Traffic Big Data Based Path Planning Strategy in Public Vehicle Systems

Ming Zhu*, Xiao-Yang Liu*, Meikang Qiu†, Ruimin Shen*, Wei Shu‡*, Min-You Wu*
*Shanghai Jiao Tong University, Shanghai, China
†Pace University, New York City, USA
‡University of New Mexico, USA
*{zhumingpassional, yanglet, rmshen, mwu}@sjtu.edu.cn, †mqiu@pace.edu, ‡shu@ece.unm.edu

***Abstract*—Public vehicle (PV) systems will be efficient traffic-management platforms in future smart cities, where PVs provide ridesharing trips with balanced QoS (quality of service). PV systems differ from traditional ridesharing due to that the paths and scheduling tasks are calculated by a server according to passengers' requests, and all PVs corporate with each other to achieve higher transportation efficiency. Path planning is the primary problem. The current path planning strategies become inefficient especially for traffic big data in cities of large population and urban area. To ensure real-time scheduling, we propose one efficient path planning strategy with balanced QoS (*e.g.*, waiting time, detour) by restricting search area for each PV, so that a large number of computation is saved. Simulation results based on the Shanghai (China) urban road network show that, the computation can be reduced by 34% compared with the exhaustive search method since many requests violating QoS are excluded.**

I. Introduction

With the increasing of mobile computing [1] and urban area, traffic big data has attracted more attention. More efficient algorithms are required in future smart cities. Traffic big data is characterized by four "V" features [2] [3]. *Volume*: the large urban area, large population, large road network, and a large amount of trip requests lead to a large amount of trip data. *Variety*: The information of traffic condition should be detected to provide better service for passengers, *e.g.*, speed, path. And the data are collected by various sensors from vehicles and roads. *Veracity*: some traffic data (*e.g.*, speed, GPS) obtained from multiple sensors may have noise. *Velocity*: the requests of cities should be handled in real-time to satisfy trip demands since passengers want immediate replies from data center.

As an efficient traffic-management platform, the public vehicle (PV) system [4] provides low-cost ridesharing trips with balanced QoS (quality of service) for passengers, thus the traffic efficiency can be improved dramatically. PV systems consist of three parts: a data center (server), PVs, and passengers. PVs may be driverless electric vehicles [5] [6] in the future. The path planning for PVs is named as PV path (PVP) problem, which is NP-Complete [4]. In addition, PVs can corporate with each other to provide multi-hop paths for passengers [7]. The running flow of PV systems is as follows. If one passenger needs trip service, he/she sends a request (including his/her origin, destination, and time window, etc.) via a mobile internet device (*e.g.*, smart phone) to the server. The server calculates the matching between PVs and the passenger, and the paths of PVs to serve him/her.

PV systems have high traffic efficiency since they provide ridesharing service. Less research focus on how to reduce the computation in the routing problem of ridesharing. Although some solutions have been proposed for this problem [8], the whole road networks should be partitioned into multiple cells, and the status of each vehicle should be updated according to preset intervals.

How to reduce computation based on traffic big data is challenging for several reasons. *First*, with the increasing of population and urban area, it is hard to provide a real-time scheduling for a large number of requests. *Second*, once the server receives new requests from passengers, new scheduling strategies will be calculated and paths of some PVs may be changed, resulting in larger computation requirement.

II. Basic Idea and Solution

Assume that at some time, there are a large number of trip requests of passengers. PVs will provide service for them according to the scheduling strategies calculated by a server. Let $r = (t, o, d)$ denote a request, where t is the request birth time, o is origin, and d is destination. Make sure o is visited before d since the passengers should be first be picked up and then dropped off. Let p denote a PV or its current location. Let δ be the detour ratio of r, which is the percentage of additional distance compared with the shortest distance from its origin to destination. Make sure that δ does not exceed a threshold Δ. Assume p serves r. We name r as the *furthest request* if its destination d is the last point on the path of p.

Potential search area, *PSA*, is an area where the origins or/and destinations of possible requests whose QoS may be in constraints, while the other requests not in this PSA should be excluded. If we try the requests one by one, a large amount of computation will be needed. PSA can efficiently improve the computation speed.

1) PSA determined by the furthest request which has been picked up

According to properties of ellipses, we know that, any point (origin or destination of any request) inserted between o and d should fall in an ellipse determined by o and d, otherwise, the QoS will be violated. Therefore the corresponding rectangle of the ellipse is the PSA to restrict search area.

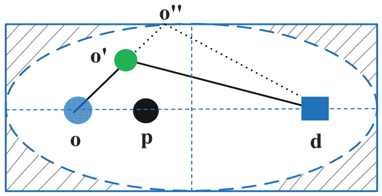

Fig. 1. PSA determined by the furthest request r which has been picked up.

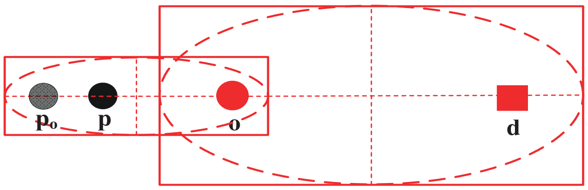

Fig. 2. PSA determined by the furthest request r which has not been picked up.

2) PSA determined by the furthest request which has not been picked up

To limit the waiting time of r, we introduce *buffer distance*, w, which denotes the travel distance of p from the time when schedule of r is determined to its pickup time. Its threshold is W. Make sure that $w \leq W$. Here, we should record the location of p when schedule of r is determined, which is denoted by p_o just as shown in Fig. 2.

3) PSA of one PV

PSA of one PV p is determined by the furthest request. And it is the union of two PSA: the one determined by p and origin of the requests, and the other one determined by the origin and destination.

4) Solution

As shown in Fig. 3, there are three insertion cases: Case $\mathbb{A}$: None of o and d becomes the last point of the new path. Case $\mathbb{B}$: d becomes the last point of the new path, while o does not precede d immediately. Case $\mathbb{C}$: d becomes the last point of the new path, and o precedes d immediately.

Our solution PSAP (PSA based path planning strategy) is as follows. With respect to an unscheduled request r and one PV p. If both the origin and destination in PSA, calculate insertion cost of Case $\mathbb{A}$. If the origin in PSA, calculate insertion cost of Case $\mathbb{B}$. If all points on path of p fall in PSA, calculate insertion cost of Case $\mathbb{C}$. Finally, calculate insertion cost among all PVs. The PV with the minimum travel distance under QoS constraints will be chosen to the request, and its path will be updated.

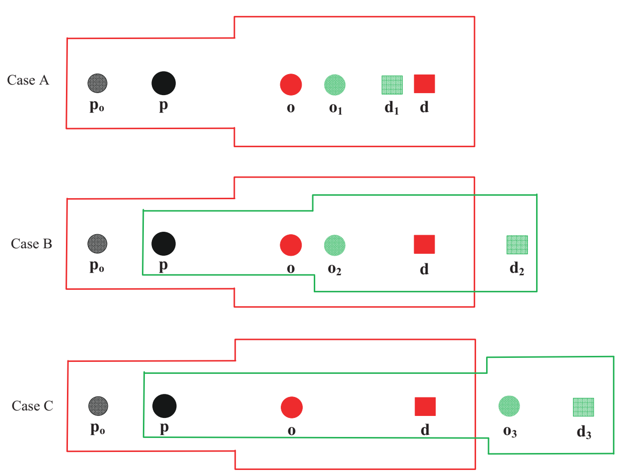

Fig. 3. Corresponding PSA in three insertion cases.

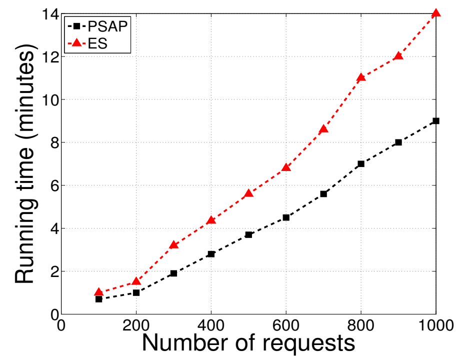

Fig. 4. Running time comparison.

III. Performance

In PV systems, the initial locations of PVs are randomly distributed in the road network which follows a uniform distribution. The compared algorithm is the exhaustive search algorithm, which is similar to the algorithm in [4] by adding checking buffer distance, *i.e.*, if the buffer distance exceeds its threshold W, the corresponding requests have to wait. We use "*ES*" to denote the "exhaustive search" algorithm. Now, we put 700 vehicles to PV systems with an urban area of 50 km^2 and compare the running time of our solution and ES. Δ is set to 0.2, and the capacity is set to 16, W is set to 7. The running time of two algorithms is shown in Fig. 4. The running time using our solution is about 34% less than ES when the number of requests varies from 100 to 1000.

References

[1] K. Mitra, A. Zaslavsky, and C. Ahlund, "Context-aware qoe modelling, measurement, and prediction in mobile computing systems," *IEEE TMC*, vol. 14, no. 5, pp. 920–936, 2015.

[2] V. Mayer-Schönberger and K. Cukier, *Big data: A revolution that will transform how we live, work, and think.* Houghton Mifflin Harcourt, 2013.

[3] M. Chen, S. Mao, and Y. Liu, "Big data: A survey," *Springer MONET*, vol. 19, no. 2, pp. 171–209, 2014.

[4] M. Zhu, X.-Y. Liu, F. Tang, M. Qiu, R. Shen, W. Shu, and M.-Y. Wu, "Public vehicles for future urban transportation," *IEEE TITS*, p. 10, 2016.

[5] "One dramatic chart shows why electric cars are about to become mainstream." http://www.techinsider.io/electric-vehicle-battery-cost-decreases-2016-3, 2016.

[6] M. Zhu, X.-Y. Liu, L. Kong, R. Shen, W. Shu, and M.-Y. Wu, "The charging-scheduling problem for electric vehicle networks," in *IEEE WCNC*, pp. 3178–3183, 2014.

[7] M. Zhu, X.-Y. Liu, M. Qiu, R. Shen, W. Shu, and M.-Y. Wu, "Transfer problem in a cloud-based public vehicle system with sustainable discomfort," *Springer MONET*, 2016.

[8] S. Ma, Y. Zheng, and O. Wolfson, "T-share: A large-scale dynamic taxi ridesharing service," in *IEEE ICDE*, pp. 410–421, 2013.