I collaborated with:

Problem

A *feedback edge set* of a graph G is a subset F of the edges such that every cycle in G contains at least one edge in F. In other words, removing every edge in F makes the graph G acyclic. Describe and analyze a fast algorithm to compute the minimum weight feedback edge set of a given edge-weighted graph. Hint. Relate this problem to some kind of spanning tree problem.

Solution

Algorithm 1 Kruskal's adaptation

$T \leftarrow (V, \emptyset)$ // Eventual maximum cost spanning tree
$R \leftarrow E$
Let F be an empty set
$MakeUnionFind(V)$
while $|E(T)| < |V| - 1$ **do**
    Remove heaviest edge $e = \{u, v\} \in R$ from R
    $uName = Find(u); vName = Find(v)$
    if $uName \neq vName$ **then**
        Add e to T
        $Union(uName, vName)$
    else
        Add e to F
    end if
end while
return F

The runtime of this algorithm is the same as Kruskal's as the extra step takes constant time: $O(|V| + |E|) * O(1) = O(|V| + |E|)$

This algorithm is adding the maximum spanning edges of the graph to T. We throw away edges that would create a cycle. In the problem we looked at in class, this edge that we threw away was the maximum edge of the cycle it would create. If we reverse the algorithm to make a maximum cost spanning tree, we put the minimum edges that will create cycles into F.