Shared Shortest Paths in Graphs

By Brooks Emerick

Mentor: Dr. Sean McCulloch, Department of Mathematics and Computer Science at Ohio Wesleyan University

Terminology:
- A graph is a set of points called vertices that are connected by lines called edges.
- An unweighted graph is a graph that has no arbitrary cost assigned to each edge.
- A graph is either directed or undirected. Directed graphs, or digraphs, have arrows to specify the direction of motion; unweighted graphs have lines as edges to represent any possible movement.
- A journey is a designated starting vertex, the source, and ending vertex, the sink.
- A path is a list of vertices that describes a way to get from a source to a sink.
- A graph is a set of points called vertices that are connected by lines called edges.
- A weighted graph is a graph that has an arbitrary cost assigned to each edge. The cost can represent anything from distance to the amount of resources needed to travel an edge.
- A journey is a designated starting vertex, the source, and ending vertex, the sink.
- A path is a list of vertices that describes a way to get from a source to a sink.

Finding an Algorithm to Produce the Shared Shortest Path with Equilibrium is Known to be NP-Complete. This means that there is no known way to solve the problem in polynomial time. In the shared shortest path algorithm that we created, the problem was such that as the number of journeys increased, the time to perform the algorithm became exponential. Therefore, we examined the possibility of creating a heuristic that closely approximates the shared shortest path or finds non-equilibrium configurations in graphs, and ran in polynomial time.

Heuristics:
- NP-Complete:
- Terminology

Heuristic:
- The Approximate Shared Shortest Path Heuristic:

Step 1:
- Use Dijkstra’s Method, find the unshared shortest path from each journey’s source to sink. If any paths are overlapping, compute the cost accordingly since they will now be sharing those overlapping edges.

Step 2:
- How delete the edge that has the largest coalition of journeys on it (c). The coalition consists of the journeys that are sharing the edge that is currently deleted.
- Move the non-coalition member that unchains this deletion. A coalition member can only share with a non-coalition member if it is beneficial.

Step 3:
- How repeat step one and find the new unshared shortest path of the coalition members only.
- If they come up, compute the costs accordingly.
- If the new cost of at least one of the coalition members is shorter, then ignore the previous graph to this new graph, and keep the total costs for every journey.

Step 4:
- How repeat step 2 by deleting the edge that is most shared (d), and register the new paths that were deleted before.
- Move the new paths of the coalition members. After this, if there are still journeys that share those paths, then knowing the position of the non-coalition members, then find the new coalition members’ new paths.

Conclusions and Future Work:
As of now, we have a heuristic that runs in O(V^2E) time that can produce a shared shortest path given any amount of demand in the journeys. We would like to be able to use more accurate heuristics to create better approximations. To do this we will have to run the programs on a sample of graphs to gain more experience. We would also like to use more data to find the heuristics approximate the true solution. Creating different variations to the algorithm can only help create better approximations. We would also like to use more data to find the heuristics approximate the true solution. Creating different variations to the algorithm can only help create better approximations.

Acknowledgements:
- Dr. Sean McCulloch
- The Physics and Mathematics/Computer Science faculty that were involved in the REU program for providing free food and valuable information at the Friday lectures.
- The Ohio Wesleyan University for paying for my research expenses and graphs had journeys that saved at least 15% of their costs. Also, 60% of all updated graphs had journeys that saved at least 20% of their costs.
- The heuristic, those 90% of graphs whose costs improve, on average, save at least 3% of their starting cost.
- We noticed that if a group of journeys is sharing and then split into two separate groups, then the two groups will never meet up at vertices t and b.
- If the weight is a, b, and c as pictured on the graph. The smallest arrow shows which groups are traveling to meet up during indicated edges.
- Since we are assuming the statement is true, we have

$$a - b + c - t = \frac{x - y}{t - x}$$

for the group t-x. This inequality holds for the t-x group to want to travel along the a weighted edge. Also, for the case of the group a and group y, we have

$$a - b + c - t = \frac{x - y}{t - x}$$

Conclusions and Future Work:
As of now, we have a heuristic that runs in O(V^2E) time that can produce a shared shortest path given any amount of demand in the journeys. We would like to be able to use more accurate heuristics to create better approximations. To do this we will have to run the programs on a sample of graphs to gain more experience. We would also like to use more data to find the heuristics approximate the true solution. Creating different variations to the algorithm can only help create better approximations. We would also like to use more data to find the heuristics approximate the true solution. Creating different variations to the algorithm can only help create better approximations.

Acknowledgements:
- Dr. Sean McCulloch
- The Physics and Mathematics/Computer Science faculty that were involved in the REU program for providing free food and valuable information at the Friday lectures.
- The Ohio Wesleyan University for paying for my research expenses and graphs had journeys that saved at least 15% of their costs. Also, 60% of all updated graphs had journeys that saved at least 20% of their costs.
- The heuristic, those 90% of graphs whose costs improve, on average, save at least 3% of their starting cost.
- We noticed that if a group of journeys is sharing and then split into two separate groups, then the two groups will never meet up at vertices t and b.
- If the weight is a, b, and c as pictured on the graph. The smallest arrow shows which groups are traveling to meet up during indicated edges.
- Since we are assuming the statement is true, we have

$$a - b + c - t = \frac{x - y}{t - x}$$

for the group t-x. This inequality holds for the t-x group to want to travel along the a weighted edge. Also, for the case of the group a and group y, we have

$$a - b + c - t = \frac{x - y}{t - x}$$

Conclusions and Future Work:
As of now, we have a heuristic that runs in O(V^2E) time that can produce a shared shortest path given any amount of demand in the journeys. We would like to be able to use more accurate heuristics to create better approximations. To do this we will have to run the programs on a sample of graphs to gain more experience. We would also like to use more data to find the heuristics approximate the true solution. Creating different variations to the algorithm can only help create better approximations. We would also like to use more data to find the heuristics approximate the true solution. Creating different variations to the algorithm can only help create better approximations. We would also like to use more data to find the heuristics approximate the true solution. Creating different variations to the algorithm can only help create better approximations.