Progettazione di Algoritmi (9 CFU)
10/6/2020

Cognome e Nome:
Numero di Matricola:

Spazio riservato alla correzione

1	2	3	4	5	Totale
/18	/20	/22	/18	22/	/100

Si ricorda che per i punti che richiedono l'analisi di un algoritmo occorre fornire un limite superiore asintotico quanto migliore e` possibile al tempo di esecuzione dell'algoritmo giustificando la risposta.

1. Analisi degli algoritmi e notazione asintotica

a) Indicare quali delle seguenti affermazioni sono vere e quali sono false.
   1. $n (\log n)^4+1000n^2 = O(n^4)$ V
   2. $n^n = \Omega(n^{3/4} n^{1/4})$ V
   3. $n^{1/2} = O(\log n)$ F
   4. $n^3+1000n^2+100=\Theta(n^3)$ V
   5. $\log(\log n)= \Omega((\log n)^{1/2})$ F $\log(\log n)$ $\sqrt{\log n}$

1
$\log\log n$
$\log n$
$\sqrt{n}$
$n\log n$
n

b) Si dimostri che se $0<f(n)=O(h(n))$ e $0<g(n)= O(p(n))$ allora $f(n)g(n)= O(h(n)\, p(n))$. Occorre utilizzare solo la definizione di O e nessuna altra proprieta`.

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.**

```
FOR(i=1; i<n; i=i+1){
  FOR(j=1; j<2^n; j=j*2) {
      print(j);
   }
}
```

Progettazione di Algoritmi (9 CFU)
10/6/2020

2. Divide et Impera

a) Si scriva lo pseudocodice di un algoritmo ricorsivo **cerca** che restituisce il numero di occorrenze di x in un array A. L'elemento x e l'array A vengono passati in input all'algoritmo. Ad esempio, se l'array A contiene <3 1 4 1 2 3 1 4 5 2 1 4 5 1> e x= 4 allora l'algoritmo restituisce 3.

Se non si è in grado di scrivere l'algoritmo richiesto, si scriva lo pseudocodice dell'algoritmo ricorsivo **ricercaBinaria** che effettua la ricerca binaria in un array ordinato. **Il punteggio massimo per l'algoritmo ricercaBinaria è la meta` di quello per l'algoritmo cerca.**

b) Si fornisca la relazione di ricorrenza che esprime un limite superiore al tempo di esecuzione dell'algoritmo da voi fornito al punto a) .

c) A partire dalla relazione di ricorrenza da voi fornita al punto b), si fornisca una funzione h(n) tale T(n)=O(h(n)). Giustificare la risposta usando o il metodo iterativo o quello della sostituzione (induzione).

```
int main {
 cerca(a[], x, 0, a.lenght-1)
}
 cerca(a[], x, sx, dx){
    if(sx == dx ) {
       if (a[sx] == x)
          return 1
       return 0
    }
    c = (sx+dx)/2
 return cerca(a[], x, sx, c) + cerca(a[], x, c+1, dx)
 }
```

$$T(n) \begin{cases} c_0 & \text{se } n \le 1 \\ 2T(n/2) \end{cases}$$

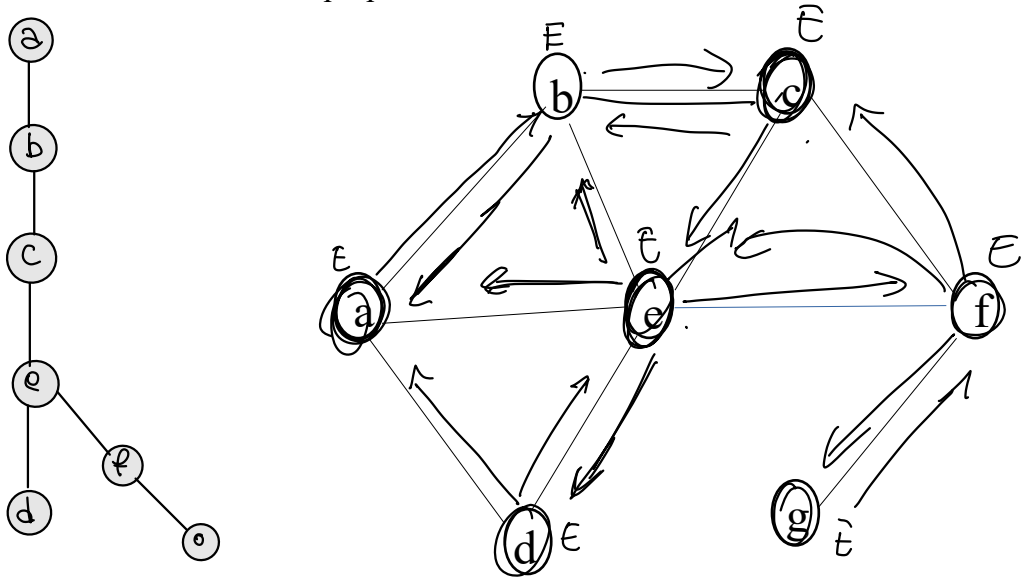
$$2T(n) \le 2T(n/2) \le 4T(n/4) \le \dots \le 2^i \left(\frac{n}{2^i}\right)$$

$$2T(n) \le \quad \frac{n}{2^i} \le 1 \quad n \le 2^i \quad 2^i \ge n \quad i \ge \log n$$

3. Grafi

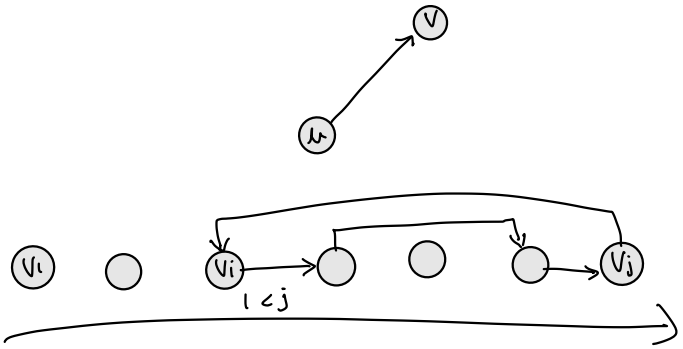
a) Si scriva lo pseudocodice **dell'algoritmo di Prim** per il minimo albero ricoprente aggiungendo anche le linee di codice per la costruzione dell'albero. Si analizzi il tempo di esecuzione dell'algoritmo proposto quando la coda a priorita` è implementata con un heap binario.

a) Si disegni l'albero DFS generato da una visita DFS del seguente grafo a partire dal nodo sorgente **a**. Si assuma che i nodi siano disposti nelle liste di adiacenza in base all'ordine crescente delle proprie etichette.



b) Si definisca il concetto di ordinamento topologico di un grafo direzionato e si dimostri che se un grafo direzionato G è un DAG allora G ha un ordinamento topologico.

L'ordinamento Topologico è una etichettatura dei vertici di un grafo $V_1, V_2, \ldots, V_n$ tale che per ogni coppia $V_i, V_j \Rightarrow i < j$.



Prim (G, c)

setta S = set dei nodi esplorati

setta T = vuoto

for each nodo u non in S

a[u] = costo di u

setta h = nuovo min heap di coppia (a[u], u)

nb, probabilmente u = radice in quanto sta in S → O(n)

foreach u in V (nodi non esplorati)

insert in minheap (∞, u)

while (h != empty)

(a[u], u) = estraiamo il minimo da h → $O(n \log n + m \log n)$

Aggiungiamo u ad S

if (parent[u] != null) inserisco in T arco (parent[u], u)

foreach arco e = (u, v)

if ((v non è in S) and (ce < a[v]))

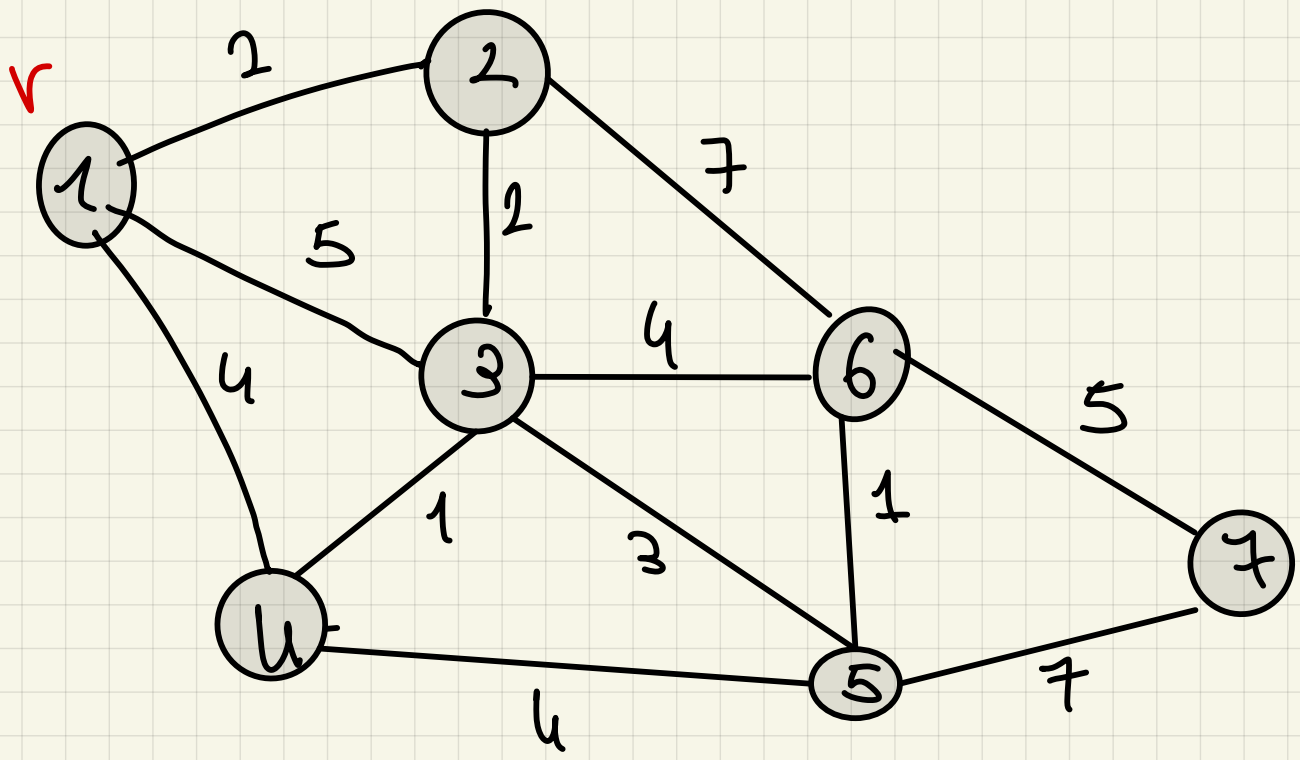
changekey (h, v, ce)

parent[v] = u

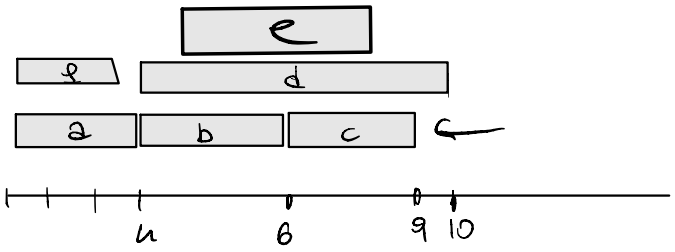
endif

end foreach

end while



Progettazione di Algoritmi (9 CFU)
10/6/2020



4. **Algoritmi greedy**

a) Si spieghi in che cosa consiste un'istanza (input) del problema dell'interval scheduling 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 dell'interval sheduling, i punti successivi dell'esercizio non saranno valutati.

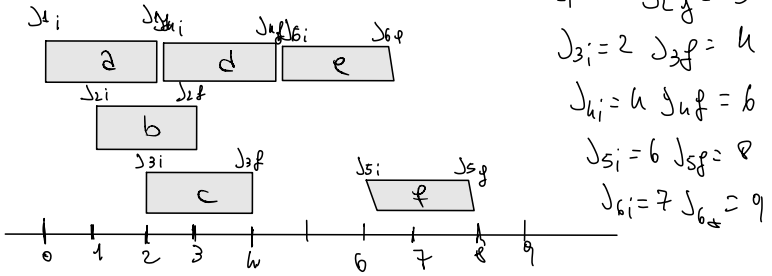
b) Si fornisca un'istanza del problema dell'interval scheduling con n=6 per cui il valore della soluzione ottima è 3. Si specifichino i valori numerici che descrivono l'input. **Non è sufficiente fornire un disegno che descriva l'input.**

c) Si scriva lo pseudocodice dell'algoritmo greedy che restituisce la soluzione ottima per il problema dell'interval scheduling.

a) Abbiamo n-job che abbiamo un inizio e una durata, $[s_i - f_j]$ possono essere eseguite più attività contemporaneamente e l'obiettivo è di eseguire in questa durata + attività possibili.
2 job, i, j sono compatibili quando $f_j \leq s_i$ o $f_i \leq s_j$
quindi l'obiettivo è quello di prendere la maggior numero di lavori compatibili

---

b) Algoritmo



$J_{1i} = 0$ $J_{1f} = 2$
$J_{2i} = 1$ $J_{2f} = 3$
$J_{3i} = 2$ $J_{3f} = 4$
$J_{4i} = 4$ $J_{4f} = 6$
$J_{5i} = 6$ $J_{5f} = 8$
$J_{6i} = 7$ $J_{6f} = 9$

Prendiamo 6 attività tali che $[a_s = 0, a_f = 2]$, $[b_s = 1, b_f = 3]$, $[c_s = 2, c_f = 4]$ $[d_s = 4, d_f = 6]$, $[e_s = 7, e_f = 9]$, $[f_s = 6, f_e = 8]$, quindi possiamo notare che gli insieni compatibili, in quanto $j'_f \leq j'_s$ oppure $j''_f \leq j'_s$ sono
$S_1 = \{a, b, e\}$, $S_2 = \{a, c, f\}$, $S_3 = \{a, c, e\}$

5. **Programmazione dinamica**

a) Fornire una formula per il calcolo del valore della soluzione ottima OPT del problema subset sums in termini di valori delle soluzioni ottime per sottoproblemi di taglia piu` piccola. Spiegare **in modo chiaro**
   1. cosa rappresenta la funzione OPT e cosa rappresentanto i suoi parametri
   2. come si arriva alla formula da voi fornita.

b) Scrivere l'algoritmo di programmazione dinamica per subset sums. Questo punto` sara` valutato solo se dalla risposta al punto a) si evincera` che lo studente sa in cosa consiste il problema di subset sums.

c) Si forniscano i valori p(j) per la seguente istanza del problema dell'Interval Scheduling Pesato.

s1=7 f1 =9	w(1)=4
s2=1 f2=4	w(2)=3
s3=5 f3=6	w(3)=6
s4=4 f4=8	w(4)=1

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