Algoritmi e Strutture Dati - Prova d'esame
11/01/12

Esercizio 1

Cerchiamo di dimostrare che $\exists d >=: T(n) \leq d \log n$.

$$\begin{aligned} T(n) &= T(\lfloor n/c \rfloor) + \Theta(1) \\ &\leq T(n/c) + \Theta(1) \\ &= T(n/c) + b \\ &\leq d \log n/c + b \\ &= d \log n - d \log c + b \\ &\leq d \log n \end{aligned}$$

Questo implica che $d \geq b/\log c$, il che è soddisfacibile per qualunque $c > 1$.
Per quanto riguarda il caso base, si noti che non è possibile dimostrare che

$$T(1) = 1 \leq d \log 1 = 0$$

Bisogna quindi dimostrare il caso base per valori superiori a 2, il cui numero dipende però da c. La tecnica non è dissimile a quella già usata tante volte.

Esercizio 2

Ovviamente è possibile confrontare tutti i dadi con tutti i bulloni, ottenendo un algoritmo di complessità $O(n^2)$. Ma in realtà è possibile confrontare ogni dado e ogni bullone al massimo una volta (complessità $O(n)$), mantenendo due variabili che contengono il minimo dado e il minimo bullone.

```
findMin(DADO[] D, BULLONE[] B, integer n)
  BULLONE min_b ← B[1]
  DADO min_d ← D[1]
  integer d ← 2
  integer b ← 2
  while d ≤ n and b ≤ n do
      if try(min_b, min_d) ≤ 0 then
          { Il bullone è più piccolo o uguale al dado }
          if d ≤ n and try(min_b, D[d]) ≥ 0 then
              { Nuovo dado }
              min_d ← D[d]
          d ← d + 1
      if try(min_b, min_d) ≥ 0 then
          { Il dado è più piccolo o uguale al bullone }
          if b ≤ n and try(B[b], min_d) ≤ 0 then
              { Nuovo bullone }
              min_b ← B[b]
          b ← b + 1
  return (min_b, min_d)
```

Nella procedura sopra, ad ogni passo almeno uno dei due indici b e d avanza di 1; dopo che uno degli indici ha raggiunto n, l'altro avanza sempre. In questo modo, il numero di iterazioni all'interno del ciclo **while** è $O(n)$.

Esercizio 3

Si noti innanzitutto che essendo $2n$ valori, il mediano non è un singolo valore, ma una coppia. Restituiremo quindi una coppia di valori, non un valore singolo.
Se n è dispari, vi è un solo mediano per entrambi i vettori e si può considerare il mediano in posizione m_x di X e il mediano m_y di Y; se n è pari, vi sono due mediani in entrambi i vettori, e consideriamo il mediano "sinistro" in posizione m_x per X e il mediano "destro"

in posizione m_y per Y. Supponendo di considerare il vettore X dall'indice b_x (begin) all'indice e_x (end) e il vettore Y dall'indice b_y all'indice e_y, possiamo ottenere le seguenti formule:

$$m_x = \lfloor (b_x + e_x)/2 \rfloor m_y \qquad = \lceil (b_x + e_x)/2 \rceil$$

A questo punto possono darsi tre casi:

- Se $X[m_x] < Y[m_y]$, tutti i valori a "sinistra" di m_x sono più piccoli di $X[m_x]$ e tutti i valori a "destra" di m_y sono più grandi di $Y[m_y]$; ovvero $X[i] < X[m_x] < Y[m_y] < Y[j]$, per $i < m_x$ e $j > m_y$. Inoltre per costruzione i valori a destra e a sinistra sono in numero uguale, quindi possiamo ridurci al sottoproblema che si ottiene scartando i valori a "sinistra" di m_x e a "destra" di m_y.
- Se $Y[m_y] < X[m_x]$, tutti i valori a "destra" di m_x sono più grandi di $X[m_x]$ e tutti i valori a "sinistra" di m_y sono più piccoli di $Y[m_y]$; ovvero $Y[i] < Y[m_y] < X[m_x] < X[j]$, per $i < m_y$ e $j > m_y$. Inoltre per costruzione i valori a destra e a sinistra sono in numero uguale, quindi possiamo ridurci al sottoproblema che si ottiene scartando i valori a "destra" di m_x e a "sinistra" di m_y.
- Se $X[m_x] = Y[m_y]$, allora tutti i valori a "sinistra" sia di m_x che di m_y sono minori di $X[m_x] = Y[m_y]$, e tutti i valori a "destra" sia di m_x che di m_y sono maggiori di $X[m_x] = Y[m_y]$, e per costruzioni il numero di valori a destra è uguale al numero di valori a sinistra. Quindi $X[m_x] = Y[m_y]$ sono i due valori mediani.

Il caso base si ha quando rimangono quattro valori (due sul lato X e due sul lato Y); a questo punto i valori mediani possono trovarsi ovunque, entrambi in X, entrambi in Y oppure divisi fra i due vettori. E' sufficiente trovare i mediani fra i quattro valori rimasti, operazione che richiede tempo $O(1)$ ed è identificata da mediana4 nel codice sottostante.
La descrizione è più lunga del codice:

```
mediana(integer[] X, integer[] Y, integer b_x, e_x, b_y, e_y)
    if e_x − b_x = 1 then return mediana4(X, Y, b_x, e_x, b_y, e_y)
    integer m_x = ⌊(b_x + e_x)/2⌋
    integer m_y = ⌈(b_x + e_x)/2⌉
    if X[m_x] < Y[m_y] then return mediana(X, Y, m_x, e_x, b_y, m_y)
    if Y[m_y] > X[m_x] then return mediana(X, Y, b_x, m_x, m_y, e_y)
    return (X[m_x], Y[m_y])
```

Il costo computazionale è $O(\log n)$.

Esercizio 4

La soluzione non viene al momento proposta; questo esercizio verrà utilizzato per il laboratorio dell'a.a. 2011/2012.