Introduzione agli Algoritmi

Primo esonero secondo canale con spunti per la soluzione

docenti: A. Monti

Sapienza Università di Roma

Aprile 2024

Esercizio 1 (10 punti): Si calcoli la complessità asintotica della seguente funzione iterativa:

```
def es1(n):
    z, x, y = 0, 0, n
    while y > 1:
        x += 1
        y = y//3
    x = x * x
    while y < n:
        z += 5
        y += x
    return z
```

a) Ad ogni iterazione del primo while il valore di y viene diviso per 3. Poiché all'inizio del while y vale n, dopo i iterazioni si avrà $y \approx \frac{n}{3^i}$ ed il while termina quando $\frac{n}{3^i} \approx 1$ vale a dire $i = \log_3 n$. Quindi la complessità del primo while è $\Theta(\log n)$.

All'uscita del primo while il valore di x sarà circa $\log_3 n$ e dopo essere stato moltiplicato per se stesso diverrà $\log_3^2 n$. Ad ogni iterazione del secondo while il valore di y (che parte da circa 1) verrà incrementato di circa $\log_3^2 n$ quindi dopo i iterazione y varrà circa $i \log_3^2 n$ e si uscirà dal while quando $i \log_3^2 n \approx n$ il che significa $i \approx \frac{n}{\log_3^2 n}$. Il secondo while ha dunque complessità $\Theta\left(\frac{n}{\log^2 n}\right)$. Per quanto detto la complessità della funzione $es1$ sarà $\Theta(\log n) + \Theta\left(\frac{n}{\log^2 n}\right) = \Theta\left(\frac{n}{\log^2 n}\right)$.

Esercizio 2 (20 punti): Si consideri la seguente funzione ricorsiva:

```
def es2(n):
    if n <= 1:
        return 5
    j= x = 0
    while j*j<n:
        if j < 2:
            x += 2*es2(n//4) + 3
        j += 2
        x += 3
    return x
```

a) Si imposti la relazione di ricorrenza che ne definisce il tempo di esecuzione **giustificando dettagliatamente l'equazione ottenuta**.

b) Si risolva la ricorrenza usando prima il metodo principale e poi a vostra scelta uno trai il metodo iterativo e il metodo di sostituzione.

a) Nel while si parte con $y = 0$ ma gia dopo la prima iterazione si ha $j = 2 > 1$ quindi il programma effettua un'unica chiamata ricorsiva e su un sottoproblema di dimensione $n//4$. Ad ogni iterazione del while la variabile j viene incrementata di 2, dopo i iterazioni avremo $j = 2i$ quindi si uscirà dal while quando $(2i)^2 \approx n$ il che vuol dire $i \approx \frac{\sqrt{n}}{4}$. Il numero di iterazioni effettuate sarà dunque $\Theta(\sqrt{n})$.
Per quanto detto l'equazione di ricorrenza che caratterizza il programma è:

$T(n) = T\left(\frac{n}{4}\right) + \Theta(\sqrt{n})$ se $n > 1$
$T(n) = \Theta(1)$ altrimenti.

b) Per risolvere con il metodo principale notiamo che $a = 1$ e $b = 4$ da cui $n^{log_b a} = n^{log_4 1} = n^0 = 1$ poiché $f(n) = \Theta(n^{1/2})$ si ha che $f(n)$ domina fortemente $n^{log_b a}$ (basta ad esempio prendere $\epsilon = 1/4$) inoltre $a \cdot f\left(\frac{n}{b}\right) = \sqrt{\frac{n}{4}} = \frac{1}{2}\sqrt{n} \leq c \cdot \sqrt{n}$ con $c = \frac{1}{2} < 1$. Siamo dunque nel terzo caso del teorema e possiamo concludere che $T(n) = \Theta(\sqrt{n})$.

- **(Metodo iterativo.)**

$$\begin{aligned} T(n) &= T\left(\frac{n}{4}\right) + \Theta(\sqrt{n}) \\ &= \left(T\left(\frac{n}{4^2}\right) + \Theta\left(\sqrt{\frac{n}{4}}\right)\right) + \Theta(\sqrt{n}) \\ &= \left(T\left(\frac{n}{4^3}\right) + \Theta\left(\sqrt{\frac{n}{4^2}}\right)\right) + \Theta\left(\sqrt{\frac{n}{4}}\right) + \Theta(\sqrt{n}) \\ &= \ldots\ldots \\ &= T\left(\frac{n}{4^i}\right) + \sum_{j=0}^{i-1} \Theta\left(\sqrt{\frac{n}{4^j}}\right) \end{aligned}$$

Ci fermiamo quando $\frac{n}{4^i} \approx 1$ **vale a dire** $i = \log_4 n$ **ed otteniamo**

$$T(n) = T(1) + \Theta\left(\sqrt{n} \sum_{j=0}^{\log_4 n - 1} \frac{1}{2^j}\right) = \Theta(1) + \Theta(\sqrt{n})\Theta(1) = \Theta(\sqrt{n})$$

Dove abbiamo usato che $\sum_{j=0}^{x} c^j = \Theta(1)$ **quando** $c < 1$**.**

· **(Metodo di sostituzione.) Per prima cosa ci liberiamo dell'asintotica:**
$T(n) = T\left(\frac{n}{4}\right) + b\sqrt{n}$ **se** $n > 1$
$T(n) = a$ **altrimenti.**
dove a **e** b **sono costanti positive.**

- **per dimostrare** $T(n) = O(\sqrt{n})$ **proviamo per induzione che vale** $T(n) \leq c\sqrt{n}$ **per una costante** c **maggiore di zero da definire.**
  Per il caso base abbiamo $T(1) = a \leq c \cdot 1$ **che è vera se prendiamo** $c \geq a$**.**
  Per il passo induttivo:
  $T(n) \leq c\sqrt{\frac{n}{4}} + b\sqrt{n} = \frac{c}{2}\sqrt{n} + b\sqrt{n} \leq c\sqrt{n}$ **dove l'ultima diseguaglianza vale se prendiamo** $c \geq 2b$**. In pratica la prova funziona prendendo** $c = \max\{a, 2b\}$**.**
- **per dimostrare** $T(n) = \Omega(\sqrt{n})$ **proviamo per induzione che vale** $T(n) \geq c\sqrt{n}$ **per una costante** c **maggiore di zero da definire.**
  La prova è del tutto simmetrica al caso precedente, la riportiamo comunque qui per completezza. Per il caso base abbiamo $T(1) = a \geq c \cdot 1$ **che è vera se prendiamo** $c \leq a$**.**
  Per il passo induttivo:
  $T(n) \geq c\sqrt{\frac{n}{4}} + b\sqrt{n} = \frac{c}{2}\sqrt{n} + b\sqrt{n} \geq c\sqrt{n}$ **dove l'ultima**

diseguaglianza vale se prendiamo $c \leq 2b$. In pratica la prova funziona prendendo $c = \min\{a, 2b\}$.