| The Design and Analysis of Algorithms | Problem Set 4 |
|---|---|
| CS 681 Fall 2003 | Due Friday, November 7, 2003 |

Solve 3 of the following 4 problems. You may solve all 4 for extra credit. We will maintain a FAQ for the problem set on the course Web page.

**(1)** (15 points) Suppose you are given a directed graph $G = (V, E)$ with costs on the edges $c_e$ for $e \in E$ and a sink $t$ (costs may be negative). Assume that you also have values $d(v)$ for $v \in V$. Someone claims that for each node $v \in V$, $d(v)$ is the costs of the minimum cost path from node $v$ to the sink $t$.

**(a)** Give a linear time algorithm (time $O(m)$ if the graph has $m$ edges) that verifies if this claim is correct.

**(b)** Assume that the distances are correct, and $d(v)$ is finite for all $v \in V$. Now you need to compute distances to a different sink $t'$. Give an $O(m \log n)$ algorithm for computing distances $d'(v)$ for all nodes $v \in V$ to the terminal $t'$. **Hint:** It is useful to consider a new cost function defined as follows: for edge $e = (v, w)$ let $c'_e = c_e - d(v) + d(w)$. Is there a relation between costs of paths for costs $c$ and $c'$?

**(2)** (15 points) You're consulting for a group of people, whose jobs consist of monitoring and analyzing electronic signals coming from ships in coastal Atlantic waters. They want a fast algorithm for a basic primitive that arises frequently: "untangling" a superposition of two known signals. Specifically, they're picturing a situation in which each of two ships is emitting a short sequence of 0's and 1's over and over, and they want to make sure that the signal they're hearing is simply an *interleaving* of these two emissions, with nothing extra added in.

This describes the whole problem; we can make it a little more explicit as follows. Given a string $x$ consisting of 0's and 1's, we write $x^k$ to denote $k$ copies of $x$ concatenated together. We say that a string $x'$ is a *repetition* of $x$ if it is a prefix of $x^k$ for some number $k$. So $x' = 10110110110$ is a prefix of $x = 101$.

We say that a string $s$ is an *interleaving* of $x$ and $y$ if its symbols can be partitioned into two (not necessarily contiguous) subsequences $s'$ and $s''$, so that $s'$ is a repetition of $x$ and $s''$ is a repetition of $y$. (So each symbol in $s$ must belong to exactly one of $s'$ or $s''$.) For example, if $x = 101$ and $y = 00$, then $s = 100010101$ is an interleaving of $x$ and $y$, since characters 1,2,5,7,8,9 form 101101 — a repetition of $x$ — and the remaining characters 3,4,6 form 000 — a repetition of $y$.

In terms of our application, $x$ and $y$ are the repeating sequences from the two ships, and $s$ is the signal we're listening to: we want to make sure it "unravels" into simple repetitions of $x$ and $y$.

**(3)** (15 points) Consider a network of workstations modeled as an undirected graph $G$, where each node is a workstation, and the edges represent direct communication links. We'd like to place copies of a database at nodes in $G$, so that each node is close to at least one copy.

Specifically, assume that each node $v$ in $G$ has a cost $c_v$ charged for placing a copy of the database at node $v$. The MIN-COST SERVER PLACEMENT problem is as follows. Given the network $G$, and costs $\{c_v\}$, find a set of nodes $S \subseteq V$ of minimum total cost $\sum_{v \in S} c_v$, so that if we place copies of a database at each node in $S$, then every workstation either has a copy of the database, or is connected by a direct link to a workstation that has a copy of the database.

Give a polynomial time algorithm for the special case of the MIN-COST SERVER PLACEMENT where the graph $G$ is a tree.

Note the difference between SERVER PLACEMENT and VERTEX COVER. If the graph $G$ is a path of consisting of 6 nodes, then VERTEX COVER If the graph $G$ is a path of consisting of 6 nodes, then VERTEX COVER needs to select at least 3 of the 6 nodes, while the second and the 5th node form a valid solution of the MIN-COST SERVER PLACEMENT problem, requiring only two nodes.

**(4)** (15 points) A *$k$-coloring* of an undirected graph $G = (V, E)$ is an assignment of one of the numbers $\{1, 2, \ldots, k\}$ to each node, so that if two nodes are joined by an edge, then they are assigned different numbers. The *chromatic number* of $G$ is the minimum $k$ such that it has a $k$-coloring. For $k \geq 3$, it is NP-complete to decide whether a given input graph has chromatic number $\leq k$. (You don't have to prove this.)

**(a)** Show that for every natural number $w \geq 1$, there is a number $k(w)$ so that the following holds. If $G$ is a graph of tree-width at most $w$, then $G$ has chromatic number at most $k(w)$. (The point is that $k(w)$ depends only on $w$, not on the number of nodes in $G$.)

**(b)** Given an undirected $n$-node graph $G = (V, E)$ of tree-width at most $w$, show how to compute the chromatic number of $G$ in time $O(f(w) \cdot p(n))$, where $p(\cdot)$ is a polynomial but $f(\cdot)$ can be an arbitrary function.