CS 161 (Stanford, Fall 2025) Section 5 Extra

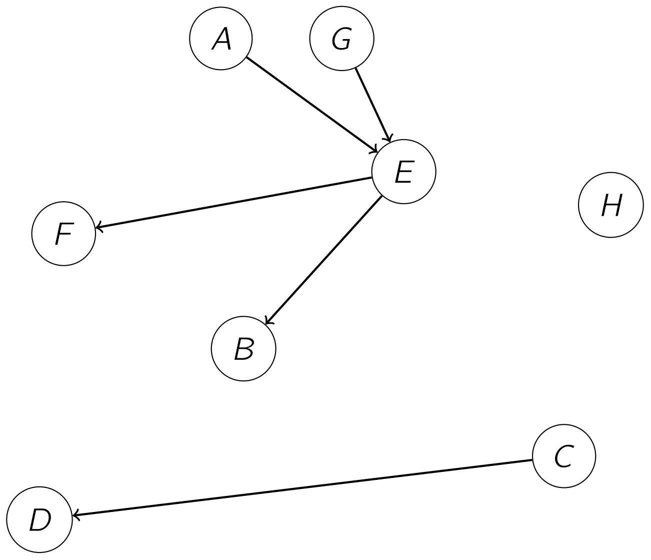
1 True or False

1. Dijkstra's can be used for successfully finding the shortest path from a source to all other vertices in a graph with negative edges, but not with negative cycles.
2. To find the shortest path from one vertex to another in an unweighted graph, you should use Dijkstra's algorithm as it is the most efficient solution.
3. Adding a new positive edge to an undirected weighted graph with positive edges cannot lead to the output values of Dijkstra's increasing.

2 Hyperlinks can go backward?

On the internet, many pages have links pointing to other pages, but sometimes it's not possible to reach a site you were on previously without clicking the "back" button in your browser. Elgoog can model their website as a directed graph G with n pages, and each page has some number of links to other pages. There are a total of m links over all these pages. Currently, it's not possible to get from some pages to some other pages without clicking the back button, and sometimes not possible at all! They want your help in designing an algorithm which can output the **minimum** total number of extra links they need to add so that every page is reachable from every other page.

1. In the given graph, find the minimum number of links that Elgoog must add.



2. Suppose G is a directed, acyclic graph (DAG) where every sink is reachable from every source. Define $S \subseteq V$ as the set of *source* nodes: those vertices with no incoming

edges and $T \subseteq V$ as the set of *sink* nodes: those vertices with no outgoing edges. **Prove** that the minimum number of links which have to be added is $\max(|S|, |T|)$.

3 Currency conversion

Suppose the various economies of the world use a set of currencies $C_1, C_2, \ldots, C_n$ – 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.

3.1 Flat fees

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 $i, j \in \{1, 2, \ldots, n\}$, computes the cheapest way (that is, incurring the least in fees) to exchange all of our currency in C_s to currency C_t. Justify the correctness of your algorithm and its runtime.

3.2 Exchange rates

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} units of C_j, i.e. you receive r_{ij} units of C_j in exchange of one unit of C_i. 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.

3.3 Making money

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

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

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