CS 473G: Graduate Algorithms, Spring 2007
Homework 3

Due Friday, March 9, 2007

Remember to submit **separate, individually stapled** solutions to each problem.

As a general rule, a complete, full-credit solution to any homework problem should fit into two typeset pages (or five hand-written pages). If your solution is significantly longer than this, you may be including too much detail.

---

1. (a) Let $X[1..m]$ and $Y[1..n]$ be two arbitrary arrays. A *common supersequence* of X and Y is another sequence that contains both X and Y as subsequences. Describe and analyze an efficient algorithm to compute the function $scs(X, Y)$, which gives the length of the *shortest* common supersequence of X and Y.

   (b) Call a sequence $X[1..n]$ *oscillating* if $X[i] < X[i+1]$ for all even i, and $X[i] > X[i+1]$ for all odd i. Describe and analyze an efficient algorithm to compute the function $los(X)$, which gives the length of the longest oscillating subsequence of an arbitrary array X of integers.

   (c) Call a sequence $X[1..n]$ of integers *accelerating* if $2 \cdot X[i] < X[i-1] + X[i+1]$ for all i. Describe and analyze an efficient algorithm to compute the function $lxs(X)$, which gives the length of the longest accelerating subsequence of an arbitrary array X of integers.

   *[Hint: Use the recurrences you found in Homework 2. You do not need to prove **again** that these recurrences are correct.]*

2. Describe and analyze an algorithm to solve the traveling salesman problem in $O(2^n \operatorname{poly}(n))$ time. Given an undirected n-vertex graph G with weighted edges, your algorithm should return the weight of the lightest Hamiltonian cycle in G (or ∞ if G has no Hamiltonian cycles).

3. Let G be an arbitrary undirected graph. A set of cycles $\{c_1, \ldots, c_k\}$ in G is *redundant* if it is non-empty and every edge in G appears in an even number of c_i's. A set of cycles is *independent* if it contains no redundant subsets. (In particular, the empty set is independent.) A maximal independent set of cycles is called a *cycle basis* for G.

   (a) Let C be any cycle basis for G. Prove that for any cycle γ in G **that is not an element of C**, there is a subset $A \subseteq C$ such that $A \cup \{\gamma\}$ is redundant. In other words, prove that γ is the 'exclusive or' of some subset of basis cycles.

   Solution: The claim follows directly from the definitions. A cycle basis is a *maximal* independent set, so if C is a cycle basis, then for any cycle $\gamma \notin C$, the larger set $C \cup \{\gamma\}$ cannot be an independent set, so it must contain a redundant subset. On the other hand, if C is a basis, then C is independent, so C contains no redundant subsets. Thus, $C \cup \{\gamma\}$ must have a redundant subset B that contains γ. Let $A = B \setminus \{\gamma\}$. ■

(b) Prove that the set of independent cycle sets form a matroid.

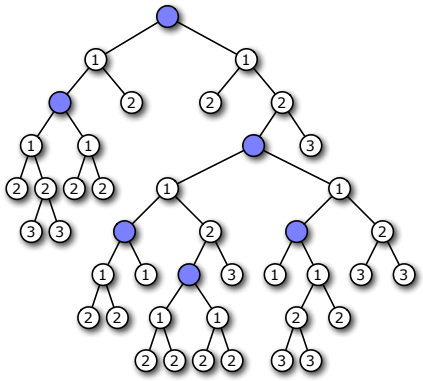
(c) Now suppose each edge of G has a weight. Define the weight of a cycle to be the total weight of its edges, and the weight of a *set* of cycles to be the total weight of all cycles in the set. (Thus, each edge is counted once for every cycle in which it appears.) Describe and analyze an efficient algorithm to compute the minimum-weight cycle basis of G.

4. Let T be a rooted binary tree with n vertices, and let $k \leq n$ be a positive integer. We would like to mark k vertices in T so that every vertex has a nearby marked ancestor. More formally, we define the *clustering cost* of a clustering of any subset K of vertices as

$$\mathit{cost}(K) = \max_v \mathit{cost}(v, K),$$

where the maximum is taken over all vertices v in the tree, and

$$\mathit{cost}(v, K) = \begin{cases} 0 & \text{if } v \in K \\ \infty & \text{if } v \text{ is the root of } T \text{ and } v \notin K \\ 1 + \mathit{cost}(\mathit{parent}(v)) & \text{otherwise} \end{cases}$$

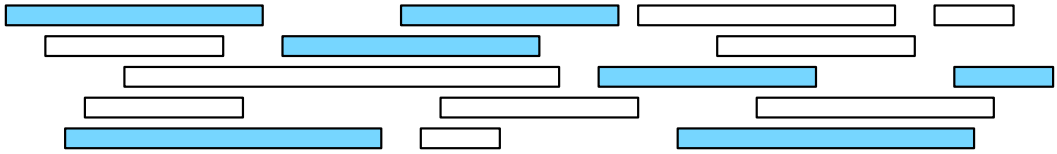


A subset of 5 vertices with clustering cost 3

Describe and analyze a dynamic-programming algorithm to compute the minimum clustering cost of any subset of k vertices in T. For full credit, your algorithm should run in $O(n^2k^2)$ time.

5. Let X be a set of n intervals on the real line. A subset of intervals $Y \subseteq X$ is called a *tiling path* if the intervals in Y cover the intervals in X, that is, any real value that is contained in some interval in X is also contained in some interval in Y. The *size* of a tiling cover is just the number of intervals.

Describe and analyze an algorithm to compute the smallest tiling path of X as quickly as possible. Assume that your input consists of two arrays $X_L[1..n]$ and $X_R[1..n]$, representing the left and right endpoints of the intervals in X. If you use a greedy algorithm, you must prove that it is correct.



A set of intervals. The seven shaded intervals form a tiling path.