Algoritmi e Strutture Dati (modulo I) - testo prova scritta 4/07/2023

docenti: Luciano Gualà & Andrea Clementi

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

Esercizio 1 [16 punti]

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

$n^{1/5}\log n + \sqrt{\log n} = o(n^{1/4})$; $\frac{n}{\log n} = \omega(\frac{n+3}{\log^3 n})$; $\frac{n^3+\log n}{\sqrt{n}} = \Theta(n^{2.5})$; $\log\log n = o(\sqrt[4]{\log n})$;

$2^{\sqrt{\log n}} = o(n^{1/3})$; $2^n = \Theta(2^{n+\log n})$; $2^{n+2} = \Theta(2^{n/2})$; $2^{2n} = \Theta(4^n + n^2)$;

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

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

$T(n) = T(n-1) + \sqrt{n}$; Soluzione:

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

- In un grafo diretto rappresentato con *matrice di adiacenza*, calcolare i nodi raggiungibili da un nodo specifico s:
- In un grafo non orientato e pesato, individuare il nodo a distanza massima da un nodo v:
- In un albero AVL di n nodi, trovare il secondo minimo:
- In un vettore ordinato, calcolare il numero di nodi di valore minimo:

Esercizio 2 [8 punti]

Sia T un albero binario con n, dove ogni nodo v di T ha un colore $v.col$ che può essere *Blu (B)* o *Giallo (G)*, e quindi $v.col \in \{B, G\}$. Diciamo che un nodo v ha *antenati ben colorati* se il cammino dalla radice al nodo v è composto da una sequenza (potenzialmente vuota) di nodi di colore Blu seguita da una sequenza (potenzialmente vuota) di nodi di colore Giallo.[1]

Si progetti un algoritmo che dato T, restituisca il numero di nodi di T che hanno antenati ben colorati. Si assuma che T è rappresentato tramite una struttura dati collegata, con record e puntatori, dove il record di ogni nodo, oltre al campo $v.col$, contiene anche il puntatore al figlio sinistro e al figlio destro del nodo. L'algoritmo deve avere complessità $O(n)$. Si fornisca lo pseudocodice dettagliato.

Esercizio 3 [8 punti]

Sia $G = (V, E)$ un grafo diretto con n nodi ed m archi. Ci sono Alice e Bob che vogliono incontrarsi in un nodo di G. Inizialmente, Alice si trova sul nodo s_A ed ha a disposizione Δ_A monete di tipo A, mentre Bob si trova sul nodo s_B ed ha a disposizione Δ_B monete di tipo B. Ad ogni arco $e \in E$, sono associati due interi, c_e^A e c_e^B, che rappresentano rispettivamente il numero di monete di tipo A che Alice deve pagare per attraversare e, e il numero di monete di tipo B che Bob deve pagare per poter attraversare e.

Progettate un algoritmo di complessità $O(m + n\log n)$ che calcola, se esiste, un modo per far incontrare Alice e Bob.

[1] Quindi un nodo v che ha tutti gli antenati di colore Giallo o tutti gli antenati di colore Blu ha antenati ben colorati. Si ricorda inoltre che un nodo è per definizione antenato di se stesso.

1 (a)

- $n^{1/5}\log(n) + \sqrt{\log n} = o(n^{1/4})$ FALSO

- $\dfrac{n}{\log n} = \omega\left(\dfrac{n+3}{\log^3 n}\right)$ VERO

- $\dfrac{n^3+\log n}{\sqrt{n}} = \Theta(n^{5/2})$ VERO

$\hookrightarrow \lim_{n\to+\infty} \dfrac{n^3+\log n}{n^{1/2}\cdot n^{5/2}} = \dfrac{n^3+\log n}{n^3} =$

$= 1 + \dfrac{\log n}{\boxed{n^3}} \rightarrow = 0 \quad \rightarrow = 1$

- $\log\log n = o(\sqrt[4]{\log n})$ VERO

LIMITE CON DE L'HOPITÀL.

- $2^{\sqrt{\log n}} = O(n^{1/3})$ FALSO

- $2^n = \Theta(2^{n+\log n})$ FALSO

$2^n = o(2^{n+\log n})$

- $2^{n+2} = \Theta(2^{n/2})$ FALSO

$2^{n+2} = \omega(2^{n/2})$

- $2^{2n} = \Theta(4^n + n^2)$ VERO

(b)

- $T(n) = T(n-1) + \sqrt{n}$

ALBERO

n
$|$
$n-1$
$\vdots$
1

- h ALBERO $= n$
- COSTO DI OGNI NODO? $=$ AL PIÙ $\sqrt{n}$

$T(n) = O(n\sqrt{n}) \to \Theta$?

$\left\{ \begin{array}{l} n \\ \vdots \\ n/2 \end{array} \right. \to n/2$ el. $\to$ COSTA $\geq \frac{\sqrt{n}}{2}$

$$T(n/2) \geq \frac{n}{2} \cdot \frac{\sqrt{n}}{2} = \frac{n\sqrt{n}}{4} = \Omega(n\sqrt{n})$$

$$T(n) = \Theta(n\sqrt{n})$$

- $T(n) = 2T(n/4) + \sqrt{n}$

$n^{\frac{1}{2}} \sim \sqrt{n}$

$n^{1/2} = \Theta(\sqrt{n}) \to$ CASO ②

$$T(n) = \Theta(\sqrt{n} \lg n)$$

(c)

① • ALGO $\to$ VISITA <u>DFS</u> DA SORGENTE ②
• COSTO $\to O(n^2)$

② • ALGO : ALGO DIJKSTRA CON v COME SORGENTE,
POI RICERCA ESAUSTIVA PER TROVARE $u \in V$
CON $\Delta(v, u) = MAX$

• COSTO $\rightarrow O(m + n \log n + n) = O(m + n \log n)$.

③ • ALGO : RICERCA MINIMO, IN + SE IL NODO MINIMO
HA UN FIGLIO, RESTITUISCE IL FIGLIO DESTRO,
ALTRIMENTI RESTITUISCE IL PADRE.

• COSTO : $O(\log n)$.

④ ALGO: VISITA LINEARE

COSTO: $O(n)$

2

CAMMINI APPROVATI → BLU - GIALLO
- BLU
- GIALLO

PROPIETA': 1 NODO, IF LAST DI SEQUENZA BUONA,
I SUOI FIGLI SICURO NON HA ANTENATI BUONI.

POSSO CREARE 2 PROCEDURE
- PERCORSO_B_G (T, V) → INTERO.

CONTA I NODI DISC. DI V CHE HANNO BUONI
ALTRIMENTI (DOVE IL PERCORSO È BLU OPP.
BLU - GIALLO).

- PERCORSO G (T, V) → INTERO

CONTA I NODI DISCENDENTI DI U CHE SONO BUONI ANTENATI DI U (DOVE IL PERCORSO È GIALLO)

```
PERCORSO_B_G(ALBERO T, NODO V) :  →  QUINDI PADRE(V).COL = B
    IF V = NULL
        RETURN 0

    ELSE IF V.COL = B :
        RETURN 1 + PERCORSO_B_G(T, SIN(V)) +
                 + PERCORSO_B_G(T, DES(V))

    ELSE
        RETURN 1 + PERCORSO_G(T, SIN(V)) +
                 + PERCORSO_G(T, DES(V))


PERCORSO_G(T, V):
    IF V = NULL OR V.COL = B
        RETURN 0

    ELSE
        RETURN 1 + PERC_G(T, SIN(V)) + PER_G(T, DES(V))
```

ALGO PRINCIPALE

```
BUONI_ANTENATI(T):
    RADICE r, INTERO RES
    IF r.COL = B :
        RES = 1 + PERCORSO_B_G(T, SIN(r)) +
                + PERCORSO_B_G(T, DES(r)).
```

ELSE IF r.COL = G

RES $1 +$ PERCORSO_G(T, SIN(r)) +

\+ PERCORSO_G(T, DES(r))

RETURN RES

CORRETTEZZA: CONTROLLO OGNI NODO CHE AMMETTE UN PERCORSO VALIDO.

COMPLESSITÀ:

Esercizio 3 [8 punti]

Sia $G = (V, E)$ un grafo diretto con n nodi ed m archi. Ci sono Alice e Bob che vogliono incontrarsi in un nodo di G. Inizialmente, Alice si trova sul nodo s_A ed ha a disposizione Δ_A monete di tipo A, mentre Bob si trova sul nodo s_B ed ha a disposizione Δ_B monete di tipo B. Ad ogni arco $e \in E$, sono associati due interi, c_e^A e c_e^B, che rappresentano rispettivamente il numero di monete di tipo A che Alice deve pagare per attraversare e, e il numero di monete di tipo B che Bob deve pagare per poter attraversare e.

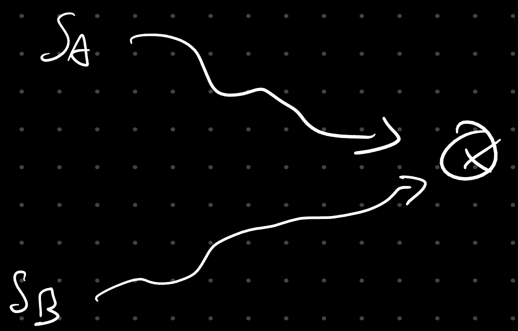
Progettate un algoritmo di complessità $O(m + n \log n)$ che calcola, se esiste, un modo per far incontrare Alice e Bob.

[1]Quindi un nodo v che ha tutti gli antenati di colore Giallo o tutti gli antenati di colore Blu ha antenati ben colorati. Si ricorda inoltre che un nodo è per definizione antenato di se stesso.

$\boxed{3}$

IDEA, SIMILE AL 'ESERCIZIO DI "ROCCA MAGGICA"

TROVARE 1 NODO CHE ABBIA QUESTA PROPIETÀ:



$$COST(x) = d_A(s_A, x) + d_B(s_B, x)$$

$$\begin{cases} e \ d_A(s_A, x) \leq \Delta_A \\ e \\ d_B(s_A, x) \leq \Delta_B \end{cases}$$

CALCOLABILI E ,O IN TIME COSTANTE.

l'idea è fare due grafi ausiliari, dove sono tutti e due pesati, uno usando i pesi di Ca e l'atro di Cb, calcolarmi l'SPT dell'uno e dell'altro e con i valori ottenuti, cercare un nodo che rispetti i vincoli, se esiste
lo ritorniamo altrimenti ritorna nullo.

la creazione dei due grafi costerebbe O(2n+2m) = O(n+m), i due SPT, 2*O(m+nlog(n)) = O(m+n(log(n)) e la ricerca = O(n)

costo totale = O(m+n(log(n))