$.

**Ricorrenza:** $T(n) = T(n/4) + T(3n/4) + n$

Qui, il master theorem non ci aiuta. Utilizziamo il metodo dell'albero di ricorsione, e vediamo che

- Al primo livello abbiamo un costo $cn$, dovuto alla componente $n$ della ricorrenza;
- Al secondo livello abbiamo un costo $\frac{1}{4}cn + \frac{3}{4}n = cn$, dovuti alla componente $n$ della ricorrenza sulle due sottoparti;
- Al terzo livello abbiamo un costo $\frac{1}{16}cn + \frac{3}{16}cn + \frac{3}{16}cn + \frac{9}{16}cn = cn$, dovuti alla componente $n$ della ricorrenza sulle quattro sottoparti

e così via. È facile vedere che questo andrà avanti per almeno $\log_4 n$ livelli, quando $n/4^i$ scenderà sotto il valore 1. I livelli successivi saranno comunque limitati superiormente da un costo $cn$, fino al livello $\log_{4/3} n$, quando $\frac{3}{4}^i$ scenderà sotto il valore 1. Quindi abbiamo che:

$$cn \log_4 n \leq T(n) \leq cn \log_{4/3} n$$

e quindi possiamo dimostrare che $T(n) = \Theta(n \log n)$.

**Ricorrenza:** $T(n) = T(n-d) + T(d) + n^2 \quad d \geq 1$ **costante**

Innanzitutto, assumiamo che il costo di $T(d)$ sia costante, in quanto $d$ è costante. Quindi riscriviamo l'equazione come:

$$T(n) = T(n-d) + cd + n^2$$

Utilizziamo il metodo dell'iterazione:

$$\begin{aligned}
T(n) &= T(n-d) + d + n^2 \\
&= T(n-2d) + d + (n-d)^2 + d + n^2 \\
&= T(n-3d) + d + (n-2d)^2 + d + (n-d)^2 + d + n^2 \\
&= \dots \\
&= n + \sum_{i=0}^{n/d} (n - id)^2 \\
&= n + \sum_{i=0}^{n/d} (n^2 - 2nid + i^2d^2) \\
&= n + \sum_{i=0}^{n/d} n^2 - \sum_{i=0}^{n/d} 2nid + \sum_{i=0}^{n/d} i^2d^2 \\
&= n + n^3/d + n^2 - 2nd\frac{(n/d+1)n/d}{2} + d^2 \sum_{i=0}^{n/d} i^2 \\
&= n + n^3/d + n^2 - n^3/d - n^2 + d^2 \frac{n/d(n/d+1)(2nd/d+1)}{6} \\
&= \Theta(n^3)
\end{aligned}$$

**Ricorrenza:** $T(n) = T(n-1) + \log n$

Sulla base dell'esercizio precedente, proviamo a dimostrare che $T(n) = O(n \log n)$. Utilizziamo il metodo di sostituzione. La nostra ipotesi induttiva è che $T(m) \leq cm \log m$ per tutti gli $m < n$, e vogliamo dimostare che la proprietà è vera per $n$.

$$\begin{aligned}
T(n) &= T(n-1) + \log n \\
&\leq c(n-1)\log(n-1) + \log n \\
&\leq c(n-1)\log n + \log n \\
&= cn \log n - c \log n + \log n
\end{aligned}$$

Abbiamo che $cn \log n - c \log n + \log n \leq cn \log n$ se $c \geq 1$.
Per quanto riguarda il caso base, il valore 1 non puó essere utilizzato perché $\log 1 = 0$. Ma se prendiamo $T(2) = T(1) + \log 2 = 1 + 1 = 2 \leq c2 \log 2 = 2c$, abbiamo ancora una volta che $c \geq 1$.

**Ricorrenza:** $T(n) = 2T(n^{1/2}) + \log n$

Questo non è banale: dobbiamo utilizzare qualche trucco algebrico. Quindi, poniamo $n = 2^m$. La ricorrenza viene riscritta nel modo seguente:

$$T(2^m) = 2T(2^{m/2}) + \log 2^m$$

Ora poniamo $S(m) = T(2^m)$, sostituiamo e semplifichiamo:

$$S(m) = 2S(m/2) + m$$

la cui soluzione sappiamo essere $S(m) = \Theta(m \log m)$. Sapendo che $m = \log n$, otteniamo

$$T(n) = \Theta(\log n \log \log n)$$

**Ricorrenza:** $T(n) = \sqrt{n}T(\sqrt{n}) + n\log\sqrt{n}$

Poniamo $n = 2^k$ (oppure $k = \log n$). Sostituendo nella ricorrenza otteniamo:

$$\begin{aligned} T(2^k) &= 2^{\frac{k}{2}}T(2^{\frac{k}{2}}) + 2^k \log 2^{\frac{k}{2}} \\ &= 2^{\frac{k}{2}}T(2^{\frac{k}{2}}) + 2^k\frac{k}{2} \end{aligned}$$

C'è questo termine $2^k$ che dà fastidio; dividiamo tutto per $2^k$ e otteniamo:

$$\frac{T(2^k)}{2^k} = \frac{T(2^{\frac{k}{2}})}{2^{\frac{k}{2}}} + \frac{k}{2}$$

Poniamo ora $S(k) = \frac{T(2^k)}{2^k}$; otteniamo quindi:

$$S(k) = S(k/2) + \frac{k}{2}$$

A questo punto, possiamo utilizzare il Master Theorem, e ottenere che

$$S(k) = \Theta(k)$$

A questo punto, possiamo tornare indietro e ottenere:

$$\begin{aligned} \frac{T(2^k)}{2^k} &= \Theta(k) \Rightarrow \\ T(2^k) &= \Theta(k)2^k \Rightarrow \\ T(n) &= \Theta(\log n)n \Rightarrow \\ T(n) &= \Theta(n\log n) \end{aligned}$$

**Ricorrenza:** $T(n) = \sqrt{n}T(\sqrt{n}) + O(n)$

Esplicitando le costanti, e ponendo $n = 2^k$ (oppure $k = \log n$) nella ricorrenza, otteniamo:

$$\begin{aligned} T(n) &\leq \sqrt{n}T(\sqrt{n}) + cn \Rightarrow \\ T(2^k) &\leq 2^{\frac{k}{2}}T(2^{\frac{k}{2}}) + c2^k \end{aligned}$$

Dividiamo tutto per $2^k$, e otteniamo:

$$\frac{T(2^k)}{2^k} \leq \frac{T(2^{\frac{k}{2}})}{2^{\frac{k}{2}}} + c$$

Poniamo ora $S(k) = \frac{T(2^k)}{2^k}$; otteniamo quindi:

$$S(k) = S(k/2) + \Theta(1)$$

A questo punto, possiamo utilizzare il Master Theorem, e ottenere che

$$S(k) = \Theta(\log k)$$

A questo punto, possiamo tornare indietro e ottenere:

$$\begin{aligned} \frac{T(2^k)}{2^k} &= \Theta(\log k) \Rightarrow \\ T(2^k) &= \Theta(\log k)2^k \Rightarrow \\ T(n) &= \Theta(\log\log n)n \Rightarrow \\ T(n) &= \Theta(n\log\log n) \end{aligned}$$

## 2.6 Limite superiore per le sommatorie

- Caso base ($h = 0$):
$$\sum_{i=1}^{n} i^0 = \sum_{i=1}^{n} 1 = n = n^{h+1}$$
- Passo induttivo. Supponiamo che la proprietà sia vera per ogni $k < h$; vogliamo dimostrare che la proprietà è vera per $h$:
$$\sum_{i=1}^{n} i^h = \sum_{i=1}^{n} i^{h-1} i \leq \sum_{i=1}^{n} i^{h-1} n = n \sum_{i=1}^{n} i^{h-1} = nO(n^h) = O(n^{h+1})$$

## 2.7 Stima di fattoriale

Per dimostrare che $\log n! = \Theta(n \log n)$, dobbiamo dimostrare che $\log n! = O(n \log n)$ e $\log n! = \Omega(n \log n)$.

- $\log n! = O(n \log n)$
  È banalmente vero che:
$$n! = n \cdot (n-1) \cdot \ldots \cdot 1 \leq \underbrace{n \cdot n \cdot \ldots n}_{n} = n^n$$
  Quindi,
$$\log n! \leq \log n^n = n \log n$$
- Metà dei fattori in $n!$ superano $n/2$, l'altra metà supera 1. Quindi:
$$n! \geq \underbrace{n/2 \cdot \ldots \cdot n/2}_{\frac{n}{2}} \cdot \underbrace{1 \cdot \ldots \cdot 1}_{\frac{n}{2}} = \left(\frac{n}{2}\right)^{\frac{n}{2}}.$$
  Applicando il logaritmo, otteniamo:
$$\log n! \geq \log \left(\frac{n}{2}\right)^{\frac{n}{2}} = \frac{n}{2} \log \frac{n}{2} = \frac{n}{2} \log n - \frac{n}{2} \log 2 = \Omega(n \log n)$$

## 2.8 Valutazione complessità

La funzione di ricorrenza può essere descritta in questo modo:

$$T(n) = \begin{cases} T(n-2) + n & n > 44 \\ n & n \leq 44 \end{cases}$$

Supponiamo che $n$ sia pari. Utilizzando il metodo dell'iterazione, possiamo scrivere:

$$\begin{aligned} T(n) &= T(n-2) + n \\ T(n) &= T(n-4) + n - 2 + n \\ T(n) &= T(n-6) + n - 4 + n - 2 + n \end{aligned}$$

Svolgendo completamente l'iterazione, possiamo ottenere il seguente risultato

$$\begin{aligned} T(n) &= \sum_{i=23}^{n/2} 2i \\ &= 2(\sum_{i=0}^{n/2} i - \sum_{i=0}^{22} i) \\ &= O(n^2) \end{aligned}$$

## 2.9 Ordinamento funzioni

Le funzioni da ordinare:

$$\begin{aligned}
f_1(n) &= 2^{n+2} = 4 \cdot 2^n = \Theta(2^n) \\
f_2(n) &= \log^2 n = \Theta(\log^2 n) \\
f_3(n) &= (\log_n(\sqrt{n})^2 n) + 1/n^2 = (\log_n n^2) + 1/n^2 = 2 + 1/n^2 = \Theta(1) \\
f_4(n) &= 3n^{0.5} = \Theta(n^{1/2}) \\
f_5(n) &= 16^{n/4} = (2^4)^{n/4} = 2^{4n/4} = \Theta(2^n) \\
f_6(n) &= 2\sqrt{n} + 4n^{1/4} + 8n^{1/8} + 16n^{1/16} = \Theta(n^{1/2}) \\
f_7(n) &= \sqrt{(\log n)(\log n)} = \Theta(\log n) \\
f_8(n) &= \frac{n^3}{(n+1)(n+3)} = \Theta(n) \\
f_9(n) &= 2^n = \Theta(2^n)
\end{aligned}$$

Una volta stabilito l'ordine $\Theta$ delle funzioni, è abbastanza semplice stabilire l'ordine corretto:

$$f_3 < f_7 < f_2 < f_4 = f_6 < f_8 < f_1 = f_5 = f_9$$

## 2.10 Limite superiore-inferiore

È facile osservare le seguenti relazioni:

$$T(n) = 2T(n/3) + T(2n/3) + 1 \geq 2T(n/3) + T(n/3) + 1 = 3T(n/3) + 1$$

e

$$T(n) = 2T(n/3) + T(2n/3) + 1 \leq 2T(2n/3) + T(2n/3) + 1 = 3T(2n/3) + 1$$

Applicando il master theorem alle due relazioni, si ottiene che:

$$T(n) \geq 3T(n/3) + 1 = \Omega(n)$$

e

$$T(n) \leq 3T(2n/3) + 1 = O(n^{\log_{3/2} 3}) = O(n^{2.70\ldots})$$

È possibile dimostrare un limite più stretto provando a dimostrare che $T(n) = O(n^2)$ per sostituzione. Nel caso generale, dobbiamo dimostrare che esistono $c > 0$, $m \geq 0$ tale per cui $T(n) \leq cn^2$ per ogni $n \geq m$.

$$\begin{aligned}
T(n) &= 2T(n/3) + T(2n/3) + 1 \\
&\leq 2cn^2/9 + 4cn^2/9 + 1 \\
&= 2/3cn^2 + 1 \leq cn^2.
\end{aligned}$$

L'ultima condizione è vera per $n \geq m = 1$, $c \geq 3$.
Nel caso base, $T(1) = 1 \leq c \cdot 1^2$, che è vera per $c \geq 1$.

## 2.11 Esercizio

Il limite inferiore è banalmente $\Omega(n \log n)$:

$$T(n) = 2T(n/8) + 2T(n/4) + n \log n \geq n \log n = \Omega(n \log n)$$

Il limite superiore è ancora $n \log n$. Lo dimostriamo per sostituzione; supponiamo quindi che esistano $c, n_0$ tali che $T(n) \leq cn \log n$ per ogni $n \geq n_0$.

$$\begin{aligned} T(n) &= 2T(n/8) + 2T(n/4) + n \\ &\leq 2cn/8 \log n/8 + 2cn/4 \log n/4 + n \log n \\ &\leq cn/4 \log n + cn/2 \log n + n \log n \\ &= 3/4cn \log n + n \log n \leq cn \log n \end{aligned}$$

L'ultima disequazione è vera per $c \geq 4$.
Il caso base non è dimostrabile per $n = 1$, ma è facile vedere che è rispettato per $n \leq 16$.

## 2.12 Esercizio

È facile dimostrare che ovviamente, la funzione $T(n)$ è $\Omega(n)$; infatti,

$$T(n) = 2T(n/8) + 2T(n/4) + n \geq n \geq c_1 n$$

per $c_1 \leq 1$. Non è necessario dimostrare il caso base, perché abbiamo rimosso gli elementi ricorsivi e quindi non abbiamo bisogno di induzione.
Una prima osservazione che si potrebbe fare per "indovinare" un limite superiore è la seguente:

$$\begin{aligned} T(N) &= 2T(n/8) + 2T(n/4) + n \\ &\leq 2T(n/4) + 2T(n/4) + n \\ &\leq 4T(n/4) + n \end{aligned}$$

In base a questo risultato, il master theorem dice che $T(n) = O(n \log n)$. E' facile dimostrare questo risultato.
A questo punto, non ci resta che vedere quale dei due risultati ($\Omega(n)$ e $O(n \log n)$ è stretto. Proviamo con $O(n)$.
Ipotesi induttiva: $\forall n < n' : T(n) \leq cn$. Proviamo che il risultato è valido anche per $n$.

$$\begin{aligned} T(n) &= 2T(n/8) + 2T(n/4) + n \\ &\leq 2cn/8 + 2cn/4 + n \\ &= 3/4cn + n \\ &\leq cn \end{aligned}$$

L'ultima disequazione è vera se $3/4c + 1 \leq c$, ovvero se $c \geq 4$.
Quindi la nostra funzione è $\Theta(n)$.

## 2.13 Limite sommatoria

**Soluzione, parte $O$** Dalla definzione di $\Theta$, dobbiamo provare che esistono due valori $c > 0$ e $n_0 > 0$ tali che:

$$\sum_{i=1}^{n} \log i \leq cn \log n, \forall n > n_0$$

Facile:

$$\begin{aligned} \sum_{i=1}^{n} \log i &\leq \sum_{i=1}^{n} \log n \\ &= n \log n \end{aligned}$$

**Soluzione, parte $\Omega$** Dalla definzione di $\Theta$, dobbiamo provare che esistono due valori $c > 0$ e $n_0 > 0$ tali che:

$$\sum_{i=1}^{n} \log i \geq cn \log n, \forall n > n_0$$

Lo proviamo in questo modo:

$$\begin{aligned}
\sum_{i=1}^{n} \log i &= \log 1 + \ldots + \log(\frac{n}{2} - 1) + \log \frac{n}{2} + \ldots + \log n \\
&\geq 0 + \ldots + 0 + \log \frac{n}{2} + \ldots + \log \frac{n}{2} \\
&= \frac{n}{2} \log \frac{n}{2} \\
&= \frac{n}{2} \log n - n \log 2 \\
&= \frac{n}{2} \log n - n \\
&= \Omega(n \log n)
\end{aligned}$$

## 2.14 Trova la costante

1. La disequazione è vera per qualsiasi costante $c \geq 7/16$; (si può ottenere notando che per $n > 1$, $f(n)$ è sempre maggiore di $g(n)$; quindi basta sostituire $n = 2$ per ottenere il valore minore di $c$)
2. Simili considerazioni si possono fare per questa disequazione, ottenendo $c \geq \frac{\sqrt{2}+2}{2}$
3. $c \geq 2$

## 2.15 Scegli la funzione

Per $a \leq 16$ si ottiene una complessità pari a $O(n^2)$, quindi è l'algoritmo ricorsivo è preferibile (o uguale) all'algoritmo iterativo. Per valori di $a$ maggiori di 16, l'algoritmo iterativo è preferibile.

## 2.16 Dimostrazione

Per dimostrare che $(n + a)^b = \Theta(n^b)$, dobbiamo trovare tre costanti $c_1$, $c_2$ , $n_0 > 0$ tali che

$$0 \leq c_1 n^b \leq (n + a)^b \leq c_2 n^b, \forall n \geq n_0$$

Si noti che $n + a \leq 2n$, per $a \leq n$; e che $n + a \geq \frac{1}{2}n$, per qualsiasi $n$. Quindi, per $n \geq a$,

$$0 \leq \frac{1}{2}n \leq n + a \leq 2n$$

Poiché $b > 0$, si ha che

$$0 \leq \frac{1}{2}^b n^b \leq (n + a)^b \leq 2^b n^b$$

quindi $c_1 = \frac{1}{2}^b$, $c_2 = 2^b$ e $n_0 = a$.

## 2.17 Funzione misteriosa

La funzione $T(n)$ è definita dalla seguente equazione di ricorrenza:

$$T(n) = \begin{cases} \alpha & n = 0 \vee (n \mod 2) = 0 \\ 2T(n/2) + \beta & \text{altrimenti} \end{cases}$$

Il caso pessimo si ha quando $n$ è una potenza di 2. In quel caso, è possibile applicare il Master theorem, ed ottenere che la complessità è pari a $O(n)$.

## 2.18 Ordinamento funzioni

$1/n < 2^{100} < \log\log n < \sqrt{\log n} < \log^2 n < n^{0.01} < 3n^{0.5} = \sqrt{n} < 2^{\log n} = 5n < 6n\log n = n\log_4 n < 2n\log^2 n < 4n^{3/2} < 4^{\log n} < n^2\log n < n^3 < 2^n < 4^n < 2^{2^n}$

## 2.19 Esercizio sostituzione

Poniamo $n = 2^k$, ovvero $k = \log n$. Ne segue:

$$T(2^k) = T(2^{(k/2)}) + 1$$

Definiamo $S(p) = T(2^p)$.

- Sostituendo $T$ con $S$: $S(k) = S(k/2) + 1$
- Per il caso 2 del Master Theorem: $f(n) = 1 = \Theta(1) = \Theta(k^{\log_2 1})$.
- Quindi: $S(k) = \Theta(\log k)$.
- Risostituiamo $k = \log n$: $S(k) = T(2^k) = \Theta(\log k)$
- Quindi, $T(n) = \Theta(\log\log n)$.

# 3 Problemi aperti

## 3.1 Esercizio 2.3 del libro

Si mostri che, per stabilire se una funzione $g(n)$ è $\Omega(f(n))$, si può calcolare $\lim_{n\to\infty} g(n)/f(n)$ ed affermare che se il limite esiste ed è una costante $c > 0$ oppure $\infty$, allora $g(n)$ è $\Omega(f(n))$. Come si può inoltre stabilire se $g(n)$ è $\Theta(f(n))$?

## 3.2 Ricorrenze

Risolvere le seguenti ricorrenze:

- $T(n) = 2T(n/8) + 2T(n/4) + n, \forall n > 1$
- $T(n) = 2T(2n/3) + n^3, \forall n > 1$

evitando di utilizzare tecniche per la risoluzione di equazioni di ricorrenza comuni e assumendo $T(1) = 1$.