Algoritmi e Strutture Dati

a.a. 2015/16

Compito del 31/05/2016

Cognome: _____ Nome: _____

Matricola: _____ E-mail: _____

Parte I

(30 minuti; ogni esercizio vale 2 punti)

1. Dato un nodo x in un albero binario di ricerca T, scrivere la funzione *Tree-Predecessor(Node x)* che restituisce il predecessore di x nell'ordine stabilito da un attraversamento simmetrico. Qual è la complessità della funzione?

2. Il Prof. Pelillo sostiene di aver sviluppato un algoritmo di complessità

$$T(n) = 3T(n/4) + n \log n$$

che riceve in ingresso un grafo non orientato G e una costante k e stabilisce se il grafo contiene o meno una clique di k vertici (n rappresenta il numero di vertici di G). Si dica, **giustificando tecnicamente la risposta**, se l'affermazione è verosimile.

3. La Prof.ssa Raffaetà sostiene che, al fine di utilizzare l'algoritmo di Dijkstra su grafi aventi pesi negativi, è sufficiente sommare a tutti i pesi degli archi una costante k in modo da renderli positivi (per esempio, k potrebbe coincidere con il peso più piccolo). Si dica, **giustificando tecnicamente la risposta**, se l'affermazione è corretta.

Algoritmi e Strutture Dati

a.a. 2015/16

Compito del 31/05/2016

Cognome: _____ Nome: _____

Matricola: _____ E-mail: _____

Parte II

(2.5 ore; ogni esercizio vale 6 punti)

1. Sia *T* un albero binario di ricerca di altezza *h* e avente *n* nodi con chiavi intere eventualmente ripetute. Scrivere una funzione **efficiente** che, ricevuto in ingresso *T* e un intero *k*, restituisce il nodo con la **prima occorrenza** di *k* in *T* nell'ordine stabilito da un attraversamento simmetrico se *k* è presente in *T, NULL* altrimenti.
   Analizzare la complessità della funzione.

   Per l'esame da **12 CFU**, deve essere fornita **una funzione C**.
   Per l'esame da **9 CFU**, è sufficiente specificare lo pseudocodice.

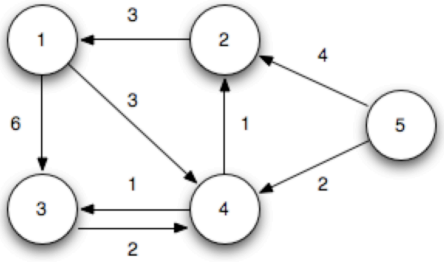
2. Scrivere un algoritmo che, dato un vettore di intervalli e la sua dimensione, verifica se gli intervalli sono a due a due disgiunti, restituendo *1* se sono disgiunti, *0* altrimenti. **Tutti gli intervalli sono NON vuoti.**
   Analizzare la complessità dell'algoritmo.

   Esempio:
   Sia v = ⟨(-2,11), (-5,-4), (-1, 7), (10,15)⟩ *sonodisgiunti(v,4) =0*
   Sia v = ⟨(-2,11), (-5,-4), (12, 18)⟩ *sonodisgiunti(v,3) =1*

3. Si stabilisca quale problema risolve il seguente algoritmo, che accetta in ingresso un grafo orientato e pesato $G = (V, E)$, la sua funzione peso $w : E \rightarrow \mathrm{R}$, e un vertice $s \in V$:

```
MyAlgorithm(G, w, s)
1.  for each (u,v) ∈ E[G] do
2.     if w(u,v) < 0 then
3.        print "Failure"
4.        return
5.  d[s] = 0
6.  for u ∈ V[G] \ {s}
7.     d[u] = +∞
8.  for i = 1 to n - 1
9.     for each (u,v) ∈ E[G] do
10.       d[v] = min { d[v], d[u] + w(u,v) }
11. for u ∈ V[G]
12.    print d[u]
13. return
```

   Si dimostri la correttezza dell'algoritmo, si determini la sua complessità e si simuli infine la sua esecuzione sul seguente grafo (si usi, come vertice *s,* il vertice 1):



4. Sia $G = (V, E)$ un grafo non orientato connesso, con funzione peso $w : E \rightarrow \mathrm{R}$. Si consideri un taglio arbitrario di G in due parti $(S, V \setminus S)$, con $S \not\subset V$ e $S \neq \emptyset$, e sia (u, v) un arco che attraversa il taglio avente peso minimo, ovvero $w(u, v) \leq w(x, y)$, per tutti gli archi (x, y) che attraversano il taglio. Quali delle seguenti affermazioni sono vere?

   a) Tutti gli alberi di copertura di G conterrano l'arco (u, v);

   b) Tutti gli alberi di copertura minimi di G conterrano l'arco (u, v);

   c) Almeno un albero di copertura di G conterrà l'arco (u, v);

   d) Almeno un albero di copertura minimo di G conterrà l'arco (u, v).

   Se invece $w(u, v) < w(x, y)$, per tutti gli archi (x, y) che attraversano il taglio diversi da (u, v), quali delle affermazioni precedenti risultano essere vere?

   Si giustifichino **formalmente** le risposte, fornendo una dimostrazione in caso affermativo e un controesempio in caso negativo.