Name: Luke Joyce

ID: 107355873

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**

Problem Set 5b (48 points) **Fall 2019, CU-Boulder**

Instructions for submitting your solution:

- The solutions **should be typed** and we cannot accept hand-written solutions. Here's a short intro to Latex.
- You should submit your work through **Gradescope** only.
- If you don't have an account on it, sign up for one using your CU email. You should have gotten an email to sign up. If your name based CU email doesn't work, try the identikey@colorado.edu version.
- Gradescope will only accept **.pdf** files (except for code files that should be submitted separately on Gradescope if a problem set has them) and **try to fit your work in the box provided**.
- You cannot submit a pdf which has less pages than what we provided you as Gradescope won't allow it.
- Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.
- For every problem in this class, you must justify your answer: show how you arrived at it and why it is correct. If there are assumptions you need to make along the way, state those clearly.
- You may work with other students. However, **all solutions must be written independently and in your own words.** Referencing solutions of any sort is strictly prohibited. You must explicitly cite any sources, as well as any collaborators.

Name: Luke Joyce
ID: 107355873

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**
Problem Set 5b (48 points) **Fall 2019, CU-Boulder**

1. (25 pts) For this question, you are going to implement Kruskal's algorithm and union-find to build an MST from supplied data. Refer to the python starter code **MST_Q1_starter_code.py** on Canvas that generates a graph of US cities, where the cities are the vertices and the edges are the distances between them. The code requires **miles_dat.txt.gz** file as the graph data source so keep it in the same folder as the code. Before you start writing any code, make sure you can build the code that's been supplied. The code uses the networkx library. You may need to install this library for the code to run.

   Read all instructions for this question carefully.

   (a) (5 pts) Complete the code to find the edges that are part of the MST. You should add these edges in the list *kruskal_selected_edges*. Do not change the existing format of the edges. They are represented as a tuple of vertices and a vertex is represented like v = "Waukegan, IL". Read the comments in the code for more information. You don't need to read/understand the *miles_graph*() and *draw_graph*() functions.

   (b) (10 pts) Implement the *union*() function to implement Kruskal's.

   (c) (10 pts) Modify your code slightly so that you can produce disconnected components. Let's call these components clusters. The "spacing" of any particular clustering (group of clusters) is defined as the smallest edge between vertices in any pair of different clusters. If we stop Kruskal's k iterations before the algorithm completes, what is the spacing value? Run your code for $k = 2...10$ to generate spacing for all these k values. Your code needs to have this calculation for your answer to receive credit.

   (d) In the pdf that you submit for this assignment, please include the following:

      i. One of the generated graphs **MST.png** that your code produces that shows the MST for that run. Note that on each run, you can get a different number of edges to begin with. Thus, you can expect a different answer each time you run.

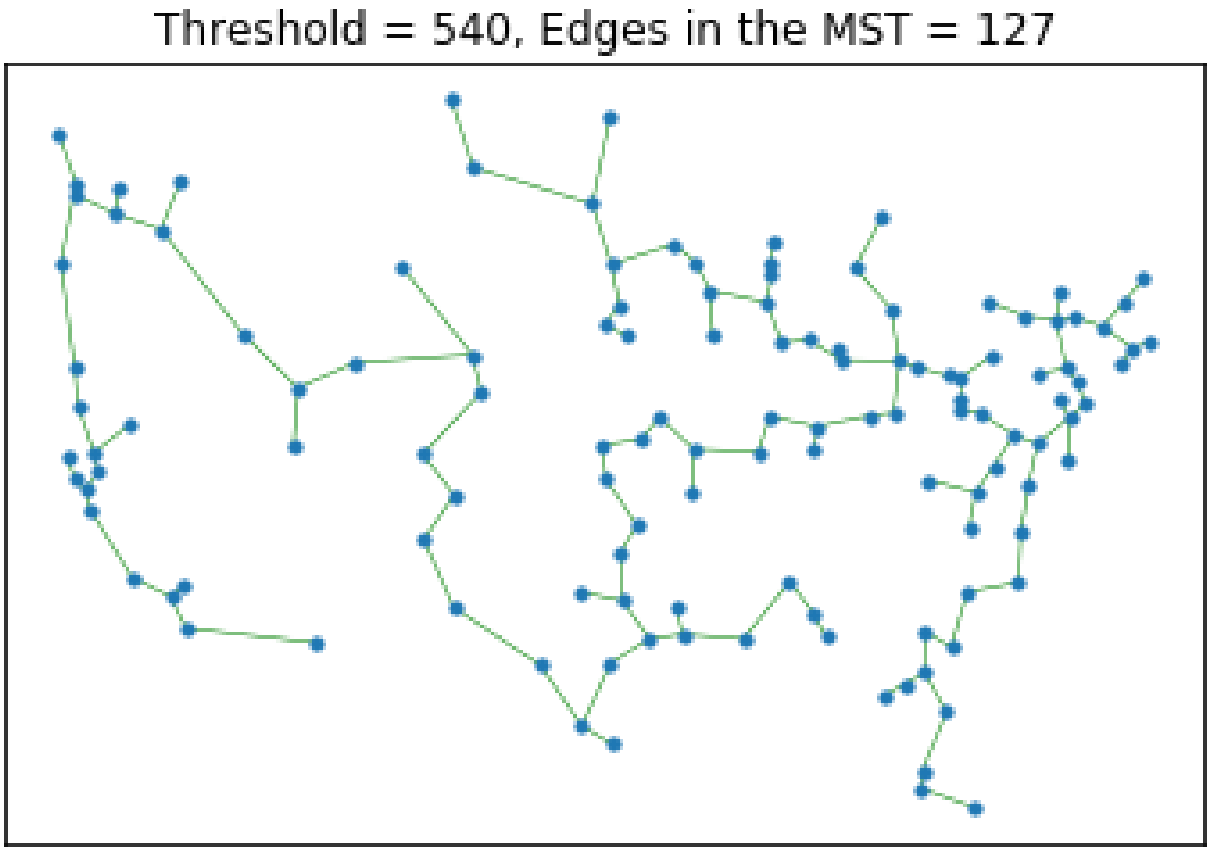
      ii. The spacing values for each k value that you use.

      iii. Your .py file for this question needs to be submitted to Canvas.

Name: Luke Joyce
ID: 107355873

CSCI 3104, Algorithms **Profs. Hoenigman & Agrawal**
Problem Set 5b (48 points) **Fall 2019, CU-Boulder**

(Space for Q1 image and spacing values)




```
k = 2, spacing = 418
k = 3, spacing = 358
k = 4, spacing = 357
k = 5, spacing = 344
k = 6, spacing = 324
k = 7, spacing = 320
k = 8, spacing = 278
k = 9, spacing = 271
k = 10, spacing = 236
```

2. (3 pts) How many disconnected components are there when you stop Kruskal's k round before you complete the MST? Justify your answer.

   Solution. The number of disconnected components after you stop the algorithm k iterations before the MST is completed is k+1. This is the case because if you stop Kruskal's algorithm 1 iteration before the MST is finished there are still 2 components that need to be connected.

3. (5 pts) Consider the recurrence $F_n = 2F_{n-1} + F_{n-2}$, with the base cases $F_0 = 1$ and $F_1 = 2$. Suppose we have letters $v_0, \ldots, v_7$; where for $i \in \{0, \ldots, 7\}$, the frequency of v_i is given by F_i. Draw a Huffman tree for $v_0, \ldots, v_7$.

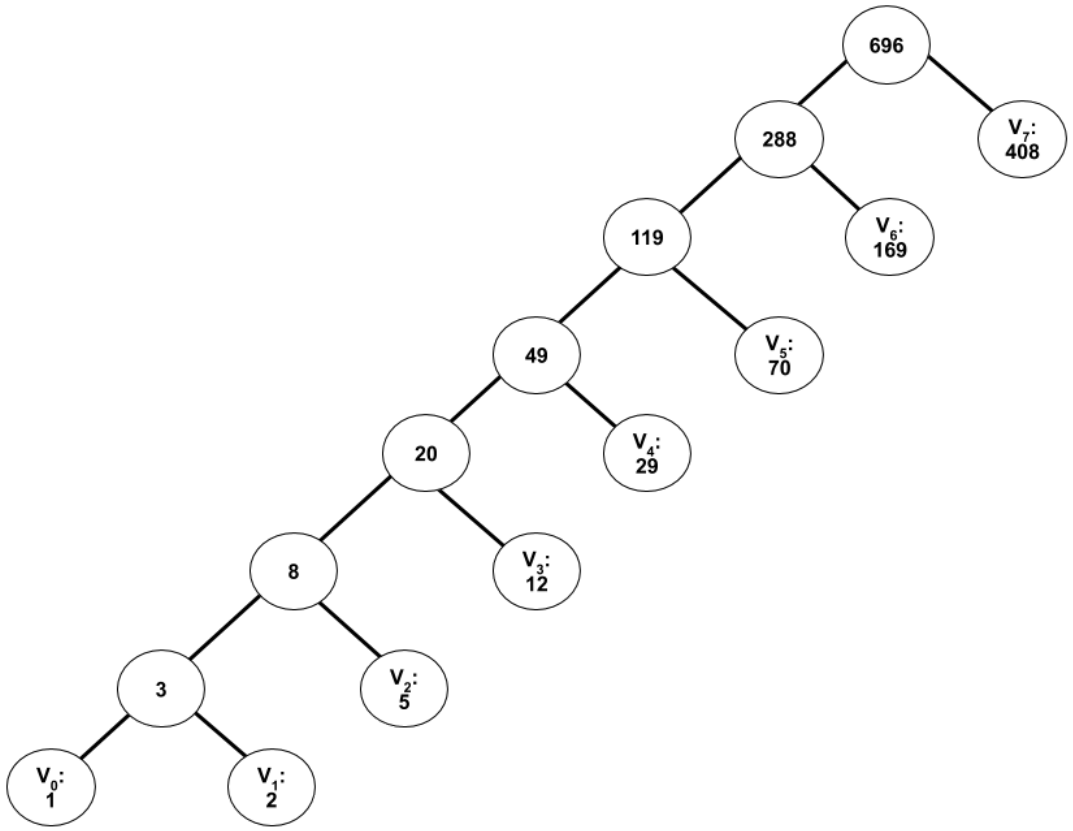


Figure 1: Huffman tree for $v_0, ..., v_7$

Solution.

4. (5 pts) Assume you run your Huffman tree algorithm and you produce the following pre-fix codes. Describe why there must be an error in your algorithm.

```
S = 00
c = 01
i = 001
e = 011
n = 101
```

Solution.
Letter 'S' is encoded as 00, which means it is the left most element two branches down from the root node. Letter 'i' is encoded as 001, which means it is one branch down to the right of the left-most node, or two branches down to the left and one branch down to the right. This presents a conflict, because in the Huffman tree representation, characters are always leaf nodes. 'S' being 00 and 'i' being 001 means that 'S' is the parent node of 'i', which doesn't make sense because trees are merged.

5. (10 pts) Assume you're given an integer matrix that represents a plot of land, where the value at that location in the matrix represents the height above sea level. A value of zero indicates water. A pond is a region of water connected vertically, horizontally, or diagonally. The size of the pond is the total number of connected water cells. Write an algorithm to compute the sizes of all ponds in the matrix.

   Example:

   ```
   0 2 1 0
   0 1 0 1
   1 1 0 1
   0 1 0 1
   ```

   would output 1, 2, 4.

   (a) (3 pts) Describe the graph data structure that your algorithm will use for this problem.

   Solution. The data structure that I would use for the graph would be a dictionary of all the vertices, similar to the way we implemented properties of the vertices in Kruskal's algorithm on this homework.

(b) (2 pts) Provide a 3-4 sentence description of how your algorithm works, including how the matrix is converted to the graph, how adjacent vertices are identified, and how the algorithm traverses the graph to identify connected vertices.

Solution.
In Kruskal's algorithm, each edge is added to a list in ascending order and is checked to see if their two vertices are part of the same tree or not. Similarly, we add all edges between adjacent 0's to a list, in no specific order. Edges are between two elements that are above, below, next to or diagonal to each other. Then we use Union-find to determine what each 'pond' a zero is a part of. This is needed so that we don't end up witht wo separate ponds that should be part of the same one. (There are specific cases here this could happen if we don't use union-find.)

(c) (5 pts) Write an algorithm to solve this problem.

Solution.
```
PondsInMatrix(G):
For each vertex v = 0 in G.V:
    MAKE-SET(v)
For each edge (u, v) in G.edges where both vertices = 0:
    if FIND-SET(u) != FIND-SET(v):
        UNION(u, v)
For each vertex v = 0 in G.V
    increment number of 0's in each set
```