Esercitazione

6 aprile 2022

Nelle prossime pagine, gli esercizi svolti

Prima prova intercorso 2017/18

1) 1 ☐

Qual è il tempo di esecuzione del seguente frammento di pseudocodice?

```
x=0
for i=1 to n-1
    for j=1 to logn
        x=i+j
return x
```

A. Θ (n)
B. Θ (n log n)
C. Θ (n^2)
D. Nessuna delle risposte precedenti

2) 2 ☐

Qual è la corretta successione delle funzioni seguenti affinché compaiano da sinistra a destra in ordine crescente di crescita asintotica: $\boldsymbol{n^2}$, $\boldsymbol{n(\log n)^3}$, $\boldsymbol{n\sqrt{n}}$?

A. $n (\log n)^3$, $n\sqrt{n}$, n^2
B. n^2, $n (\log n)^3$, $n\sqrt{n}$
C. $n (\log n)^3$, n^2 , $n\sqrt{n}$
D. Nessuna delle risposte precedenti.

Prima prova intercorso 2017/18

3) 3 ☐

Quale relazione di ricorrenza soddisfa il tempo di esecuzione T(n) del seguente algoritmo ricorsivo, supponendo che l'esecuzione di `qualcosa` richieda tempo costante?

```
PROVA(n)
if n=1 then do qualcosa
       else
           return PROVA(n/2)+n-1
```

A. $T(n) = T(n/2) + \Theta(n)$ con $T(1)= \Theta(1)$

B. $T(n) = T(n-1) + \Theta(n)$ con $T(1)= \Theta(1)$

C. $T(n) = T(n-1) + \Theta(1)$ con $T(1)= \Theta(1)$

D. Nessuna delle risposte precedenti

Prima prova intercorso 2017/18

4) 4 ☐

Una chiamata alla procedura `PARTITION` (come studiata) su A[1..5] = [3, 9, 8, 1, 2] restituisce

A. 2 B. 3 C. 5 D. Nessuna delle risposte precedenti

5) 5 ☐

Quali delle seguenti affermazioni è vera?

Il tempo di esecuzione dell'algoritmo `MERGE` per la fusione di due array ordinati di n elementi è

A. $O(n)$
B. $\Theta(n \log n)$
C. $O(\log n)$
D. Nessuna delle risposte precedenti

Prima prova intercorso 2017/18

Quesito 1 (16 punti)
Si risolva la seguente relazione di ricorrenza. Si potrà avere il massimo del punteggio se si mostrano **due diversi** modi di risolverla.

$$T(n) = 2\,T(n/2) + \Theta(1) \text{ con } T(1)=\Theta(1)$$

Prima prova intercorso 2017/18

Quesito 2 (18 punti)

a) Descrivere un algoritmo efficiente basato sul paradigma **divide et impera** che dato un vettore **ordinato** A[1..n] di interi strettamente positivi (cioè per ogni $1 \leq i \leq n$, $A[i] \geq 1$), restituisca il numero di occorrenze di 1 nel vettore A. Commentare il funzionamento dell'algoritmo.

b) Sia T(n) il tempo di esecuzione dell'algoritmo proposto al punto precedente. Scrivere la relazione di ricorrenza soddisfatta da T(n). Non è necessario mostrarne la soluzione.

Nota: Può essere utile sapere che esiste un algoritmo che risolve il problema in tempo O(log n).

Discusso in aula.
Siete pregati di inserire la soluzione completa sulla piattaforma di e-learning, per il Compito Conta 1 (D&I) .

Nelle prossime pagine, esercizi da svolgere

Prima prova intercorso 2017/18

Quesito 2 (18 punti)

a) Descrivere un algoritmo efficiente basato sul paradigma **divide et impera** che dato un vettore **ordinato** A[1..n] di interi strettamente positivi (cioè per ogni $1 \leq i \leq n$, $A[i] \geq 1$), restituisca il numero di occorrenze di 1 nel vettore A. Commentare il funzionamento dell'algoritmo.

b) Sia $T(n)$ il tempo di esecuzione dell'algoritmo proposto al punto precedente. Scrivere la relazione di ricorrenza soddisfatta da $T(n)$. Non è necessario mostrarne la soluzione.

Nota: Può essere utile sapere che esiste un algoritmo che risolve il problema in tempo $O(\log n)$.

Prima prova intercorso 2017/18

Quesito 3 (16 punti)
Si consideri il problema dello *scheduling* di intervalli pesato.
Ricostruire uno *scheduling* ottimale per il problema dato su un insieme di intervalli S={1, 2, … , 6} ordinato secondo il tempo di fine crescente degli intervalli (cioè $f_1 \leq f_2 \leq \ldots \leq f_6$), sapendo che i pesi degli intervalli sono rispettivamente w_1 =12, w_2 =2, w_3 =8, w_4 =9, w_5 =3, w_6 =10, che i valori della funzione p sono p(1)=0, p(2)=0, p(3)=2, p(4)=0, p(5)=1, p(6)=4 e l'array M calcolato dall'algoritmo di programmazione dinamica studiato è M[0 .. 6] = [0, 12, 12, 20, 20, 20, 30]. E' necessario giustificare la risposta.
Si noti che non occorre conoscere i valori dei tempi di inizio e di fine degli intervalli per ricostruire la soluzione.

Prima prova intercorso 2017/18

Quesito 4 (18 punti)
La soluzione ad un problema (a noi ignoto) è data da OPT(m,n) dove OPT(i,j) per i=0,1, …, m e j=0,1, …, n soddisfa la seguente relazione di ricorrenza dove v_1, v_2, … , v_m, sono dei valori dati.

$$\begin{aligned}
&OPT(i, 0) = 0 \text{ per ogni } i = 0,1, \dots, m\\
&OPT(0, j) = j \text{ per ogni } j=1, \dots, n\\
&OPT(i,j) = \max \{ v_i + OPT(i-1, j),\ OPT(i, j-1) \} \text{ altrimenti.}
\end{aligned}$$

a) Scrivere lo pseudocodice di un algoritmo di **programmazione dinamica** per il calcolo di OPT(m,n) ed analizzarne la complessità di tempo e di spazio, giustificando la risposta.
b) Indicare quali sono i punti salienti dell'algoritmo descritto al punto precedente che lo rendono un algoritmo di programmazione dinamica.

Si consideri il problema dello *scheduling* di attività (non pesato). Quale delle seguenti affermazioni è vera?

A. Ogni soluzione ottimale contiene l'intervallo che finisce per primo
B. Ogni soluzione ottimale contiene l'intervallo che inizia per primo
C. Esiste una soluzione ottimale che contiene l'intervallo che finisce per primo
D. Nessuna delle risposte precedenti

- Si considerino le stringhe X= $x_1\, x_2\, x_3\, x_4\, x_5$ = mamma e Y= $y_1\, y_2\, y_3$ = mia. Se il costo di un *gap* è 5, il costo di un *mismatch* fra due vocali differenti è 3, fra due consonanti differenti è 4 e fra vocale e consonante è 7, allora il costo dell'allineamento $x_1 – y_1$, $x_3 – y_2$, $x_5 – y_3$ è:

A. 7

B. 14

C. 17

D. Nessuna delle risposte precedenti

Il tempo di esecuzione del seguente frammento di pseudocodice è

```
for i=1 to logn
   for j=1 to logn
      x=i*j
return x
```

A. O(log n)
B. O(n log n), ma non Θ (n log n)
C. Θ (n^2)
D. Nessuna delle risposte precedenti

Il problema dello zaino è quello di determinare, dato un insieme di oggetti S={1,2,…,n}, dove l'oggetto *i* ha peso p_i e valore v_i, e uno zaino di capacità W:

A. La massima $\sum_{i=1}^{n} v_i$ tale che $\sum_{i=1}^{n} p_i \leq$ W
B. Un sottoinsieme S' di S tale che sia massima $\sum_{i=1}^{n} v_i$ e $\sum_{i=1}^{n} p_i \leq$ W
C. Un sottoinsieme S' di S tale che sia massima $\sum_{i\ S'} v_i$ e $\sum_{i\ S'} p_i \leq$ W

D. Nessuna delle risposte precedenti

La soluzione dell'equazione di ricorrenza $T(n) = 2T(n/2)+ \Theta(n)$ é

A. Θ (n log n)

B. O(n log n) ma non Θ(n log n)

C. Ω(n log n) ma non Θ(n log n)

D. Nessuna delle risposte precedenti

Nel problema dello *scheduling* di intervalli, due attività i e j, dove i ha tempo di inizio s_i e tempo di fine f_i, e j ha tempo di inizio s_j e tempo di fine f_j, sono definite compatibili se

A. $f_i \leq s_j$

B. $f_i \leq s_j$ e $f_j \leq s_i$

C. $f_i \leq s_j$ oppure $f_j \leq s_i$

D. Nessuna delle risposte precedenti

Si consideri il problema dello *scheduling* di intervalli pesato e la soluzione studiata. Supponendo che all'intervallo i sia associato un peso v_i, la relazione di ricorrenza per OPT(i) è:

A. $OPT(i) = \max \{ OPT(i-1),\ v_i + OPT(p(i)) \}$ con $OPT(0) = 0$

B. $OPT(i) = \max \{ OPT(i-1),\ v_i + OPT(p(i)) \}$ con $OPT(1) = 0$

C. $OPT(i) = \max \{ OPT(i-1),\ v_i + OPT(w - w_i) \}$ con $OPT(0) = 0$

D. Nessuna delle risposte precedenti

La complessità di tempo dell'algoritmo Knapsack studiato per il problema dello zaino su n oggetti e zaino di capacità W è:

A. $\Theta(n^2)$
B. $\Theta(nW)$
C. $\Theta(n+W)$
D. Nessuna delle risposte precedenti

Quesito 1 (26 punti) (*k-esimo minimo*)

a) Indicare le varie fasi in cui è suddiviso un algoritmo basato sulla tecnica *Divide et Impera*.

b) Descrivere un algoritmo basato sulla tecnica *Divide et Impera* che, dati un array A[1..n] di interi **positivi distinti** e un intero k, con $1 \leq k \leq n$, calcoli il k-esimo minimo di A. Tale algoritmo **non** dovrà fare alcun ricorso ad algoritmi di ordinamento, ma dovrà utilizzare la procedura `Partition` nella sua prima fase. Si potrà ottenere il massimo del punteggio solo se l'algoritmo è descritto tramite pseudo-codice. In ogni caso, è necessario spiegare verbalmente il funzionamento dell'algoritmo proposto e giustificarne la correttezza.

Esempio: se A[1..7] = [6, 9, 3, 2, 5, 7, 8] e k = 3 l'algoritmo dovrà restituire 5.

c) Analizzare la complessità di tempo nel caso peggiore dell'algoritmo proposto al punto b).

Appello del 10 novembre 2017

Quesito 1 (23 punti) (*Zaino*)

a) Definire il problema computazionale dello zaino (0-1), specificando i dati in ingresso e quelli in uscita.

b) Definire la funzione OPT(i, x) studiata per risolvere il problema e scrivere la relativa relazione di ricorrenza. E' necessario giustificare la risposta.

c) Eseguire l'algoritmo studiato sui seguenti dati: {1, 2, 3}, $w_1 = 2$, $w_2 = 3$, $w_3 = 1$; $v_1 = 4$, $v_2 = 2$, $v_3 = 5$ e $W=5$. E' necessario mostrare e commentare i passi salienti dell'esecuzione.

d) Mostrare come ottenere un insieme di oggetti ottimale per i dati del punto c), a partire dai valori ottimi calcolati al punto c).

Appello del 4 aprile 2018

Quesito 2 (24 punti) (*Programmazione dinamica*)

Si supponga che la soluzione ad un certo problema (a noi ignoto) sia data, per un certo intero n positivo, dal massimo fra i valori OPT(n,R) e OPT(n,B) definiti ricorsivamente come segue (R sta per Rosso e B sta per Blu):

$$OPT(1, R) = 2$$

$$OPT(1, B) = 1$$

$$OPT(i, R) = OPT(i-1, B) + 1, \text{ se } i > 1$$

$$OPT(i, B) = \max\{OPT(i, R) - 1, OPT(i-1, R)\}, \text{ se } i > 1$$

a) Calcolare i valori di OPT(i, R) e OPT(i, B) per ogni i=1, 2, …, 5, organizzandoli in una tabella.

b) Scrivere lo pseudocodice di un algoritmo **ricorsivo** per il calcolo della soluzione al problema.

c) Scrivere lo pseudocodice di un algoritmo di **programmazione dinamica** per il calcolo della soluzione al problema. Analizzarne la complessità di tempo e di spazio, giustificando la risposta.

Appello del 21 marzo 2019

Quesito 1 (26 punti) (*Giornata di seminari*)

Vi state occupando di organizzare una giornata di seminari nell'Aula Magna della vostra università. Avete avuto la disponibilità di vari relatori a tenere un loro intervento; ognuno ha specificato da che ora a che ora si terrebbe il suo seminario. Purtroppo, non riuscite ad organizzare la giornata in modo da inserire tutti i relatori. Dovete perciò scegliere alcuni fra i relatori in modo che i loro seminari possano essere svolti nell'aula senza sovrapposizioni di orario. Volete inoltre fare in modo che sia massimo il **tempo** totale di utilizzo effettivo dell'aula.

a) Definire il problema computazionale, specificandone i dati in ingresso e in uscita.

b) Indicare se si tratta di un problema studiato e, se sì, quale.

c) Si consideri un algoritmo *greedy* basato sul criterio di scelta del seminario che utilizza l'aula per più tempo. Mostrare, con un contro-esempio, che tale algoritmo non sempre porta ad una soluzione ottimale.

d) Progettare un algoritmo di **programmazione dinamica** che risolve il problema e valutarne la complessità. Si potrà ottenere il massimo della votazione solo se l'algoritmo è descritto tramite pseudo-codice ed è discussa la sua correttezza.

Appello 9 febbraio 2011

I numeri di Tribonacci sono cosi' definiti:

$R(0) = 0$

$R(1) = 0$

$R(2) = 1$

$R(n) = R(n-1) + R(n-2) + R(n-3)$ se $n \geq 3$.

a) Scrivere lo pseudocodice di un algoritmo di programmazione dinamica per il calcolo dell'n-esimo numero di Tribonacci $R(n)$.

b) Analizzare la complessita' di tempo e di spazio dell'algoritmo proposto.

c) E' possibile realizzare l'algoritmo con spazio $O(1)$? Giustificare la risposta.