CIS 502 - Algorithms

Fall 2015 Homework 5 Solutions

Problem 1 *Question 10, pg 419–420.*

Solution: We are given the f_e values. Suppose the flow values are integral. Observe that if $f_{e^*} \leq c_{e^*} - 1$ then there is nothing to do. The maxflow cannot increase (because the mincut cannot increase) when we decrease capacities.

Suppose $f_{e^*} = c_{e^*}$. Let $e^* = (u, v)$. We now need to decrease the flow from s to u by 1 unit, decrease the flow from v to t by 1 unit and the check if we can send another unit of flow from s to t. The first two steps create a feasible flow, and the last step verifies if the maxflow has decreased or stayed the same. To decrease the flow from s to u we perform a DFS/BFS starting from s in the set of edges $\{e | f_e > 0\}$, once find a path then we decrease the flow along that path by 1. Likewise from v to t. The final step requires $O(m + n)$ time as well to check if there is an s–t path.

Non Integral flow values: we can produce a flow which is feasible quite easily. But there is no easy way that we can send one extra unit of flow given fractional flow values (if we knew how to do that fast, then we can solve the maxflow problem using that algorithm). The acyclicity of the flow graph does not imply integral flow. It is likely that the authors did not intend the 10^{th} question in the chapter to be this hard.

Problem 2 *Question 12, pg 420.*

Solution: We first compute the maximum flow using the Ford-Fulkerson algorithm in time $O(m^2)$ (since the maximum flow is m, based on each $c_e = 1$). Now we compute the residual graph and find the min-cut. (You can directly state that using $O(m^2)$ time max-flow algorithm you would find the min-cut.) Let this cut be C. We delete k edges from this cut. In the new graph the minimum cut now has size $|C| - k$. Hence the max-flow has decreased by k.

Proof of optimality: Suppose not. Suppose by deleting k other edges, we achieved a max-flow value $f < |C| - k$. Let the corresponding min-cut (after deleting those edges) be C' where $|C'| = f$. Then C' corresponds to a cut C'' in the original graph and $|C''| \leq |C'| + k$. Which implies that $|C''| \leq f + k < |C| - k + k = |C|$. Thus this cut C'' corresponds to a better min-cut in the original graph which contradicts the minimality of C.

Problem 3 *Question 14, pg 421.*

Solution: For part (a) we construct a graph G' where we add a new source vertex s and add directed edges (s, v) for all $v \in X$. We add a new sink vertex t and add edges (u, t) for all $u \in S$. All edges are given capacity 1. We then compute the maxflow from s to t using the Ford–Fulkerson algorithm (integral flow) and see if the maxflow is $|X|$, using time $O(|X|(m + n))$. We can now decompose the flow into flow paths in time $O(|X|(m + n))$. These flow paths provide the escape routes.

Proof of correctness: If there exists escape routes then max flow in G' is $|X|$ because we can send one unit flow from s to every $v \in X$ and then follow the escape routes. Since the routes are edge disjoint we will not violate the capacity constraint anywhere. If the maxflow is $|X|$ then we can decompose the flow into $|X|$ edge disjoint paths. Therefore we will have escape routes for each vertex in X such that the routes do not share edges.

Part b. We now perform the following transformation: we split every vertex v in G' except s, t into 2 pieces v_{in}, v_{out}. All edges (u, v) in G' become (u_{out}, v_{in}). We now add the edges (v_{in}, v_{out}) for every v. Once again every edge has capacity 1. Let this graph be G''. We compute the maxflow and the maxflow provides us the escape routes.

Proof of correctness: If there exists node disjoint escape routes then max flow in G'' is $|X|$ because we can send one unit flow from s to every $v \in X$ and then follow the escape routes. Since the paths are node disjoint we will not violate capacities in G''. Likewise if we find $|X|$ edge disjoint paths in G'' then they must correspond to node disjoint paths in G' (because there is exactly one way to traverse any node v, using the v_{in} to v_{out} edge).

Problem 4 *Problem 17, page 423–424.*

Solution: Observe that if the mincut in the graph was unique then we already knew which edges were cut! So there are multiple min-cuts and the goal is to find which mincut has been destroyed.

We first run Ford-Fulkerson to find a maxflow of k; we then perform a flow decomposition and get k edge disjoint paths from s to t. Each of these steps require $O(mk)$ time.

Now every cut contains one of the edges from each of these paths – otherwise it is not separating s and t. Therefore every mincut has exactly one edge from each of the paths – otherwise that cut will have more than k edges and will not be a mincut!

Therefore the adversary has destroyed exactly one edge on each of the paths we found. We can now perform *binary search* to find the specific edge which has been cut. Since the paths are of length at most n we need $O(\log n)$ pings. Therefore in total we need $O(k \log n)$ pings.