# Optimizing Ride Hailing Under Uncertainty

Saakaar Bhatnagar (06340885)*

**This paper aims to develop and demonstrate a method to improve the efficiency of ride hailing services in starting and completing trips during demand surges, by integrating aspects of customer mobility into route planning and decision making algorithms. This allows for better accessibility to cabs in high-traffic zones, which often exist in areas undergoing demand surges.**

## I. Introduction

Ride hailing services try their best to guarantee arrival times when using their cabs. But in some cases, such promises are hard to keep, especially during big events in crowded cities like San Francisco, where traffic jams and surge pricing together cause massive inconvenience to customers. Even if one was to put aside surge pricing, sometimes the cab cannot reach the customer due to traffic issues and has to eventually cancel. It has been found that the performance of Uber's matching algorithm plummets significantly during demand surges[1], and a major part of this is due to the uncertainty of a customer on whether he/she can get a ride or not[2].

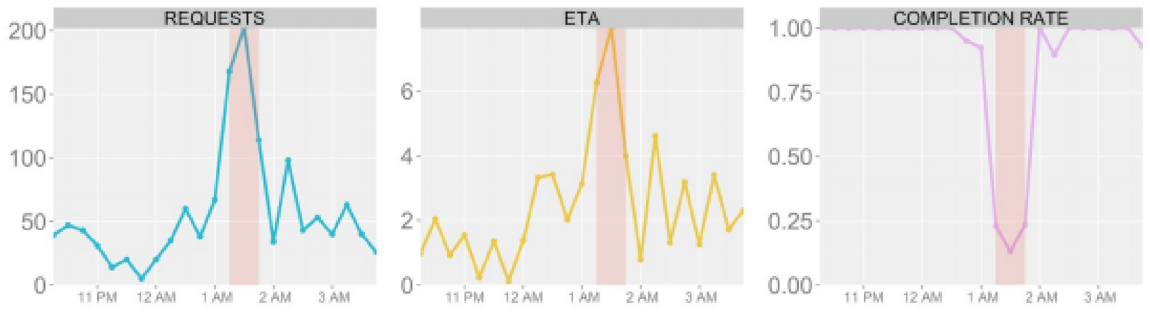

**Fig. 1 Study by Hall[1] showing the effect of demand spikes on ride completion rate. Surge pricing helps mitigate this, but is a purely economic solution that does not account for physical difficulties of heavy traffic**

The project aims at developing a method to improve the odds of a customer getting a cab in congested spaces, and subsequently making it to their destination on time in the presence of disruptive traffic and crowd. This is done by enabling that consumers can walk certain distances if told to by the app, which at present is not a common feature in most ride hailing services. Consumers can try going to less crowded parts of a city themselves, but this would involve using third party apps to try and figure out where to go (like Google Maps) and the lack of integration between cab route planning and customer movement would lead to uncertainty in outcome.

## II. Solution Approach

### A. Solution Philosophy

As described, the algorithm aims to find a method to reduce uncertainty of outcome in the ride. In this work, this is done by placing some "control" in the hands of the user. If the user gives information to the software about how much distance/time he/she is willing to walk, the app can incorporate that information while making it's routing decision and suggest a pickup point to the user, rather than the user choosing it. This feature's utility is extremely useful when users might be in crowded parts of a city like San Francisco or New York, with dense roads and not a lot of leeway room for

*MS Candidate, Aero/Astro, Stanford University cars in the case of high traffic.

The software can incorporate near-real time maps information around the user to suggest a route which the user can "walk" potentially taking him/her out of a congested zone the cab can reach easily. As mentioned before, it is better to have the user mobility and cab route-planning algorithm integrated into one rather than them working independently, which this work aims to accomplish.

This can hence be setup as an optimization problem, with the algorithm trying to minimize a certain metric while making the routing/planning decision for both the consumer and the cab. In the present work, this metric is set to be the total time of travel, i.e time to pickup plus time to drop off from pickup point.

## B. Mathematical Model

There are several aspects to the model in order to set this up as an optimization problem.

### *1. Grid*

Ideally a maps software could tell us (in near real time) a mean and variance in travel time for given road. In the absence of this, I have modelled a region around a customer as a "grid" of interconnected roads, each with their own mean and variance in travel time. There is a central disturbance somewhere on the grid contributing to demand surge and delays in traffic. Shown below is a demonstration of a grid setup, with the heatmap showing distribution of travel time across the grid:

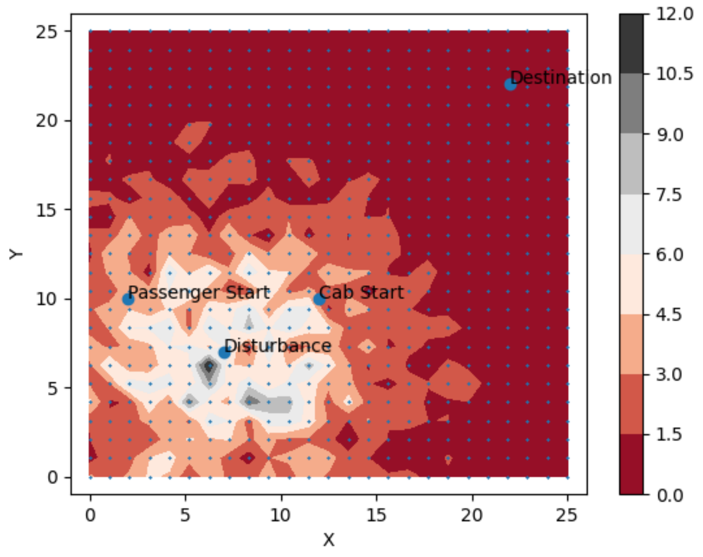

**Fig. 2 Heatmap showing the Upper Confidence Bound of travel times on the grid, and an example of the important points on the map**

This map is created a having an exponentially decaying mean and variance from the centre of the disturbance, and adding a random component to both of them. The idea is for it to be similar to a heavy traffic situation.

## C. Optimization Framework

To pose this as an optimization problem, one must recognize that this is a nested optimization problem. It can verbally be described as:

Outer Problem: Find the best pickup point according to the metric(total time of travel) given. In the current work, due to the potentially small size of the discrete feasible set, I solve this using random search, which is nearly exhaustive in nature. In order to scale this method, one can ideally use a particle based method in order to come up with the best pickup point.

Inner Problem: Given a pickup point, find the optimal route to take for a cab to minimize the metric, which is the total time of travel. This is again broken into two parts, namely time for cab to reach pickup point, and time to go from pickup point to destination. Both of them are essentially shortest path sub problems that are individually solved. These problems are solved using **Dijkstra's S.P algorithm**.

It is important to note, however, that for this (or the real time) application, we cannot really know that travel time on the road, since it is a stochastic quantity, particularly in the presence of disturbed traffic flow. In the optimization problem, we must optimize for known quantities in order to have a tractable objective function; this is hence done by optimizing the **Upper Confidence Bound**(UCB)[2] of the travel time. This ensures that the worst case scenario is being considered while planning (i.e finding the min of max time), to make sure the trip time would meet certain total time constraints if required. This formulation, makes a few assumptions:

1) The travel time on a particular road is normally distributed: the mean and variance of which is normally given by maps data, and for this model is given by an exponentially decaying radial function, centred at the disturbance point
2) The mean and variance of "consecutive" roads travelled on are independent of each other, to enable additive computation of normal distributions. This is a major assumption since it is likely that the travel times on consecutive roads are likely correlated.

Hence, mathematically, the problem can be written as:
Optimize:

$$t_{total} = (\sum_k \mu_k + \sigma_k)_{\text{(Cab to Pickup)}} + (\sum_k \mu_k + \sigma_k)_{\text{(Pickup to Dest.)}}$$

where $u_k, \sigma_k$ represent the UCB of travel time for road k. This is done by solving the outer optimization problem:

$$X_{pickup} = min_{t_{total}}\{(x, y) \in X_p\}$$

$$X_p = \{(x, y)|\|(x - x_{start}, y - y_{start})\|_2 < d_{constraint}\}$$

which depends on the inner optimization problem:

$$X_j = min_{t_{total,j}}\{(x_{i,j}, y_{i,j})|i = (1, 2...n), (x_{1,j}, x_{2,j}) \in X_p\}$$

Here $X_j$ is the set of trajectories for every feasible pickup point, $X_p$ is the set of feasible pickup points, $X_{pickup}$ is the optimally selected pickup point from the set of feasible points and $t_{total}$ is the objective function (UCB of travel time), and $(x_{i,j}, y_{i,j})$ is interpreted as coordinates of trajectory j corresponding to step i.

*1. Dijkstra's S.P Algorithm[3]*

Dijkstra's algorithm if a form of best first search, and is a greedy path finding algorithm between nodes of a graph, that greedily selects the next best node from a given point on the graph.

## III. Results and Discussion

### A. Results

In order to demonstrate the usefulness of the application, presented below are a few test cases. The plot on the left demonstrates the travel time and optimal path for the case where the user does not move at all. This is usually the case in today's ride hailing applications, where the user either waits where he/she is, or selects a pickup point, without knowing the traffic dynamics around him/her.
The plot on the right shows the path taken when user mobility and route planning are combined into one optimization problem.

**Chart of Travel Times for Cases Shown, in Order**

| Test Case | No Cust. Movement | Cust. Movement |
|---|---|---|
| Case 1 | 104.4 | 83.3 |
| Case 2 | 149.4 | 132.71 |
| Case 3 | 135.34 | 115.25 |

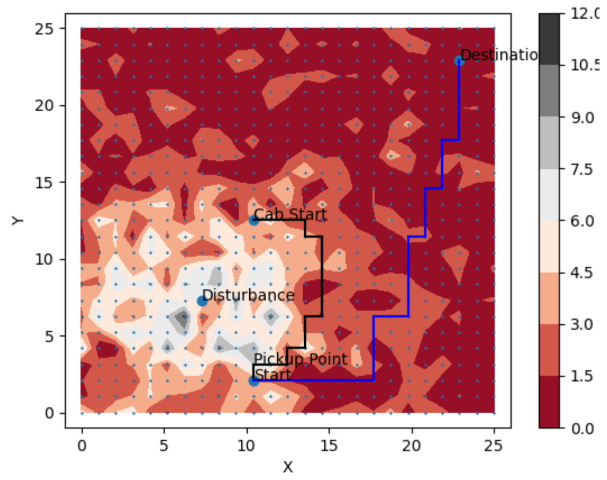

**Fig. 3 No customer movement, optimal trajectories planned (Case 1)**

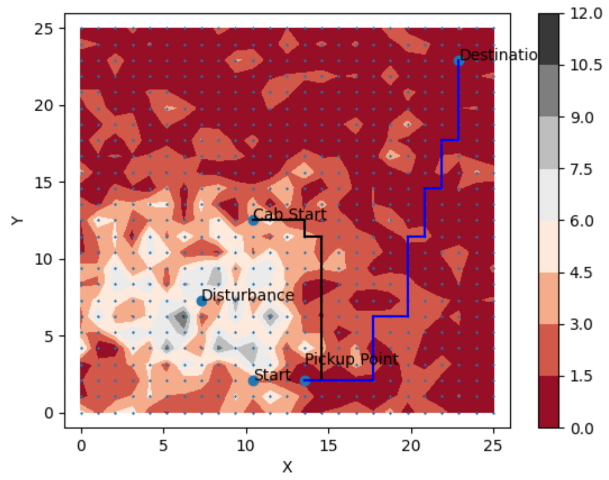

**Fig. 4 Customer movement-coupled trajectory planning(Case 1)**

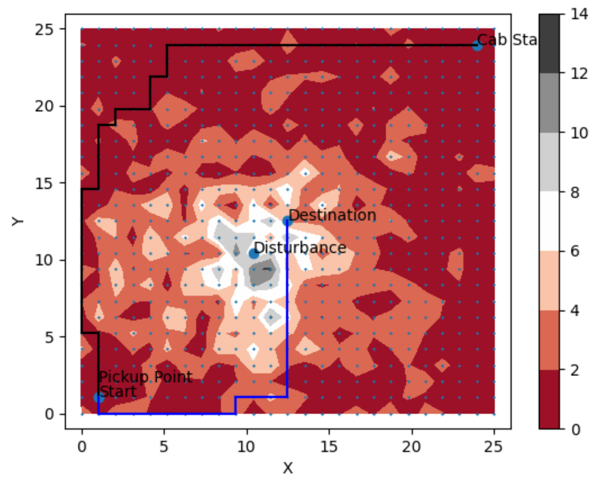

**Fig. 5 No customer movement, optimal trajectories planned (Case 2)**

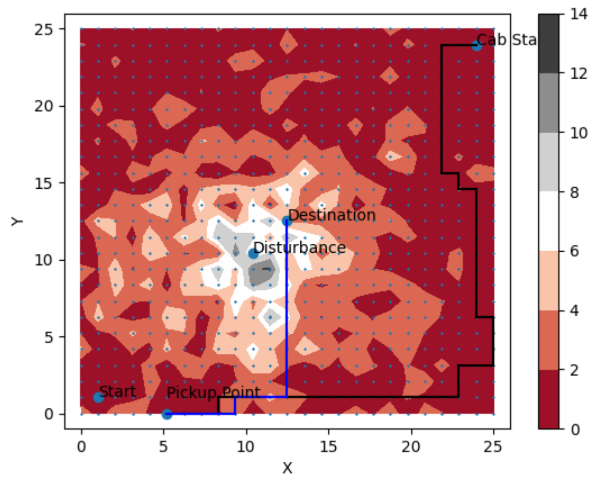

**Fig. 6 Customer movement-coupled trajectory planning(Case 2)**

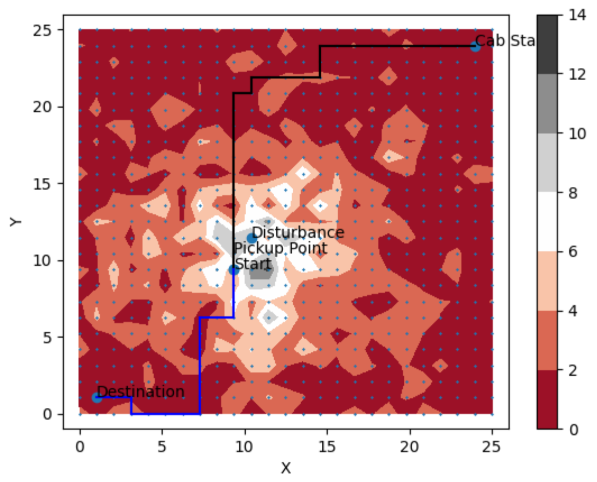

**Fig. 7 No customer movement, optimal trajectories planned (Case 3)**

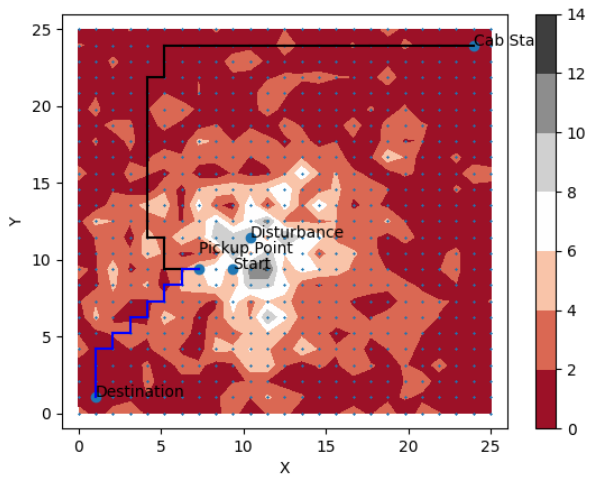

**Fig. 8 Customer movement-coupled trajectory planning(Case 3)**

## B. Discussion

In the plots shown above, the black path corresponds to the path taken by the cab going from it's start point to an "optimal" pickup point, and the blue line corresponds to going from the pickup point to the destination. This is the determined best path by the algorithm to take, to minimize the UCB of total travel time, from cab booking, to drop off of customer.These plots are, as described before, overlayed on a heat map showing the UCB of time taken on that part of the map.

The benefit of using coupled customer mobility and route planning can clearly be seen from the table of travel times: The planned routes when enabling customer mobility might differ significantly from that when mobility is not allowed. Some customer movements might not even be the most intuitive (as we can see in case 2, where the recommended movement is not in the same direction as destination).

## IV. Conclusions and Future Work

The work demonstrates the utility of coupling customer mobility and route planning in certain situations, by posing this as a discrete optimization problem and subsequently solving using Dijkstra's Shortest Path algorithm. It is intended as a complement to surge pricing strategies that ride hailing services usually follow, that enables them to instill more confidence in consumers while booking rides, that their demand will be met.

The algorithm in its present form comes with a set of limitations:

1) Open Loop Algorithm: One of the major drawbacks of the work currently is that the policy of travel is executed as planned in the beginning; this is essentially open loop and this policy might become sub-optimal during execution. Solving this issue is more complex than simply re-solving the problem at fixed time intervals, because the algorithm cannot keep asking the customer to move to a new point every few minutes, it is not convenient and not practical.
2) In real implementation, this work would use real maps data, which tends to be fairly accurate but also tends to be a few minutes behind the true state. This further increases the chances of calculated trajectories becoming sub-optimal during execution
3) One major assumption made during formulating the objective function is the additivity of Gaussians; this is possible only if the distributions they represent are independent of each other. This is not the case for, say adjacent road travel times; it is highly likely that a jam on one road has an effect on the adjacent road.
4) Finally, the grid-map travel times are modelled as Gaussians, which may be inappropriate by itself.

## References

[1] Hall, J., Kendrick, C., and Nosko, C., “The Effects of Uber’s Surge Pricing: A Case Study,” *Uber Research*, 2015.

[2] Kochenderfer, M. J., and Wheeler, T. A., *Algorithms for Optimization*, MIT Press, Cambridge, MA, 2018.

[3] Dijkstra, E., “A note on two problems in connexion with graphs,” *Numerische Mathematik*, 1959.