Programming Assignment 1
Report

Balaji Karedla

March 9, 2025

1 Problem Statement

Consider a graph G with weights on edges and each edge colored red or blue. Given a threshold T, we want to find a spanning tree whose weight is at most T and want to minimize the number of red edges. Formally, we want to find

$$\min \{k : \text{ there is a spanning tree with } k \text{ red edges and weight at most } T\}$$

Output both the number of red edges and the weight of the optimal tree (which will be at most T). You can assume that the input graph has a spanning tree with weight at most T.

2 Algorithm

My algorithm follows a greedy apprach on sorted edges according to weights. The algorithm first sorts the edges according to weights irrespective of color. It then finds the MST of the graph while prioritizing blue edges over red edges for tie-breaks. I now replace a red edge with blue edge such that the two form a pair with least difference among the pairs such that the red edge lies in the cycle in the tree formed by adding the blue edge.

An algorithm to find all such replacement edges of every tree-edge was given by David Bader and Paul Burkhardt which runs in time complexity of $O(E + V)$[1]. But I implemented a simpler version of this algorithm (an $O(ElogV + V)$ instead of $O(E + V)$ with a simpler implementation of Disjoint Sets over the Gabow-Tarjan implementation) This replacement of minimum difference pair I do until the threshold is reached or no more red edges are left.

This algorithm is a greedy algorithm and is guaranteed to give the optimal solution as the replacement of red edges with blue edges is done in a way that the weight of the tree is minimized and the number of red edges is minimized.

The time complexity of the sorting is $O(ElogV)$ and for each replacement of edges, I run the algorithm in $O(ElogV + V)$ time and there are a maximum of V replacements until the threshold is reached, as there are at most V red edges in the MST and the red edge count keeps decreasing at every replacement. Hence

the overall time complexity of the algorithm is $O(ElogV + V(ElogV + V)) = O(EVlogV)$.

References

[1] David Bader and Paul Burkhardt. A simple and efficient algorithm for finding minimum spanning tree replacement edges. *Journal of Graph Algorithms and Applications*, 26(4):577–588, July 2022.