

Ride-Sharing: A Multi Source-Destination Path Planning Approach

Jamal Yousaf¹, Juanzi Li¹, Lu Chen², Jie Tang¹, Xiaowen Dai², and John Du²

¹ Department of Computer Science and Technology,
Tsinghua National Laboratory for Information Science and Technology, Tsinghua
University, Beijing, 100084, China

² General Motors, China Science Lab, Shanghai
{jamalyousaf,ljz,jietang}@keg.tsinghua.edu.cn,
{lu.chen,xiaowen.dai,john.du}@gm.com

Abstract. Ride-sharing is considered as one of the promising solutions for reducing fuel consumption of fuel and reducing the congestion in urban cities, hence reducing the environmental pollution. With the advancement of mobile social networking technologies, it is necessary to reconsider the principles and desired characteristics of ride-sharing systems. Ride-sharing systems can be popular among people if we can provide more flexible and adaptive solution according to preferences of the participants and solve the social challenges. In this paper, we focus on encouraging people to use a ride-sharing system by satisfying their demands in terms of safety, privacy, convenience and also provide enough incentives for drivers and riders. We formalized the ride sharing problem as a multi source-destination path planning problem. An objective function is developed which models different conflicting objectives in a unified framework. We provide the flexibility to each driver that he can generate the sub-optimal paths according to his own requirements by suitably adjusting the weights. These sub-optimal paths are generated in an order of priority (optimality). The simulation results have shown that the system has the potential to compute multiple optimal paths.

Keywords: Ride-sharing, Path Planning, Dynamic Optimization.

1 Introduction

We are entering “a new era” enabled with new smart phone applications and more intuitive software that will help us find efficient and easy-to-use shared rides. Technology thus plays a major role in overcoming carpool barriers such as perceived ease-of-use and flexibility. As our society adapts to these new tools, our perception of carpooling will change as well. Apparently, there is plenty of room for improvements, for example, when considering the fact that on average cars carry 1.58 passengers, i.e., only an estimated 25% of emissions are caused by people traveling, and the rest by moving empty seats [6]. The growing ubiquity of mobile Internet technology has created new opportunities to bring together

people with similar itineraries and time schedules to share rides on short-notice. Encouraging ride-share is, after all, an easy and cost-effective solution to helping reduce congestion and gas emissions.

Internet-enabled smart-phones allow people to offer and request trips whenever they want wherever they are. By dynamic ride-sharing, we refer to a system where an automated process employed by a ride-share provider matches up drivers and riders on very short notice [5], which can range from a few minutes to a few hours before departure time. Dynamic ride-sharing systems are developed to publicize formal ride-sharing in urban areas. Most of the current ride-sharing systems are covering formal requirements of users. Safety, security, flexibility, enough incentives for drivers and riders are the critical issues for users and it is also necessary to understand that riding and meeting with people are often related activities. Gathering enough participants is also major concern for web based and mobile ride-sharing systems. This paper addresses the following questions.

- Q1)** *How can we provide more flexibility to drivers in finalizing the path based on his own preferences?*
- Q2)** *Can we use social networking features like social strength or closeness in the path computation?*
- Q3)** *Could we provide a ride-sharing system that encourages the people and satisfying their demands in terms of safety, flexibility, privacy and convenience?*

In our approach for ride-sharing system, we particularly focus on encouraging people to use a ride-sharing system by satisfying their demands addressed above. We provide the flexibility to each driver that he can generate the sub-optimal paths according to his own requirements by suitably adjusting the weights. An objective function is developed which models different conflicting objectives in a unified framework, different objectives can be assigned with different weights.

With a GPS-enabled phone, a user can select his current location as the origin of the trip. For security purpose, the system will generate an auto pin number for the riders and also provides an option to see the complete description of the vehicle and the proposed path before finalizing the trip. The system allows the riders to rate the driver and the ride including the vehicle (zero to five stars), and optionally a comment just after the ride. Such ratings are converted into the feedback score. In addition, the driver could also rate participants by monitoring cancellations, no-shows and late arrivals. Such ratings are converted into a reliability score. Both the reliability and feedback scores are computed and recorded by the ride-sharing system and have great importance on the trust building of the system. If the driver and the rider both agree on the proposed arrangement, the driver picks up the rider at the agreed time and location.

The rest of the paper is structured as follows. In Section 2, we will introduce the related work. In Section 3 and 4, we will formulate the ride-sharing problem and discusses the route matching factors and finally the cost function is developed. In Section 5, we explain our approach to solve the dynamic ride-share problem and the simulation results are discussed in Section 6. Finally, in Section 7, we summarize our main insights in the form of conclusion.

2 Related Work

Many solutions concerning to the Ride-sharing System have been proposed in the literature and many of them were implemented and are used on the Web. In this section we will summarize the related work in cotext of ride sharing systems and the path planning.

Ride Sharing System: Real-time ride-sharing is becoming a new, mainstream mode of transport and a lot of work has already been done in this regard. In a recent work, Trasarti et al. [2] introduces a methodology for extracting mobility profiles of individuals from raw digital GPS traces. In his experiments, the impact of car pooling application based on profile matching is measured and showed as it may foster an intelligent car pooling service. In another work Kleiner et al. [3] presents an incentive compatible dynamic ride-sharing solution based on auctions. It also provides tradeoff the minimization of vehicle kilometers travelled with overall probability of successful ride-share.

Resnick [16] suggests to drop the concept of a few standardized pickup and drop off locations and replace them with dynamically chosen collecting points regarding the current position of the rider and driver. Kelley [11] introduces an automatic system by which needed information is retrieved by special RFID tags. Smart Jitney [12] is a slightly different system design approach. Murphy et al. suggests to equip every vehicle with identifiable and trackable hardware. Just as a cab, this device can be contacted to reserve a seat.

SafeRide [9] is a draft of a dynamic ride-sharing system for modern Smart-phones. Morris proposes to use existing Web technologies such as Google Transit and combine them in order to facilitate user interaction. Further, he recommends the riders to rate the driver (and vice versa) in an also included eBay-like reputation system. With Ride Now! Kirshner [8] introduces a testable computer- and telephone-based dynamic ride-sharing system. The system finds a feasible driver for this request and informs the rider about the drivers contact details. Vanoutrive et al. [1] find that the most carpool-oriented sectors are construction and manufacturing and also in the wholesale and retail sectors carpool is popular. It is rather unpopular at universities, health sector and in public transport companies, and seems to be an alternative at locations where rail is no real alternative.

Multi Objective Path Planning: In this section, we will give a brief overview of some efforts directed towards solving the multi objective shortest path finding problem. There are of two main broad categories of multi objective shortest path problem. The first category is the deterministic (global optimal) approaches which are generalizations of the traditional single objective shortest path label setting and label correcting algorithms, and the second category is the approximation schemes, mainly evolutionary algorithms.

Herbwai et al. [4] presents evolutionay multiobjective route planning algorithm for solving the route planning problem in the dynamic multi-hop ride-sharing. They have used the time expanded graph to represents the drivers offers. In another independent work, Martins et al. [18] have provided a detailed

theoretical study for different types of labeling algorithms to solve the multi objective path problem. They have proved that it is bounded if and only if absorbent cycles are not present in the network. Müller-Hannemann and Weihe [17] have identified a set of key characteristics that exists in many applications of shortest path problem that makes the number of Pareto optimal paths at each visited node to be limited by a small constant.

Shad et al. [15] have analyzed and concluded that the genetic algorithm performs poorly on large size networks. However, Mooney et al. [14] have developed an evolutionary algorithm for the multi objective shortest path problem and showed its feasibility through a set of experiments over real road networks. Pangilinan et al. [13] have explored the behavior of a multi objective evolutionary algorithm when applied to solve the shortest path planning problem. The behavior of the algorithm is described in terms of the diversity and optimality of the solutions in addition to its computational complexity.

Summary: To summarize the literature review, the majority of the existing systems provides one fixed path to the driver and the rider based on their locations. The drivers do not have the flexibility to change or compute the path according to his own requirement. This is one of the main motivations of this work because the driver can be any person from any discipline of life therefore his choice of preferences are different at different times and on different days.

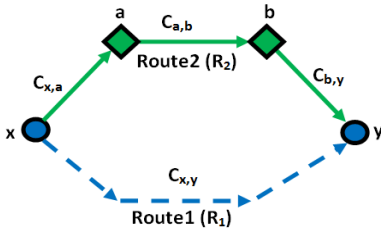


Fig. 1. A shared trip between driver(circles) and rider(diamonds)

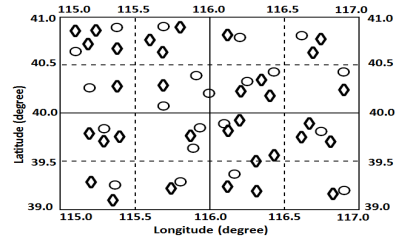


Fig. 2. A grid showing the drivers(circles) and riders(diamonds)

3 Problem Formulation

The target system of our Ride-sharing System approach is a smart-phone application for dynamic ride-sharing, does not require pre-declared paths or pickup points, and can be used from virtually any location at any time. The system promotes ride-shares among people with connections within social networks (e.g., Facebook) in order to favor ride-shares among users that trust each other.

3.1 Ride Sharing

Ride-sharing system continuously receives a sequence Z of trip announcements over time from potential participants and stored this information in the database.

A trip announcement contains an origin and a destination location, and additional information that specify its potential timing. With this information, the provider automatically establishes ride-shares over time, matching potential drivers and riders. Now the goal of our system is twofold, on the one hand the system needs to minimize the distance or path length and on the other hand it increases the number of passengers for optimal utilization of the resource.

We suppose that the database has set of drivers $D = \{d_1, d_2, \dots, d_n\}$, a set of stations $S = \{s_1, s_2, \dots, s_m\}$ and a set of driver's offers $O = \{o_1, o_2, \dots, o_k\}$. Each offer consists of a starting station s_1 , ending station s_2 , departure time t_1 and arrival time t_2 , and a cost c such that $\forall o \in O : o = \{s_1, s_2, t_1, t_2, c : s_1, s_2 \in S, s.t. s_1 \neq s_2 \wedge t_1 < t_2\}$. We also have set of riders $R = \{r_1, r_2, \dots, r_n\}$ with request q for sharing a ride, it contains the source station s_1 , destination station s_2 , and the earliest departure time t_1 and the latest arrival time t_2 . Similarly for all requests $\forall q \in Q : q = \{s_1, s_2, t_1, t_2 : s_1, s_2 \in S, s.t. s_1 \neq s_2 \wedge t_1 < t_2\}$. Now we have the multiple drivers' and the riders' presents as agents in the system.

The basic motivation behind the ride sharing system usage is to reduce the travel cost. In this work, we focus on systems designed to enable users to share variable trip costs. When such costs are roughly proportional to distance traveled, cost reduction is only possible when the length of a ride-share trip is shorter than the sum of the lengths of the separate trips. We consider this as a first condition for a feasible match only if it provides positive cost savings: i.e a ride-share between driver d and rider r is feasible iff $C(P) > 0$, where

$$C(P) = C_{(x,y)} + C_{(a,b)} - (C_{(x,a)} + C_{(a,b)} + C_{(b,y)}) \quad (1)$$

3.2 Modeling Drivers' Offers

Here we will model the two sets of group to solve the route matching problem. We are using the geodetic grid (latitude/longitude) system with a resolution of 0.5 degree in each axes to efficiently map the drivers' offer and riders' request in each grid element as shown in the Fig. 2. Whenever a new request $r \in R$ and offer $o \in O$ comes from the rider and driver respectively, it will be first directly assigned to some grid element and then it is processed. Each grid element has its own local list of the drivers and riders based on the trip announcements. If there is no driver already present to any grid element or no successful match found then the respective grid element is expanded to neighboring eight grid elements to find suitable match for the ride-sharing. The request of the rider is evaluated with the driver's list based on the following factors as discussed below.

4 Route Matching

In this section we will discuss about the various factors which will be used in matching the new request $r \in R$ from the rider to each driver $d \in D$ present on the network.

4.1 Detour and Range Factors

At this stage we make use of the detour distance and the range factor as an initial measurement criteria of route matching and further it is used for calculating the relevancy rate. The ride request which has higher relevancy rate value will occupy the higher position in the recommendation list of drivers. The number of riders assigned to each driver depends upon the nature and capacity of vehicle. Moreover the detour distance and range factor between new request from rider and default paths of online drivers are computed in respective grid elements.

Definition (Detour and the Range Factor): Given two ride requests R_1 and R_2 which have start and destination locations $\langle a, b \rangle$ and $\langle x, y \rangle$ as shown in the Fig. 1. Now if $\mu(x, y)$ denotes the length of the shortest path of the driver from point x to y and $\mu(a, b)$ denotes the length of the shortest path of the rider. The detour factor is then defined as:

$$f_d(R_1, R_2) = \frac{(\mu(x, a) + \mu(a, b) + \mu(b, y) - \mu(x, y))}{\mu(x, y)} \quad (2)$$

and the range factor is defined as:

$$f_r(R_1, R_2) = \frac{\text{mean}(\mu(x, a), \mu(b, y))}{\mu(x, y)} \quad (3)$$

The value range of detour factor will be $[0, +\infty)$. Now we define a transformation $E : R^{[0, +\infty)} \rightarrow (0, 1]$ for uniform comparison of the values between driver and different riders. We define this transformation as relevancy rate:

$$E(R_1, R_2) = e^{-f_d(R_1, R_2)} \quad (4)$$

The range of the values of relevancy is between $(0, 1]$ and 1 means the maximum matching of the routes. The value of the range factor needs to be less than 0.5 as the first step for suitable match. The system determines drivers that are online and compatible for this ride based on the relevancy of both the paths, the value of the range factor, condition of the Eq. 1 and whether the deadlines will comply. In this way system assigns multiple riders to single driver based on these factors.

Graph Generation: From drivers' perspective, each driver may have single or multiple riders associated with him based on vehicle capacity. Now just before starting the drive, the driver computes the path for the whole ride. The system will generate a number of sub-optimal paths for the driver based on the multiple riders requests and the weights of the objective function set by the driver as shown in the Fig. 3.

Now each driver has its own local network, rider nodes are sorted in ascending order with respect to the distances, the drivers' initial point I and final point F are then inserted at the start and end of the list. Each start point node in this sorted vector is a candidate for connection to all start nodes ahead of it and no

Drivers Setting Dialog

Path Planning Settings

Vehicle Speed :	40	Vehicle Capacity :	4
Number of Paths :	5	Safety Margin Time :	30

Note : speed is in km/hr and time is in minutes whereas the capacity and number of paths are counts.

Weight Factors

Length :	10
Time :	90
Occupancy :	0
Social Strength :	0

OK Cancel

Fig. 3. The drivers' setting dialog

unrealistic edge (e.g inserting destination node without the source node and vice versa) is allowed into the grid. Therefore, the node N_i is connected to all source nodes and hence with their corresponding destination nodes $I, N_{i+1}, \dots, N_N, F$. Each edge is evaluated under several constraints like length, time etc and assigned the edge cost based on the objective function as described below.

4.2 Cost Function

In the ride-sharing system, new riders and drivers continuously enter and leave the system. Some drivers want the shortest route and some may want the maximum riders on the weekends or holidays. Some people may only feel safe sharing rides with friends or acquaintances and possibly also with the friends of friends. The different objectives need to be considered when defining the edge cost. Often these objectives are conflicting, and thus there is a need to incorporate them into a single multi-objective function that satisfies route requirements. In this study the objectives considered are: **1) Minimization of the path lengths** **2) Minimization of the travel time** **3) Maximization of the available seats (occupancy)** **4) Maximization of social closeness.**

The first objective can be modelled as $\frac{L_E}{L_{max}}$, where L_E is the edge length and normalized by the maximum length L_{max} . The travel time is the time spent in the vehicle while actually traveling from origin to destination. The second objective of minimization of travel time can be transformed into an equivalent objective of minimizing the path length. It can be modelled by dividing the edge distance by the average velocity of the vehicle i.e $\frac{L_E}{Avg_{vel}}$. The third objective of maximum utilization of the available seats(i.e occupancy) can be modeled as $\frac{c}{(Sa)_{max}}$, where c is the constant usually taken to be 1 (the same for all the edges) and the $(Sa)_{max}$ is the maximum number of the riders used for the normalization of the cost. This objective may be beneficial for a ride-share provider whose revenues are linked to the number of successful ride-shares. In social closeness calculation, we adopt the same method as defined by Gilbert et al. [10]. From their method we consider some measures like social distance,

demographic and some structural information to compute the social closeness. Finally we normalize the net social closeness as S . We assign the $-S$ to the edge cost, where $0 \leq S \leq 1$. The edge cost can be written as:

$$E_{cost} = \frac{W_L \times C_1 + W_T \times C_2 + W_{Sa} \times C_3 + W_{Sd} \times C_4}{W_L + W_T + W_{Sa} + W_{Sd}} \quad (5)$$

Here E_{cost} is the edge cost and W_L , W_T , W_{Sa} and W_{Sd} are user defined weights for four objectives as shown in the Fig. 3. where

$$C_1 = \frac{L_E}{L_{max}}, C_2 = \frac{L_E}{Avg_{vel}}, C_3 = \frac{c}{(Sa)_{max}}, C_4 = -S$$

5 Path Planning

In this section we will discuss how we have exploited the greedy algorithm to compute a number of suboptimal paths available to the driver.

5.1 Path Optimization

Our goal is to select an algorithm that solves the minimum cost problem given in Eq. 5 above, furthermore we would like to work out a number of sub-optimal paths also. The edge cost components discussed above can be negative, the choice of the optimization algorithm therefore needs to be done accordingly. Since the Bellman-Ford algorithm works well with the negative edge costs and computes the shortest path between two given nodes in a weighted digraph. The Bellman-Ford algorithm in the process of finding the optimal path from the start node I to the final node F, also finds lowest cost paths from the start node to all other nodes. This feature of the algorithm is used here in the computation of a number of sub-optimal paths. We have extended the Bellman-Ford algorithm and incorporate the constraints directly into the runtime. In the algorithm, we have assigned the labels to nodes and the edges. Each label consists of three tags i.e cost, name and the time. The cost tag indicates the cost in going from the node I to that node and the name tag contains the name of edge that connects to the node I on the shortest path found so far and the third tag contains the time.

The algorithm can be divided into four parts. The graph generated in above section is a directed graph with an edge j starting from node $b(j)$ and ending at node $e(j)$ with cost E^j . Let C_I^i be the cost of the shortest path of the i^{th} node from the node I, P_I^i is the edge previous to the i^{th} node and T_I^i is the time to the i^{th} node on that path.

1) During the initialization part, first assign a label $(0,0,0)$ to the initial node I, and the labels $(\infty, 0, 0)$ to all other nodes and the edges respectively. The zero present in the name tag indicates that neither preceding edge assignment has been made, nor the edge originates from the initial node I.

2) The forward and reverse phase computations are done for generating the multiple paths. At the start of the algorithm we check if the start node of an

edge is the initial node I, in which case the cost of the end node of the edge is compared with the cost of the edge itself. If the cost of the edge is smaller, the label of the end node and the time tags are updated accordingly. If an edge does not originate from the initial node I, we check if a preceding edge assignment for the edge start node exists. Then previously stored cost at the end node of the edge is compared to that accrued by going through this edge. In case of a lower cost, the end node label and the edge label are both updated whenever there is an end node assignment, there is also an edge assignment as the reverse is not possible. If however the cost at the end node while going through the edge is not lower, then this cost is compared with the cost stored in the edge label. In case of a lower cost, only the edge label is updated. This implies that this new path from I to $e(j)$ via the j^{th} edge results in a lower cost, and hence the label of the j^{th} edge now contains this cost. During forward phase computation, we have traced for each edge j , the shortest path vector $j2I$ from $e(j)$ to initial node I by linking each node to preceding node as:

$$j2I = [e(j), b(j), b(P_I^j), b(P_I^{P_I^j}), b(P_I^{P_I^{P_I^j}}), \dots, I]$$

The shortest path vector $I2j$ from I to $e(j)$ can be obtained by reversing this.

3) Similarly treatment is done in the reverse phase computation, we interchange the roles of $b(j)$ and $e(j)$ and also the nodes I and F, the shortest path vector from $I2j$ from I to $e(j)$ is obtained along with the corresponding cost C_I^j and the time T_I^j . The paths are generated in an order of priority if one path does not meet user satisfaction for some reason, the next path can be considered.

4) Now during the multiple path computation, we find the corresponding list of the path vectors $[I2j \cup \overline{j2F}]$, where $\overline{j2F}$ is the shortest path vector from edge j to the final node F, computed during the reverse phase computation. This will give the multiple best paths in the order of priority defined by the driver that fulfils all the required constraints.

6 Simulation Results

The above algorithm is coded into a software application for generating the optimal paths from given set of offers and requests at any time t . We utilized the OpenStreetMaps (openstreetmap.org) data together with the MoNav planner that offers exact routing without heuristic assumptions at very little computational demand due to a routing core based on contraction hierarchies Vetter [7]. As a test bed for our experiments we used the following case study.

6.1 Case Study

In this section, we will discuss the different scenarios and monitor the performance of our algorithm. Optimal paths (satisfying all constraints) are computed using the above algorithm and the effect of varying the weights is studied here.

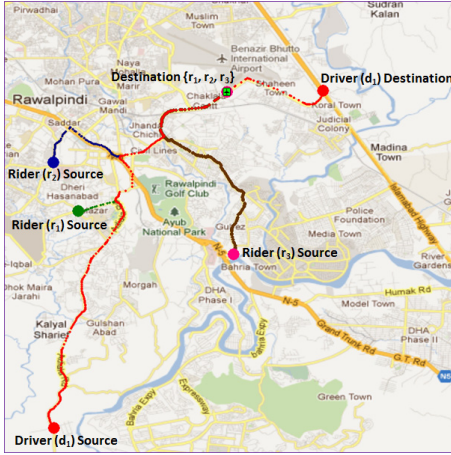


Fig. 4. Individual paths of driver and riders

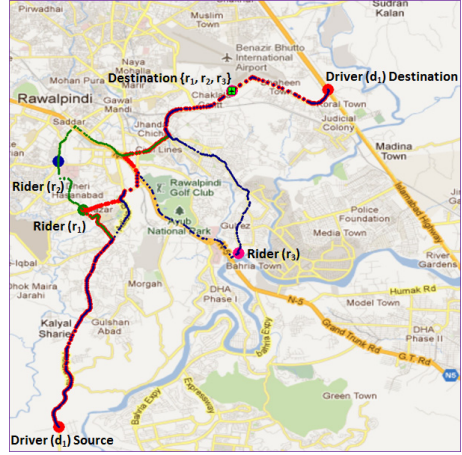


Fig. 5. Shortest length paths

The case study considers a person (driver) who has uploaded his schedule of departure from home to the office with the route map. Similarly, numerous people, who intend to reach airport for catching a flight have also uploaded their schedule along with departure point. Based on the information uploaded by the riders, the riders are automatically assigned to a suitable driver as per defined criteria described in section 4. All the scenarios consider $r_1, r_2, r_3 \in R$ are three riders with requests $q_1, q_2, q_3 \in Q$ and one driver $d_1 \in D$ with offer $o_1 \in O$ and their default paths are shown in Fig. 4.

In the first scenario, the objective function is optimized for the shortest path with weight settings $W_L = 1$, assuming that all other weights are considered to be zero. Similarly in the second scenario, the objective function is optimized for the maximum occupancy by adjusting weight as $W_{Sa} = 1$. The third scenario is worked around optimizing the minimum time constraint, with the weight $W_T = 1$. In the fourth scenario, the paths are computed on the basis of the social closeness value by mining the riders social network. It can be achieved by setting the weight as $W_{Sd} = 1$. Finally we applied multiple constraints like time and social closeness by assigning the weights as $W_T = 0.4, W_{Sd} = 0.6$.

In all the five scenarios, the optimal paths are generated by the application based on priority settings adjusted by the driver and the paths which are not complying with the constraints are simply rejected. The path computed closest to the objective function will be ranked the highest and is shown in red, similarly the next path in green and the third path is shown in blue color as shown in figures Fig. 5, 6, 7, 8, 9 respectively. Moreover Fig. 9 shows that the shortest closeness path (red) comes as a first choice in comparison with Fig. 8 output.



Fig. 6. Paths with maximum occupancy

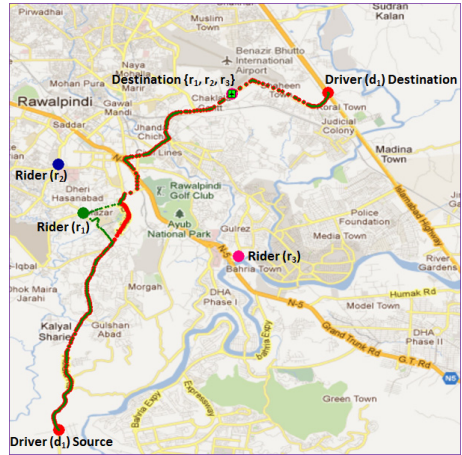


Fig. 7. Paths with time constraint



Fig. 8. Paths with Social Closeness



Fig. 9. Time constraint on Social Closeness

7 Conclusion

Internet-enabled mobile technology allows car travelers to announce trip requests and ride offer on short notice. In this paper we have presented a novel path planning approach that: 1) Allows more flexibility than state of the art system, as it gives more freedom to the users in the choice of ride sharing partner such as social interests matching and provides more security and convenience. 2) Furthermore it provides multiple feasible paths along with the ranking that minimizes a multi-objective cost function and weights of the factors can be specified by the driver to achieve the optimal objectives, such as the length, time, occupancy and social closeness. 3) Our simulation study results have shown that the proposed system

may have the potential to compute that multiple optimal paths against different user defined objective function.

Acknowledgments. The work is supported by the General Motors, China Science Lab, Shanghai.

References

1. Vanoutrive, T., Van De Vijver, E., Malderen, L.V., Jourquin, B., Thomas, I., Verhetsel, A., Witlox, F.: What determines carpooling to workplaces in Belgium: location, organisation, or promotion? *Journal of Transport Geography* 22, 77–86 (2012)
2. Trasarti, R., Pinelli, F., Nanni, M., Giannotti, F.: Mining mobility user profiles for car pooling. In: *KDD 2011*, pp. 1190–1198 (2011)
3. Kleiner, A., Nebel, B., Ziparo, V.A.: A Mechanism for Dynamic Ride Sharing Based on Parallel Auctions. In: *IJCAI 2011*, pp. 266–272 (2011)
4. Herbawi, W., Weber, M.: Evolutionary multiobjective route planning in dynamic multi-hop ridesharing. In: Merz, P., Hao, J.-K. (eds.) *EvoCOP 2011*, pp. 84–95. Springer, Heidelberg (2011)
5. Agatz, N., Erera, A., Savelsbergh, M., Wang, X.: Sustainable passenger transportation: Dynamic ride-sharing. Tech. rep., Erasmus Research Inst. of Management (ERIM), Erasmus Uni., Rotterdam (2010)
6. Jeanes, P.: Can total car emissions be reduced enough by 2020? In: *Traffic Engineering & Control*. PA Associates (January 2010)
7. Vetter, C.: Fast and Exact Mobile Navigation with OpenStreetMap Data. Diploma thesis, Algorithmics II, University of Karlsruhe, Germany (2010)
8. Kirshner, D.: Ride Now (2010), <http://www.ridenow.org>
9. Morris, J.H.: SafeRide: Reducing Single Occupancy Vehicles. Carnegie Mellon school of computer science, Tech. Rep. (2009)
10. Gilbert, E., Karahalios, K.: Predicting tie strength with social media. In: *CHI 2009*, pp. 211–220. ACM, New York (2009)
11. Kelley, K.L.: Casual Carpooling - Enhanced. *Journal of Public Transportation* 10(4), 119–130 (2007)
12. Murphy, P.: The Smart Jitney: Rapid, Realistic Transport. *New Solutions* 12, 12 (2007)
13. Pangilinan, J., Janssens, G.: Evolutionary algorithms for the multi-objective shortest path problem. *IJASET* 4(1), 205–210 (2007)
14. Mooney, P., Winstanley, A.: An evolutionary algorithm for multicriteria path optimization problems. *IJGIS* 20, 401–423 (2006)
15. Roozbeh, S., Hamid, E., Mohsen, G.: Evaluation of route finding methods in GIS application. In: *Map Asia 2003*, Beijing, China, pp. 12–20 (2003)
16. Resnick, P.: SocioTechnical Support for Ride Sharing. University of Michigan, School of Information, Tech. Rep. (2003)
17. Müller-Hannemann, M., Weihe, K.: Pareto Shortest Paths is Often Feasible in Practice. In: Brodal, G.S., Frigioni, D., Marchetti-Spaccamela, A. (eds.) *WAE 2001*. LNCS, vol. 2141, pp. 185–197. Springer, Heidelberg (2001)
18. Martins, E., Santos, J.: The labeling algorithm for the multiobjective shortest path problem. Tech. rep., University of Coimbra, Portugal (1999)