Spiegazione dell'appello di Settembre 2022

13 Settembre, 2022

Esercizio 1. Conteggio di monete:

Assumiamo che $d_1 < d_2 < d_3 < \ldots < d_t$ sono i tagli di monete a disposizione. Nell'esercizio abbiamo $t = 4$ e $d_1 = 1, d_2 = 2, d_3 = 5, d_4 = 10$.

Definiamo una funzione $c(N, i)$ che da il conto di quanti modi ci sono per rappresentare N con i tagli di monete $d_1 < \ldots < d_i$, ovvero usando monete con **al massimo** valore d_i.

Nell'esempio abbiamo visto che $c(7, 4)$ è uguale a 6, e anche $c(7, 3)$ è 6. Invece $c(7, 2)$ è 4. È facile osservare che $c(0, t) = 1$ per ogni $t \geq 1$ e che, essendo $d_1 = 1$, abbiamo che $c(N, 1) = 1$ ogni $N \geq 0$.

Risolti i casi base, possiamo osservare che ogni soluzione conteggiata in $c(N, i)$ ha esattamente una delle seguenti due forme:

- una moneta di valore d_i insieme ad una scomposizione in monete di valore al massimo d_i del numero $N - d_i$;
- una scomposizione che non contiene monete di valore d_i.

Questo indica la ricorrenza, per $N > 0$ e $t > 1$

$$c(N, i) = c(N - d_i, i) + c(N, d_{i-1}) \ .$$

Da questa ricorrenza si può scrivere un programma bottom-up che riempie la tabella di valori $c(N, i)$, di complessità $O(N \times t)$. Visto che $t = 4$ la complessità è quella desiderata.

Esercizio 2. Stampa:

Questo esercizio ha una soluzione basata su backtraking. L'idea principale non è difficile, ma è delicata da implementare. Supponiamo $n = 10$ e $I = \{2, 3, 5, 9\}$. Se abbiamo costruito una stringa parziale 01101 allora è chiaro che essendoci solo 5 posizioni libere, il numero di 1 totali potrà essere solo un dei valori $\{3, 5\}$. Quindi nella funzione ricorsiva è utile ricordare

- il numero di 1 già inseriti
- il numero di posizioni libere rimaste (che può essere calcolato)
- il segmento del vettore I contentente solo i valori "raggiungibili" da quella stringa parziale.

Ad esempio si possono mantenere due indici `low` e `high` che puntano alle posizioni rispettivamente più piccola e più grande tra quelle ancora raggiungibili, mantenendo l'invariante che la stringa parziale possa essere completata almeno in un modo. All'inizio `low=0` e `high=`$(n - 1)$. La funzione può iniziare così:

Assumendo che I sia ordinato e che sia costituito da numeri distinti e tra 0 e n, la chiamata iniziale è `RicStampa(n,[],I,0,0,len(I)-1)`. Vediamo come gestire i due rami.

È possibile aggiungere uno 0 alla stringa? Il numero massimo di 1 nella stringa finale è uguale al numero di 1 già nella stringa parziale più il numero di posizioni attualmente libere, meno la posizione a cui verrebbe applicato lo 0. Se questo numero è strettamente inferiore a `I[low]` allora qualunque stringa generata non raggiungerebbe un valore accettabile, e quindi il ramo va tagliato. Se invece non è inferiore a `I[low]` **esiste almeno una stringa accettabile che estende quel ramo** e quindi lo zero può essere aggiunto.

Osservazione: vogliamo mantenere che tutti i valori nel segmento di I indicato da `low` and `high` siano raggiungibili. Aver aggiunto uno zero alla stringa potrebbe aver reso irragiungibile `I[high]` e in questo caso bisognerebbe scartare quel valore.

Il codice potrebbe essere una cosa simile:

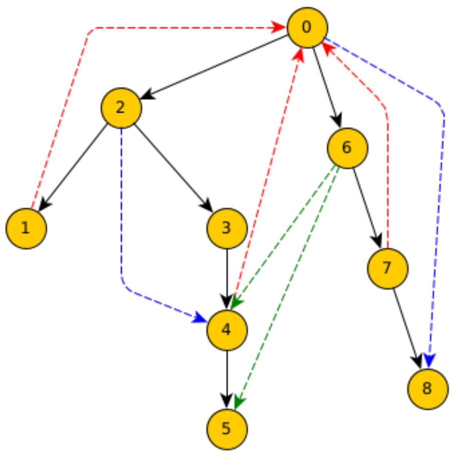
Il codice per il ramo 1 segue un ragionamento analogo.

Osservate che non si tagliano mai entrambi i rami e quindi ogni nodo foglia della ricorsione corrisponde ad una stringa che viene stampata. Il costo della stampa nei nodi foglia, il costo delle operazioni nei nodi interni, e la relazione tra il numero di nodi foglia e nodi interni determina la complessità.

Esercizio 3. Grafi:

La DFS produce a partire da 0 produce un singolo albero di visita con

- archi all'indietro: $(1, 0), (4, 0), (7, 0)$;
- archi in avanti: $(2, 4), (0, 8)$;
- archi di attraversamento $(6, 5), (6, 4)$.



Eliminati gli archi all'indietro non c'è bisogno di rifare la visita in profondità sul nuovo grafo. L'albero di visita è lo stesso, così come i tempi di inizio e fine visita di ogni nodo. L'ordine topologico risultante è

$$0, 6, 7, 8, 2, 3, 4, 5, 1 \ .$$