EE189 Project: Phylogeny

Daragh Crowley, Theodoris Papaiakovou

I. Introduction

In this project the subject of reconstructing the phylogenetic tree of a segment of the Sars-Cov2 genome is considered. The data are generated by simulating mutations and an algorithm based on the minimum spanning tree problem is used to reconstruct the minimum edge cost phylogenetic tree. Then an analysis on the similarity, the total edge cost difference between the ground truth tree and the minimum cost tree and the running time required for the program to construct the minimum cost tree for different segment lengths, maximum number of generations and number of nodes is made.

II. Discussion

A. Generating Data

To generate the ground truth tree we took an *n* length segment from the SARS-CoV-2 genome and performed random insertions, substitutions and deletions on the segment. The mutation probability, number of mutations per generation and number of generations are adjustable. We construct the ground truth tree for comparison with the minimum cost tree we later find.

B. Our Algorithm

Kruskal's algorithm [1] can be used to find the minimum spanning tree in a graph. The Python graphing package, NetworkX [2], includes a function that finds the Steiner Tree using Kruskal's algorithm. We used this function to find the minimum cost tree which explains the observed genomes. We start by creating a complete graph with all of the observed vertices. The edit distance of each pair of vertices is computed and stored in a dictionary using our function from Homework 2. Kruskal's algorithm takes the following steps to find the minimum cost spanning tree from the complete graph:

- Arrange edge costs in ascending order.
- Start by adding the lowest cost edges to the tree.
- Add the next lowest cost edge to the tree as long as adding this edge does not create a cycle.
- Continue this procedure until all vertices have been added to the tree graph.

After the minimum spanning tree is created, we plot the tree using the NetworkX package. The root node is known from when the data is generated. The total edge cost can be computed by adding the edit distances stored in the dictionary from before.

There can be many minimum spanning trees for a graph. This is especially true in our case because we are finding the minimum spanning tree of a complete graph which has n^{n-2} spanning trees, where *n* is the number of vertices. Our function returns just one of these minimum spanning trees even though there may be more.

III. Results

First of all, some examples of ground truth phylogenetic trees (Figures 1 and 3) are constructed for maximum number of generations of 4 and 6 and they are compared with the minimum Steiner tree found (Figures 2 and 4).

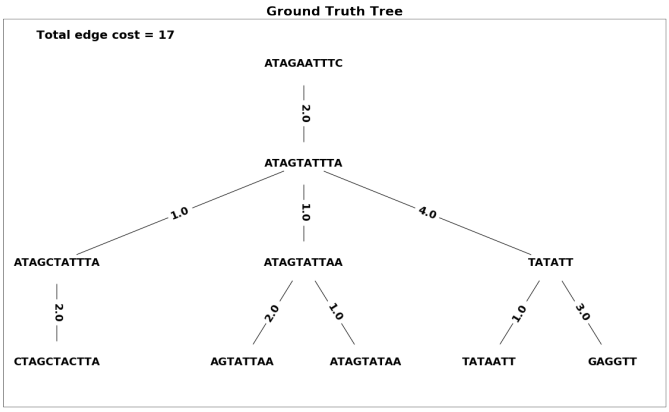

Fig. 1: Ground truth tree with 4 generations

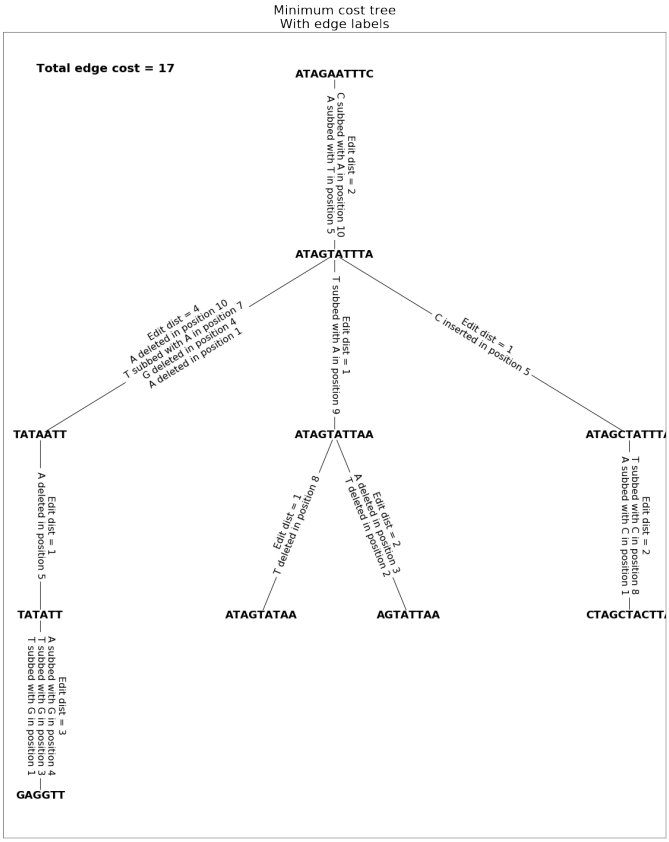

Fig. 2: Minimum cost tree found from vertices in Figure 1

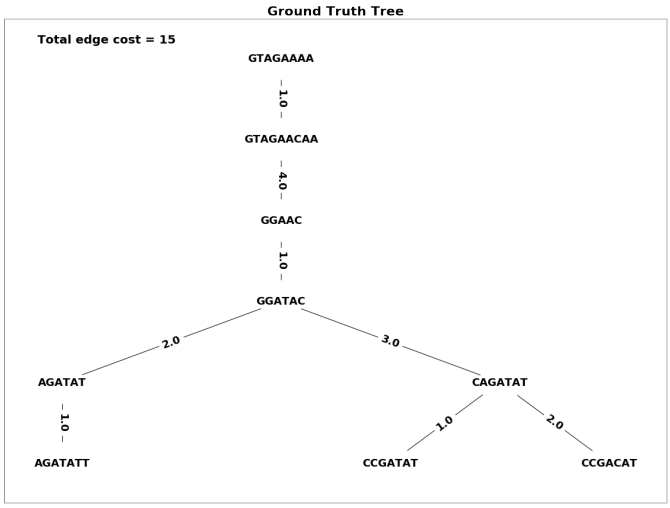

Fig. 3: Ground truth tree with 6 generations

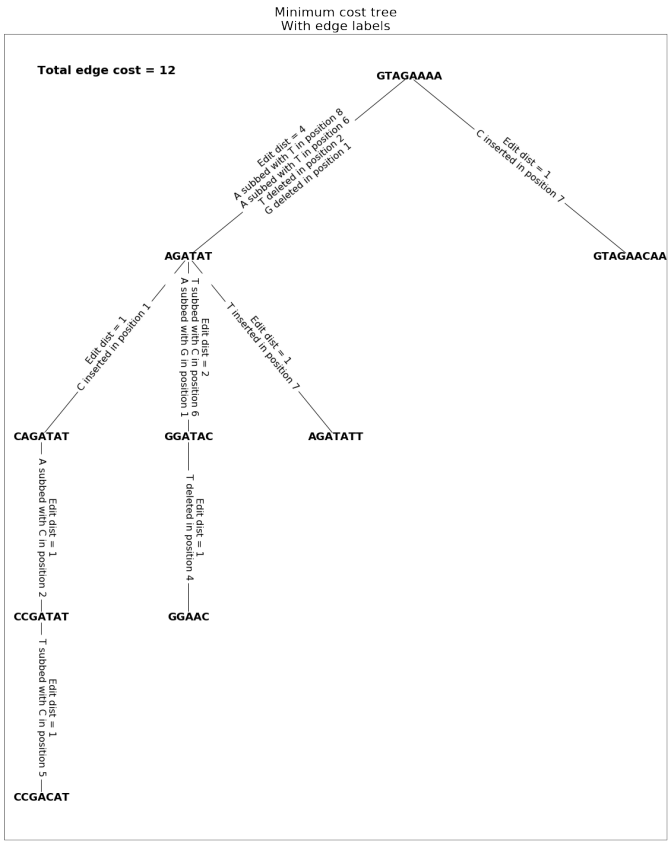

Fig. 4: Minimum cost tree found from vertices in Figure 3

Moreover, the similarity between the structure of the ground truth tree and the minimum cost tree, the total edge cost difference between the ground truth tree and the minimum cost tree and the running time required for the program to construct the Steiner tree were measured and plotted over the number of segment lengths (5,10,15,20,40,100) for different maximum number of generations (4 ,6 ,8). The values are averages over 50 repetitions.

From Figure 5 it is observed that as the segment length increases the time required to find the Steiner tree and the cost difference is increased and this relationship was more prominent for higher max number of generations. The similarity data is inconsistent to suggest a relationship between the segment length or the number of generations. It is clear that as the number of generations increases the total time required and the cost difference increase as for 4 generations the time is in between 0.5-1.5 milliseconds and the cost difference is in between 0-15 while for the 8 generations the time is in between 0-0.2 seconds and the cost difference is in between 0-80.

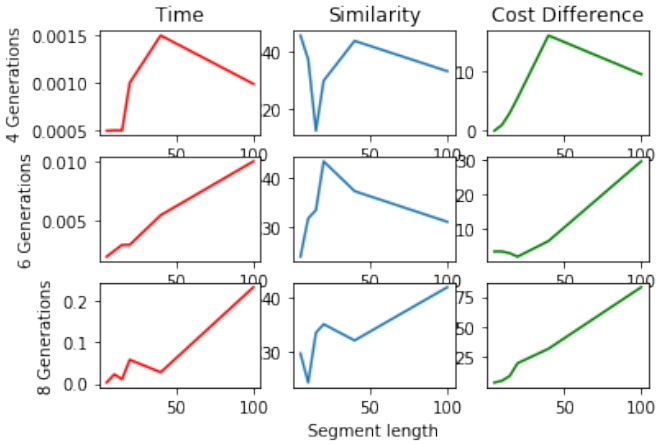

Fig. 5: Test results

In addition, these values were plotted over the number of nodes (vertices) of each graph, shown in Figures 6, 7 and 8.

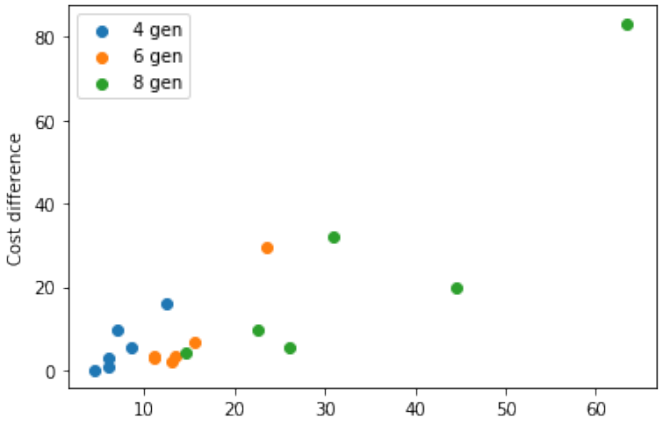

Fig. 6: Number of vertices vs cost difference

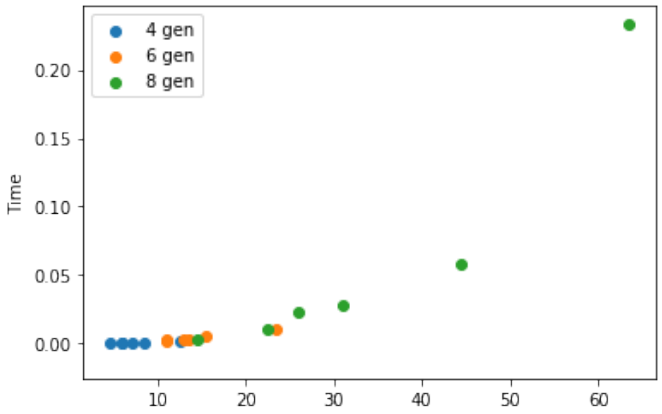

Fig. 7: Number of vertices vs computation time

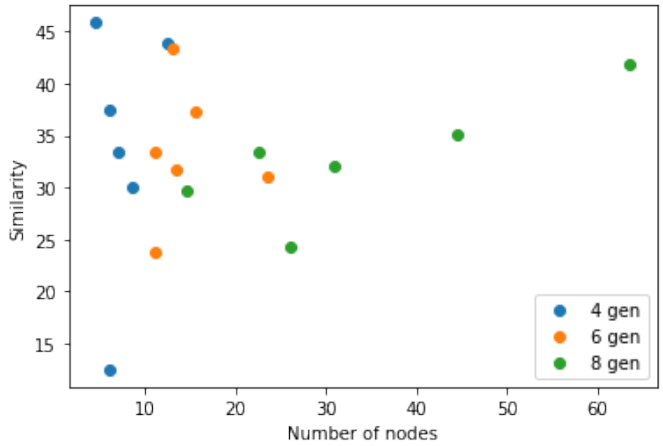

Fig. 8: Number of vertices vs similarity

From Figure 8, it is seen that the similarity is independent on the number of nodes and the max number of generations as the data are too scattered. Furthermore, it is suggested in Figures 6 and 7 that the cost difference and the running time have an exponential relationship with the number of nodes of the graph with more nodes naturally occurring for trees of greater max number of generations.

IV. Analysis and Potential Improvements

Our algorithm works because Kruskal's algorithm is a "greedy" algorithm. You can always find a better spanning tree if there is an unused edge that has a lower cost than any edge already on the tree, provided it does not form a cycle. In the case of our problem, the genomes are observed in a random order, allowing us to reconstruct the tree in the order that provides the minimum cost. Our project shows one way to find the minimum cost tree of a set of observed genomes using a well known algorithm. Some improvements that could be made are as follows:

- Remove unlikely edges (e.g. high edit distance) in the complete graph before calling the Steiner tree function. The max number of mutations per generation is known when the data is generated so this information could be used to make better decisions in reconstructing the minimum cost tree.
- Include extra constraints on the minimum cost tree. For example, keep the leaves on the ground truth tree the same as the leaves on the minimum cost tree. Vertices further down the tree should represent more recently observed genome reads. This could be used to more accurately examine how a virus mutates over time.
- Our code works well for trees with up to about 10 generations. After this point the code takes a long time to run. We noticed that generating the data for long trees was very slow while the length of time to compute the Steiner tree still remained reasonably fast. Thus, if the data could be generated in a more efficient way we would have been better able to test our algorithm that finds the minimum spanning tree for very large trees (e.g. 10+ generations, 100+ nodes).

V. Conclusion

To summarize, we used Kruskal's algorithm to reconstruct the minimum edge-cost trees. It was seen that the algorithm achieved our goal as the total edge cost of the reconstructed minimum cost trees was always smaller than the ground truth trees. Our analysis also showed that the running time and the total edge cost difference are proportional to the segment length and the number of generations in the ground truth tree. Running time increases exponentially with the number of nodes which is expected as a greater number of nodes results in more potential minimum trees which requires more computation. The similarity between the ground truth tree and the minimum cost tree appear to be uncorrelated to the number of nodes, the segment length or the number of generations. This can be explained as the minimum cost tree constructed is randomly chosen between all the minimum cost trees that could be implemented.

References

[1] Kruskal, Joseph B, On the shortest spanning subtree of a graph and the traveling salesman problem, *Proceedings of the American Mathematical society*, vol 7(1), pp.48-50, 1956

[2] Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, Exploring network structure, dynamics, and function using NetworkX, *Proceedings of the 7th Python in Science Conference (SciPy2008)*, pp. 11-15, Aug 2008.