A Quantitative Comparison of Solo and Shared Ride

Sachin K Salim
under the guidance of Swaprava Nath
Department of Computer Science and Engineering, IIT Kanpur

February 2, 2018

Abstract

The motive of this paper is to quantify the improvement achieved in terms of total distance covered by a cab when a group of passengers in a city choose to share ride instead of riding solo. We call the ratio of the length of the shortest possible route in solo ride to that of shared ride as the *Environmental Improvement Factor* (EIF), which we want to minimize. However, we also need to ensure that the suggestion is *individually rational* for the passenger, i.e., this gives her at least the same utility as a solo ride. We then discuss about some approximation algorithms for the solo riding problem and attempts to do the same for shared riding.

1 Introduction

Environmental degradation and its protection has become a hot topic in the last few decades. The people and their government began to acknowledge the impact human activity is causing on the environment and policies and actions have been coming up to reduce the emission of greenhouse gases and other toxic substances to the atmosphere.

As the technology progresses and buying power of population increases over years, more and more automobiles are seen on the road. People tend to avoid public transports and prefer private vehicles. Taxi-cabs constitute a major share of automobiles especially in urban areas where pollution is prominent. In India, it's observed that people prefer travelling alone rather than pooling their ride even though the latter offers cheaper fare. This could be attributed to a variety of reasons including comfort, waiting time and security.

It's intuitive that when more and more people prefer to pool their rides, the total distance travelled by all the cars in that locality reduces, thereby saving fuel and making a lesser impact on environment. In this paper, we make an attempt to quantify this improvement by modelling the locations in a city on a graph comparing the expected lengths of shortest routes when all passengers travel solo to that of shared ride.

2 Literature Review

In a paper by Furuhuta et. al, the authors discuss about the history of ridesharing and classify it based on the target market, service type, matching activity, pricing policy, etc. Ma et. al, proposed a new pricing mechanism called *Spatio-Temporal Pricing* that is smooth in space and time. This ensures that drivers will not reject the trip dispatched to them and keeps them from flocking in a particular area for higher prices. Almost the entire literature in ridesharing is focused on optimizing the ride for passengers and drivers. Here, our focus is on matching the passengers to a driver such that there is least impact on the environment.

3 Problem statement

Consider a set of locations $\mathcal{L}$ that represents the possible locations in a city, and serve as potential pickup and drop-off points of every passenger. Given a specific demand, i.e., the sources and destinations of k passengers, we want to solve the constrained optimization problem that minimizes the total cost of travel subject to the individual rationality constraint, i.e., the cost of traveling via the proposed algorithm is at most as expensive as a solo ride – and holds for every passenger.

The transportation network of a city is modeled as an undirected graph $G = (V, E)$, where the costs are identical for any directional traversal. The function δ gives the costs of the edges of the graph. Passenger $i \in \mathcal{P}$, the set of passengers, reports her demand (p_i, d_i) where p_i is the pickup location and d_i is the drop-off location. Driver $\mathcal{D}$ has an initial location $o_{\mathcal{D}} \in \mathcal{L}$.

Solo ride An order $(\pi_1, \ldots, \pi_k)$ is decided over these k passengers and the driver $\mathcal{D}$ picks up each of these passengers and then drops them off before picking another passenger in the order given by π. The problem here is what order of the passengers should be followed so that the total length of the travel route is minimized.

Shared ride All permutations of $(p_1, d_1, \ldots, p_k, d_k)$ are considered such that p_i comes before d_i for all $i \in \mathcal{P}$ and the occupied passengers do not cross the capacity of cab when the passengers are served in the given order. Again, the problem here is to find such a path that minimizes the cost of the total travel.

The problem in hand is to compare the costs of the following two cases: (a) the minimum cost path in solo-riding, and (b) the minimum cost path in shared-riding. Are these problems NP-hard? If so, what is a good approximation scheme (particularly for special types of graphs, say grid or a weakly connected cliques or star)? The analysis has to take care of the fact that the worst case analysis is done on same instances.

3.1 Ridesharing problem (RSP)

Given a city with a list of passengers having a pickup point and a drop-off point, what is the shortest route of a driver to pick up all passengers and drop them

off given the driver's current location?

Notations

- $\mathcal{P} = 1, 2, \ldots, k$: Set of passengers in ridesharing.
- $\mathcal{L} \subset \mathbb{N} \times \mathbb{N}$, Set of locations.
- $p_i, d_i \in \mathcal{L}$: pickup and drop-off locations of passenger i
- $\delta : \mathcal{L} \times \mathcal{L} \to \mathbb{N}$ where $\delta(a, b)$ gives the distance between locations a and b. For simplicity we take δ to be the manhattan distance.
- $\mathcal{D}$: the driver.
- $o_{\mathcal{D}} \in \mathcal{L}$: origin of the driver $\mathcal{D}$, i.e., the initial location.
- $\mathcal{C}$: Capacity of $\mathcal{D}$'s vehicle.
- $\mathcal{R}$: Set of feasible routes of $\mathcal{D}$.
- $r \in \mathcal{R}$: A sequence of locations in $\mathcal{L}$

A route r is feasible iff it satisfies the following three conditions:

1. The route r covers all the passengers' pickup and drop-off point. Thus $|r| = 2|\mathcal{P}|$.
2. The pickup point of a passenger i comes before the drop-off point for all $i \in \mathcal{P}$.
3. At any location $a \in r$, number of pickups until a must not be more than the sum of number of drop-offs until a and the capacity $\mathcal{C}$.

Objective Our objective is to find the shortest length route $r^* \in \mathcal{R}$ such that

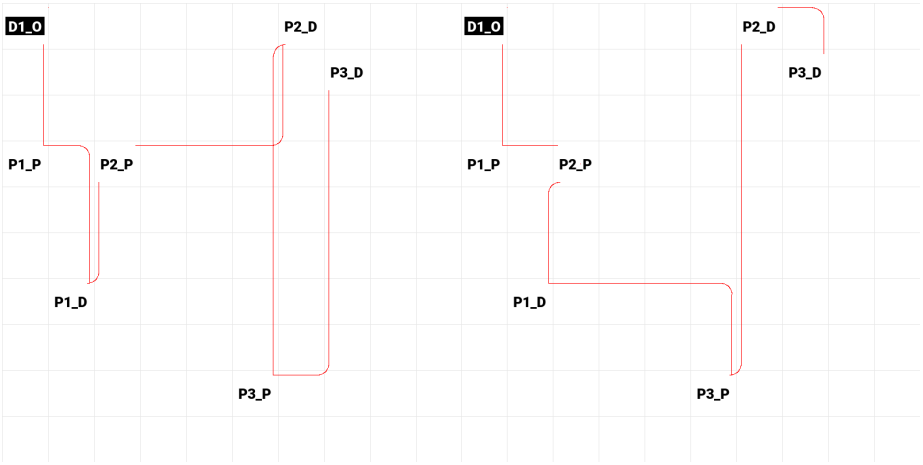
$$r^* = \underset{r \in \mathcal{R}}{\operatorname{argmin}} \sum_{i=1}^{|r|-1} \delta(r_i, r_{i+1})$$

where r_i is the i^{th} location in the route r.

Solo riding problem When the driver is required to drop-off each passenger immediately after she is picked up, it is an instance of solo-riding problem. In other words, it can be stated as a subclass of shared riding problem with the vehicle's capacity to be one.

Shared riding problem When the capacity of vehicle is a small finite natural number, it is a shared-riding problem. In this case, a driver can pick-up multiple passengers before dropping them off.

We would prove here that both ridesharing problems are NP-hard problems in combinatorial optimization by showing it is a generalization of the Travelling salesman problem (TSP) which is already proven to be NP-hard. Then we'll show a dynamic programming approach to compute the optimal solution followed by a 2-approximate algorithm.



(a) (b)

Figure 1: Left: A sample solo-ride route (Distance = 36)
Right: A sample shared-ride route (Distance = 26)

4 Proving NP-hardness

4.1 Open Travelling Salesman Problem (OTSP)

OTSP is a variation of TSP in which the salesman does not return to the starting point. The problem is to find out the shortest Hamiltonian path in a graph G starting from a given vertex.

4.1.1 Formal definition of OTSP

Given a complete graph $G = (V, E)$ with positive weighted edges and a starting vertex $s \in V$, determine if there exists a Hamiltonian path whose sum of weights of edges in the path is not more than a given number k.

4.1.2 NP-hardness of OTSP

We'll first show that OTSP is in NP. Given a hamiltonian path in G, we just need to sum the weights to determine if it's less than k. This could be done in $O(|V|)$ time and is thus polynomial in complexity.

Now we'll prove it is NP-hard. For this, we'll use the knowledge that the Hamiltonian path problem (to determine if a Hamiltonian path exists in a given graph with a fixed starting vertex) is NP-hard.

Takes as input a graph G which is an instance of Hamiltonian path Problem. We now construct a graph G' by the reduction of HP to OTSP as follows:

1. Create a complete graph $G'(V', E')$ with $V' = V$, $s' = s$ and $E' = \{(i, j) : i, j \in V,\ i \neq j\}$.

2. Define the weight function $w : E \to \mathbb{N}$ as:

$$w(e) = \begin{cases} 0 & \text{if } e \in E \\ 1 & \text{if } e \notin E \end{cases}$$

3. Pass the instance $(G', 0)$ as input to the OTSP problem.

If a Hamiltonian path exists in G starting from s, then the cost of each corresponding edge in G' would be 0 and thus total cost of HP would be 0. If no such HP exists in G, then the cost of any HP in G' must be more than 0. So G has a Hamiltonian path starting at s iff G' has a Hamiltonian path of cost at most 0 starting at s' . Hence, OTSP is NP-hard.

4.2 NP-hardness of RSP

solo-RSP We are given a graph $G(V, E)$, an instance of OTSP problem and we show here how to convert this to an instance of solo-RSP.

- Define a bijective map $\psi : V \to \mathcal{L}$, from vertex set V to locations L.
- Construct a driver $\mathcal{D}$ with origin as $o_{\mathcal{D}} := \psi(s)$.
- Construct $|V| - 1$ passengers with each passenger's pickup and drop-off location, $p_i = d_i = \psi(v_i), v_i \in V/s$.
- Define $\delta(a, b) := w(\psi^{-1}(a), \psi^{-1}(b))$

It is evident from the above definition that an HP with length $\leq$ k starting at s exists in OTSP instance iff a route exists in the translated solo-RSP instance with length $\leq$ k and driver's origin at $o_{\mathcal{D}}$. Hence, solo-RSP is NP-hard.

shared-RSP Shared RSP is a superclass of solo-RSP problem as it deals with capacity of cab with any finite natural number. Since the instances over which the minimum is to be found out in shared-RSP is a superset of that of solo-RSP, it's intuitive that shared-RSP is also NP-hard. To solve the solo-RSP problem, it could call the shared RSP problem with capacity as 1. Since we know that there exists no polynomial time algorithm for solo-RSP, it can be seen from here no such polynomial time algorithm can exist for shared RSP.

5 Optimal solution algorithm

Let's discuss the DP approach [U12] to solve OTSP problem. This algorithm runs in $O(n^2 * 2^n)$ time where n is the number of vertices of the graph G.

Algorithm
$\mathcal{V}$ = set of vertices;
s = starting vertex;
$w(a, b)$ = weight of the edge (a, b);
for $\mathcal{S} \subseteq \mathcal{V}$ *with* $|\mathcal{S}| = 2$ *and* $s \in S$ **do**
  $C(\mathcal{S}, i) = w(s, i)$;
end
for $k \leftarrow 2$ **to** $n - 1$ **do**
  for $\mathcal{S} \subseteq \mathcal{V}$ *with* $|\mathcal{S}| = k$ *and* $s \in S$ **do**
    for $i \in \mathcal{S} \setminus \{s\}$ **do**
      $C(\mathcal{S}, i) = \min\{C(\mathcal{S} \setminus \{i\}, j) + w(j, i)\} \quad \forall j \in \mathcal{S} \setminus \{i, s\}$;
    end
  end
end
$C(\mathcal{V}) = \infty$;
for $i \in V \setminus \{s\}$ **do**
  $C(\mathcal{V}, i) = \min\{C(\mathcal{V} \setminus \{i\}, j) + w(j, i)\} \quad \forall j \in \mathcal{V} \setminus \{i, s\}$;
  $C(\mathcal{V}) = \min(C(\mathcal{V}), C(\mathcal{V}, i))$;
end
Result: $C(\mathcal{V})$

5.1 Pricing

The fare of passenger $i \in \mathcal{P}$ is given by Shapley value [S88] defined as follows:

$$Sh_i(\mathcal{P}, v) = \frac{\alpha}{n!} \sum_{S \subseteq \mathcal{P} \setminus \{i\}} |S|! \; (|\mathcal{P}| - |S| - 1)! \; \Big(v(S \cup \{i\}) - v(S) \Big)$$

where α is the fare per unit distance and $v(S)$ for any $S \subseteq \mathcal{P}$ is the length of the shortest route to serve all the passengers in S and none of the passengers not in S.

Since the total distance distance travelled in a shared ride is always at most the total distance distance travelled in solo ride for a given set of passenger locations, the Shapley value fare ensures individual rationality by distributing the total fare among the passengers in such a way that no passenger pays more for solo ride than shared ride.

6 Approximate algorithms

We'll show here an approximate algorithm to OTSP. Using the solution to this algorithm, we could compute the solution to the **solo RSP** problem by translating this solution.

6.1 Approximate algorithm using MST

We describe a method [A14] to get a 2-approximate solution to solo-ride ridesharing problem.

Algorithm

1. Construct Minimum Spanning Tree T of G with $o_{\mathcal{S}}$ as root using Prim's Algorithm.
2. Let $\mathcal{W}$ be the pre-order walk in T. A pre-order walk is a sequence of vertices visited in the order of DFS of T along with its return. See fig: 2 for an example.
3. Return the Hamiltonian tour $\mathcal{R}_{apx}$ visited in the order of the above pre-order walk.

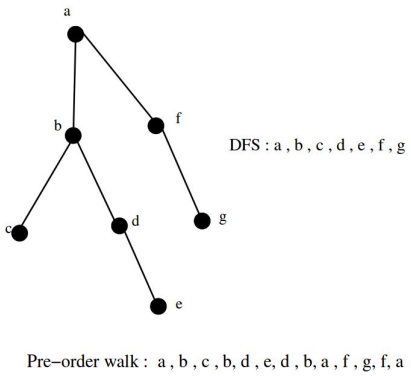


Figure 2: Example: Pre-order walk (credits: [A14])

Proof The above algorithm runs in polynomial time since Prim's Algorithm has a polynomial complexity.

Let $\mathcal{R}_{opt}$ be the shortest Hamiltonian tour in G. Let $C(T)$ is the sum of the weights of the edges in T. Since $\mathcal{T}$ is the MST,

$$C(T) \leq C(\mathcal{R}_{opt})$$

since $\mathcal{R}_{opt}$ is also a spanning tree.

In the pre-order walk $\mathcal{W}$, every edge is visited twice except the final few edges since it doesn't come back to the starting point,

$$c(\mathcal{W}) \leq 2 * C(T)$$

. Combining both the inequalities above, we get,

$$c(\mathcal{W}) \leq 2 * C(\mathcal{R}_{opt})$$

Now, observe that $\mathcal{R}$ is effectively a 'short-cut' of $\mathcal{W}$ since no vertices are visited twice in $\mathcal{R}$ unlike $\mathcal{W}$ because the vertices are visited directly without tracing the route back. Under the assumption of triangle inequality, this would give

$$c(\mathcal{R}_{apx}) \leq C(\mathcal{W})$$

and thus we have proved that

$$c(\mathcal{R}_{apx}) \leq 2 * C(\mathcal{R}_{opt})$$

Note: If the triangle inequality doesn't hold in G, we can return $\mathcal{W}$ instead of $\mathcal{R}_{apx}$ since our original problem doesn't prevent us from visiting the same vertex twice.

6.2 A probabilistic approach

Ant Colony Optimization This is a probabilistic technique to compute close-to-optimal path in a graph, originally proposed by Marco Dorigo[DB92] in 1992. This technique is inspired from the way the ants move in search of food leaving behind a chemical called pheromones in the trail.

We can apply this technique to solve both the ridesharing problems. We can increase the number of iterations to get a more and more closer solution to the optimal solution.

7 Simulation results

The minimum total distance was calculated for a 1000X1000 grid for 1 driver and passengers varying from 1 to 5. The locations of driver and passengers were generated at random and the average of shortest distances over 1 lakh iterations were calculated. The below plots show these results.

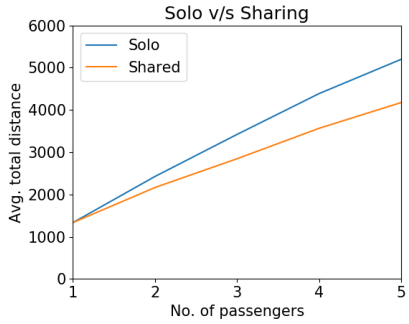


Figure 3: Avg. total distance for solo and shared ride

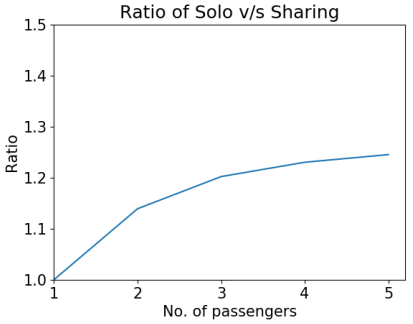


Figure 4: Ratio of avg. total distance for solo and shared ride

As we could see from the ratio plot, the ratio

$$r = \frac{\text{length of shortest solo-ride route}}{\text{length of shortest shared-ride route}} \simeq 1.25$$

for one driver and 5 passengers. We can see from the plot that this value is converging to a ratio $\simeq 1.25$

8 Future directions

The exponential time complexity of the optimal solution algorithm is a hindrance in implementing this algorithm in real case scenario. The NP-hardness of shared-riding wasn't proven formally in this paper and that's one of the future work. An approximation algorithm for shared riding is to be found out such that the deviation from this approximate solution from optimal solution is correlated with that of solo-ride. This will enable us to give a ratio of them for larger number of passengers.

References

[MPF17] Hongyao Ma, David C. Parkes and Fei Fang "Spatio-Temporal Pricing for Ridesharing Platforms"

[FDO13] M Furuhata, M Dessouky, F Ordonez, ME Brunet, X Wang, S Koenig "Ridesharing: the State-of-the-art and Future Directions"

[S88] Lloyd S Shapley, Chapter 2, "The Shapley value", Cambridge University Press, 1988

[A14] Arash Rafiey "Approximation Algorithms (Travelling Salesman Problem)" (URL: `http://www.sfu.ca/~A14/lecture25.pdf`)

[U12] Umesh Vazirani "Dynamic Programming" (URL: `https://people.eecs.berkeley.edu/~vazirani/algorithms/chap6.pdf`)

[DB92] M Dorigo C Blum "Ant colony optimization theory: A survey"