CIS 675 Homework 6

Design and Analysis of Algorithms

Saket Kiran Thombre

SU ID: 899913802

NetID: sthombre

Email: sthombre@syr.edu

Collaborators for Q1, Q2:

1. Saurav Shashank Narvekar

   NetID: sanarvek

2. Chandan Pothumarthi

NetID: cpothuma

Problem 1 Populist spanning tree (Point 5)

Let G = (V, E) be a connected graph with distinct positive edge costs, and let T be a spanning tree of G (not necessarily the minimum spanning tree). The elite edge of T is the edge with the greatest cost. A spanning tree is said to be populist if there is no other spanning tree with a less costly elite edge. In other words, such a tree minimizes the cost of the most costly edge (instead of minimizing the overall cost).

• Is every minimum spanning tree of G a populist spanning tree of G? (Explain (1 point) and prove (2 point) your answer)

Hint: The proof is very similar to MST proof.

Solution:

Yes

Every populist spanning tree G is a minimal spanning tree G. This is needed since the lightest elite edge should always be included in the minimum spanning tree.

Proof:

Let us solve the proof using method of contradiction.

- Let us assume a graph G, where T is a Minimum Spanning Tree (MST). However, it is not a populist tree.
- Considering this assumption, we can derive that there will exist spanning tree “L” with a lighter edge.
- Now that we have shown that edge e = (x,y) in T is an MST of the graph G, it will be heavier than all the edges in L. T will split up into two distinct portions if we remove that edge "e1," which will divide it into 2 halves (the halves may not be equal). These parts include "Q," which holds x, and (V-Q), which holds y.
- This edge, e1, will have the most weight if we add it to L, that will result in at least 1 cycle between x and y.
- Let's now assume that K would be the cycle with the fewest edges between x and y. Let e2 be the edge that will be used to split K into two parts (Q, V-Q).
- Let's now look at a tree M, where M is constituted of the edges T - (e1) + (e2). By consequence, this M will now be a spanning tree.
- Given that weight(e1) > weight(e2) > 0, it is reasonable to assume that weight(T) > weight (M).

Yet, this contradicts with our hypothesis that T is a Minimum Spanning Tree (MST). Hence, we can say that, every populist tree should also be MST. □

• Is every populist spanning tree of G a minimum spanning tree? (Explain in detail (2 point), no need for a proof)

Solution:

No, a Minimal Spanning Tree (MST) and popular spanning tree are not the same.

This is due to the fact that there is no rule indicating that what tree will be the minimum spanning tree, even if it is a populist spanning tree (meaning the elite edge is the lowest among the trees). Some other tree with same elite edge value could end up having a lower total weight making it the minimum spanning tree. So, the populist spanning tree we looked at will not be the minimum spanning tree.

For example:

Consider that a populist spanning tree's edge has a weight of 6, 8, 15 and that our elite edge value is 15. If we take another populist spanning tree with edge weights of 5,6,15 into consideration. Given the overall weight of the edges, the second tree will be the Minimum Spanning Tree (MST). This however contradicts our question.

Problem 2 Coca-Cola formula (Point 5)

The Coca-Cola Company's formula for Coca-Cola syrup, which bottlers combine with carbonated water to create the company's flagship coke soft drink, is a closely guarded trade secret. Company founder Asa Candler initiated the veil of secrecy that surrounds the formula in 1891 as a publicity, marketing, and intellectual property protection strategy. While several recipes, each purporting to be the authentic formula, have been published, the company maintains that the actual formula remains a secret, known only to a very few select (and anonymous) employees; Coca-Cola contains hundreds of different chemical compounds that contribute to its flavor. There is an ongoing debate about which Coke is better: US Coke or Mexican Coke. Suppose a group of Coca-Cola enthusiasts are given n samples of Coke: s1, ...,sn by a sceptic. The sample is either a US Coke specimen or a Mexican Coke specimen. The enthusiasts are given each pair (si ,sj) of coke to taste, and they must collectively decide whether (a) both are the same type of coke, (b) they are different types of coke, or (c) they cannot decide. Note: all pairs are tested, including pairs such as (si ,si), (sj ,si) and (si ,sj) for i, j ∈ 1, 2, 3, .., n, but not all pairs have 'same' or 'different' decisions. At the end of the tasting, suppose the Coca-Cola enthusiasts have made m judgements of 'same' or 'different'. Give an algorithm that takes these m judgements and determines whether they are consistent. The m judgements are consistent if there is a way to label each sample si with 'Mexican' or 'US' such that for every taste-test (si ,sj) labelled 'same', both si and sj have the same label, and for every taste-test labelled 'different,' both si and sj are labelled differently. Your algorithm should run in time O(m + n). Give a correctness argument of you algorithm (1 point is assigned to correctness argument/proof)

Hint: You can consider the samples Si s as nodes/vertices. Testing decisions as edge between two nodes (si ,sj). The problem is solvable using the BFS and graph coloring concepts.

Solution:

In this question, we will make judgements for a graph and check if the given graph will contain a contradiction cycle.

Subsequently, Breadth First Search (BFS) algorithm will tag the nodes and then we will use those tags to further analyze the edges of the given graph.

Algorithm:

1. First, let's make a single graph using the samples (si, sj, si, …., si). The graph's vertices are these samples.
2. We will create an edge connecting two vertices if there is a judgment between them.
3. We will verify to see if an edge j' = (q, p) exists in the graph for each edge j = (p, q).
    a. If it exists, we determine whether e and e' are the same or different.
        i. If not, this output will be returned as a contradiction.
    b. If it does not already exist, we should include this.
    c. We shall run a scanning operation for this step to make it undirected.

4. At this point, each connected graph component will be subjected to a Breadth First Search (BFS) operation. To achieve this, we would choose start node and run BFS operations till all nodes have been visited.
5. We will label node as 0 when Breadth First Search (BFS) is launched. The graph structure uses adjacency list for BFS. Therefore, when we look at an edge (s,r):
    a. We shall label r with the same node s if r is not labeled but corresponds to the same judgment (0).
    b. Otherwise, we'll classify it as different (1).
6. Now in the last step, we scan
    a. If the edge with same edge but two vertices with different labels is considered inconsistent, we halt and print it as inconsistent.
    b. If not, since it now meets all the requirements, we print it as consistent.

Correctness Proof:

- Edge judgment is indicated by the integers 0 and 1, respectively.
- Only if the judgement graph has a cycle with an odd parity are the judgements inconsistent. The XOR operation on its edges is exactly what this parity is.
- Let's look at an odd parity cycle, x = (si, sj, si,..., sj). If x is larger than 2, the algorithm's third step won't detect an inconsistency. If not, the BFS will proceed and tag the nodes x in a specific sequence.
- WOLOG, put (sx, sy) the last edge and let s1 be the first node which the BFS analyzes. Since they were already marked by BFS, it would be the ignored.
- The BFS algorithm will tag these paths consistently since they follow the paths si->sx and sj->sy, respectively. However, since c is in the odd parity cycle, executing an EXOR would return the result 1. As a result, the sixth step of the algorithm will also turn inconsistent.

Running Time:

The fourth and fifth steps of the BFS algorithm will require the most execution time.

All remaining phases will require O(m) or O(n), and the algorithm will run in O (m + n).