Solutions to Practice Midterm

K. Subramani
Department of Computer Science and Electrical Engineering,
West Virginia University,
Morgantown, WV
ksmani@csee.wvu.edu

1. Minimum Spanning Trees

   (a) Problem 2.7 - The following algorithm solves the problem: The algorithm definitely produces a connected subgraph. Let us call it $\mathbf{G_A}$ and denote its weight by $w(\mathbf{G_A})$. Let $\mathbf{G_B}$ denote the optimal weight connector with weight $w(\mathbf{G_B})$. Observe that $\mathbf{G_B}$ *must* have all the negative weight edges which are present in $\mathbf{G}$ and hence $\mathbf{G_A}$. Otherwise we can decrease the weight of $\mathbf{G_B}$ by adding such an edge. So the only way that $w(\mathbf{G_B})$ can be less than $w(\mathbf{G_A})$ is if $\mathbf{G_A}$ contains a postive weight edge not in $\mathbf{G_B}$. However the removal of such an edge from $\mathbf{G}$ disconnects it and thus $\mathbf{G_B}$ cannot be connected...[Q.E.D.]

Algorithm 1 Minimum Connector with negative edge weights

Function MINIMUM CONNECTOR ($\mathbf{G} =< \mathbf{V}, \mathbf{E} >$)

```
{The key idea is that we only throw out the positive weighted edges, since throwing out edges with negative
 weight increases the cost of the minimum-connector!}
for ( i = 1 to m) do
  if ( w(e_i) > 0) then
    if ( {G − {e_i}} is connected ) then
      G = G − {e_i}
    end if
  end if
end for
```

   (b) Problem 2.8 - Let $|c_e|$ denote the weight of the largest negative weight (in magnitude) edge in the given graph $\mathbf{G}$ [1] Add $|c_e| + 1$ to the weight of every edge in the graph. Let us call the new graph $\mathbf{G'}$.

   Claim: 0.1 *The edges corresponding to an MST in* $\mathbf{G'}$ *also correspond to and MST in* $\mathbf{G}$.

   Proof: *Identical to the greedy proof of Kruskal's algorithm!* □

   (c) Consider the heaviest edge e' in the MST T of a graph $\mathbf{G}$. Removal of this edge, creates a partition (cut) of the vertex set $V \setminus S$ and S. Now from the Blue-Rule, we know that the lightest weight edge of *any cut* must be part of the MST; thus e' is the lightest weight edge crossing S. Consequently, if the minmax spanning tree solution does not include e', then either it is not connected or the maximum weight of an edge in it has weight greater than the weight of e'. For the second part, consider a triangle with weights $2, 2, 1$. The MST solution is $\{1, 2\}$, whereas a valid solution to the minmax spanning tree solution is $\{2, 2\}$, which is clearly not an MST.

[1] In other words c_e is the smallest weight of any edge in the graph.

2. Shortest Path Trees

   (a) Problem 2.23 - We need to consider the following 3 cases:

   - Both arcs incident on w are directed out of w. Clearly, $d(w) = \infty$ and we can work on the subgraph $\mathbf{G} - w$;
   - Both arcs are directed into w. Solve the shortest path problem on $\mathbf{G} - w$. Let a and b be the 2 vertices which are connected to w. Then $d(w) = \min\{d(a) + w_{aw}, d(b) + w_{bw}\}$;
   - One arc (say e_1) enters w, while the other arc (say e_2) leaves w. First delete e_2 and solve the shortest path problem on $\mathbf{G} - e_2$. Clearly, we get the correct value of $d(w)$. Now delete e_2 and solve the shortest path problem on $\mathbf{G} - e_1$. We now get the correct distances to all the other vertices in the graph.

   The key point is that we solved the shortest path problem on a subgraph of $\mathbf{G}$ in all three cases.

   (b) Problem 2.27 - Proved in class!

3. Max flows and Min-cuts

   (a) Problem 3.3 - It is clear that if there is a path P from r to s in $\mathbf{G}$ such that the capacity of every edge is ∞, then there cannot be a maximum flow. We now need to argue that the converse, i.e. if for every path P between r and s, there is at least one graph of finite capacity, then there must exist a finite maximum flow. To see this, let us use the augmentation path algorithm on $\mathbf{G}$. Observe that in each iteration one path is lost and the amount of flow is incremented by the smallest residual capacity of any edge on that path (which is finite). Since the number of paths is finite and in each iteration, the flow increases by at most a finite amount, the claim follows.

   (b) Problem 3.7 - Without loss of generality, we assume that the discussion is about (r, s) mincuts. Let us restate Corollary 3.8 from the book:

   Corollary: 0.1 *Let $\vec{x}$ denote a feasible flow and $\delta(R)$ denote an (r, s) cut. Then $\vec{x}$ is a maximum flow and $\delta(R)$ is a min-cut if and only if:*
   $x_e = u_e, \forall e \in \delta(R)$ and $x_e = 0, \forall e \in \delta(\bar{R})$

   In other words, a maximum flow *saturates* every (r, s) minimum cut. It follows that under any maximum flow every edge $e \in \delta(R_1)$ and every edge $e \in \delta(R_2)$ is saturated.

   - Consider any edge $e \in \delta(R_1 \cup R_2)$. Clearly $e \in \delta(R_1)$ or $e \in deta(R_2)$ or both. It is possible that not all edges in $\{\delta(R_1) \cup \delta(R_2)\}$ are in $\delta(R_1 \cup R_2)$, but that is not our concern. It therefore follows that every edge $e \in \delta(R_1 \cup R_2)$ is saturated under any flow and hence $\delta(R_1 \cup R_2)$ is a min-cut. ...[Q.E.D.]
   - Consider any edge $e \in \delta(R_1 \cup R_2)$. e has its tail in both R_1 and R_2; it must be one of the following types: (a) The edge has its head in R_1; clearly it must belong to $\delta(R_2)$; (b)The edge has its head in R_2; clearly it must belong to $\delta(R_1$; (c) the edge has its head in $V \backslash \{R_1 \cup R_2\}$; in this case $e \in \delta(R_1) \cap \delta(R_2)$.
     In all three cases, e is saturated. It follows that the cut defined by $\delta(R_1 \cap R_2)$ is saturated by any flow and is hence a min-cut.

   (c) Problem 3.8 - Form an (r, s) min-cut corresponding to maximum flow $\vec{x^1}$; let the cut be represented by $\delta(R_1)$ for suitably chosen R_1. From the hypothesis $v \in R_1$, since there is an (r, v) path in $G(\vec{x^1})$. Similarly form an (r, s) min-cut corresponding to maximum flow $\vec{x^2}$; let the cut be represented by $\delta(R_2)$, for suitable chosen R_2. From the previous problem, we know that $\vec{x^2}$ can be a maximum flow if and only if it saturate $\delta(R_1 \cup R_2)$; it follows that even under flow $\vec{x^2}$, v must be reachable from r in $G(\vec{x^2})$; i.e. there is an $\vec{x^2}$-incrementing path (r, v).

   (d) Problem 3.11 - If there is no $\vec{x}$-augmenting path of width ≥ 0, it means that there is no path from r to s in $G(\vec{x})$ and it follows that $\vec{x}$ is a maximum flow. Let us now consider the case when the maximum width over any $\vec{x}$-augmenting is K. Draw the residual graph $G(\vec{x})$; also associate the residual capacity with each edge. If the maximum flow in $G(\vec{x})$ is $\vec{y}$, then the maximum flow in the original graph is $\vec{x} + \vec{y}$ [2] Now in the residual graph the maximum width is K; which means that the smallest weight min-cut $\geq K.m$, since a min-cut could have all m edges. It follows that a max-flow in $G(\vec{x})$ cannot be larger than $K.m$, which proves the claim.

[2] Use a few examples to convince yourself that this is true!