Prova in itinere per l'esame di Algoritmi e Strutture Dati

Corso di Laurea in Informatica

6 Marzo 2012

Siano date n scatole $B_1, \ldots, B_n$. Ogni scatola B_i è descritta da una tripla (a_i, b_i, c_i), le cui componenti denotano rispettivamente lunghezza, larghezza e altezza della scatola. La scatola B_i può essere inserita nella scatola B_j se e solo se $a_i < a_j$, $b_i < b_j$ e $c_i < c_j$; in particolare, le scatole non possono essere ruotate. Per brevità indichiamo con $B_i \subset B_j$ il fatto che la scatola B_i può essere inserita nella scatola B_j. Descrivere un algoritmo efficiente per determinare il massimo valore di k tale che esiste una sequenza $B_{i_1}, \ldots, B_{i_k}$ che soddisfa le condizioni: $B_{i_1} \subset B_{i_2} \subset \cdots \subset B_{i_k}$ e $i_1 < i_2 < \cdots < i_k$ (ossia le scatole vanno scelte nell'ordine in cui compaiono).

1. scrivere una funzione di programmazione dinamica che calcola la più lunga sequenza di scatole che possono essere inserite l'una dentro l'altra;
2. determinare il tempo empirico che occorre per eseguire la procedura;
3. scrivere in commento la complessità di tempo della procedura da voi scritta, in funzione del numero n delle scatole considerate.

Commentare opportunamente il codice implementato.

Suggerimento: Conviene costruire un array Z di lunghezza n tale che $Z[i]$ indica la lunghezza della più lunga sottosequenza di scatole della sequenza $B_1, B_2, \ldots, B_i$ e che contiene B_i (ossia di cui B_i è la scatola più interna).

Fissato i, il valore di $Z[i]$ è calcolato facendo la seguente considerazione: per ogni $j < i$, se $a_i < a_j$, $b_i < b_j$, $c_i < c_j$ allora la più lunga sottosequenza di scatole che termina con la scatola B_j può essere prolungata con la scatola B_i. La più lunga sottosequenza di scatole di $B_1, B_2, \ldots, B_i$ che include la scatola B_i è ottenuto aggiungendo la scatola B_i alla più lunga sottosequenza di scatole che termina in un B_j con $j < i$. Quindi il problema è risolto dalla seguente equazione di ricorrenza:

$$Z[i] = \begin{cases} 1 & \text{se } i = 1 \text{ oppure se } B[i] \text{ non può essere inserito in nessuna scatola precedente} \\ 1 + \{\max_{0 \leq j < i} Z[j] \mid a_i < a_j, b_i < b_j, c_i < c_j\} & \end{cases}$$