CS 161 Fall 2017: Section 6

Currency Exchange

Suppose the various economies of the world use a set of currencies $C_1, \ldots, C_n$—think of these as dollars, pounds, bitcoins, etc. Your bank allows you to trade each currency C_i for any other currency C_j, and finds some way to charge you for this service (in a manner to be elaborated in the subparts below). We will devise algorithms to trade currencies to maximize the amount we end up with.

(a) Suppose that for each ordered pair of currencies (C_i, C_j), the bank charges a flat fee of $f_{ij} > 0$ dollars to exchange C_i for C_j (regardless of the quantity of currency being exchanged). Devise an efficient algorithm which, given a starting currency C_s, a target currency C_t, and a list of fees f_{ij} for all $i, j \in \{1, \ldots, n\}$, computes the cheapest way (that is, incurring the least in fees) to exchange all of our currency in C_s into currency C_t. Justify the correctness of your algorithm and its runtime.

(b) Consider the more realistic setting where the bank does not charge flat fees, but instead uses exchange *rates*. In particular, for each ordered pair (C_i, C_j), the bank lets you trade one unit of C_i for $r_{ij} > 0$ units of C_j. Devise an efficient algorithm which, given starting currency C_s, target currency C_t, and a list of rates r_{ij}, computes a sequence of exchanges that results in the greatest amount of C_t. Justify the correctness of your algorithm and its runtime.[Hint: How can you turn a product of terms into a sum?]

(c) Due to fluctuations in the markets, it is occasionally possible to find a sequence of exchanges that lets you start with currency A, change into currencies, B, C, D, etc., and then end up changing back to currency A in such a way that you end up with more money than you started with—that is, there are currencies $C_{i_1}, \ldots, C_{i_k}$ such that

$$r_{i_1 i_2} \times r_{i_2 i_3} \times \cdots \times r_{i_{k-1} i_k} \times r_{i_k i_1} > 1.$$

Devise an efficient algorithm that finds such an anomaly if one exists. Justify the correctness of your algorithm and its runtime.

Traveling Across the Country

We have a graph representation of the country, where nodes u_i are on the east coast and nodes v_j are on the west coast, with n nodes in total. We also have $|E|$ undirected edges representing distances between these cities.

(a) Design an efficient algorithm to compute the shortest path starting at *any* city on the east coast and ending at *any* city on the west coast.

(b) This time, we start from a specific city u_i and end at a specific city v_j. However, we impose an additional restriction that we must traverse one of the edges between two cities 3 times: that is, for some w, w', we must traverse $w \to w'$, $w' \to w$, and then $w \to w'$ again. Design an efficient algorithm to find the shortest path from u_i to v_j with this additional constraint.