Algoritmi e Strutture Dati (modulo I) - testo prova scritta 20/02/2023

docenti: Luciano Gualà & Andrea Clementi

Cognome:..... Nome:..... Matr.:.....

Esercizio 1 [16 punti]

A: notazione asintotica. Dire quali delle seguenti relazioni asintotiche sono vere:

F $\sqrt{\log n} = \Theta(\log \sqrt{n})$; V $n \log n = o(n^2)$; F $\frac{\sqrt{n^3+\log n}}{\sqrt{n}} = o(n)$; F $\sqrt[4]{\log n} = O(\log \log n)$;

V $3^n = o(2^{2n})$; F $2^n = \Theta(2^{n+\log n})$; V $2^{n+2} = \omega(2^{n/2})$; V $\frac{2^n}{n^2} = \omega(2^{n/2})$;

B: equazioni di ricorrenza. Fornire la soluzione asintotica alle seguenti relazioni di ricorrenza:

$T(n) = 2T(n/4) + n\sqrt{n}$; Soluzione: $\Theta(n\sqrt{n})$

$T(n) = 2T(n-4) + 1$; Soluzione: $\Theta(2^n)$

C: algoritmi e complessità. Quale algoritmo useresti e quanto costa se devi:

- In un grafo non orientato e pesato, calcolare la distanza fra tutte le coppie di nodi:

- In un grafo orientato, capire se uno specifico nodo s può essere raggiunto da tutti gli altri nodi:

- Ordinare un vettore $V[1:n]$ di n bit ($V[i] \in \{0,1\}$): integer sort: $T(n) = O(n+k)$

$k = 1 \Rightarrow O(n)$

- aggiungere $\sqrt{n}$ elementi ad un heap binario di n elementi: inserimenti ripetuti

poiché $\sqrt{n} = o\left(\frac{n}{\log n}\right) \Rightarrow T(n) = O(n)$

Esercizio 2 [8 punti]

Sia $A[1:n]$ un vettore di n bit, ovvero $A[i] \in \{0,1\}$ per ogni i. Si progetti un *oracolo* (struttura dati) che può essere costruito in tempo $O(n)$ e che sia in grado di rispondere in tempo costante a domande del tipo:

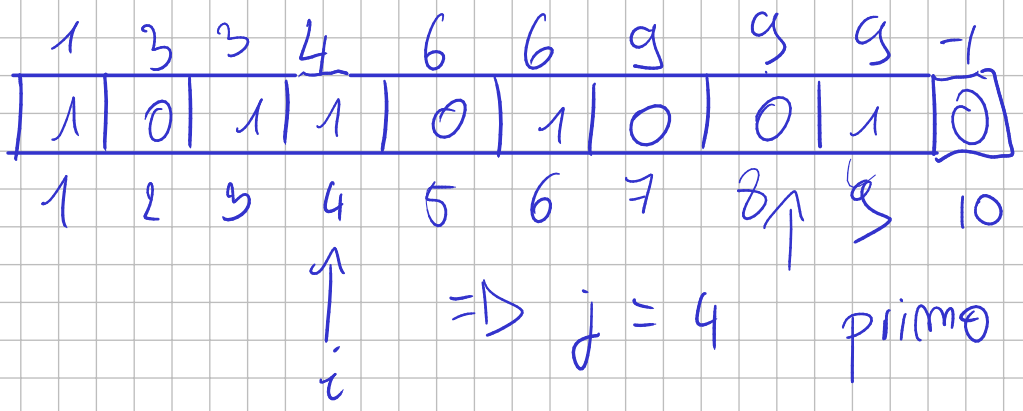
- $q(i)$: dato $i \in 1, \ldots, n$, restituire il più piccolo indice $j \geq i$ tale che $A[j] = 1$. Se tale indice j non esiste la risposta alla domanda è -1.

Si forniscano gli pseudocodici dettagliati dell'algoritmo che, preso A, costruisce in tempo $O(n)$ l'oracolo, e dell'algoritmo che, dato l'oracolo e un indice i, restituisce in tempo $O(1)$ la risposta alla domanda $q(i)$.

Esercizio 3 [10 punti]

La vostra città è modellata come un grafo diretto e pesato $G = (V, E, w)$. Voi siete nel nodo s e dovete raggiungere il nodo t dove si svolgerà l'esonero del corso di ASD. Ma siete in ritardo. Dovete fare in fretta. Per fortuna avete una bicicletta. Con la vostra bicicletta, attraversare un arco $e \in E$ richiede tempo $w(e)$. La bicicletta non è il solo mezzo che potete usare. Sapete che ci sono dei nodi del grafo, i nodi nell'insieme $X \subseteq V$, in cui potete affittare scooter, monopattini e altra roba. Avete soldi per affittare un solo mezzo. Per ogni nodo $x \in X$, conoscete due cose: il tempo τ_x che vi richiede lo scambio fra la bicicletta e il mezzo che si trova in x, e il fattore di *speed-up* $\sigma_x \leq 1$ del mezzo: con il mezzo preso nel nodo x il tempo di attraversamento di un arco e scende da $w(e)$ a $\sigma_x w(e)$.

Progettate un algoritmo che in tempo $O(m + n \log n)$, calcola la strategia che vi porta a t nel minor tempo possibile.

② 

Idea: creare array Y tale che
Y[i] = primo $j \geq i$ tale che A[j] = 1

creaOracolo(A)
  sia n la lunghezza di A
  Sia Y un array aux di dim n
  if (A[n] == 0) Y[n] = -1
  for k = n-1 down to 2
    if A[k] == 1 Y[k] = K
    else Y[k] = Y[k+1]

queryOracolo(Y, i)
  return Y[i]