Rishad Rahman (Deepak's Recitation) April 9, 2015

6.046 Problem 7-1

Collaborators: *None*

Assume for the problem that u is reachable from s and t is reachable from v otherwise nothing happens (we can check this condition in $O(V+E)$ time using BFS).

(a) The flow network after the weight change is $F' = (G, c')$.

1. Follows from 2.
2. Before the modification, consider the min-cut of F into S and T i.e. $c(S,T)$. Suppose $(u,v) \notin S \times T$. We look at the $S-T$ cut in the new flow network: $|f'| = f'(S,T) \leq c'(S,T) = c(S,T) = |f|$. Now suppose wlog $u \in S$ and $v \in T$, then $|f'| = f'(S,T) \leq c'(S,T) = c(S,T) + k = |f| + k$.
3. Follows from 4.
4. Consider the min-cut of F' into S and T. Suppose $(u,v) \notin S \times T$ then $|f'| = c'(S',T') = c(S,T) \geq |f|$. Now suppose wlog $u \in S$ and $v \in T$, then $|f'| = f'(S,T) = c'(S,T) = c(S,T) - k \geq |f| - k$.

(b) **Algorithm**: Run Edmonds-Karp on F' using f as the initial flow until we have found the new max flow.

Correctness: Since Edmonds-Karp cannot do anything but increase the flow given a valid flow, we are safe as long as f is still valid after $r \leftarrow c(u,v)$. However this is obviously true since $f(u,v) \leq c(u,v) < r$.

Runtime: On each iteration, Edmonds-Karp takes $\Theta(V+E)$ time because BFS, and the number if iterations is bounded by k since we cannot increase the flow by more than k hence the runtime is $O(k \cdot (V+E))$.

(c) **Algorithm**: If $f(u,v) \leq r$ do nothing (max flow is the same). Otherwise before the capacity update, we pre-modify F with the following until $f(u,v) = r$:

- $f(u,v) \leftarrow f(u,v) - 1$
- DFS backwards from u to s by selecting edges with positive flow and decrease the flow of edges on the found path 1.
- DFS forwards from v to t by selecting edges with positive flow and decrease the flow of edges on the found path 1.

Then run Edmonds-Karp on this new flow network until termination.

Correctness: If $f(u, v) \leq r$ then we still have a valid flow after the update and we cannot have a bigger one otherwise f' would be a valid flow on F such that $|f'| > |f|$. This along with (a) tells us $|f| - k \leq |f'| \leq |f|$. Decreasing $f(u, v)$ until it equals r guarantees that there will be no overflow once we update $c(u, v)$ to r so we have to make sure that on each iteration in the pre-modification we have a valid flow network. Obviously each node on the found DFS path (except s or t) preserves 0 total flow since we decrease both an outgoing flow and incoming flow by 1 and since we're decreasing flow we can't have overflow. Now we just have to argue that we will always have a positive flow path from s to u and v to t on each iteration otherwise DFS won't work as expected. However this is guaranteed by the fact that there is still a positive flow $f(u, v)$ so it has to be connected to a source and a sink. Therefore we always maintain a flow network on each iteration and running Edmonds-Karp after the capacity will give expected results.

Runtime: On each iteration of the pre-modification we decrease $f(s, V)$ by 1 hence our initial flow is no less than $|f| - k$ but since we are bounded by $|f|$, Edmonds-Karp will not take more than $O(k \cdot (V + E))$ time. Since the pre-modification is $O(k)$ DFS's we have the total is runtime is $O(k \cdot (V + E))$.

Rishad Rahman (Deepak's Recitation)	April 9, 2015
6.046 Problem 7-2	
Collaborators: *None*	

Algorithm: Create a flow network, F, with a new vertex s as the source, connected to all of the k companies. Give every edge a capacity of 1. If the max-flow $|f| = k$ then use the flow from c_n as the path for c_n, otherwise it is impossible.

Correctness: Suppose we have that k disjoint paths are possible. Give all edges in these paths a flow of 1 in F along with (s, c_n) for all n. Obviously there is no overflow so we just have to check that the net flow for each vertex that's not s nor t is 0. This is true because for each of those vertices, if a truck comes, a flow of -1, in then it must also go out, a flow of $+1$, Multiple trucks use different edges to come in and go out hence we have a net flow of 0. Therefore this is a legitimate flow but we use all the edges coming out of s hence the max-flow $|f| = k$ which our algorithm detects as expected. Suppose on the other hand the algorithm finds a max-flow $|f| = k$. Then we must have all the nodes c_n have an outgoing flow of 1 unit, i.e. a truck, which reaches t without going over the edge capacity of 1 therefore each truck can indeed reach the destination without occupying the same road.

Runtime: The most significant work comes from running Edmond-Karps. The flow network has $V + 1$ vertices and $E + k$ edges with $k \leq E$ so our runtime is $O(VE^2)$

Rishad Rahman (Deepak's Recitation) April 9, 2015

6.046 Problem 7-3

Collaborators: *None*

Algorithm: We create a flow network with intermediate vertices $a_1, a_2 ..., a_n, b_1, ba_2 ..., b_m$. We have the source, s, is connected to all a_i through edges with capacity 1. Then for each a_i, and $b_j \in A_i$, put an edge from a_i to b_j with capacity 1. Then for each b_j we connect b_j to sink t with capacity q_j. Run a max-flow algorithm. The a_i with a flow going into them are the customers who get their choices and the flow out of a_i points towards which choice that customer got. All the other customers get vouchers.

Correctness: Treat the units of flow as customers. Flow from s to a_i means a_i got an order i.e. did not get a voucher. Flow from a_i to b_j means a_i got order b_j. As a result, flow from b_j to t means the number of times b_j was given. As a result of the initialized capacities, this flow network is equivalent to the problem conditions and minimizing the number of vouchers is the same as maximizing the number of a_i which get orders i.e. the flow out of s which is what our algorithm does.

Runtime: We have $m + n + 2$ vertices and $m + n + \sum A_i = O(mn)$ edges. Therefore the runtime using Edmond-Karp is $O((m + n)m^2n^2)$.