

Esercizi di Programmazione Greedy

- **Selezione di file**

Si supponga di avere n files di lunghezze l_1, \dots, l_n (interi positivi) che bisogna memorizzare su un disco di capacità data D . Si assuma che la somma delle lunghezze di questi files ecceda la capacità del disco. Il problema sta nel selezionare un sottoinsieme degli n files che abbia cardinalità massima e che possa essere memorizzato sul disco. Descrivere un algoritmo greedy che risolve il problema, provarne la correttezza e valutare la complessità di una sua efficiente implementazione.

- **Tappe per il rifornimento**

Si supponga di dover effettuare un viaggio dalla località A alla località B con un'auto che ha un'autonomia di k chilometri. Lungo il percorso, a partire da A sono presenti n distributori di benzina ciascuno distante dal precedente meno di k chilometri e l'ultimo dista meno di k chilometri da B . Sia d_i la distanza che separa il distributore i dal distributore $i + 1$ per $i = 1, 2, \dots, n - 1$ e sia d_n la distanza da B dell'ultimo distributore. Descrivere un algoritmo greedy che seleziona un numero minimo di distributori in cui far tappa durante il viaggio. Descrivere un algoritmo greedy che risolve il problema, provarne la correttezza e valutare la complessità di una sua efficiente implementazione.

- **L'insieme univoco**

In un grafo orientato G un sottoinsieme A degli archi è detto univoco se non contiene 2 o più archi uscenti dallo stesso nodo. Descrivere un algoritmo greedy che preso in input un grafo G orientato con pesi positivi sugli archi, trovi un insieme univoco A di archi di G di peso massimo. Provare la correttezza dell'algoritmo proposto e valutare la complessità di una sua efficiente implementazione.

- **Intervalli**

Dato un vettore V di n numeri reali, bisogna calcolare il numero minimo di intervalli di lunghezza unitaria in grado di ricoprire tutti i valori di V . Ad esempio se $n = 5$ e $V[1] = 2.5$, $V[2] = 3.8$, $V[3] = 1.5$, $V[4] = 3.1$, $V[5] = 1.8$, allora il numero minimo di intervalli unitari è 2 e una possibile soluzione è $\{[1.5, 2.5], [3, 4]\}$. Descrivere un algoritmo greedy che risolve il problema, provarne la correttezza e valutare la complessità di una sua efficiente implementazione.

- **Aule**

Si assuma di dover assegnare aule in cui tenere n lezioni e che ciascuna lezione i è caratterizzata da un tempo di inizio s_i ed un tempo di fine f_i (dove naturalmente si ha $s_i < f_i$). Tenendo conto che in una stessa aula non possono tenersi più lezioni contemporaneamente, si

trovi un algoritmo greedy per determinare un assegnazione delle lezioni alle aule che minimizzi il numero di aule utilizzate. Provare la correttezza dell'algoritmo proposto e valutare la complessità di una sua efficiente implementazione.

- **Utilizzazione d'aula**

Date n attività (lezioni, seminari, ecc.) ognuna delle quali caratterizzata da un tempo di inizio s_i ed un tempo di fine f_i (dove naturalmente si ha $s_i < f_i$), si vuole trovare un insieme di attività che possono essere svolte in un'unica aula senza sovrapposizioni e che massimizzi il tempo totale di utilizzo dell'aula. Proporre un algoritmo greedy per questo problema. Discutere la correttezza dell'algoritmo: se non risolve il problema valutarne il rapporto d'approssimazione.