Optimization II Project: Emergency Dispatch Optimization

Inés De Remedios, Javier Sanchez-Prieto, Jorge Rivero, Margherita Tonon

May 2025

1 Project Goal and Dataset Overview

1.1 Project Goal

In large, high-traffic urban areas like Manhattan, it is crucial to ensure that emergency vehicles, such as ambulances and fire trucks, arrive at their emergency destinations efficiently to avoid life-threatening delays.

In a graph, a *path* of length n from u to v is a sequence of n edges $e_1, e_2, ..., e_n$ such that there exists a sequence of vertices $x_0 = u, x_1, ..., x_{n-1}, x_n = v$ where each edge e_i connects x_{i-1} to x_i. In a weighted graph, a *shortest path* is the path from u to v with minimal sum of edge weights.

Therefore, optimizing emergency routes consists in modeling Manhattan's roads in a weighted graph, and subsequently finding the shortest path from the source node, the dispatch center, to the goal node, the emergency location.

The goal of this project is to evaluate shortest path routes from Manhattan fire stations to simulated emergency locations. We will utilize different distance metrics – for example, road length and travel time – with the ultimate goal of identifying routes that minimize emergency response time.

1.2 Dataset Overview

To meet the project goal, we utilized the `OSMnx` library as well as a dataset listing all of the firehouses in New York City.

The `OSMnx` library enables the downloading, modeling, analysis, and visualization of street networks and other geospatial features from OpenStreetMap [1]. Using the `graph_from_place` function, we downloaded the drivable Manhattan street map, where edges represent roads and nodes represent road intersections. Edges have attributes representing road properties, namely `osmid, highway, maxspeed, name, oneway, reversed, length, geometry`, and `lanes`, some of which were used to calculate various measures of distance to evaluate route efficiencies. Moreover, additional `OSMnx` functions were utilized to add edge attributes, such as `ox.add_edge_speeds` and `ox.add_edge_travel_times`.

The firehouse dataset is a `.csv` file which contains the names, addresses, latitudes, longitudes, postcodes, and other geographical information of firehouses in all of New York City [2]. Extracting exclusively the Manhattan firehouses, along with their latitudes and longitudes, we mapped all the Manhattan fire stations onto nodes of the `OSMnx` Manhattan graph.

We ultimately worked with a directed, weighted graph of Manhattan in which some nodes represented fire stations.

2 Explanation of chosen algorithms

To find the shortest paths between fire stations and emergency locations, we utilized Dijkstra's algorithm.

Dijkstra's algorithm finds the shortest path between a start node and all other nodes in the graph. The algorithm starts by labeling all vertices as "unvisited"; the algorithm iteratively selects the unvisited vertex with the smallest distance from the source. Then, neighbors of this vertex are visited and the distance is updated if a shorter path is found. The vertex is then marked as "visited". This process repeats until all vertices are visited. Dijkstra's algorithm is guaranteed to find the global optimal shortest path [3]. Although A* is another well known shortest path algorithm, it requires a heuristic function and is often used for large graphs. Preliminary tests showing Dijkstra's performed just as well as A*, as well as the advantage of not requiring a heuristic, supported our choice to use Dijkstra's throughout the investigation.

3 Results and Reflections

We simulated an emergency scenario by selecting three nodes in the network – one in the north, one in the middle, and one in the south of Manhattan – representing the locations where a fire truck is needed (Figure 1).

Figure 1: Manhattan map with fire stations (red) and simulated emergency locations (blue)

Firstly, we utilized road length (in kilometers) as the weight of the graph. Applying Dijkstra's to find the shortest paths with this standard weight, we obtained the routes seen in Figure 2.

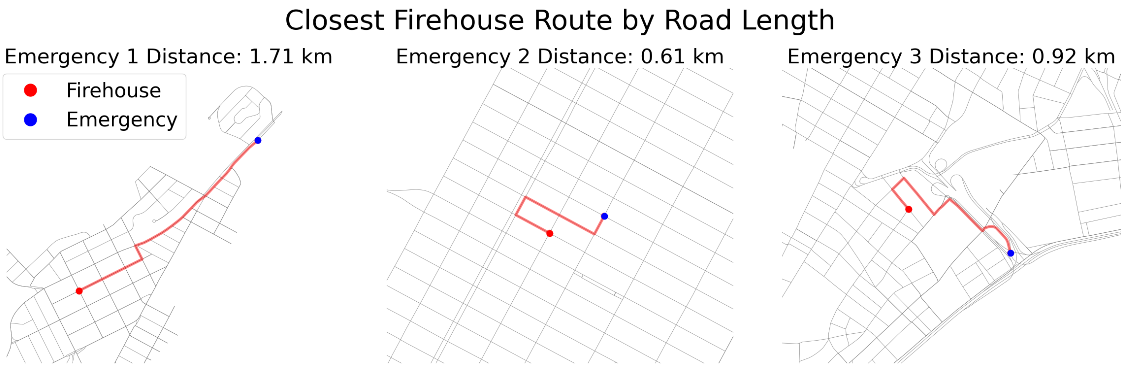

Figure 2: Shortest paths to emergency locations based on road length, in kilometers.

At first glance, some of these routes appear unnecessarily long; for example, in emergency 2 the fire truck could simply travel one block southeast and two blocks northeast to reach its destination. However, `OSMnx` takes into account whether streets are one-way, whether they are limited to pedestrian access only, whether there are certain turn restrictions, etc.. Therefore, some streets that would make the path shorter cannot be crossed due to the `OSMnx` constraints, although in practice emergency vehicles could potentially bypass these restrictions and be able to cross anyway.

Next, using the `ox.add_edge_travel_times` function, we utilized travel time as the edge weight and computed shortest paths (Figure 3).

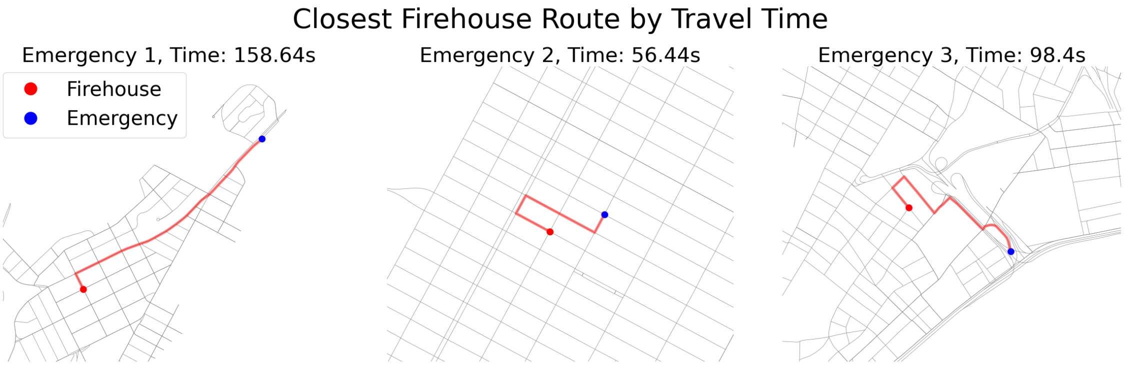

Figure 3: Shortest paths to emergency locations based on road travel time.

Compared to the shortest paths calculated with road length, emergency 2 and emergency 3 paths did not change. Road length is related to the travel time – usually, the longer the road, the longer the travel time – and therefore it makes sense that the paths did not differ dramatically. Emergency 1 differed by one turn southeast. When comparing the highway types in the road-length and travel-time paths of Emergency 1, we find that the time-based path primarily uses primary highways, with only one tertiary highway, while the road-length-based path includes four tertiary highways. Primary highways are larger and allow for higher speeds compared to tertiary highways, thus reducing the travel time and explaining why the travel-time based path prefers traveling on primary roads even if it results in covering more distance.

We then created a weight function that inflated the road travel time based on the number of lanes. Roads with fewer lanes offer less space for fire trucks, making it harder to pass through or overtake other cars. Therefore, roads with fewer lanes were assigned higher weights. The shortest paths calculated with the lane-penalized weight are shown in Figure 4.

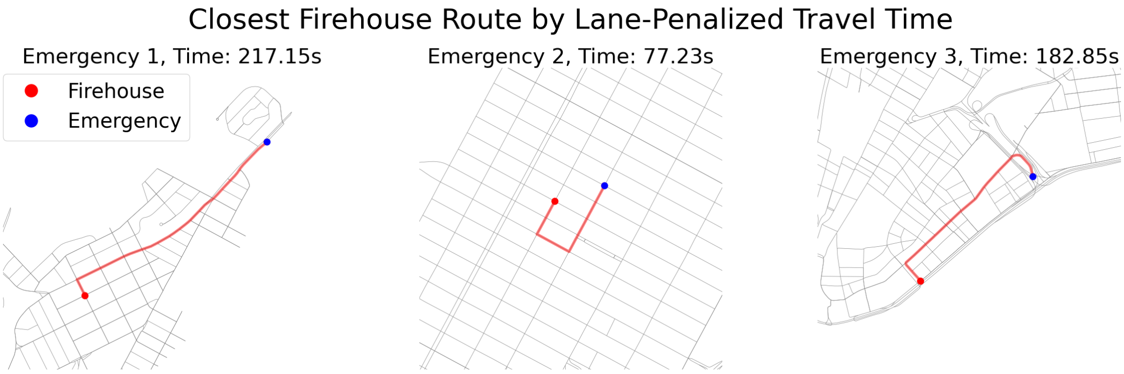

Figure 4: Shortest paths to emergency locations based on lane-penalized travel time.

The average number of lanes for the entire network was 2.04. For emergency 1, the penalized path remained unchanged compared to the pure travel-time path because the pure travel-time path already had an average of 3.21 lanes, which was higher than the network average. For emergency 2, the pure travel-time-based path had an average of 2.1 lanes, while the lane-penalized path had an average of 3.62 lanes – explaining the visible change in the route, as roads with more lanes were prioritized. For emergency 3, the closest firehouse changed. The average number of lanes for the pure travel-time path was 1.55 and for the lane-penalized path was 3.14; the dispatch center from which the vehicles depart changed because the new firehouse location enables traveling on roads with more lanes.

Another edge attribute is the highway type. The highway types include motorway, motorway link, trunk, trunk link, primary, primary link, secondary, secondary link, tertiary, tertiary link, residential, living street, and unclassified, ordered from largest/most important to smallest/least important [4]. Larger roads (motorways, trunks, primary roads) tend to have a higher number of lanes and fewer intersections and stops, allowing trucks to speed through and quickly reach their destination. However, in the case of peak hour high-traffic conditions, traveling on these roads may no longer be beneficial due to congestion and traffic jams. In this case, fire trucks may prefer to travel on smaller, "back" roads which are likely less heavily trafficked. We therefore looked at two different weight functions, one which penalized the travel time on smaller roads (Figure 5) and one which penalized on larger highways (Figure 6).

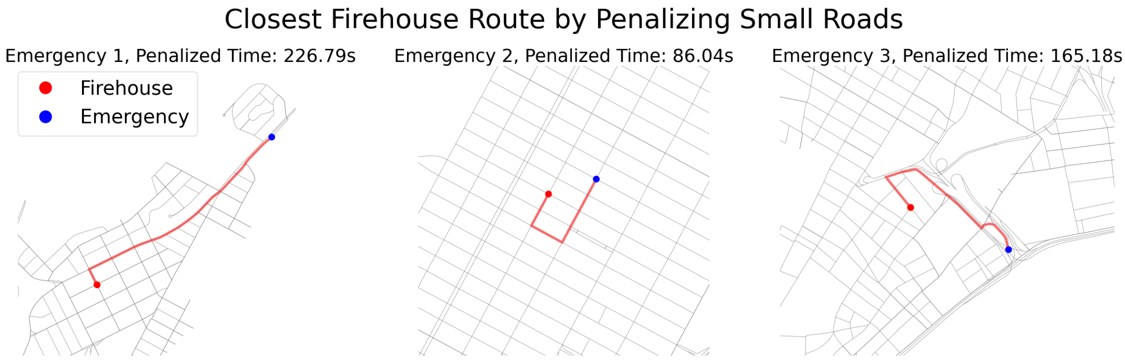

Figure 5: Shortest paths to emergency locations based on highway-penalized travel time, penalizing smaller roads.

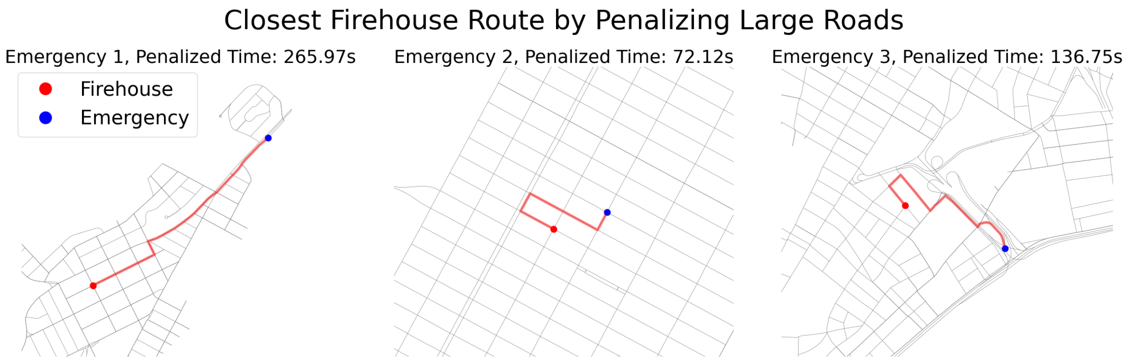

Figure 6: Shortest paths to emergency locations based on highway-penalized travel time, penalizing larger roads.

For emergency 2, the small-road penalized path had 5 primary highways and 3 secondary highways, whereas the large-road penalized path had 2 primary highways and 3 residential roads. We can see that the path in Figure 5 has only 2 turning points whereas the path in Figure 6 has 3, which is congruent with the fact that larger roads have less intersections and are less wiggly. For emergency 3, the small-road penalized path included 1 unclassified road, 1 residential road, 5 secondary roads, 1 primary road, and 2 motorway links, while the large-road penalized path had 1 unclassified road, 4 residential roads, 2 secondary roads, 1 primary road, and 2 motorway links. Although the small-road penalized path had slightly more larger roads, it is unlikely that a path can consist only of the most favorable highway types due to the structure of the roads around an emergency location. An emergency could be located right by a major highway and thus a large road must be crossed to arrive, or an emergency location could be in a more residential area surrounded by smaller roads. This explains why both small and large roads appeared in the paths for emergency 3 despite the penalization. For emergency 1, the small-road penalized path had 1 tertiary road and 16 primary roads, while the large-road penalized path had 4 tertiary roads and 13 primary roads. Similarly to emergency 2, we observe large roads even when penalizing for them. This indicates that even when larger roads are penalized, as they would be in high traffic situations, the network topology still sometimes pushes fire trucks to travel through these roads, and similarly, smaller roads can also be prioritized despite penalties.

4 Conclusion

Through this project, we identified and discussed various shortest paths from firehouses to emergency destinations based on road length, road travel time, the number of lanes a road had, and the type of highway a road was. We obtained various routes that dispatch centers can select from based on various priorities – whether this be minimizing distance, maximizing the number of lanes, etc.

Nonetheless, various assumptions were made throughout the project, mainly due to the `OSMnx` library structure. Firstly, `OSMnx` assumes that vehicles follow the speed limit, which is not the case for emergency vehicles that speed through roads. As a result, the estimated travel time, calculated assuming the speed limit is met, may overestimate the true travel time a fire truck experiences. `OSMnx` also enforces restrictions like one-way streets and pedestrian-only zones, which fire trucks often ignore; there may exist shorter, more direct paths if these constraints were not applied. Finally, the model does not take into account whether certain roads are under construction or temporarily blocked off. If a fire truck were to follow the suggested route, it could encounter unexpected obstructions and lose time, making the calculated path suboptimal.

Further investigation could focus on refining a model that more accurately reflects the behaviour of emergency vehicles, such as removing certain constraints imposed by `OSMnx`. Additionally, it would be insightful to investigate traffic patterns across different areas of NYC. By identifying which roads have heavier traffic at specific times of day, a time dependent-weight favouring less congested areas could be used to generate more realistic paths for fire trucks at any hour.

References

[1] Boeing, Geoff. "OSMNX: New methods for acquiring, constructing, analyzing, and Visualizing Complex Street Networks." Computers, Environment and Urban Systems, vol. 65, Sept. 2017, pp. 126–139, https://doi.org/10.1016/j.compenvurbsys.2017.05.004.

[2] "FDNY Firehouse Listing: NYC Open Data." FDNY Firehouse Listing — NYC Open Data, 8 Apr. 2022, data.cityofnewyork.us/Public-Safety/FDNY-Firehouse-Listing/hc8x-tcnd/about_data. Accessed 8 Apr. 2025.

[3] "What Is Dijkstra's Algorithm?: Introduction to Dijkstra's Shortest Path Algorithm." GeeksforGeeks, GeeksforGeeks, 9 Apr. 2025, www.geeksforgeeks.org/introduction-to-dijkstras-shortest-path-algorithm/. Accessed 29 Apr. 2025.

[4] "Key:Highway." Key:Highway - OpenStreetMap Wiki, wiki.openstreetmap.org/wiki/Key:highway. Accessed 22 Apr. 2025.