Example sheet week 6

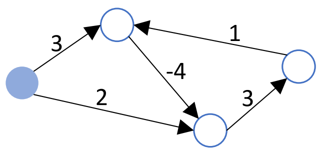
Paths. Trees. Flows. Matchings.
Algorithms—DJW—2016/2017

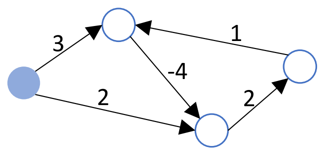
Questions labelled FS are from Dr Stajano's list of exercises. Questions labelled CLRS are from *Introduction to Algorithms, 3rd ed.* by Cormen, Leiserson, Rivest and Stein. Questions labelled $*$ involve more coding or more thinking and are best tackled after the other questions, if you have time.

Question 1 (CLRS-24.3-4). A contractor has written a program that she claims solves the shortest path problem, on directed graphs with non-negative edge weights. The program produces v.`distance` and v.`come_from` for every vertex v in a graph. Give an $O(V+E)$-time algorithm to check the output of the contractor's program.

Question 2*. The Bellman-Ford code given in the handout will report "Negative cycle detected" if there is a negative-weight cycle reachable from the start vertex. Modify the code so that, in such cases, it returns a negative-weight cycle, rather than just reporting that one exists.

Question 3. By hand, run both Dijkstra's algorithm and the Bellman-Ford algorithm on each of the graphs below, starting from the shaded vertex. The labels indicate edge costs, and one is negative. Does Dijkstra's algorithm correctly compute minimum weights?





Question 4 (CLRS-25.3-4)*. An engineer friend tells you there is a simpler way to reweight edges than the method used in Johnson's algorithm. Let w^* be the minimum weight of all edges in the graph, and just define $w'(u \to v) = w(u \to v) - w^*$ for all edges $u \to v$. What is wrong with your friend's idea?

Question 5 (FS58, FS59). In Section 5.8, we defined $M_{ij}^{(n)}$ to be the minimum weight among all paths from i to j that have n or fewer edges, and we derived the recursion

$$M^{(1)} = W, \qquad M^{(n)} = M^{(n-1)} \otimes W$$

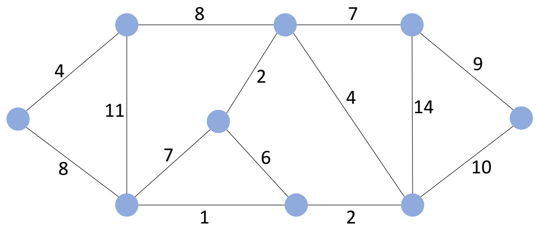
where $W_{ij} = 0$ if $i = j$, $W_{ij} = \text{weight}(i \to j)$ if there is an edge $i \to j$, and $w_{ij} = \infty$ otherwise. How can we define $M^{(0)}$ so that $M^{(1)} = M^{(0)} \otimes W$? What is the relationship between $M^{(0)}$ and the identity matrix I used in standard matrix multiplication?

Question 6*. The *incidence matrix* of a directed graph is a $V \times E$ matrix B defined by

$$B_{ve} = \begin{cases} 1 & \text{if } e = u \to v \text{ for some vertex } u \\ -1 & \text{if } e = v \to w \text{ for some vertex } w \\ 0 & \text{otherwise.} \end{cases}$$

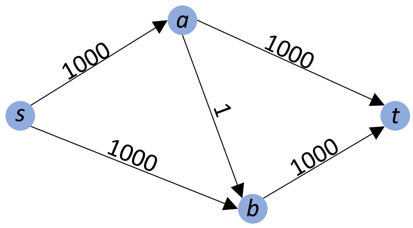
From a directed graph g, we can produce a dual graph g' whose vertices correspond to edges of g. Draw an edge connecting two vertices e and f in g', if the corresponding edges in g share a vertex, i.e. if $e = u \to v$ and $f = v \to w$ for some vertices u, v and w in g. Give a formula for the adjacency matrix of g' in terms of the incidence matrix of g.

Question 7 (FS55, FS56). Try to find, by hand, a minimum spanning tree for this graph. Now run, by hand, the Kruskal and Prim algorithms. *Note how, even when they reach the same end result, they may get to it via wildly different intermediate stages.*



Question 8. In an undirected graph with positive edge weights, let u–v be a minimum-weight edge. Show that u–v belongs to a minimum spanning tree.

Question 9. Use the Ford-Fulkerson algorithm, by hand, to find the maximum flow from s to t in the following graph. How many iterations did you take? What is the largest number of iterations it might take, with unfortunate choice of augmenting path?



Question 10*. Devise an algorithm that takes as input a flow f on a network, and produces as output a decomposition $[(f_1, p_1), \ldots, (f_n, p_n)]$ where each p_i is a path from the source to the sink, and each f_i is a positive number. The decomposition must satisfy $f = \sum_i f_i p_i$, by which we mean "put flow f_i along path p_i, and add together all these flows-along-paths, and the answer must be equal to f". Explain why your algorithm works.

Question 11. The Russian mathematician A.N. Tolstoy introduced the following problem in 1930. Consider a directed graph with edge capacities, representing the rail network. There are three types of vertex: supplies, demands, and ordinary interconnection points. There is a single type of cargo we wish to carry. Each demand vertex v has a requirement $d_v > 0$. Each supply vertex v has a maximum amount it can produce $s_v > 0$. Tolstoy asked: can the demands be met, given the supplies and graph and capacities, and if so then what flow will achieve this?

Explain how to translate Tolstoy's problem into a max-flow problem of the sort we have studied.

Question 12*. In the London tube system (including DLR and Overground), there are occasional signal failures that prevent travel in either direction between a pair of adjacent stations. Find the minimum number of disruptions that will prevent travel between Kings Cross and Embankment. Justify your answer carefully using a max-flow formulation.

Question 13*. Consider a bipartite graph, in which edges go between the left vertex set L and the right vertex set R. A matching is called *complete* if every vertex in L is matched to a vertex in R, and vice versa. For a complete matching to exist, we obviously need $|L| = |R|$. The following result is known as Hall's Theorem:

> A complete matching exists if and only if, for every subset $X \subseteq L$, the set of vertices in R connected to a vertex in X is at least as big as X.

Prove Hall's Theorem, using a max-flow formulation. *[Hint: Use the same construction as we used in lectures, except with capacity ∞ on the edges between L and R. In this graph, some cuts have infinite capacity, and some cuts have finite capacity. If a cut has finite capacity, what can you deduce about its capacity?]*