Project Checkpoint 2 (Group 9): Traffic Network Simulation in Pluvial Flash Flood: Rerouting and Drainage system recommendations

Summary of current work

We have developed a basic simulation model to generate the shortest paths for a given traffic network under certain traffic demands (trips within the network). Usage of each link under this travel demand is visualized by plotting the traffic flow (number of vehicles) with direction on the network. The shortest paths between nodes are also visualized. Please refer to the example section.

The objective function of the optimization is the shortest path, which is defined as the shortest travel time for each trip. For the time being, a very simple assumption is currently made here: the travel time of each link is constant no matter how many vehicles are currently on it, which would be updated by considering the capacity of the link and the flash flood situation.

Model Development Progress

Github repository: https://github.gatech.edu/glee388/2023Fall-CSE6730-Group9

- **Network**

Current progress: We currently generate a grid like road network with N x N nodes utilizing **networkx** package (the function `create_grid_network` in the code). The function takes into the following inputs and converts them as attributes of the network: grid dimension, link length, speed limit, vehicle length, lane per direction, reaction time of vehicle. The output of the function includes a network with these attributes, position of nodes for visualization and node labels as ID.

Future work: the road network could be in more realistic shapes other than N x N grids. There are also some published **networkx** network representations of real-word road networks that we might consider using.

- **Traffic demand**

Current progress: The traffic demand is represented as an Origin-Destination (OD) Matrix. We developed a function ('generate_random_od_matrix' in the code) to randomly generate OD matrix using Poisson distribution.

- **Routing optimization – Shortest path finding for each trip**

Current progress: We currently use Dijkstra's algorithm for all OD pairs to generate the shortest path for each trip, given travel time for each link the weight ('find_optimal_paths' in the code). We have now implemented an optimization algorithm to minimize the total approximate travel time for all the vehicles trying to traverse the map. Dijkstra's algorithm provides a good starting point, and with a 10x10 grid, this algorithm produces a total travel time of 4.312e10 minutes. We have since implemented a Simulated Annealing algorithm which has effectively reduced the total travel time to 8.321e8 minutes,

reducing the total travel time by 98%. This algorithm does take quite a while to converge depending on the size of the problem, however.

Future work: we will take into account the capacity of the link and the flash flood situation by updating the travel time of each link when each vehicle arrives at a node and run the 'find_optimal_paths' again.

Capacity: When the flow on a link hits its maximum capacity, this link is blocked so the travel time of this link is updated as infinity or a certain large number to make sure it is at the least priority in path finding.

Flash Flood: the flash flood's impact on the link depends on the inundation and could be interperated as the lower travel time in the model. The impact could be categorized in 3 levels with corresponding coefficients of capacity. For example, 0 for road closure due to high inundation, 0.5-0.6 for moderate inundation, 0.85 for wet surface. When a flash flood happens, the whole system will be wet and all link capacity is updated with the coefficient of 0.85, some by 0.5 and a few by zero according to the flash flood scenario.

Basically, each time the vehicle arrives at the node, the network is updated with a new travel time for each link and routing for the rest of the trip is based on the status of the link. The ultimate optimization would be the shortest travel time for each trip to complete in the OD matrix given the road capacity limit and flash flood influence.

Visualization Example

Figure 1 shows a visualization of the shortest path between node 22 to node 55.
Figure 2 shows the total traffic flow of the network after optimization based on the shortest path algorithm.

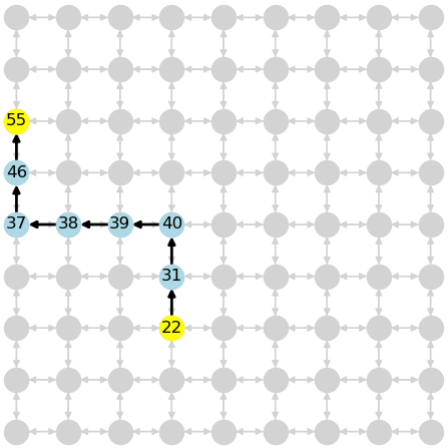

Figure 1

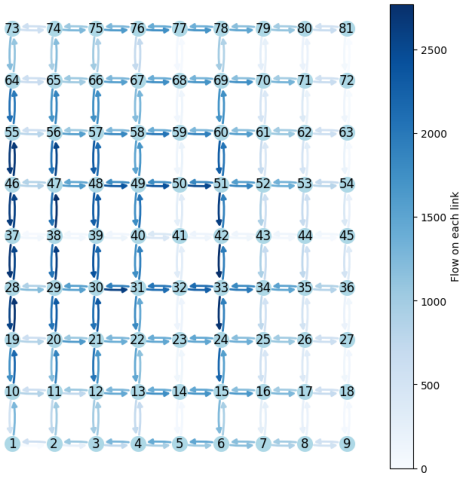

Figure 2

Work division

Discussion: Atticus, Li-Yen (Zoey), Yifan, Garyoung

Base code programming: Garyoung
Backbone programming: Li-Yen (Zoey), Atticus
Writing: Yifan, Atticus, Garyoung, Zoey