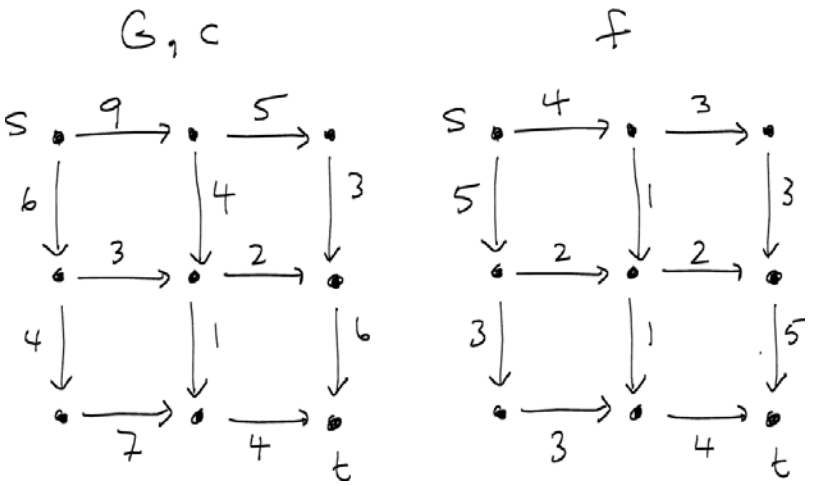
Exercises: Network Flows 1

Questions

1. On the left below is flow network (G,c) with s at the top left corner and t at the lower right. On the right is a flow f. Verify for yourself that the flow satisfies the conservation conditions. Draw the residual graph (G_f, c_f). You do not need to label which edges are forwards or backwards.



2. See the definition of the bottleneck value β of an augmenting path P. Show that when you add the bottleneck value β to the flow on each edge of the path P, you indeed respect the capacity constraints of the original graph G. This may seem so obvious that there is nothing to prove. Its one of the those cases where the proof is just required to make sure you understand all the definitions.
Hint: You should consider the cases of forward and backwards edge separately.

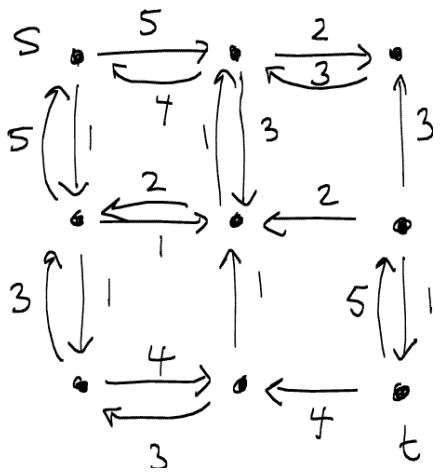
By the way, I could also ask you to show that the augmented flow (i.e. the original flow + the flow added on the augmenting path) respects the conservation constraints. But I won't. This one turns out to be tedious to prove since there are several cases to consider, having to do with combinations of forward or backward edges "arriving at" or "leaving" each vertex.

3. Prove that f^out(s) = f^in(t), that is, the value of the flow could equivalently be defined by the flow leaving the source or arriving at the terminal. Again, this fact is no surprise at all, given the physical interpretation of the flow network problem. We just prove it to make sure we understand the formulation of that problem.

4. Suppose the capacities in a network flow are not integers. How would this change the O( ) runtime of the Ford Fulkerson algorithm? Does the algorithm terminate?

Answers

1.



2. Since β was the smallest capacity of the edges, we know $0 < \beta <= f(e)$. So, first consider a forward edge in the residual graph. The capacity of this edge is c_f(e) = c(e) – f(e) and it follows that $0 < f(e) + \beta <= c(e)$ which is indeed satisfies the capacity constraint. Second, consider a backwards edge. The capacity of this edge is c_f(e) = f(e). We modify the flow f but subtracting β from f(e), and we know $c(e) > f(e) - \beta >= 0$ which indeed satisfies the capacity constraint.

3. Use the conservation constraint that, for a fixed vertex v,

   sum_{ (u, v) in E } c(u, v) = sum_{ (v, w) in E } c(v, w).

   What happens when we add together these conservation constraint equations for all vertices v in V \ {s, t}. For each edge joining two vertices in V \ {s,t}, there will be two terms in the sum, one when the edge is incoming (into v) and one when the edge is outgoing (from v), one of each side of the equals sign. The only remaining terms in the sum will be those edges out of s (on the left side) and those into t (on the right side), which is what we wanted to prove.

4. The stated runtime of Ford-Fulkerson is O(C m) as explained in the lecture, where C is an integer namely the sum of integer capacities of edges out of s. If the capacities are not integers, then this O( ) no longer makes sense because O( ) always assumes we are talking about integers i.e. number of operations. More concretely, if we applied the Ford Fulkerson algorithm to a flow network having non-integer capacities, we would still find that the flow increases in every pass through the main loop. However, the amount by which the flow increases might be a very small number and this number might get smaller and smaller indefinitely. There is no reason why the while loop would ever terminate. (Here I am ignoring the precision limits of representing floating point numbers on real computers, which you will learn about in COMP 273 or ECSE 221, if you haven't already.)