GrabShare: The Construction of a Realtime Ridesharing Service

Dominic Widdows, Jacob Lucas, Muchen Tang, Weilun Wu
Grab Technologies, Inc.
`[dominic.widdows|jacob.lucas|muchen.tang|weilun.wu]@grab.com`

Abstract—Ridesharing is a natural option for increasing the efficiency and availability of transportation, and many factors need to align for ridesharing to successfully meet user needs in the marketplace. This paper explains how some of these issues have been addressed in the creation of GrabShare, a realtime ridesharing service available in an increasing number of Southeast Asian cities including Singapore, Manila, Kuala Lumpur, and Jakarta.

From an algorithmic point of view, the central topic is the scheduling system, which, given passenger bookings and vehicle locations, assigns passengers to vehicles and creates routes for those vehicles to follow.

Other crucial factors include pricing, a navigable and reliable user experience, a system architecture robust to rejections and cancellations, and computationally tractable use of maps and traffic resources. A continuing dedication to understanding each city's individual needs and challenges, and persistent attention to user feedback, is also vital.

This paper gives an account of these areas, and attempts to give an organic overview of how GrabShare helps to serve customers as part of an integrated suite of transportation services throughout Southeast Asia.

I. Introduction

Since vehicles were invented, vehicles have been shared. Wagons and carriages are designed for this purpose, and even two-wheeled vehicles can be used for shared transportation by the enterprising. Typically the weight of an extra passenger or package adds little to the cost of operation, so carrying extra passengers or cargo is entirely useful so long as there is ample space and the route of the vehicle (which places it visits at which times) meets the needs of the various users. Sharing transportation within the capacity of available vehicles makes sense. Making the best use of these resources involves a network of geometry, information, social, and economics problems.

The incentives for tackling these problem are huge. An estimated 2-5% of Southeast Asia's economic output is wasted on traffic congestion (Asia Development Bank figure). Though the cost of congestion is less in some other parts of the world (e.g., 1% of GDP in the United States [1]), reducing congestion with single-passenger motor cars involves astronomical investment in roads and vehicles, involving a commitment of land, capital, and environmental resources which is neither globally available nor sustainable. Shared transportation has always been part of the solution, and the need is more pressing than ever.

To reduce the number of vehicles needed, carsharing programs were introduced in many countries, growing considerably to incorporate an estimated 340,000 members and 11,700 vehicles worldwide by 2007 [2]. Since then, the number of cars available for realtime requests, and the number of passengers using them, has exploded thanks particularly to the ubiquitous use of mobile devices, particularly smartphones. Using a mobile computer to request immediate transportation would have still been a novelty 20 years ago. Today, it is ubiquitous, with companies such as Didi, Ola, Lyft, Uber, and Grab serving millions of passengers on a daily basis. Thus, over the course of a day, a single vehicle can serve many more passengers, effectively providing an economy on capital investment through sharing the vehicle.

Ridesharing is an extra step, enabling passengers to share the vehicle *simultaneously* because they want to go the same way. For larger vehicles, having several passengers sharing the vehicle at the same time has always been the rule, with rail, bus, and air transportation being key examples. But routes involving these larger vehicles typically involve considerable advance planning, either on the part of the transportation company in preparing timetables and selling tickets, or on the part of the consumer to organize enough passengers to charter a large vehicle.

Compared with these precedents, realtime ridesharing is a genuinely new development, depending on the availability of smartphones for communicating between passengers and drivers, and fast scheduling services which make use of highly-available parallel computing in the cloud. These resources have enabled us to launch GrabShare successfully and quickly in several markets throughout Southeast Asia, and to provide more than 2 million rides to customers in Singapore alone within the first two months of launching the service. Each GrabShare journey may currently include up to two bookings, each of which allows up to two passengers, so we are able to make much greater utilization of available seats than in most other privately-operated cars.

As well as the demand volatility inherent in realtime scheduling, a key challenge with implementing GrabShare is that, being part of the so-called sharing economy, the supply of

vehicles is also volatile and sometimes unpredictable. Drivers are not bound by a contractual arrangement or employment agreement that commits them to driving at particular times, or to accepting every booking that is proposed by the system. These considerations are crucial in understanding why the system is built as it is, and why its scheduling approach is sometimes different from others proposed in the literature.

This paper is organized in such a way as to give a summary of how these various core considerations affect the design of GrabShare. Section II outlines the core workflow of system components, which is important to understanding why some of the algorithmic and design challenges arise. Section III explains the scheduling system, which houses the core algorithm for assigning bookings to vehicles. Section IV outlines some of the economic factors and pricing decisions, which must work effectively and clearly for all parties at all times. Finally, section VI summarizes related work.

II. GRABSHARE CORE CONCEPTS AND WORKFLOW

From a practical engineering point-of-view, GrabShare is an enormously distributed, asynchronous system. In any one city, we may have several thousand active drivers, including dedicated full-time drivers, part time drivers attracted by higher earnings during peak-hour surges, and casual drivers who drive well but are only occasionally available. Passengers are a much larger and even more varied, unpredictable, and sporadic user group. These users connect to the network using a mobile device, and each user is an independent agent. It is crucial to realise that when saying 'distributed' and 'asynchronous', it is not in the sense that the computations can be distributed for higher performance. The authority and decisions in the system are distributed, and they may be made and communicated at any time. If we think of the whole network as a collection of processors, then each processor is sporadically unavailable, and as soon as it becomes available, its owner usually wants to earn money or be taken somewhere immediately!

The core building blocks of the system in terms of information architecture are as follows:

Passenger	Customer who wants to travel from some pickup location to some dropoff location.
Driver	User who controls a vehicle that is able to carry passengers.
Booking	An individual transportation request, placed by a passenger.
Step	An instruction to go to a particular location and pickup or dropoff a passenger. Each booking is composed of a pickup step and a dropoff step.
Driver Plan	A sequence of steps to be performed in order by an individual driver.

Thus far, the definitions are deliberately general, and are aligned with standard introductions in the vehicle routing literature (e.g., [3, Ch 1]). A traditional package delivery scheduling system is somewhat different, in that a customer

does not want to transport themselves, and a booking may contain several packages each of which entails a separate transportation request. While these are important differences, GrabShare's architecture is in an abstract sense more like that of a realtime delivery network that that of a mass passenger transit system or even a single-booking taxi dispatch system.

The design of GrabShare as a whole takes into account a complex existing collection of systems and services, in particular, the behavior of drivers. Sometimes in software engineering, "must interoperate with existing systems" comes as a requirement to avoid the cost of replacing legacy code: in our case, it came much more from the requirement that GrabShare should work as naturally as possible for our drivers. This is crucial, because drivers operate quite independently, and neither network communication nor driver behavior is something we can take for granted. To avoid large-scale disruption, the expected user experience for drivers had to change as little as possible: especially since some large changes were unavoidable.

Along with the notion of having two parties in the vehicle at the same time, the key difference between GrabShare and a single-rider taxi dispatch system is that a GrabShare driver can be assigned more than one booking at once. The driver does not always proceed from one booking's pickup to the corresponding dropoff. This is mathematically obvious, but it is a significant change in the behaviour expected of drivers. This is why it is necessary to decompose each booking into a pickup step and a dropoff step, and to introduce the driver plan as a list of steps.

III. THE SCHEDULING SYSTEM

A. Overall Process

The scheduling system is responsible for taking new bookings and proposing appropriate driver plans. To make the relationship between bookings, steps, and driver plans explicit, the GrabShare team internally uses *trip notation*, whereby a booking A has a pickup step $\overline{A}$ and a dropoff step $\underline{A}$, and then a driver plan "pickup A, pickup B, dropoff A, dropoff B" is written $\overline{A}\,\overline{B}\underline{A}\,\underline{B}$. If a vehicle has no bookings assigned, its driver plan starts as empty, and the only plan that can be proposed is of the form $(\overline{A}, \underline{A})$.

If a new booking, B, is introduced, the possible driver plans where any of the route is shared are $(\overline{A}, \overline{B}, \underline{A}, \underline{B})$, $(\overline{B}, \overline{A}, \underline{B}, \underline{A})$, $(\overline{A}, \overline{B}, \underline{B}, \underline{A})$, and $(\overline{B}, \overline{A}, \underline{A}, \underline{B})$. If the step $\overline{A}$ has already been completed, then A has already been picked up and it follows that any driver plans that would put $\overline{B}$ before $\overline{A}$ are impossible. In practice, GrabShare currently avoids plans where $\overline{B}$ would be placed before $\overline{A}$, because A has already been quoted an approximate ETA, and we did not wish to disrupt this. Such rules-of-thumb are not typically SLA-level commitments to passengers — if the efficiency and cost-savings that could be passed on to passengers were found to justify considering plans where active pickup steps are interrupted, this would be considered. Two other rules in precisely this category were also introduced: do not allow dropoff steps to be scheduled any more than 3 steps after their

corresponding pickup steps; and do not make driver plans with more than 2 bookings sharing. (The latter implies the former, but not vice-versa.)

Because drivers may be experiencing network difficulties or unexpected traffic incidents, the scheduling system has to propose several possible driver plans for each booking, and then Grab's existing booking service and dispatch system is responsible for assigning each booking to the best available driver, and confirming when the driver has responded accepting the booking. To make this efficient and tractable, we consider only the nearest candidate drivers. Grab's existing services for tracking and updating known driver locations and getting the drivers nearest to a given location are used for this. For the nearby drivers who already have GrabShare bookings assigned, there is an existing driver plan, otherwise an empty driver plan is created.

The scheduling system thus takes a booking A, a set of existing driver plans (some typically empty), and returns a set of candidate driver plans, ranked in order of preference. This process is executed as follows:

- For each driver plan, enumerate the possible new driver plans that arise from adding the steps $\overline{A}$ and $\underline{A}$, following the restrictions outlined above.
- Estimate how long each step will take to carry out using a course geometric estimate of travel time for each step.
- Score each candidate plan according to a number of features that characterize good plans.
- For the k best-scoring plans, recompute the estimated step travel times using a more accurate but more computationally costly estimate from a directions web service.
- Rescore each of these plans using the new travel time estimates.
- Return these candidate plans to the booking service, which is then responsible for offering the booking to the drivers in order until an acceptance is confirmed.

B. Scoring Features

The scoring features used currently include:

- Minimize ETA to passenger pickup.
- Minimize the amount of expected additional journey time, over and above the predicted direct travel time. (This is particularly important to passengers, to give good quality of service.)
- Maximize the overlap between bookings. (This is particularly important to drivers, because the overlapping part of the trip is the part for which they are earning two fares.)
- Minimize the angle between the bookings, construed as straight lines from pickup to dropoff.

Each of these scoring functions is easily configurable using a web interface managed by Grab country teams. (For example, Jakarta, being a much larger and more congested city, may allow a larger added travel time than Singapore.) The scoring functions are parametrized using intuitive physical measures: for example, there is a maximum expected time to pickup, given in minutes.

Internally, the scoring features are all normalized to return scores between -1 (bad) and 1 (good). For example, if the maximum expected time to pickup is set to t minutes, a normalized scoring function might be $1 - 2x/t$, so that at $x = 0$ minutes, the score is a perfect 1, at $x = t/2$ minutes a neutral 0 score, and at $x = t$ minutes, a negative -1 score. Scores below -1 are considered to be vetoes — such a plan is not allowed.

C. Performance Measures

The scheduling system is thus designed to be fast, robust, and comparatively simple. For simplicity, the parameters are designed to capture intuitions from user experience and business insights (such as "never allow bookings whose directions on the map are more than 90 degrees apart to be matched"). For robustness, the system has comparatively little internal state: the crucial state of the system is the currently-committed driver plans, which crucially is managed without adding any extra new driver states or booking states to existing Grab services. This allows most of the existing allocation, dispatch, and location-tracking systems to be reused without alteration. For processing speed, the scheduling system is produces candidate driver plans for a booking typically within a few hundred milliseconds, and almost always with a p99 latency under 1 second.

Of course, the accuracy of the system is also crucial, and constantly measured. Key performance metrics used include match rate (the proportion of bookings where the system finds an proposed match), match quality (a combination of low detour and large overlap), efficiency (how much driving did the driver save compared with taking each passenger separately). These numbers are constantly changing as different cities try to iterate towards the balance of factors and tradeoffs found to be most appropriate for their usage and traffic patterns. As a example, during the first two months of running GrabShare in production in Singapore, the average match rate was just over 40%, and the average efficiency just over 1.2, during a period where over 2 million rides were given.

The system is constantly under review, and many alternatives are possible. As we gather more experience, particularly more known good and bad user experiences (from sampling and rating, and from user feedback), a machine-learned approach to parameter setting becomes more desirable. Also, several more combinatoric approaches that consider a batch of bookings together can lead to more optimal outcomes in principle than our current one-at-a-time approach that seeks just the best candidates for a single booking.

IV. PRICING AND INCENTIVES

For a service such as GrabShare to succeed in the marketplace, it has to be in the best interests of drivers and passengers. This is complicated somewhat by the network effect — until there are a large volume of booking requests, there are comparatively few good matches, and if most passengers are riding alone, drivers are typically collecting only one fare.

A. Passenger Discounts from Predicted Efficiency Pricing

The efficiency from sharing has to be passed on to passengers for them to be interested in sharing at all. Hence one of GrabShare's main marketing slogans, "Share the ride, and the fare." This appropriate fare is computed as a discount on the fare for a direct single-rider trip using GrabCar (Grab's on-demand ride service using private-hire cars). The discount offered is proportional to an estimate of the efficiency likely to be gained from matching this booking. That is, if bookings on the ground are currently sparse, there is comparatively little chance of a booking being matched, and the discount should be small. (Effectively, in sparse conditions, passengers aren't offered a discount for sharing — they are offered a discount for being willing to share in the unlikely event that a good match is found.)

The predicted efficiency algorithm proceeds as follows:

- Given a new booking A, find the existing bookings whose match with the new booking would give a high efficiency.
- Rank the top k such bookings to give a set $\{B_i\}$.
- Compute a weighted score, e.g., $\sum_1^k \frac{1}{k} E(A, B_i)$, where $E(A, B)$ measures the efficiency that can be gained by carrying out bookings A and B together.
- Normalize this score to a discount in the range 0 to 1.

There are several choices available in the number of bookings k, the precise efficiency function, and the corpus of bookings to be used to begin with. (To begin with of course there were no GrabShare bookings, so direct GrabCar bookings were used — this can gradually change in cities where a suitable number of current or historical GrabShare bookings are now recorded.)

Because ridesharing is still new to many passengers, extra discounts and sometimes promotions are then offered to encourage uptake. This may take the form of money off, or a minimum setting for the predicted efficiency discount (e.g., by making GrabShare discounts to be at least 20% for each ride).

B. Driver Incentives

Compensating drivers for investing their time in a growing product is standard throughout the ridehailing industry — for example, drivers are sometimes guaranteed a minimum fare.

The main naturally occurring incentive for GrabShare is the economy of time and distance — by carrying more than one passenger at a time, drivers can complete more bookings in a day. With more passenger bookings completed, driver monthly incomes have also increased 10 per cent on average [4].

However, sometimes other top-up incentives are appropriate, especially for unmatched rides. Balancing the right goals and making incentives that promote the needs of drivers and passengers with shared transportation remains an active area of research and development.

V. OTHER AREAS VITAL TO GRABSHARE

The areas presented thus far are form only some parts of a successful system. For GrabShare to meet the needs of passengers in several countries over a long period of time, many related teams and disciplines have worked together. A distinct paper could probably be written on each of the below areas, and due to space constraints, the summaries here are brief.

A. Routing and Traffic Resources

Southeast Asia has weaker technology resources than several other regions, and this is certainly true with mapping and traffic data. Of course, GrabShare depends crucially on such resources, and Grab as a whole is committed to improving them. Illustrative examples include:

- OpenTraffic shares Grab's anonymized traffic data in Malaysia and the Philippines with local governments to assist with traffic planning and infrastructure investments [5].
- Grab's driver location information can be segmented by vehicle type, leading to directions and ETA estimates that are tailor-made for different vehicle types. This is crucial in Jakarta, for example, where bike and car journeys often take different routes and nearly always take very different times to complete.

B. Scheduling Research and Simulation

Thanks partly to in-house routing support, many large-scale simulations could be carried out before and after launch, to give an estimate of (for example) expected match rates at different booking volumes. As GrabShare grows to more cities, simulation has become more important than ever. For example, when faced with a report of bad matching behavior, we may sometimes decide to introduce a new scoring feature or change a feature weight or parameter. Simulation is important for predicting the likely consequences of such attempted improvements.

Cloud computing has helped considerably as the system has grown. Thanks to efficient use of cloud resources, and particularly, making sure that the scheduling operations are collocated with the lower-level travel time estimation services, has enabled us to run large scale simulations (several hours of bookings in a large city) within a short time (less that ten minutes).

C. User Experience Design

Grab conducted extensive research on consumer behaviour and preferences based on other carpooling services prior to launch, to map out and ensure a logical and simple user experience with GrabShare.

For example, for passengers, the booking flow and being picked up is just the same. But there may be other passengers in the car, and the driver may go to someone else's destination first, so the passenger app gives information about the order-of-events for each passenger in the car.

For drivers, the assignment experience is largely the same, but there are significant changes required to previous behavior. The direction of where to go next, and the need to remain aware of new bookings being assigned, are both departures from direct point-to-point passenger transport. Drivers are trained to expect this, and the user experience in the driver

app is designed to make (for example) accepting one booking while carrying out another explicit and clear.

For making the riding experience predictable, we have introduced some clear rules-of-thumb, some mentioned earlier. For example, having only two rides matched in any plan ensures that drivers and passengers have a clear sense that the ride will not become interminable, a frequent worry (according to internal research) and sometimes complaint of users of carpooling products.

D. Feedback from Passengers and Drivers

Large scale statistical systems make mistakes sometimes. We encourage feedback about matches and driver plans from drivers and passengers, paying particular attention to feedback from Grab employees and other highly active passengers who have used the system during alpha and beta testing phases (like many tech products, GrabShare typically goes through several gradually more public testing phases before being opened for general availability).

Learning from our mistakes is a long, gradual process, requiring local knowledge and expertise from country teams, and algorithmic expertise from scientific and engineering staff to propose solutions to specific problems and predict the macro effects of particular decisions. Gathering and using data in this fashion requires regular commitment from many team members, and can feel sometimes like a never-ending process. As with many data-driven systems, this long-term devotion to carefully gathering and analyzing good cases and problem cases makes the difference between success and failure.

Thus far, the passenger experience overall appears to have been well-received — the differential in average star-ratings between GrabCar and GrabShare during the first two-months of launch was around 0.02.

E. Geographic Diversity

Every city GrabShare serves (at the time of writing, Singapore, Manila, Kuala Lumpur, Jakarta, and Ho Chi Minh City) is different. At one extreme, Singapore is highly developed: and being an island, its geographic boundaries could hardly be more obvious! At another, Jakarta is huge, sprawling, and its transportation infrastructure has often lagged behind its population growth.

The region's diversity is a challenge to most, but Grab's hyperlocal approach is making this our strength, as we ensure we have strong local teams who understand local complexities and navigating around differences. Globally, research studies of traffic and potential resource-sharing patterns are readily available for other cities such as New York [1], London [6], Sydney [7], and many more: but as so often, Southeast Asia is poorly-covered and even absent from this literature, apart from the exceptional case of highly-developed Singapore [8]. An experience report easily the length of this paper could be written on the traffic patterns, booking volumes at various times, and scheduling and matching behaviors that arise from this, for each individual city. As a very brief initial
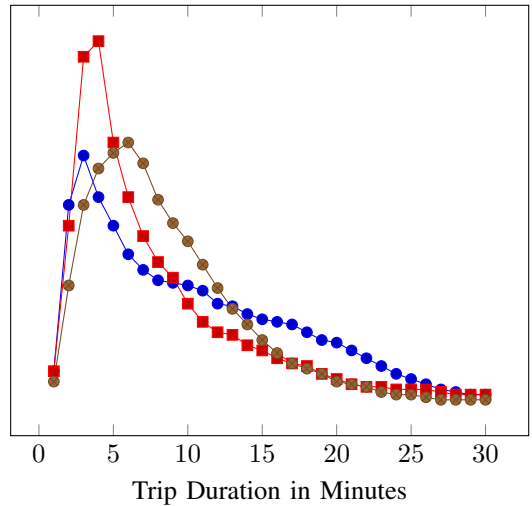

Fig. 1. Relative match rates at different trip durations for various cities

example, Figure 1 shows that in three different cities, matched GrabShare rides tend to have different durations.

Formerly often overlooked, we hope that Grab's rapid growth as a distinctly Southeast Asian technology company will help to put Southeast Asia firmly in the sights of research and development contributors worldwide.

VI. RELATED WORK

The creation of GrabShare depended on many factors, and in decades to come we expect that it will be historically obvious that these ingredients came together in 2016 and 2017.

To begin with, many years of research have gone into the vehicle routing problem (see the summary in [3]), involving formulations such as the travelling salesman and capacitated vehicle routing problems. While most of the traditional operations research involves optimization for supply networks where considerable time is available for planning, interest has grown more recently in the area of dynamic vehicle routing [9].

The particular problem of dynamic ridesharing appears to be underrepresented in the literature, though there are good summaries and experiments in some PhD theses and tech reports [10], [11], [12]. For example, the last of these work compares one-at-a-time 'greedy' approaches with batch optimizations, and predicts a large increase (28% to 74%) in match rates when a batch optimization is introduced. However, the catch is that these results are obtained with a 10 minute planning window, which could not compete effectively in today's smartphone ridehailing market where passengers are used to placing and confirming bookings in well under a minute. The shorter the planning window, the smaller the benefit from batch optimizations, and when bookings are to be confirmed more-or-less immediately, the greedy or myopic approaches are found to perform similarly [13]. This finding was taken into account when building the initial GrabShare scheduling system.

Further recent research directly in dynamic ride-sharing has demonstrated even more impressive potential savings through ridesharing, this time using more sophisticated combinatoric optimization for Manhattan [1]. This work still uses some beneficial simplifications, in its assumptions of where drivers are located, that all assignments are accepted by drivers and passengers, and most importantly, the futuristic that all rides in the city are open to sharing on the same platform. This gives companies like Grab that are currently maintaining live services a useful guide to what can be achieved in theory.

As well as smartphones, huge infrastructural improvements have contributed directly to making all of Grab's services possible. Some cloud computing strategies and their applicability for realtime ridesharing are suggested in [14], which (for example) correctly anticipates a preference for NoSql storage coupled with the use of a distributed RAM cache for fast realtime data storage.

In social sciences, at least one study is available that analyzes adoption of ridesharing services by different demographic groups, showing that younger and higher income demographics are most likely to use these services [15]. This survey considers seven North American cities, which yet again for Grab is good motivation but highlights the need for such research in Southeast Asia.

The most immediately comparable systems to GrabShare are the carpooling services from other smartphone ridehailing operators mentioned earlier. The single most relevant and useful publication for thinking through the design of GrabShare was in fact a series of Lyft blog posts [16]. We would like to thank the author of those articles and Lyft engineering for making this prior work available.

VII. Conclusion

The construction of GrabShare demonstrates many teams and disciplines working together to provide intelligent transportation solutions that make the best use of available technology to meet the needs of today's rapidly growing population of smartphone users in Southeast Asia.

Thus far, GrabShare's scheduling component has used an entirely greedy approach to allocating bookings to drivers, which has so far been found to be quite adequate. The booking protocols and interface design leave the way open for more sophisticated scheduling operations in the future. Pricing strategies must balance the needs of passengers and drivers, and promotions and incentives are necessary while attracting the kind of volumes the system needs to reach a self-sustaining network-effect. User feedback — and internal teams paying attention to user feedback — is crucial to improving and maintaining the quality of matches. Road data and traffic information are also vital, and since such data is often lacking or poor quality for most Southeast Asian cities, Grab has built and is improving its own platform for meeting these needs.

The product is still in its infancy, and in spite of its success so far, there is little doubt that an architectural summary of GrabShare written in subsequent years would have considerable differences. Nonetheless, we believe the current paper demonstrates some key features that will persist: in particular, a unique dedication to improving transportation throughout Southeast Asia, a region hitherto overlooked in research and technology papers, and whose vast opportunity and growing importance should be increasingly recognized worldwide.

References

[1] J. Alonso-Mora, S. Samanaranayake, A. Wallar, E. Frazzoli, and D. Rus, "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment," *PNAS*, vol. 114, no. 3, pp. 462–467, 2017.

[2] S. Shaheen and A. Cohen, "Growth in worldwide carsharing: An international comparison," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1992, pp. 81–89, 2007.

[3] P. Toth and D. Vigo, *Vehicle routing: problems, methods, and applications.* SIAM, 2014.

[4] Grab, "Grab extends GrabShare regionally with Malaysia's first on-demand carpooling service," https://www.grab.com/my/press/business/grabsharemalaysia/, 2016.

[5] D. N. Asia, "Grab, MDEC, World Bank Group launch OpenTraffic platform to combat local traffic woes," https://www.digitalnewsasia.com/, 2017.

[6] S. Le Vine, M. Lee-Gosselin, A. Sivakumar, and J. Polak, "A new approach to predict the market and impacts of round-trip and point-to-point carsharing systems: case study of London," *Transportation Research Part D: Transport and Environment*, vol. 32, pp. 218–229, 2014.

[7] R. Dowling and J. Kent, "Practice and public–private partnerships in sustainable transport governance: The case of car sharing in Sydney, Australia," *Transport Policy*, vol. 40, pp. 58–64, 2015.

[8] K. Spieser, K. Treleaven, R. Zhang, E. Frazzoli, D. Morton, and M. Pavone, "Toward a systematic approach to the design and evaluation of automated mobility-on-demand systems: A case study in Singapore," in *Road Vehicle Automation.* Springer, 2014, pp. 229–245.

[9] V. S. Zeimpekis, C. D. Tarantilis, G. M. Giaglis, and I. E. Minis, *Dynamic fleet management: concepts, systems, algorithms & case studies.* Springer Science & Business Media, 2007, vol. 38.

[10] K. Ghoseiri, *Dynamic rideshare optimized matching problem.* University of Maryland, College Park, 2012.

[11] N. Masoud and R. Jayakrishnan, "A real-time algorithm to solve the peer-to-peer ride-matching problem in a flexible ridesharing system," University of California, Irvine, Tech. Rep., 2015.

[12] X. Wang, "Optimizing ride matches for dynamic ride-sharing systems," Ph.D. dissertation, Georgia Institute of Technology, 2013.

[13] W. B. Powell, M. T. Towns, and A. Marar, "On the value of optimal myopic solutions for dynamic routing and scheduling problems in the presence of user noncompliance," *Transportation Science*, vol. 34, no. 1, pp. 67–85, 2000.

[14] D. Dimitrijevic, N. Nedic, and V. Dimitrieski, "Real-time carpooling and ride-sharing: Position paper on design concepts, distribution and cloud computing strategies," in *Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on.* IEEE, 2013, pp. 781–786.

[15] R. Clewlow, G. S. Mishra, and P. E. E. Center, "Shared mobility: Current adoption, use, and potential impacts on travel behavior," in *Annual Meetings of the Transportation Research Board, Washington DC*, 2017.

[16] T. Brown, "Matchmaking in Lyft Line," https://eng.lyft.com/matchmaking-in-lyft-line-part-3-d8f9497c0e51, 2016.