Name: Harshit Jain
Access ID: hmj5262
Recitation: 8

Problem 0

Points:

Acknowledgements

(a) I did not work in a group.

(b) I did not consult without anyone my group members.

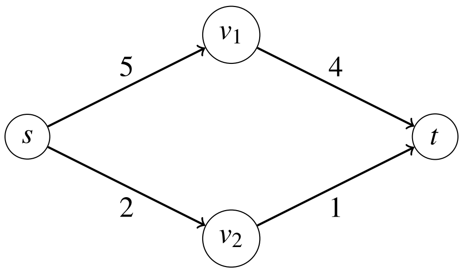
(c) I did not consult any non-class materials.

Problem 1

Points:

The given statement is **False**. If f is a maximum $s-t$ flow in G, then f need not to saturate every edge out of s with flow.

Counter Example:



Here, the maximum flow on the upper branch will be 4 since the bottleneck capacity for the path $s \Rightarrow v_1 \Rightarrow t$ is 4. The maximum flow on the lower branch will be 1 since the bottleneck capacity for the path $s \Rightarrow v_2 \Rightarrow t$ is 1.

Clearly, both the edges out of s have flow value $f(e) < c_e$ where c_e is the capacity of the edges coming out of node s. Therefore,

$$v(f) = \sum_{e \text{ out of } s} f(e) < \sum_{e \text{ out of } s} c_e = C$$

Problem 2

Points:

Create a flow network graph. Firstly, take the vertices of a graph to be the set of patients (p_i where $i \in [1,n]$) and hospitals (h_j where $j \in [1,k]$). For each patient, we add a directed edge (p_i, h_j) from that patient to each hospital to which she/he can be evacuated. Then, to turn this directed graph into a flow network, we add a source vertex s and connect s to each patient node. And we add a sink vertex t and connect each hospital to t.

We can observe that all patients have an edge to all hospitals. Having created this graph, we can simply apply the Ford-Fulkerson algorithm to find the maximum possible flow from s to t. We know that if the maximum flow from s to t is n, then every single person can be brought to a hospital.

Runtime: To find the runtime, we must determine the runtime of the Ford-Fulkerson algorithm. We know that the runtime of the algorithm is $O(C \cdot |E|)$, where C is the sum of capacities coming out of source node, which is n in this case. Since there is an edge from every patient to every hospital, $|E| = n \cdot k$. We see that initialization takes $\theta(n \cdot k + n + k)$ for creating all the edges of the graph. So $\theta(n \cdot k + n + k) + O(n \cdot n \cdot k) = O(n^2 k)$, which is polynomial time.

Problem 3

Points:

Proof: The functions f_1 and f_2 are flows, so it follows that $f_1(u,v) <= c(u,v)$ and $f_2(u,v) <= c(u,v)\ \forall\ u,v$.

Now,

$$\Rightarrow (\alpha f_1 + (1-\alpha) f_2)(u,v) = (\alpha f_1)(u,v) + ((1-\alpha) f_2)(u,v)$$

$$\Rightarrow (\alpha f_1 + (1-\alpha) f_2)(u,v) = \alpha * f_1(u,v) + (1-\alpha) * f_2(u,v)$$

$$\Rightarrow (\alpha f_1 + (1-\alpha) f_2)(u,v) <= \alpha * c(u,v) + (1-\alpha) * c(u,v)$$

$$\Rightarrow (\alpha f_1 + (1-\alpha) f_2)(u,v) <= 1 * c(u,v) = \mathbf{c(u,v)}$$

Therefore, $(\alpha f_1 + (1-\alpha) f_2)(u,v) <= c(u,v)$ which means $(\alpha f_1 + (1-\alpha) f_2)(u,v)$ is a flow $\forall\ \alpha$ in the range $0 \leq \alpha \leq 1$.