Anthony Rusignuolo

CS 600WS – Advanced Algorithms

Reza Peyrovian

Homework 7

I pledge my honor that I have abided by the Stevens Honor System.

1. C-15.2 Suppose G is a weighted, connected, undirected, simple graph and e is a largest-weight edge in G. Prove or disprove the claim that there is no minimum spanning tree of G that contains e.
   1. There is no minimum spanning tree that can be created using the largest weighted edge, e, in the graph. This is proven by contradiction, for if the tree were created with e and excluding a lesser weighted edge, le, the total weight would obviously be more that the tree created under opposite circumstances, and would therefore not be the minimum spanning tree of that graph.
2. A-15.2 Suppose you are given a diagram of a telephone network, which is a graph G whose vertices represent switching centers, and whose edges represent communication lines between two centers. The edges are marked by their bandwidth, that is, the maximum speed, in bits per second, that information can be transmitted along that communication line. The bandwidth of a path in G is the bandwidth of its lowest-bandwidth edge. Give an algorithm that, given a diagram and two switching centers, a and b, will output the maximum bandwidth of a path between a and b. What is the running time of your algorithm?
   1. Running a modified version of Dijkstra’s Algorithm should return the highest bandwidth path. This algorithm should take in the graph, a, and b. It starts by initializing all bandwidths, to be infinity for the starting vertex and 0 for the remaining vertices. Then, implement a priority queue, Q, that allows for max removals instead of min removals and append all vertices with their bandwidths as keys to the queue. Then, while Q is not empty, remove the max value, u, and either terminates if u is equal to b or iterate through its adjacent vertices if not. In that iteration, the bandwidth of the adjacent vertex, z, is compared to the minimum of the bandwidth of u or the bandwidth between u and z. If the value of the minimum operation is greater than the bandwidth of z, z’s bandwidth is updated in the priority queue and the loop continues until it reaches b. This should run in O((n+m)logn) time.
3. A-15.6 Suppose you have n rooms that you would like to connect in a communication network in one of the dormitories of Flash University. You have modeled the problem using a connected, undirected graph, G, where each of the n vertices in G is a room and each of the m edges in G is a possible connection that you can form by running a cable between the rooms represented by the end vertices of that edge. In this case, however, there are only two kinds of cables that you may possibly use, a 12-foot cable, which costs $10 and is sufﬁcient to connect some pairs of rooms, and a 50-foot cable, which costs $30 and can be used to connect pairs of rooms that are farther apart. Describe an algorithm for ﬁnding a minimum-cost spanning tree for G in O(n + m) time.
   1. Run the Prim Jarnk Algorithm with two doubly linked lists for priority queues to reduce insertion, updates, and removeMin functions down to O(1), the total running time would be O(n+m) instead of O((n+m)logn).
4. R-16.2 Answer the following questions on the ﬂow network N and ﬂow f shown in  
   Figure 16.6a:  
   1. What are the forward and backward edges of augmenting path π?  
   2. How many augmenting paths are there with respect to ﬂow f? For each such path, list the sequence of vertices of the path and the residual capacity of the path.  
   3. What is the value of a maximum ﬂow in N?
   1. 1. Forward edges: (s, v2), (v2, v3), (v1, v4), (v4, t)  
      Backward edges: (v1, v3)  
        
      2. There are 6 paths  
      Path 1: s, v1, v3, v4, t. Δf(Path 1) = 3  
      Path 2: s, v1, v4, t. Δf(Path 2) = 2  
      Path 3: s, v2, v3, v1, v4, t. Δf(Path 3) = 2  
      Path 4: s, v2, v3, v4, t. Δf(Path 4) = 3  
      Path 5: s, v3, v1, v4, t. Δf(Path 5) = 1  
      Path 6: s, v3, v4, t. Δf(Path 6) = 1  
        
      3. 15
5. C-16.7 Give an algorithm that determines, in O(n + m) time, whether a graph with n   
   vertices and m edges is bipartite.
   1. Bipartite graphs, by definition, do not have edges connecting a node to the same set the node belongs to. Therefore, if the graph is bipartite and Bread First Search is run on all nodes, no node should ever share an edge with another node with the same type of level, type being odd or even. For example, if BFS is run and a given node from level 1 shares an edge with a node in level 3, or level 5, or level 7, etc., that graph is not bipartite because node 1 shares an edge with a node that is supposed to be in its own set. A simpler way to think of this is BFS without levels, but with two sets: odd and even, which correspond to X and Y for a bipartite graph. If any odd or even nodes link to odd or even nodes respectively, the graph is not bipartite. BFS is O(n+m).
6. A-16.2 The city of Irvine, California, allows for residents to own a maximum of three dogs per household without a breeder’s license. Imagine you are running an online pet adoption website for the city, as in the previous exercise, but now for n Irvine residents and m puppies. Describe an efﬁcient algorithm for assigning puppies to residents that provides for the maximum number of puppy adoptions possible while satisfying the constraints that each resident will only adopt puppies that he or she likes and that no resident can adopt more than three puppies.
   1. Assuming that residents are in set X and puppies are in set Y of a bipartite graph. A flow network can be created by setting up a source vertex, s, a sink vertex, t, edges with weight of 3 from s to all nodes in X, edges with weight one from all nodes in Y to t, and a resident’s desire to adopt a puppy as the edges connecting the two sets from X to Y. This step takes O(n+m) time. Then, run the Ford-Fulkerson Algorithm to find the max flow, which equals the maximum matching. This step runs in O(nm) time.