6.854
ADVANCED ALGORITHMS
PROBLEM SET 4

JOHN WANG

Collaborators:

Problem 1-a: Consider a second-level table, that in the static case uses $f(s) = O(s^2)$ space to store s items. We gave a construction of such a table that succeeds with probability $1/2$. Suppose that we take the following approach to insertions: if inserting the sth item violates perfection of the table, repeatedly build a table of size $f(2s)$ until you get a perfect one. Show that when s items are inserted (one at a time, with rebuilds as necessary) the expected total cost of all the rebuilds is $O(s)$.

Solution:
□

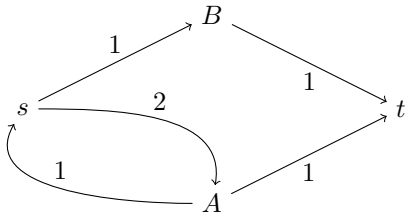
6.854 Advanced Algorithms

Problem Set 2

John Wang
Collaborators:

Problem 2-a: In any maximum flow, and for all vertices v and w, either the raw flow from v to w or the raw flow from w to v is 0.

Solution: False. Consider the following graph.



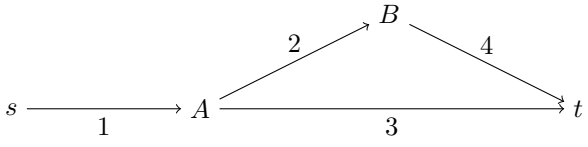
It is clear that the maximum flow is 2, where flow comes into t from nodes A and B, each with a flow of 1. However, we can see that there is non-zero raw flow from s to B and from B to s. Thus, there is a maximum flow where there exist two vertices v and w which break the above conjecture. □

Problem 2-b: There always exists a maximum flow such that, for all vertices v and w, either the raw flow from v to w or the raw flow from w to v is zero.

Solution: True. We will show this by contradiction. Suppose this is not the case. Then there is raw flow such that on some vertices v and w, the raw flow from v to w and from w to v are nonzero for all flows which are maximal. This means that any flow with zero raw flow in each direction is not a maximum flow (of flow f), so that this new flow would be less than f. Now, consider what happens when we set the raw flow from w to v to be zero, and the raw flow from v to w to be $rf(v,w) - rf(w,v)$ where rf denotes raw flow. In this way, we have created a graph G' which has the same amount of flow as before towards the sink (if not, the switch v and w) as in the original graph. This means that the new flow is also a maximum flow. However, we know that the flow on the vertices of the rest of the graph are unchanged, and we also assumed that there does not exist any maximum flow with zero raw flow between any vertices v and w. This is a contradiction. □

Problem 2-c: If all directed edges in a network have distinct capacities, then there is a unique maximum flow.

Solution: False. Consider the following counterexample.



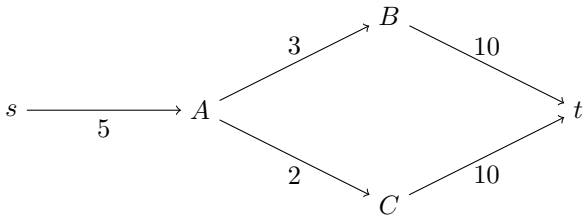
In the graph above, the maximum possible flow is $F = 1$. However, there are two possible ways that this flow can be achieved. In the first case, a flow of capacity 1 can move along the path S, A, t. In the second case, a flow of capacity 1 can move along the path S, A, B, t. □

Problem 2-d: If we replace each directed edge in a network with two directed edges in opposite directions with the same capacity and connecting the same vertices, then the value of the maximum flow remains unchanged.

Solution: False. Consider the simple graph with two nodes s and t. The only edge in this graph is a directed edge from t to s of capacity v. Clearly, the maximum flow from s to t is zero, because there is no way to reach t. However, if we augment the graph with two edges in opposite directions, both of capacity v, then the maximum flow becomes v. □

Problem 2-e: If we add the same positive number λ to the capacity of every directed edge, then the minimum cut (but not its value) remains unchanged.

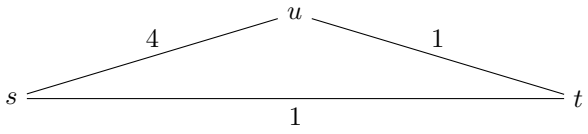
Solution: False. Consider the graph below.



We see that on the original graph, the unique $s-t$ cut is given by $S = \{s, A, B, C\}$ and $T = \{t\}$. However, when $\lambda = 1$ is added to all of the edge capacities, we see that the new unique $s - t$ cut is given by $S = \{s\}$ and $T = \{A, B, C, t\}$, since the bottleneck appears at A when its capacity is only increased by 1, and the capacities of both outgoing edges are both increased by 1, leading to a total increase of 2. □

Problem 2-f: Flow is transitive: if in a graph there is a flow of value v from s to t and there is a flow of value v from t to u, then there is a flow of value v from s to u.

Solution: False. Consider the following graph:



From s to t, there is a maximum flow of 1. From t to u there is a maximum flow of 1. However, from s to u, there is a maximum flow of 5. Thus, it is clear that the flow from s to u is not equal to the flow from s to t then from t to u, since there are other paths for the flow to travel on. □

6.854 Advanced Algorithms

Problem Set 2

John Wang
Collaborators:

Problem 5-a: Argue that the running time of Dial's algorithm is $O(m + D)$ where D is the maximum distance, and that the algorithm still works if some edges have length 0.

Solution: First we will initialize an array storing the keys from 1 to nC. We know that this can be done in $O(1)$ time. Next, we will perform Dijkstra's algorithm by successively finding the minimum element in the queue, and expanding its neighbors, and decreasing the key by storing the reduced edge lengths $l^d_{vw} = l_{vw} + d_v - d_w$. The total reduced edge length along a path from the source to some node t is $\sum_{v,w} l^d_{vw}$ where v and w are intermediate nodes on the path. Since this sum telescopes, we see that the total reduced edge length is simply d_t. This implies that removing the lowest reduced edge length will remove the current shortest path to some node and will not affect the correctness of Dijkstra's.

This means that we can perform n inserts, n delete-mins, and m decrease-keys. We know the insert and decrease key operations cost $O(1)$ each. The delete-min operations cost $O(k)$ where k is the distance between the last minimum and the next minimum (since we have to crawl our way up k buckets until we find the minimum). This means that the total amount of work performed is just $\sum_{i=1}^{n} k_i$ where k_i is the distance between the minimums at the $i-1$st and ith steps. This comes out to the maximum distance, which is D. Thus total work is just $O(m + D)$.

The algorithm clearly still works for edge length of zero because we can initialize a bucket for edges of length 0, which will then always be the minimum in the set, so they will be picked first. □

Problem 5-b: Show that the reduced edge lengths defined above are all nonnegative integers.

Solution: We need to show that $l^d_{vw} = l_{vw} + d_v - d_w$ are all nonnegative. We know that d_v, d_w and l_{vw} are all nonnegative. Therefore, all we need to do is to show that $l_{vw} \geq d_v - d_w$. Let $\{s, v_1, v_2, \ldots, v\}$ be the shortest path from s to v with distance d_v. We know that $d_v + l_{vw} \leq d_w$ since the shortest path to w from s puts a lower bound on the distance of any path to w from s. This expression rearranges to $l_{vw} \geq d_v - d_w$ which is what we wanted to show. □

Problem 5-c: Show that the shortest paths under the reduced length function are the same as those under the original length function. What is the length of shortest paths under l^d?

Solution: Suppose by contradiction that under the reduced length function, there is some path P' from s to u which is shorter than P, where P is the shortest path under the original length function. Let $P = \langle s = v_1, v_2, v_3, \ldots, u = v_n \rangle$ of length n and $P' = \langle s = v'_1, v'_2, v'_3, \ldots, u = v'_{n'} \rangle$ of length n'. If P' is shorter than P under the reduced length function, then the following is true:

$$\sum_{i=1}^{n'} l^d_{v'_{i-1}v'_i} < \sum_{i=1}^{n} l^d_{v_{i-1}v_i} \tag{1}$$

$$\sum_{i=1}^{n'} l_{v'_{i-1}v'_i} - d_{v'_i} + d_{v'_{i-1}} < \sum_{i=1}^{n} l_{v_{i-1}v_i} - d_{v_i} + d_{v_{i-1}} \tag{2}$$

$$-d_u + \sum_{i=1}^{n'} l_{v'_{i-1}v'_i} < -d_u + \sum_{i=1}^{n} l_{v_{i-1}v_i} \tag{3}$$

$$\sum_{i=1}^{n'} l_{v'_{i-1}v'_i} < \sum_{i=1}^{n} l_{v_{i-1}v_i} \tag{4}$$

This occurs because $d_{v_i} - d_{v_{i-1}}$ leads to a telescoping sum. However, this means that path P' has a shorter length than P under the original length function, which is a contradiction of the fact that P was the shortest path under the original length function.

The length of the shortest paths under l^d is therefore $-d_u + \sum_{i=1}^{n} l_{v_{i-1}v_i}$ where d_u is the distance of the shortest path under the original length function. Thus, the shortest path under the reduced length function is just 0. □

Problem 5-d: Devise a scaling algorithm for shortest paths. Use the reduced edge lengths and Dial's bucketing algorithm to keep the task of shifting in a new bit sample.

Solution: □