Progettazione di Algoritmi (9 CFU)
12/7/2019

Cognome e Nome:
Numero di Matricola:

Spazio riservato alla correzione

1	2	3	4	5	6	Totale
/18	/20	/18	20	/10	/14	/100

1.

a) Indicare quali delle seguenti affermazioni sono vere e quali sono false.
1. $(\log n)n^{1/8} = O((\log n)^2)$ F $\quad \log \sqrt[8]{n} = \log n \; \log n$
2. $n^2\, 2^{n/2} = \Omega(2^n)$ F
3. $10n+n^k = O(n)$, k e` una costante minore o uguale di 1 V
4. $1000n^4-100n^2=\Omega(n^3)$ V
5. $\frac{1}{4} n + n^{1/2}\log n = O(n)$ V

b) Si dimostri che se f(n)=O(g(n)) allora nf(n)=O(ng(n)). **Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.**

Dalla definizione di O:

1) $f(n) = O(g(n)) \iff \exists\, c>0, n_0 \geq 0 \mid f(n) \leq c\,g(n)\ \forall n \geq n_0$

abbiamo che utilizzando la 1

$nf(n) = O(ng(n)) \iff \exists\, c'>0\ n'_0 \geq 0 \mid nf(n) \leq c' n g(n)\ \forall n \geq n'_0$

dividendo entrambi i membri per n abbiamo

$f(n) \leq c' g(n)$ quindi prendendo $c = c'$ e $n_0 = n'_0$ abbiamo

che $f(n) \leq c\, g(n)$ quindi vale $nf(n) = O(ng(n))$

c) Si analizzi il tempo di esecuzione nel caso pessimo del seguente segmento di codice fornendo una stima asintotica **quanto migliore e` possibile** per esso. **Si giustifichi in modo chiaro la risposta.**

```
i=1, j=1;
WHILE( i≤n and  j≤n){
      print(i*j);
       i=i+1;
      j=j*2

  }
```

iteraz	i	j		
1	1	1	2^0	2^{i-1}
2	2	2	2^1	2^{i-1}
3	3	4	2^2	2^{i-1}
4	4	8	2^3	
5	5	16	2^4	
6	6	32	2^5	

alla K-esima iterazione $j = 2^{(k-1)}$

analizzando l'algoritmo possiamo che usciremo dal while quando $j \leq n$ poichè i alla K-esima iteraz. sarà proprio k mentre $j = 2^{(k-1)}$ che è anche $\log(k-1)$ ovvero $\log K$, pertanto possiamo dedurre che l'algoritmo ha tempo asintotico $O(\log n)$

2

a) Si scriva lo pseudocodice di un algoritmo basato sul paradigma del **Divide et Impera** che prende in input un array NON ordinato A e un elemento x e restituisce l'indice della cella dell'array contenente x. Se x non e` presente nell'array, l'algoritmo restituisce -1.

b) Si fornisca la relazione di ricorrenza che esprime il tempo di esecuzione T(n) dell'algoritmo di cui al punto a). **Spiegare in modo chiaro come si ottiene la formula da voi fornita. Se l'algoritmo al punto a) non risultera` corretto, questo punto non verra` valutato.**

c) A partire dalla relazione di ricorrenza di cui al punto b), si fornisca una stima asintotica quanto migliore e` possibile del tempo di esecuzione nel caso pessimo dell'algoritmo al punto a) . **Giustificare la risposta usando o il metodo iterativo o quello per sostituzione (induzione).**

3.

a) Si scriva lo pseudocodice di un algoritmo BFS **senza** coda FIFO che abbia tempo di esecuzione O(n+m) .

```
BFS(s)
init ℓ = ∅
init BFS-T = ∅
poni discovered[s] = true
poni discovered[v] = false per tutti i nodi        O(n)
inserisci ℓ(0) = s
i = 0
while (i ≤ n-2)
    init (ℓi+1) = ∅
    foreach u nodi in ℓi
        foreach n adiacenti ad u
            se discovered[n] = false
                poni discovered[n] = true                deg(v) = O(m)
                lo aggiungi a (ℓi+1)
                aggiungi n a BSF-T
            end if
        end for
    end for
    i = i + 1
end while
```

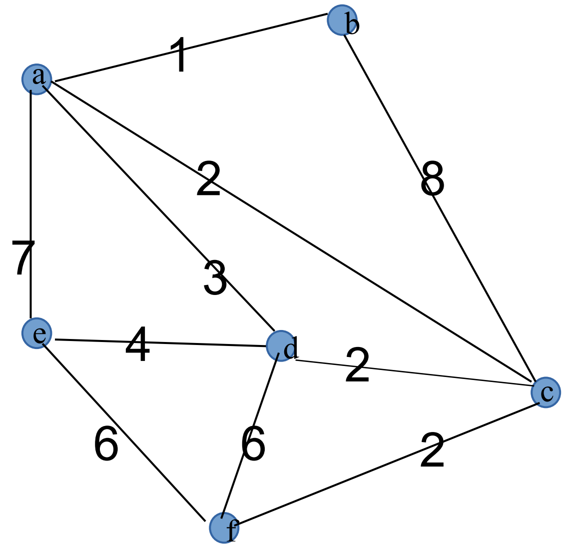
costo totale O(n+m)

b) Si scriva lo pseudocodice di un algoritmo che prende in input un grafo G non orientato e restituisce true se il grafo G e` bipartito e false altrimenti. L'algoritmo deve avere tempo di esecuzione O(n+m). Il voto dipendera`, oltre che dalla correttezza dell'algoritmo, anche da quanto dettagliato e` lo pseudocodice.

```
BFS(s)
  L(0) = s
  i=0
  Init BFS-T = Ø
  color[ ] = null per tutti i nodi dell'albero
  poni discovered[s] = true e discovered[v] = false per tutti i nodi
  bipartito = true
  while (i≤n-2)
      L(i+1) = Ø
      foreach u appartenente a L(i)
          foreach n adiacente a u
              se discovered[n]==false
                  discovered[n] = true
                  aggiungo n a L(i+1)
                  se i è pari
                      color[n] = blue
                  altrimenti
                      color[n] = red
                  end if
                  aggiungo n a BFS-T
                  se color[u]== color[n]
                      bipartito = false
                  end if
              end if
          end for
      end for
      i=i+1
  end while
  return bipartito
```

4.

a) Si mostri l’esecuzione dell’algoritmo di Krukal sul seguente grafo. **Occorre mostrare per ogni passo la foresta di alberi ottenuta fino a quel passo.**



b) Sia G un grafo non orientato connesso con pesi degli archi a due a due distinti. Si dimostri che se si esegue l'algoritmo di Kruskal su G allora ogni arco selezionato in questa esecuzione fa parte del minimo albero ricoprente di G.

c) Si spieghi in che cosa consiste un'istanza (input) del problema del **Partizionamento di Intervalli** e in cosa consiste una soluzione (output) del problema. **Se dalla risposta a questo punto si evincera` che lo studente non sa in cosa consiste il problema del Partizionamento di Intervalli, il punto successivo del quesito non sara` valutato.**

Abbiamo n intervalli $[s_1 - f_1]$, $[s_2 - f_2]$, ..., $[s_n - f_n]$ che rappresentano gli intervalli durante i quali si svolgono n-attività, l'obiettivo è di far svolgere le n-attività utilizzando il minor numero di risorse possibili

d) Si descriva la strategia greedy che permette di ottenere la soluzione ottima per il problema del **Partizionamento di Intervalli** e si dica a cosa e` uguale il valore della soluzione ottima. Fornire una definizione chiara di eventuali concetti utilizzati per rispondere al quesito.

La strategia greedy consiste nell'allocare il minor numero di risorse possibili, per farlo andiamo ad esaminare se le risorse precendemente allocate sono disponibili, in tal caso utilizziamo una di quelle risorse, altrimenti ne alloca una nuova. Il valore della soluzione ottima è sempre uguale alla profondità, ovvero al numero di attività massimo che si sovrappongono in un certo intervallo

5.

a) Si consideri la seguente istanza di Interval Scheduling Pesato:

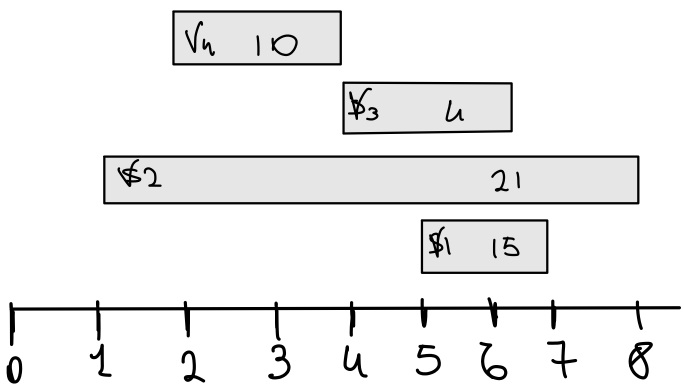
$s_1 = 5, s_2 = 1, s_3 = 4, s_4 = 2$

$f_1 = 7, f_2 = 8, f_3 = 6, f_4 = 4$

$v_1 = 15, v_2 = 21, v_3 = 4, v_4 = 10$

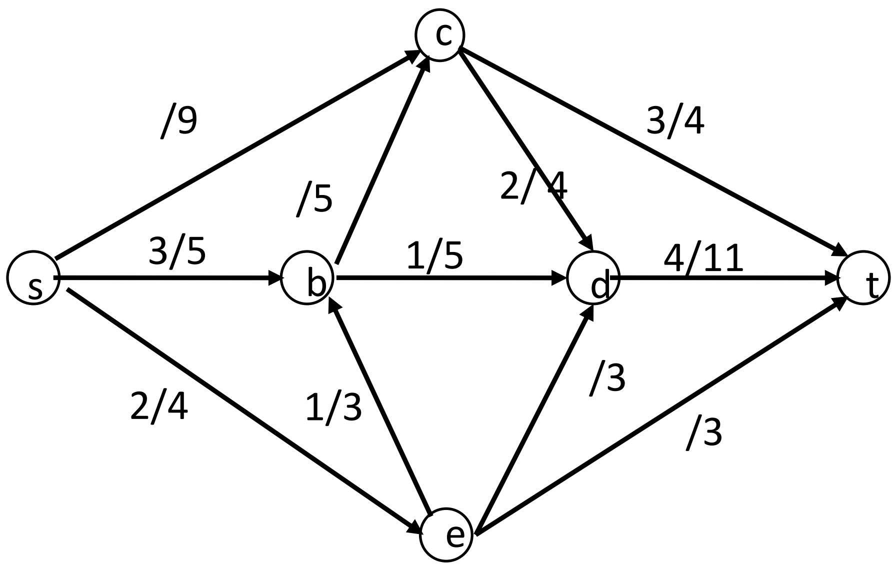
Si calcolino i valori p(j) e i valori OPT(j) per j=1,2,3,4 e si fornisca la **soluzione ottima** per la suddetta istanza del problema (non solo il suo valore).

Attenzione: gli indici j di p(j) e OPT(j) non corrispondono necessariamente agli indici j dei valori input s_j , f_j e v_j .



6.

a) Nella seguente figura sono indicate le quantita` di flusso associate ad alcuni degli archi della rete. Si associ a ciascuno dei restanti archi una quantita` di flusso in modo che i valori da voi forniti siano compatibili con quelli gia` indicati e si dica qual e` il valore della funzione flusso cosi` definita.



b) Si definisca la nozione di taglio s-t di una rete di flusso e di capacita` di un taglio s-t.

b) Si descriva un algoritmo efficiente per ottenere un taglio di capacita` minima di una rete di flusso G.