Algoritmi e Strutture Dati

a.a. 2022/23

Compito del 08/01/2024

Cognome: _____ Nome: _____

Matricola: _____ E-mail: _____

Parte I

(30 minuti; ogni esercizio vale 2 punti)

Avvertenza: Si giustifichino tecnicamente tutte le risposte. In caso di discussioni poco formali o approssimative gli esercizi non verranno valutati pienamente.

1. Sia H un Max-Heap di interi di dimensione n.
   a) Se per ogni $i \leq H.heapsize/2$ vale che $H[2i] < H[2i + 1]$, allora H è anche un albero binario di ricerca?
   b) Se H contiene solo potenze di 2, è privo di ripetizioni e $H[1] = 2^k$, allora quanto vale al massimo *heapsize*?
   c) Se $H[2] \neq H[1]$, quante occorrenze di $H[1]$ ci sono al massimo in H?

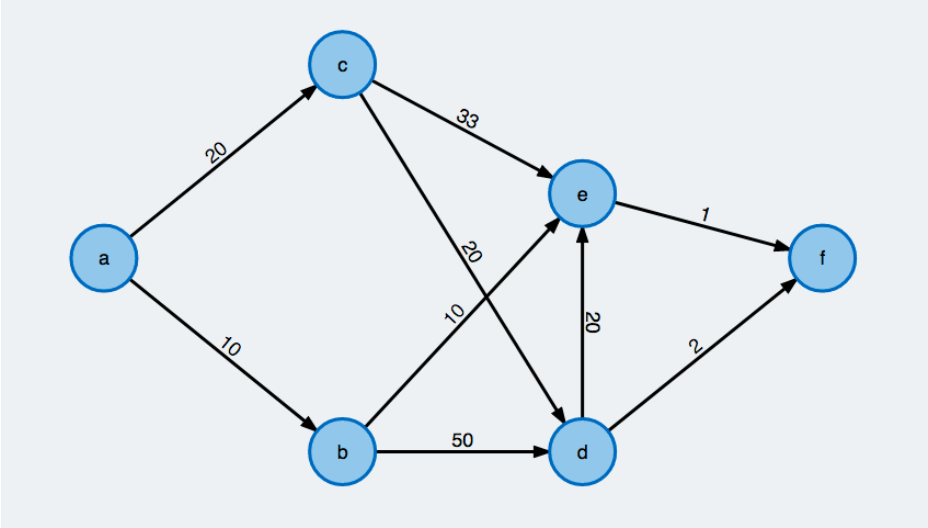
2. Il *clique number* di un grafo non orientato G, denotato con $\omega(G)$, indica la cardinalità della "clique" massima di G (ovvero il numero di vertici del più grande sottografo completo di G). Stabilire, giustificando la risposta, quali delle seguenti affermazione sono corrette:

   a) Se G è connesso e $\omega(G) = 2$, allora G è un albero

   b) Se G è un albero, allora $\omega(G) \leq 2$

   c) G è ciclico se e soltanto se $\omega(G) > 2$

3. Si simuli l'esecuzione <u>della prima iterazione</u> dell'algoritmo di Floyd-Warshall sul seguente grafo:



Algoritmi e Strutture Dati

a.a. 2022/23

Compito del 08/01/2024

Cognome: _____ Nome: _____

Matricola: _____ E-mail: _____

Parte II

(2.5 ore; ogni esercizio vale 6 punti)

Avvertenza: Si giustifichino tecnicamente tutte le risposte. In caso di discussioni poco formali o approssimative gli esercizi non verranno valutati pienamente.

1. Si consideri un albero ternario completo in cui ogni nodo ha i seguenti campi: (i) `key` chiave intera, (ii) `fruitful` valore booleano, (iii) `left` puntatore al figlio sinistro, (iv) `center` puntatore al figlio centrale, (v) `right` puntatore al figlio destro.
   a. Si scriva una procedura **efficiente** in C o C++ che assegni `True` al campo `fruitful` del nodo se e solo se la somma delle chiavi dei nodi di **ciascuno** dei sottoalberi radicati nei figli è maggiore di una costante `k` fornita in input. Il prototipo della procedura è:
      ```
      void set_fruitful(PTNode r, int k)
      ```
   b. Valutare la complessità della procedura, **indicando eventuali relazioni di ricorrenza e mostrando la loro risoluzione.**
   c. Specificare il linguaggio di programmazione scelto.

2. Questa settimana Carlotta ha ricevuto del denaro dai suoi genitori e vuole spenderlo tutto acquistando libri. Per finire un libro Carlotta impiega una settimana e poiché riceve denaro ogni due settimane, ha deciso di acquistare due libri, così potrà leggerli fino a quando riceverà altri soldi. Desidera spendere tutti i soldi così vorrebbe scegliere due libri i cui prezzi sommati sono pari ai soldi che ha ricevuto.
   Data la quantità di soldi che Carlotta ha a disposizione e un array contenente i prezzi dei libri (tutti distinti), restituire le coppie di prezzi di libri che soddisfano la condizione. Le coppie di prezzi devono contenere prima il prezzo più basso e poi quello più alto.
   a. Scrivere una funzione **efficiente** il cui prototipo è il seguente:
      ```
      int libriSelezionati(array prezzolibri, double soldi, array ris)
      ```
      La funzione restituisce la dimensione dell'array `ris`. **Si devono scrivere eventuali funzioni/procedure ausiliari utilizzate.**
   b. Valutare e giustificare la complessità della funzione.

3. Si calcoli la complessità asintontica dei seguenti algoritmi (in funzione di n) e si stabilisca quale dei due è preferibile per n sufficiente grande:

```
MyAlgorithm1( int n )
int
  a, i

if ( n > 1 ) then
   a = 0;
   for i = 1 to n
      a = a + (i+1)*(i+2)
   endfor
   for i = 1 to 3
      a = a + MyAlgorithm1(n/2)
   endfor
   return a
else
   return n-1
endif
```

```
MyAlgorithm2( int n )
int
  a, i, j

if ( n > 1 ) then
   a = 0
   for i = 1 to n
      for j = 1 to n
         a = a + (i+1)*(j+1)
      endfor
   endfor
   for i = 1 to 7
      a = a + MyAlgorithm2(n/3)
   endfor
   return a
else
   return n-1
endif
```

4. Si definiscano formalmente le classi P, NP ed NPC e si enunci e dimostri il teorema fondamentale della NP-completezza.

   Siano inoltre $\mathcal{P}$ e Q due problemi in NP e si supponga $\mathcal{P} \leq_P Q$. Si stabilisca se le seguenti affermazioni sono vere o false:

   a) $Q \in \text{P} \Rightarrow \mathcal{P} \notin \text{NPC}$
   b) $\mathcal{P} \in \text{P} \Rightarrow Q \in \text{P}$
   c) $\mathcal{P} \in \text{NPC} \Rightarrow Q \in \text{NPC}$