Esercizi su Greedy

Al fine di invogliare gli studenti a esercitarsi con il materiale visto a lezione, i seguenti esercizi sono proposti senza soluzioni.

Exercise 1.1 Let $S = \{[a_1, b_1], [a_2, b_2], \ldots, [a_n, b_n]\}$ be a set of closed intervals on the real line. We say that $C \subseteq S$ is a *covering subset* for S if, for any interval $[a, b] \in S$, there exists an interval $[a', b'] \in C$ such that $[a, b] \subseteq [a', b']$ (that is, $a \geq a'$ and $b \leq b'$).

(a) Write a greedy $O(n \log n)$ algorithm that, on input S, returns a covering subset C^* of minimum size.

(b) Prove the greedy choice property for the above algorithm.

(c) Prove the optimal substructure property for the problem.

Exercise 1.2 Let $C = \{1, 2, \ldots, n\}$ denote a set of n rectangular frames. Frame i has base $d[i].b$ and height $d[i].h$. We say that Frame i *encapsulates* Frame j if $d[i].b \geq d[j].b$ and $d[i].h \geq d[j].h$. (Note that a frame encapsulates itself). An *encapsulating subset* $C' \subseteq C$ has the property that for each $j \in C$ there exists $i \in C'$ such that i encapsulates j.

(a) Design and analyze a greedy algorithm that, on input the vector $\boldsymbol{d}$ of sizes of a set C of n frames, returns a *minimum size* encapsulating subset $C' \subseteq C$ in $O(n \log n)$ time.

(b) Prove the greedy choice property.

(c) Prove the optimal substructure property.

Exercise 1.3 Let $n > 0$. Given a set of positive real numbers $S = \{a_1, a_2, \ldots, a_n\}$, with $0 < a_i < a_j < 1$ for $1 \leq i < j \leq n$, a *(2,1)-boxing* of S of size k is a partition $\mathcal{P} = \{S_1, S_2, \ldots, S_k\}$ of S into k disjoint subsets (that is, $\bigcup_{j=1}^{k} S_j = S$ and $S_r \cap S_t = \emptyset, 1 \leq r \neq t \leq k$) which satisfies the following constraints:

$$|S_j| \leq 2 \quad \text{and} \quad \sum_{a \in S_j} a \leq 1, \ 1 \leq j \leq k.$$

A practical instance of $(2,1)$-boxing could be the following: the s_i's may correspond to the weights of different items to be boxed, where a box cannot contain more than two items whose combined weight cannot exceed 1 unit of weight. For instance, if $S = \{0.2, 0.6, 0.7\}$, an optimal 2-boxing is $\{ \{0.2, 0.7\}, \{0.6\} \}$. We want to determine a $(2,1)$-boxing of minimum size.

Point 1 Give the pseudocode of a greedy algorithm which returns an optimal $(2,1)$-boxing in linear time.

Point 2 Prove that the algorithm has the greedy choice property.

Point 3 Prove that the algorithm has the optimal substructure property.

Exercise 1.4 Given n programs $P_1, P_2, \ldots, P_n$, assume that the running time of P_i on a given machine $\mathcal{M}$ be t_i, with $1 \leq i \leq n$. An *execution order* of the n programs on $\mathcal{M}$ is a permutation of the indices $\pi : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$, prescribing that the programs are to be executed on $\mathcal{M}$ one after the other, according to the sequence $\langle P_{\pi(1)}, P_{\pi(2)}, \ldots, P_{\pi(n)} \rangle$. For a given execution order π, the *wait time* of the i-th executed program in the order, for $i \geq 2$, is clearly $\tau_i = \sum_{j=1}^{i-1} t_{\pi(j)}$, and the *cumulative wait time* π is $A_\pi = \sum_{i=2}^{n} \tau_i$. We want to determine the execution order $\pi^\star$ associated to the minimum cumulative wait time $A_{\pi^\star}$.

Prove the following greedy choice property: *every* optimal execution order $\pi^\star$ is such that $t_{\pi^\star(1)} = \min\{t_i : 1 \leq i \leq n\}$. In other words the execution order which minimizes the cumulative wait time must schedule the n programs by nondecreasing running time.

Exercise 1.5 Given a set $S = \{s_1, s_2, \ldots, s_n\} \subseteq \boldsymbol{R}^+$, with $s_1 < s_2 < \ldots < s_n$, and a positive real value $\delta < 1$, a δ-*trimming* of S is a subset $T \subseteq S$ such that

$$\forall s \in S \, \exists t \in T : s(1 - \delta) \leq t \leq s.$$

In other words, for each element $s \in S$ there is an element $t \in T$ that approximates s from below with relative error $(s - t)/s \leq \delta$. Trimming is a useful primitive to obtain small, accurate summaries of large datasets. Given S and δ, we want to determine δ-trimming of minimum-cardinality.

Point 1 Give the pseudocode of a greedy algorithm which returns such a δ-trimming in linear time.

Point 2 Prove that the algorithm has the greedy choice property.

Point 3 Prove that the algorithm has the optimal substructure property.

Exercise 1.6 Consider a variant of the *Activity Selection Problem*, where the input set of intervals $S = \{[s_1, f_1), ..., [s_n, f_n)\}$ is sorted by *non decreasing* values of the s_i's, that is, $s_1 \leq s_2... \leq s_n$. As in the original problem, we want to determine a maximum set of pairwise disjoint activities.

Point 1 Design an $O(n)$ algorithm for the above problem.

Point 2 Prove that the greedy choice property holds.

Point 3 Prove that the optimal substructure property holds.