An Energy-Efficient Ride-Sharing Algorithm Using Distributed Convex Optimization

Shashvat Jayakrishnan Adhithyan Sakthivelu

Friday 13th May, 2022

Abstract

With a rising need for transporting people from one place to another especially within urban spaces, many mobility-on-demand (MoD) systems like Uber, Lyft and the like have burgeoned to provide swift and efficient mobilty services within just a few taps through our smartphone devices. Although carpooling, an inherently fuel-efficient concept, has penetrated into this market (e.g. Uber Pool, Lyft Line), these ride-sharing algorithms are presently not centered around dispatching vehicles in a manner that optimizes rider pick-up and drop so as to minimizes fuel use/energy consumption in that phase of travel as well. With the rising level of tailpipe CO2 emissions, it is prudent to focus on fuel efficiency to curb climate impact from warming.

In this project, our focus will be on optimizing the fuel/energy use as autonomous taxi agents make decentralized rider pick-up decisions based on various practical constraints, as part of localized clusters of connected cars. We intend to use distributed convex optimization running on each of these local clusters of networked vehicles, as the backbone for our ride-sharing algorithm that optimally picks-up and drops riders. This has an objective to minimize fuel used or energy consumed in all phases of travel, thereby making carpooling a completely package of energy-efficient travel.

1 Background

The field of ride-sharing algorithms in computer science has been explored from various perspectives in the literature. [CLHM04, BMM04] solve an offline carpooling problem based on pre-bookings. Whereas, [MZW13, MZW15, GPDR16] solve a realtime ridesharing problem. In [GPDR16], Ghosh et al they use a Network Lasso framework that deploys an Alternate Direction Method of Multipliers (ADMM) method to predict costs of optimal ridesharing. In [JSZW21], Javidi et al discuss a multi-objective ridesharing algorithm that maximizes the number of ride requests satisfied while minimizing the detour each vehicle has to take in order to pick up a new rider. Their work gives further details about the constraints used, specifically ensuring a minimum delay for driver-rider matching while using a genetic algorithm based approach to arrive at a near-optimal ridesharing algorithm.

2 Introduction

Unlike the existing literature on ride-sharing, our focus in this project is on energy-efficient ride-sharing, which means our ride-sharing optimization problem formulation optimizes carpooling by minimizing energy expended in the process of picking up and dropping extra riders. Ride-sharing can be thought of as a two part problem, one part being rider pick-up and the other being scheduled drops. The former part of the problem is a rather more tractable one wherein each vehicle makes a pick-up decision based on the energy expended by that vehicle in taking a detour and picking up the new rider. This energy is a function of the average speed (or the minimum of the maximum speed limit) along the pick up route and the distance overhead (detour distance) travelled by the vehicle to complete the pick-up. While making the pick-up decision, each vehicle also runs pseudo roll-outs to predict the updated drop-off schedule and times taking into consideration the new pick-up. This is a feasibility check to ensure that if the new passenger is picked up by a vehicle then that particular vehicle would be able to drop-off all its passengers given the drop-off time constraints of each passenger. Once feasible pick-up vehicles are determined, the vehicle that minimizes pick-up energy would be scheduled for picking-up the new passenger (*Figure 1*).

This leads us to solving the latter problem of scheduled drop-offs which can be reduced to an optimal scheduling problem corresponding to each vehicle. The problem is small scale because at most each vehicle would be capped to four riders. Scheduling the drops of at most four riders is computationally tractable. Interesting insights into solving this problem was obtained from different works in the literature that have solved scheduling problems for various applications, from sensor scheduling for state estimation in [LE15] to processor speed scheduling in [BV20]. Scheduling problems are typically NP-hard and fall under the paradigm of combinatorial optimization problems. We have explored literature on concepts like permutation spaces and minimizing functions defined on a domain of permutation spaces [GG91]. We define our drop-off energy functions over the space of permutations of drop-offs schedules for each vehicle and perform the minimum energy drop-offs subject to rider drop-off time constraints. An exponential urgency profile is defined for each rider that also dictates the drop-off urgency for riders who have been in the vehicle more than 30 minutes of their regular travel time (obtained from Google's Distance Matrix API given each rider's origin and destination).

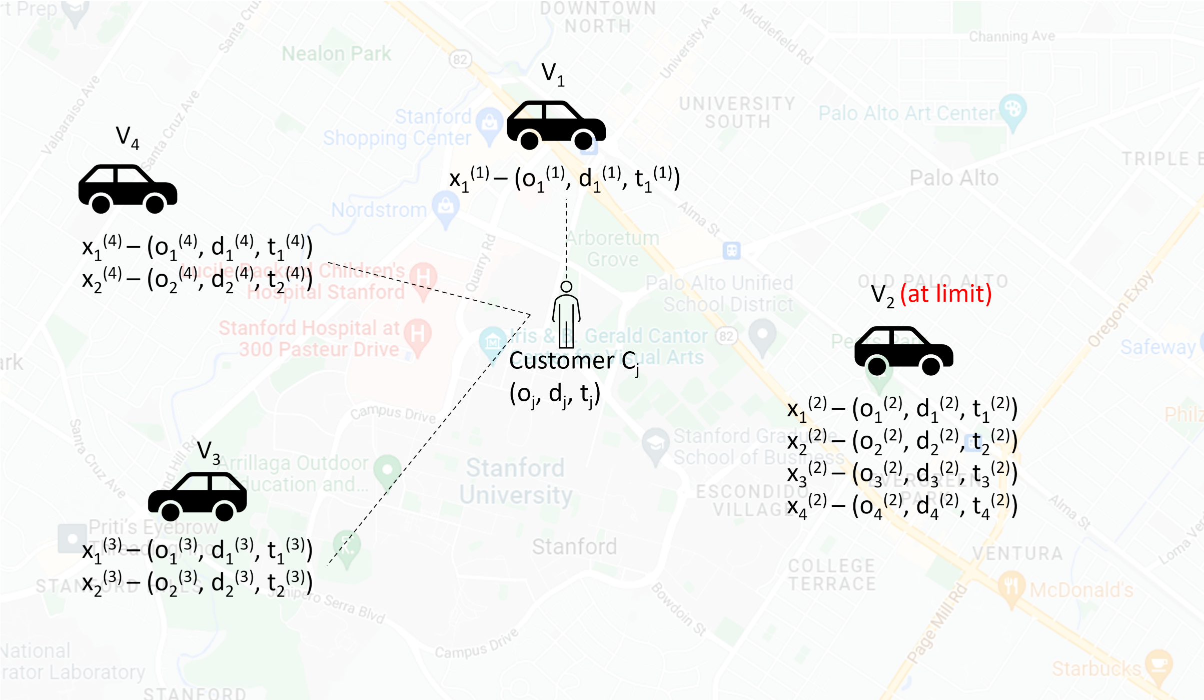
V1
x1(1) – (o1(1), d1(1), t1(1))
V4
x1(4) – (o1(4), d1(4), t1(4))
x2(4) – (o2(4), d2(4), t2(4))
V2 (at limit)
x1(2) – (o1(2), d1(2), t1(2))
x2(2) – (o2(2), d2(2), t2(2))
x3(2) – (o3(2), d3(2), t3(2))
x4(2) – (o4(2), d4(2), t4(2))
Customer Cj
(oj, dj, tj)
V3
x1(3) – (o1(3), d1(3), t1(3))
x2(3) – (o2(3), d2(3), t2(3))
DOWNTOWN NORTH
Nealon Park
Stanford Shopping Center
UNIVERSITY SOUTH
Palo Alto Art Center
Palo Alto
Nordstrom
Arboretum Grove
Palo Alto Unified School District
Stanford Hospital at 300 Pasteur Drive
Stanford University
Stanford
COLLEGE TERRACE
VENTURA
McDonald's
Starbucks
Safeway

Figure 1: The rider-vehicle matching problem

3 Mathematical Formulation

3.1 Rider Pickup Problem

Assume at a given moment (t_j (timestamp)) a customer (c_j) requests a ride at a starting location (o_j (Lat-Long)), by submitting information about the destination (d_j ((Lat-Long))). Also assume there are $\boldsymbol{n}$ cars in an x-minute vicinity of o_j that have a capacity to take one more customer while meeting the drop-off time constraint for every rider considering the new rider is picked-up.

Object c_j corresponds to the new customer and contains the parameters (o_j, d_j, t_j). Let y_i be a Boolean variable that denotes whether car V_i is assigned to the new customer or not. It is obvious that the customer would only be matched with one vehicle, i.e., $y_i \in \{e_i\}$ where e_i is a standard normal basis vector $\in \mathbf{R}^n$. So, we can formulate this into a constraint $\sum_{i=1}^{n} y_i = 1$, where $y_i \in \{0, 1\}$.

For each vehicle V_i, assuming passenger capacity $m_{cap}^{(i)}$, there would be customers already seated who need to be dropped. Let $x_k^{(i)}$, be the kth existing customer in the vehicle

(V_i) object that maps to the parameters $(o_k^{(i)}, d_k^{(i)}, t_k^{(i)}),\ k = 1, \cdots, m^{(i)},$ where $m^{(i)} \leq m_{cap}^{(i)} - 1$ is the number of riders already seated in vehicle V_i. In our project, we set $m_{cap}^{(i)} = 4$ for all vehicles, however in reality, this could be a different value for each vehicle.

Whenever a new customer c_j pops up, each vehicle in the x-minute vicinity of the customer runs pseudo roll-outs to evaluate the optimal drop-off order and drop-off feasibility had that vehicle made a pick-up decision for the new customer. Basically, each vehicle summons the scheduled drop-off optimization in consequence of prospectively picking-up the new customer. This optimization problem is discussed in the next subsection, however, for now let us assume that running this optimization yields us information about whether a particular vehicle is a feasible choice or not for the pick-up. V is the set of feasible pick-up vehicles (feasibility means they have space to take one more passenger and they can still drop-off all passengers in time if they pick-up the new passenger). $V = \{V_i \mid \forall\ i = 1, \cdots, n\}$, as defined in the start of this section.

For each vehicle V_i we run a passenger pick-up optimization to select a vehicle that minimizes the energy required to pick-up the new customer. The optimization problem is given as follows:

$$\begin{array}{ll} \text{minimize} & y^T E = \sum_{i=1}^{n} y_i E_i(v_j, s_j) \\ \text{subject to} & \sum_{i=1}^{n} y_i = 1 \text{ (i.e. } y_i \in \{e_i\}\ \forall\ i = 1, \cdots, n) \end{array}$$

Where y_i is as defined above and $E_i(v_j, s_j) = (av_j^2 + bv_j + c)s_j + k$, is the fuel consumption function for vehicle V_i given the speed v_j (which may be the average speed of the vehicle along the pick-up route or the minimum of the maximum speed limits along the pick-up route) and the detour distance s_j corresponding to picking-up the new customer c_j. The terms, a, b, c are constants that are specific to the vehicle's fuel efficiency model, while k is the minimum fuel cost associated with using a vehicle. If more vehicles are used, this factor will contribute more to the optimization cost. Therefore, it can be thought of as a parameter that incentivizes ride-sharing. This is a combinatorial optimization problem where we choose a vehicle with y_i if it yields the minimum pick-up energy (or fuel consumption).

3.2 Scheduled Drop-Off Problem

Given a vehicle V_i that contains the customers $(x_k^{(i)}), \{k = 1, \cdots, m^{(i)}\}$ we need to find the drop order that minimizes the total energy while meeting the pickup and drop constraints. Because the number of customers in a vehicle is at most 4, the problem is tractable.

In a vehicle V_i, consider all the possible permutations of customers drop off order. We denote the permutations as the variable $\tilde{x}_l^{(i)},\ l = 1, \cdots, (m^{(i)})!$, where $m^{(i)}$ is the number of customers in the vehicle V_i. We reject the permutations where the time constraint of any customer is exceeded. If every permutation is rejected, then the vehicle is considered infeasible. For all the feasible vehicles and feasible drop off combinations, we then compute the energy expenditure. Which is a function of speed and distance as the formula by Davison et al [DBBK+20].

We use the Google Maps routes API to calculate the average speed and the distance for each drop-off way-point, which is then used to measure the energy expenditure of that combination. Because we know that the energy is a function of the velocity and distance.

Therefore, the total energy consumed would be $\sum_{m=1}^{m^{(i)}} E_m(\tilde{v}, \tilde{s})$, where $\tilde{v}$ and $\tilde{s}$ is the average speed and distance between the drop-offs of customers with the give permutation.

For example, consider three passengers x_1, x_2, x_3 in a vehicle. Let the drop-off schedule be as follows: $x_2 \rightarrow x_1 \rightarrow x_3$. The total energy of this drop-off permutation is given by $E(\tilde{v}_2, \tilde{s}_2) + E(\tilde{v}_1, \tilde{s}_1) + E(\tilde{v}_3, \tilde{s}_3)$, where $\tilde{v}_2$ and $\tilde{s}_2$ are the average speed and distance (from Google's Distance Matrix API) between the vehicle's current location (l) and x_2's drop-off location (d_2) respectively, $\tilde{v}_1$ and $\tilde{s}_1$ is the average speed and distance between x_2's drop off location (d_2) and x_1's drop off location (d_1), and $\tilde{v}_3$ and $\tilde{s}_3$ is the average speed and distance between x_1's drop off location (d_1) and x_3's drop off location (d_3). So, the total energy would be $E(\tilde{v}_2, \|l - d_2\|) + E(\tilde{v}_1, \|d_2 - d_1\|) + E(\tilde{v}_3, \|d_1 - d_3\|)$, where l may be the current location of the vehicle. The distance metric $\|.\|$ is as returned by the API call. This total energy corresponds to just one permutation. Our problem is to minimize the total drop-off energy while searching through the permutation space of drop-off schedules. Therefore, it is once again a combinatorial optimization that returns the energy-optimal drop-off schedule for every vehicle.

The master problem is to minimize the system-wide energy use. The sub-problem of finding the optimal drop-off schedule for the customers $(x_k^{(i)}, c_j)\ \forall\ k = 1, ..., m^{(i)}$ for the vehicle V_i and the total energy associated with it is solved for every vehicle $V_i,\ i = 1, \cdots, n$. This is all done in parallel and the final energy expenditure is sent back to the master problem. This would make sure the system wide energy is reduced as well as a feasible vehicle is chosen for every new customer c_j.

References

[BMM04] Roberto Baldacci, Vittorio Maniezzo, and Aristide Mingozzi. An exact method for the car pooling problem based on lagrangean column generation. *Operations Research*, 52(3):422–439, 2004.

[BV20] Stephen Boyd and Lieven Vandenberghe. Convex optimization examples. `https://web.stanford.edu/class/ee364a/lectures/examples.pdf`, 2020.

[CLHM04] Roberto Wolfler Calvo, Fabio de Luigi, Palle Haastrup, and Vittorio Maniezzo. A distributed geographic information system for the daily car pooling problem. *Comput. Oper. Res.*, 31:2263–2278, 2004.

[DBBK$^+$20] Jack Davison, Yoann Bernard, Jens Borken-Kleefeld, Naomi J. Farren, Stefan Hausberger, Åke Sjödin, James E. Tate, Adam R. Vaughan, and David C. Carslaw. Distance-based emission factors from vehicle emission remote sensing measurements. *Science of The Total Environment*, 739:139688, 2020.

[GG91] Dmitri Golenko-Ginzburg. Metrics in the permutation space. *Applied Mathematics Letters*, 4(2):5–7, 1991.

[GPDR16] Shaona Ghosh, Kevin Page, and David De Roure. An application of network lasso optimization for ride sharing prediction. *arXiv preprint arXiv:1606.03276*, 2016.

[JSZW21] Hamed Javidi, Dan Simon, Ling Zhu, and Yan Wang. A multi-objective optimization framework for online ridesharing systems. In *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 252–259. IEEE, 2021.

[LE15] Chong Li and Nicola Elia. Stochastic sensor scheduling via distributed convex optimization. *Automatica*, 58:173–182, 2015.

[MZW13] Shuo Ma, Yu Zheng, and Ouri Wolfson. T-share: A large-scale dynamic taxi ridesharing service. In *Proceedings of The 29th IEEE International Conference on Data Engineering.* ICDE 2013, April 2013.

[MZW15] Shuo Ma, Yu Zheng, and Ouri Wolfson. Real-time city-scale taxi ridesharing. *IEEE Transactions on Knowledge and Data Engineering*, 27(7):1782–1795, 2015.