CS 170 HW 5

Due on 2019-02-25, at 10:00 pm

1 Study Group

List the names and SIDs of the members in your study group.

2 Updating Labels

You are given a tree $T = (V, E)$ with a designated root node r, and for each vertex $v \in V$, a non-negative integer label $l(v)$. If $l(v) = k$, we wish to relabel v, such that $l_{\text{new}}(v)$ is equal to $l(w)$, where w is the kth ancestor of v in the tree. We follow the convention that the root node, r, is its own parent. Give a linear time algorithm to compute the new label, $l_{\text{new}}(v)$ for each v in V

Slightly more formally, the *parent* of any $v \neq r$, is defined to be the node adjacent to v in the path from r to v. By convention, $p(r) = r$. For $k > 1$, define $p^k(v) = p^{k-1}(p(v))$ and $p^1(v) = p(v)$ (so p^k is the kth ancestor of v). Each vertex v of the tree has an associated non-negative integer label $l(v)$. We want to find a linear-time algorithm to update the labels of all vertices in T according to the following rule: $l_{\text{new}}(v) = l(p^{l(v)}(v))$.

3 Count Four Cycle

Given as input an undirected graph $G = (V, E)$ design an algorithm to decide whether G contains a four cycle (A cycle $v - u_1 - u_2 - u_3 - v$ where $u_1 \neq u_2 \neq u_3 \neq u_1$ and $u_i \neq v$). Your algorithm should run in time $O(|V|^3)$. You may assume that the graph is given as either an adjacency matrix or an adjacency list.

4 Constrained Dijkstra

Given as input a directed graph $G = (V, E)$, positive edge weights, ℓ_e, for each edge $e \in E$ and a particular vertex $v_o \in V$. Compute the shortest paths between all pairs of vertices in $O((|V| + |E|) \log |E|)$ time with the restriction that each of these paths pass through v_0.

5 Arbitrage

Shortest-path algorithms can also be applied to currency trading. Suppose we have n currencies $C = \{c_1, c_2, \ldots, c_n\}$: e.g., dollars, Euros, bitcoins, dogecoins, etc. For any pair i, j of currencies, there is an exchange rate $r_{i,j}$: you can buy $r_{i,j}$ units of currency c_j at the price of one unit of currency c_i. Assume that $r_{i,i} = 1$ and $r_{i,j} \geq 0$ for all i, j.

The Foreign Exchange Market Organization (FEMO) has hired Oski, a CS170 alumnus, to make sure that it is not possible to generate a profit through a cycle of exchanges; that is, for any currency $i \in C$, it is not possible to start with one unit of currency i, perform a series of exchanges, and end with more than one unit of currency i. (That is called *arbitrage*.)

More precisely, arbitrage is possible when there is a sequence of currencies $c_{i_1}, \ldots, c_{i_k}$ such that $r_{i_1,i_2} \cdot r_{i_2,i_3} \cdot \cdots \cdot r_{i_{k-1},i_k} \cdot r_{i_k,i_1} > 1$. This means that by starting with one unit of currency c_{i_1} and then successively converting it to currencies $c_{i_2}, c_{i_3}, \ldots, c_{i_k}$ and finally back to c_{i_1}, you would end up with more than one unit of currency c_{i_1}. Such anomalies last only a fraction of a minute on the currency exchange, but they provide an opportunity for profit.

We say that a set of exchange rates is arbitrage-free when there is no such sequence, i.e. it is not possible to profit by a series of exchanges.

(a) Give an efficient algorithm for the following problem: given a set of exchange rates $r_{i,j}$ which is *arbitrage-free*, and two specific currencies s, t, find the most advantageous sequence of currency exchanges for converting currency s into currency t.

Hint: represent the currencies and rates by a graph whose edge weights are real numbers.

(b) Oski is fed up of manually checking exchange rates, and has asked you for help to write a computer program to do his job for him. Give an efficient algorithm for detecting the possibility of arbitrage. You may use the same graph representation as for part (a).

6 Bounded Bellman-Ford

Modify the Bellman-Ford algorithm to find the weight of the lowest-weight path from s to t with the restriction that the path must have at most k edges.