Esercizi Complessità

1. Risolvere con l'albero della ricorsione l'equazione di ricorrenza che rappresenta la complessità in tempo del seguente algoritmo: Provate per induzione la stima fatta.

```
Pippo( int n):int
if n ≤ 5 then
    return: 3
else
    int i = 1;
    while i ≤ n do
        for j := 1; j++; j ≤ n do
            i:= i + 3
    Pippo = Pippo(n/5) + Pippo(n/7)
```

2. Risolvere la seguente equazione di ricorrenza:
   $T(n) = 4T(\frac{n}{2}) + O(n^3)$

3. Risolvere la seguente equazione di ricorrenza:
   $T(n) = \begin{cases} O(1), & n \leq 2 \\ 2T(\sqrt{n}) + \log_2 n \end{cases}$

4. Risolvere la seguente equazione di ricorrenza:
   $T(n) = \begin{cases} T(n-5) + O(1), & n > 5 \\ O(1), & n \leq 5 \end{cases}$

Soluzioni

1. Per stimare la complessità in tempo del codice proposto è necessario identificare l'equazione di ricorrenza. Quindi per prima cosa studiamo il costo del passo combina, i.e., i due cicli annidati.

   Consideriamo il costo del FOR più interno. Tale FOR parte da 1 fino ad n con passo 1, lo possiamo dunque espriremere come:

   $$\sum_{j=1}^{n} \mathcal{O}(1) = \Theta(n)$$

   Studiamo ora come cresce la variabile i, essa viene incrementata di 3 ad ogni iterazione del ciclo FOR. Pertanto, al termine del FOR più interno ($n+1$ passi) la variabile i avrà raggiunto il valore $3n+1$ che è un valore maggiore di n; ciò implica che il ciclo WHILE uscirà dopo una sola iterazione, i.e., può essere considerato di costo costante.

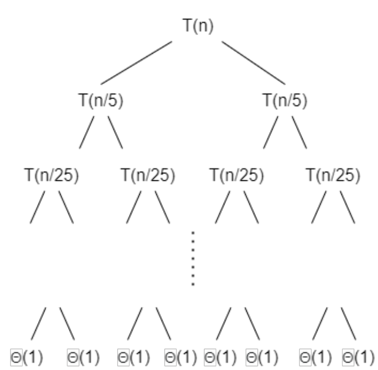
   Possiamo quindi definire l'equazione di ricorrenza:

   $$T(n) = \begin{cases} T(n/5) + T(n/7) + \mathcal{O}(n) & \text{if} \quad n > 5 \\ \Theta(1) & \text{if} \quad n \leq 5 \end{cases}$$

   Dall'equazione appena scritta è facile osservare che si avrà un albero binario sbilanciato. Dunque procediamo con una maggiorazione per semplificare la stima di un upper-bound alla nostra ricorrenza.

   $$T'(n) = \begin{cases} 2T'(n/5) + \mathcal{O}(n) & \text{if} \quad n > 5 \\ \Theta(1) & \text{if} \quad n \leq 5 \end{cases}$$

   Costruiamo l'albero di ricorrenza:



   Nodo generico: $T(n/5^i)$

   Altezza: $\frac{n}{5^i} \leq 5 \Rightarrow i \geq \log_5 n - 1$

   Costo livello generico: $2^i(\frac{n}{5^i})$

   Numero foglie: $2^{\log_5 n} = (n)^{\log_5 2}$

   Possiamo quindi stimare un costo per la nostra ricorrenza:

   $$\sum_{i=0}^{\log_5 n - 1} (2^i \frac{n}{5^i}) + n^{\log_5 2} \quad < \quad n \sum_{i=1}^{\infty} (\frac{2}{5})^i + n^{\log_5 2} \quad = \quad \frac{5}{3} n + n^{\log_5 2} \Rightarrow \quad \in \mathcal{O}(n)$$

   Proviamo il risultato ottenuto per induzione:

   Ipotesi: $\exists c > 0, n_0 > 0 : \forall n \geq n_0 T(n) \leq \mathcal{O}(n)$

   Ricordo che per fare la prova dobbiamo usare l'equazione originale, quindi:

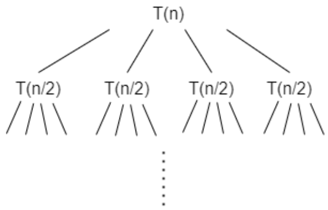
   HP: $T(\frac{n}{5}) \leq c\frac{n}{5}$ e $T(\frac{n}{7}) \leq c\frac{n}{7}$

   $$T(n) \leq c\frac{n}{5} + c\frac{n}{7} + \mathcal{O}(n)$$
   $$\leq c\frac{12n}{35} + c_1 n$$

In particolare

$$
\begin{aligned}
&c\frac{12n}{35} + c_1 n \leq cn \\
&c(1 - \frac{12n}{35}) \geq c_1 n \\
&c \geq \frac{12c_1}{35} \Rightarrow \quad T(n) \in \mathcal{O}(n)
\end{aligned}
$$

2. Costruiamo l'albero di ricorrenza:



Nodo generico: $T(n/2^i)$

Altezza: $\frac{n}{2^i} \leq 1 \Rightarrow i > \log n$

Costo livello generico: $4^i (\frac{n}{2^i})^3$

Numero foglie: **In questo caso la ricorrenza non presenta un caso base $\Rightarrow$ non abbiamo foglie**

Possiamo quindi stimare un costo per la nostra ricorrenza:

$$\sum_{i=0}^{\log n} 4^i (\frac{n}{2^i})^3 \quad = \quad n^3 \sum_{i=0}^{\log n} (\frac{1}{2^i}) \quad < \quad n^3 \sum_{i=0}^{\infty} (\frac{1}{2^i}) \leq 2n^3 \quad \Rightarrow \quad \in \mathcal{O}(n^3)$$

Proviamo il risultato ottenuto per induzione:

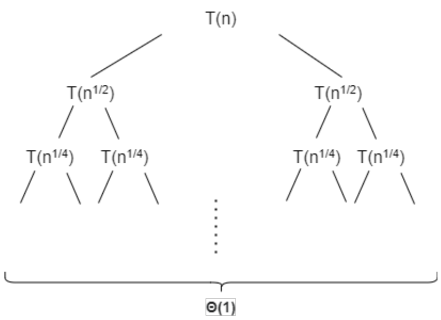
Ipotesi: $\exists c > 0, n_0 > 0 : \forall n \geq n_0 \quad T(n/2) \leq c(\frac{n}{2})^3$

$$\begin{aligned} T(n) &\leq c4(\frac{n}{2})^3 + \mathcal{O}(n^3) \\ &\leq c\frac{n^3}{2} + c_1 n^3 \end{aligned}$$

In particolare

$$c\frac{n^3}{2} + c_1 n^3 < cn^3$$
$$c \geq \frac{2c_1 n^3}{n^3} = 2c_1 \Rightarrow T(n) \in O(n^3)$$

3. Costruiamo l'albero di ricorrenza:



Nodo generico: $T(n^{\frac{1}{2}^i})$

Altezza: $n^{\frac{1}{2}i} \leq 2 \Rightarrow i > \log\log n$

Costo livello generico: $2^i(\log n^{\frac{1}{2}^i}) = \log n$

Numero foglie: $2^{\log\log n} = \log n$

Possiamo quindi stimare un costo per la nostra ricorrenza:

$$\sum_{i=0}^{\log\log n} \log n + \log n \quad = \quad \log n \sum_{i=0}^{\log\log n} \mathcal{O}(1) + \log n \quad = \quad \log n \log\log n + \log n \quad \Rightarrow \quad \in \mathcal{O}(\log n \log\log n)$$

Proviamo il risultato ottenuto per induzione:

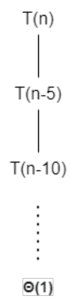
Ipotesi: $\exists c > 0, n_0 > 0 : \forall n \geq n_0 \quad T(n^{\frac{1}{2}}) \leq c \log n^{\frac{1}{2}} \log\log n^{\frac{1}{2}}$

$$\begin{aligned} T(n) &\leq c2 \log n^{\frac{1}{2}} \log\log n^{\frac{1}{2}} + \log n \\ &\leq c \log n \log(\frac{\log n}{2}) + \log n \\ &\leq c \log n \log\log n - \log 2 \log n \end{aligned}$$

In particolare

$$\begin{aligned} &c \log n \log\log n - \log 2 \log n \leq c \log n \log\log n \\ &-c \log n + \log n \leq 1 \\ &c \geq 1 \Rightarrow T(n) \in O(\log n \log\log n) \end{aligned}$$

4. Costruiamo l'albero di ricorrenza:



Nodo generico: $T(n-5i)$

Altezza: $n-5i \leq 5 \Rightarrow i \geq \frac{n}{5}-1$

Costo livello generico: $\mathcal{O}(1)$

Numero foglie: 1

Possiamo quindi stimare un costo per la nostra ricorrenza:

$$\sum_{i=0}^{\frac{n}{5}-1} \mathcal{O}(1)+1=\frac{n}{5}-1+1 \Rightarrow \quad \in \mathcal{O}(n)$$

Proviamo il risultato ottenuto per induzione:

Ipotesi: $\exists c>0, n_0>0 : \forall n \geq n_0 \quad T(n-5) \leq c(n-5)$

$$T(n) \leq c(n-5)-\mathcal{O}(1)$$

In particolare

$$cn-5c+\mathcal{O}(1) \leq cn$$
$$5c \geq \mathcal{O}(1)$$
$$c \geq \frac{\mathcal{O}(1)}{5}=\frac{c_1}{5} \Rightarrow T(n) \in O(n)$$