Progettazione di Algoritmi
27/6/2022

Cognome e Nome:
Numero di Matricola:

Spazio riservato alla correzione

1	2	3	4	5	Totale
/25	/12	/21	/18	24/	/100

Non staccate nessun foglio dal fascicolo

1. **Grafi**

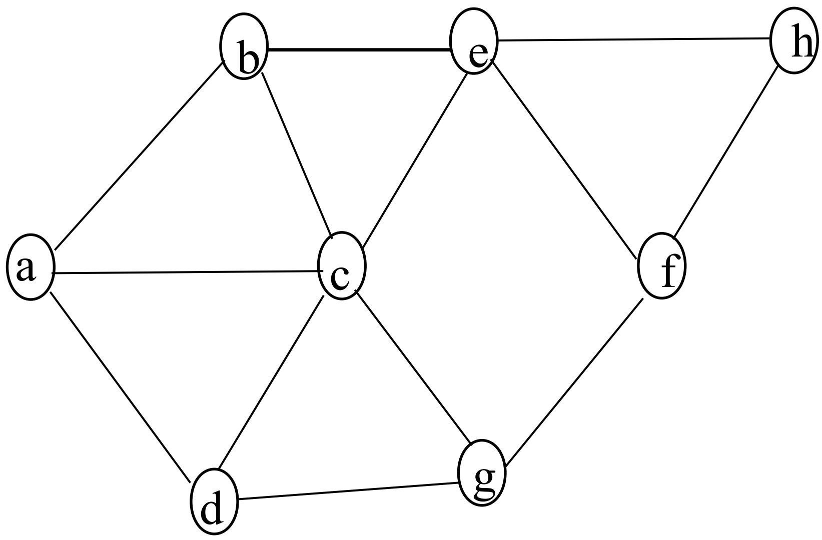
a) Cammini minimi

i. Scrivere lo pseudocodice dell'algoritmo di Dijkstra con coda a priorità, con incluse le linee per costruire l'albero dei percorsi minimi.

ii. Spiegare a cosa è uguale la chiave associata ad un nodo u quando u viene estratto dalla coda priorità.

iii. Dimostrare la risposta al punto ii. Suggerimento: potete usare la dimostrazione per l'algoritmo senza coda a priorità.

b) Si disegni l'albero BFS generato da una visita BFS 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.



c) Per il grafo dell'esercizio b,

I. si dica di che colore viene colorato ciascun nodo del grafo dall'algoritmo che verifica se il grafo è bipartito.

II. si dica se il grafo è bipartito o meno e nel caso non lo sia si indichino gli archi che devono essere rimossi affinché il grafo diventi bipartito **giustificando la risposta.**

III. si fornisca una partizione (X,Y) dei nodi da cui si evinca che il grafo così ottenuto (o il grafo di partenza a seconda di quello che avete risposto al punto b) è bipartito.

Progettazione di Algoritmi
27/6/2022

Progettazione di Algoritmi
27/6/2022

2. **Algoritmi greedy**

a) Si consideri la seguente istanza di interval scheduling: [7,9],[1,4],[3,6],[1,5],[4,9],[5,6],[4,6]. Fornire tutte le possibili soluzioni ottime per questa istanza **ottenibili con la strategia greedy.**

b) Si dimostri che la strategia greedy per la minimizzazione dei ritardi produce uno schedule ottimo usando i seguenti due fatti:
1. Tutti gli scheduling senza inversioni e privi di idle time hanno lo stesso ritardo massimo dello schedule prodotto dalla strategia greedy.
2. Ogni scheduling può essere trasformato in uno scheduling senza inversioni e privo di idle time senza che aumenti il suo ritardo massimo.
NB: non occorre dimostrare i fatti 1 e 2.

c) Si scriva lo pseudocodice dell'algoritmo per la minimizzazione dei ritardi.

Progettazione di Algoritmi
27/6/2022

3. **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 più piccola. Spiegare **in modo chiaro**
   1. cosa rappresenta la funzione OPT e cosa rappresentano i suoi parametri
   2. come si arriva alla formula da voi fornita.

b) Si fornisca la tabella M costruita dall'algoritmo che computa il valore della soluzione ottima per **il problema dello zaino** quando l'istanza input è $w_1=2$, $w_2=3$, $w_3=5$, $w_4=5$, $w_5=3$, $v_1=5$, $v_2=4$, $v_3=5$, $v_4=5$, $v_5=6$, $W=5$. Una volta costruita la tabella M, si contrassegnino con un cerchio le entrate M[i,w] corrispondenti alle coppie di indici (i,w) su cui viene invocato ricorsivamente l'algoritmo che ricostruisce la soluzione ottima e **si fornisca la soluzione ottima**.

c) Si fornisca la tabella M costruita dall'algoritmo che computa il valore della soluzione ottima per **il problema della sottosequenza comune più lunga** quando l'istanza input è formata dalle due sequenze "ABBA" e "ADBADB ". Una volta costruita la tabella M, **si contrassegnino con un cerchio** le entrate M[i,j] corrispondenti alle coppie di indici (i,j) su cui viene invocato ricorsivamente l'algoritmo che ricostruisce la soluzione ottima e **si fornisca la soluzione ottima**.

Progettazione di Algoritmi
27/6/2022

4. **Divide et Impera**

   a) Descrivere a parole il comportamento dell'algoritmo quicksort specificando quanto segue: **cosa prende in input l'algoritmo**, **in cosa consiste il lavoro di decomposizione**, **su quali sottoproblemi viene richiamato ricorsivamente l'algoritmo**, **in cosa consiste il lavoro di ricombinazione**.
   N.B. Ciò che vienè richiesto **NON** è la traduzione in italiano dello pseudocodice e cioè cose del tipo "c'è un indice i che parte da 0 e viene incrementato fino a che...". Risposte di questo tipo, o lo pseudocodice stesso, saranno valutate 0 punti.

   a) Si fornisca la relazione di ricorrenza che esprime un limite superiore al tempo di esecuzione dell'algoritmo da voi fornito al punto a) **nel caso generale, giustificando la risposta**. Si fornisca poi la relazione per il caso pessimo **giustificando anche qui la risposta**.

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

Progettazione di Algoritmi
27/6/2022

5. Analisi degli algoritmi e notazione asintotica (solo per quelli da 9 cfu)

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

b) Dimostrare che la seguente affermazione è vera giustificando la risposta. Occorre fornire le costanti c ed n_0.
   $n^4 + 100n = \Omega(n^3)$

c) Dimostrare che la seguente affermazione è falsa giustificando **matematicamente** la risposta.
   $n^n = O(n/4)^n$

d) Si dimostri che se $0 < f(n) = O(h(n))$ e se $0 < g(n) = O(q(n))$ allora $f(n)g(n) = O(h(n)q(n))$. Occorre utilizzare solo la definizione di O e nessuna altra proprietà.

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

```
i=0;
j=0;
while(i<=n and j<=m){
    if A[i]<A[j]{
            i=i+1
                }
    else{
            j=j+1
        }
}
```

Foglio minuta

Foglio minuta

Foglio minuta

Foglio minuta