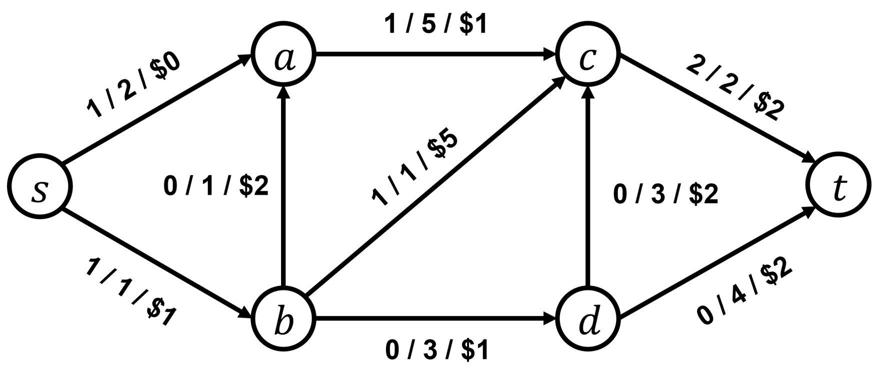
15-451/651 Algorithm Design & Analysis, Spring 2024

Extra Review Problems

Network Flows III

1. **(Short answer / multiple choice)**
   (a) Draw a flow network for which the Edmonds-Karp algorithm is guaranteed to find the maximum flow in just one iteration, but the Ford-Fulkerson algorithm could take an arbitrarily high number of iterations depending on the capacities.

   (b) Consider the following flow network. Edges are labeled with $f/c/\$$, representing the current flow, the capacity, and the cost, respectively.



   i. Draw the residual graph of the flow network with respect to the current flow.

   ii. Is the flow a maximum flow? Justify your answer using the residual graph.

   iii. Is the flow a cost-optimal flow? Justify your answer using the residual graph.

2. **(When you forget Dijkstra)** You have a directed graph G with non-negative edge weights and you want to compute the length of the shortest path from some node u to some node v. Unfortunately, you've forgotten Dijkstra's algorithm and every other shortest path algorithm you learned in 15-210. However, you have some code written up that solves the minimum-cost max flow problem. Describe how to solve your shortest path problem using minimum-cost max flow.

3. **(Transshipment)** In the transshipment problem, you are given a directed graph, where each vertex has a *balance* b, which may be positive or negative. You can think of a vertex with a positive balance (called a *supply* vertex) as having an excess of some commodity, and a vertex with a negative balance (called a *demand* vertex) needing more of that commodity delivered to it. Edges in the graph have unlimited capacity, and a cost $\$(e)$ per unit of commodity send along it. The goal is to ship commodity around the network such that each vertex ends up with a zero balance, at the minimum possible cost. Describe how to solve this problem using minimum-cost maximum flow.

4. **(Finding a spanning tree)** You are given n points $p_1, p_2, \ldots, p_n$ in the 2-D plane. Assume that there is a unique point p_r that has maximum y coordinate. For each point p other than p_r, we're going to select another point q as its "parent", with the condition that the parent of p must have strictly larger y coordinate than p. These parent pointers naturally form a spanning tree, and the cost of the spanning tree is the sum of the Euclidean distances of all the edges in it.

   Suppose that each point can have at most d children, for some d given in the input. Give an algorithm that computes the minimum cost of a valid tree, or determines that no valid tree exists.

5. **(Make the matrix better)** You are given an $n \times n$ matrix M of integers. You want to re-order the columns of the matrix in such a way that the sum of the elements of the diagonal is as large as possible. Describe an algorithm that determines the largest possible diagonal sum.

6. **(Generalized Generalized Matching)** Generate the cheapest $m \times n$ matrix that meets the following specifications

   - Each entry is either 0 or 1
   - Row i sums to a given constant r_i
   - Column j sums to a given constant c_j
   - The cost of a matrix A is $\sum_{(i,j)} d_{ij} \cdot A[i][j]$ for given constants d_{ij}