Corso di Laurea in Informatica
Corso di Algoritmi
Esercizi proposti – 16 marzo 2018

1. Chiusura transitiva

Dato un grafo orientato, memorizzato come matrice di booleani, costruire la sua chiusura transitiva, ossia il grafo che ha un arco tra ogni coppia di vertici tra i quali esiste un cammino nel grafo dato in input.

2. Probabilità di superare gli esami del semestre

Uno studente in questo semestre può frequentare n corsi e conta di dedicare allo studio un numero di ore pari a k. È importante per lui superare almeno un esame. Il problema sta quindi nel ripartire le ore di studio tra i diversi corsi in modo da minimizzare la probabilità di non superarne neppure uno. Per decidere quanto tempo di studio dedicare a ciascun corso lo studente dispone di una matrice P dove P[i,j], con i <= 1 <= n e 0 <= j <= k, rappresenta la probabilità di fallire l'esame del corso i avendogli dedicato j ore di studio. Scrivere un algoritmo di programmazione dinamica che indichi allo studente quante ore di studio dedicare a ciascuna materia in modo da minimizzare la probabilità di non superare nessun esame.
Si ricorda che se p_i è la probabilità di fallire l'esame del corso i , la probabilità di fallire tutti gli esami è $\Pi_{1<= i <=n}\, p_i$.
Ad esempio, sia n = 3, k = 4 e le probabilità siano date dalla seguente matrice P:

$$\begin{pmatrix} .8 & .7 & .65 & .62 & .6 \\ .75 & .7 & .67 & .65 & .62 \\ .9 & .7 & .6 & .55 & .5 \end{pmatrix}$$

La strategia migliore è dedicare un'ora al primo corso e tre al terzo, trascurando quindi il secondo. In tal modo la probabilità di non superare nessuno dei tre esami è circa il 29%.

3. Parentesizzazione

Data un'espressione $E = c_1 op_1 c_2 op_2 \dots c_{n-1} op_{n-1} c_n$, dove c_i (1 <= i <= n) è un intero positivo e $op_j \in \{+, *\}$ (1 <= j < n), si deve inserire una serie di parentesi in modo che il valore dell'espressione calcolato rispettando la parentetizzazione introdotta risulti minimo. Scrivere un algoritmo che, data l'espressione E, determini il valore minimo ottenibile tramite una sua parentetizzazione. L'algoritmo deve avere complessità $\mathbf{O}(n^3)$.

5. Investimenti in borsa

Supponiamo di avere *n* pacchetti di 1.000 euro ognuno da investire in *m* imprese. Il numero di pacchetti è un intero e gli investimenti devono essere fatti con un numero intero di pacchetti. La tabella “*return*” descrive il guadagno atteso dagli investimenti individuali: *return*[i, j] è il guadagno atteso per un investimento di “i” pacchetti di 1.000 euro nell’impresa “j”.

a) Fornire una definizione induttiva del problema di determinare il massimo guadagno atteso dall’investimento degli n pacchetti disponibili.
b) Scrivere l’algoritmo di programmazione dinamica che si ottiene dalla definizione precedente.
c) Quanti confronti effettua l’algoritmo proposto?

6. Blocchi di lunghezza pari

Data una stringa binaria, definiamo *blocco* una sottostringa di simboli uguali consecutivi che non può essere estesa. Scrivere un algoritmo che, dato un intero n stampi tutte le stringhe binarie di lunghezza n che non contengono blocchi di lunghezza pari.
Ad esempio se n = 5 l’algoritmo deve stampare:
00000, 00010, 01000, 01010, 01110, 10101, 10001, 11101,10111,11111.
La complessità dell’algoritmo deve essere **O**(n D(n)), dove D(n) è il numero di stringhe da stampare.