Peak-hour Vehicle Routing for First-Mile Transportation: Problem Formulation and Algorithms

Guiyuan Jiang, Siew-Kei Lam, Fangxin Ning, Peilan He, Jidong Xie

***Abstract*—The first-mile transportation provides a transit service using ridesharing based vehicles, e.g. feeder buses, for passengers to travel from their homes, workplaces or public institutions to the nearest public transportation depots (rapid-transit metro or appropriated bus stations) which are located beyond comfortable walking distance. This paper studies the vehicle routing problem (VRP) for the first-mile transportation, which aims at finding the optimal travel routes for a vehicle fleet to deliver passengers from their doorstep to the depots, where the passengers can continue their journeys using fixed-route buses or trains. We focus on the Peak-Hour VRP (PHVRP) for a limited vehicle fleet capacity to serve a large volume of travel requests, with the aim of maximizing the number of served passengers. The PHVRP generalizes the VRP with time window by considering multiple alternative depots for each travel request, such that a request is satisfied if the passenger is taken to one of his/her nearest depots. We formally formulate the PHVRP with constraints on vehicle capacity, pickup time windows, and quality of service regarding riding time, where a novel trip-based constraint model is used. We proposed an ant-colony optimization algorithm for the PHVRP, which is initialized with pheromone information that jointly considers the temporal-spatial distance as well as depot similarity among different travel requests. We introduced a novel scheme (called trip-by-trip scheme) to construct the travel routes by repeatedly forming a single trip for the vehicle with earliest end time until no vehicle can accept any more trips. In constructing a single trip, the algorithm intelligently decides whether or not to end the trip instead of taking more passengers. The effectiveness of the proposed methods is evaluated by comparing with optimal solutions on small size instances and with heuristic solutions on large size instances, using road network in Singapore and synthetic travel requests that are generated based on real bus travel demands.**

***Index Terms*—First-mile problem, peak-hour VRP, alternative depot, trip-based constraint model, trip-by-trip routing scheme.**

I. Introduction

The provision of first-mile last-mile connections is motivated by the need for increased accessibility to public transportation depots (rapid-transit metro or bus stations) in urban areas, which are typically geographically located beyond comfortable walking distance from/to the passengers' homes, workplaces or public institutions. These depots are typically positioned a mile apart on average to achieve high speed travel, and hence most of the urban area is beyond easy walking distance to a transit station.

It is envisioned that the availability of an efficient first-mile, last-mile transportation system will increase the public transport's share of urban trips which would lead to reduced road congestion and air pollution [1]. The enhanced mobility service will also benefit the aging population as well as school-children and those with physical disabilities [2]. Even bus stops are densely deployed in some cities, each of the bus stops typically belong to only a few bus lines whose direction and coverage areas are fixed. Our work addresses the first-mile transportation needs for passengers to commute from their homes, workplaces or public institutions to the nearest transportation depots so that they can continue their journeys using the necessary fixed-route transports. In particular, we focus on the vehicle routing problem for an urban first-mile transportation system, which aims at finding the optimal travel routes for a fleet of vehicles to deliver passengers to the nearest depots during peak hours. Middle-sized vehicles, such as feeder buses with capacity ranging from 10 to 20 passengers, are used to offer point-to-point services while enhancing efficiency by supporting ridesharing. The routes of the vehicles are flexible and can be changed based on the travel requests. The pickup points are spatially distributed in the urban area and are not fixed (i.e. they emerged when the passenger makes a travel requests through a smartphone application). The locations of these pickup points may include existing bus stops, vehicle pick-up points or loading areas at hotels, office buildings, shopping complexes, and crossroads near residential buildings or complexes.

We focus on the Peak Hour Vehicle Routing Problem (PHVRP) that is characterized by a limited vehicle fleet capacity to serve a large volume of travel requests, with the aim of maximizing the number of served passengers. To the best of our knowledge, this is the first work that addresses the PHVRP to support the first-mile transportation. Throughout the paper, the concept of 'destination' indicates the final point of the passenger's journey, while 'depot' indicates the transit points (rapid-transit metro or alternative bus stations), where the passenger switches from the feeder buses to the fixed-route transports (trains or buses) to continue his/her journey. Due to the limited vehicle capacity and a large volume of travel requests during peak hours, the PHVRP is addressed by selectively visiting the request nodes (pickup points) with the aim of maximizing the number of served passengers.

In addition to the above-mentioned optimization objective, i.e. maximizing the service capability (number of served passengers) for the limited vehicles during peak hours, the PHVRP extends the well-known Vehicle Routing Problem

This research project is partially funded by the National Research Foundation Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme with the Technical University of Munich at TUMCREATE. *(Corresponding author: Peilan He)*

G. Jiang, S.K. Lam, F. Ning and P. He are with the School of Computer Science and Engineering, Nanyang Technological University, 639798, Singapore. (e-mail: {gyjiang, assklam,fangxin_ning}@ntu.edu.sg, phe002@e.ntu.edu.sg). J. Xie is with the Huawei Technologies Co., Ltd., Shenzhen 518129, P.R.. China. (e-mail: xiejidong@huawei.com).

(VRP) with Time Window (VRPTW) by accommodating the willingness of public commuters to accept alternative travel routes through various depots in order to reduce their overall travel time, waiting time or walking distance. We assume that multiple alternative depots are associated with each travel request, which is supported by the fact that the existing travel planners (e.g. google map[1]) often provide multiple routes with alternative transit points. This opens up new opportunities to maximize service capacity (number of served passengers) through vehicle routing. While the PHVRP bears some similarity with existing vehicle routing problem variants such as Multi-Depot VRP (MDVRP) [3], multi-trip VRP (MTVRP) [4], [5] and Selective VRP (SVRP) [6], [7], it also exhibits significant differences. A detailed comparison between PHVRP and MDVRP, MTVRP as well as SVRP will be presented in Section II, where we discussed and highlighted the differences between the proposed ant-colony optimization (ACO) algorithm and the existing ACO-based algorithms for VRP problems.

The main contributions of this paper are as follows:

(1) To the best of our knowledge, our work is the first to investigate the PHVRP, where the number of travel requests exceeds the service capability of the limited vehicles. As such, each vehicle performs successive trips to serve as many passengers as possible during peak hours. The PHVRP aims at maximizing the number of served passengers by leveraging multiple alternative depots. This enables passengers to take alternative travel routes if they do not differ much from the optimal travel route in terms of travel time.

(2) We formally formulate the PHVRP under the constraints of vehicle capacity, time windows of travel requests, and quality of service regarding riding time. Since the objective of PHVRP problem is to maximize total served passengers instead of the number of passengers in each trip, we characterize the riding time of each vehicle trip (trip-based constraint) instead of each passenger. This may decrease the number of passengers in a trip but can potentially increase the total number of served passengers. We demonstrate through extensive experiments that this formulation can significantly improve the computation efficiency while slightly improving the solution quality for the PHVRP problem.

(3) We propose a novel demand graph consisting of travel request nodes and depot nodes which enables efficient search of routing solutions. We develop an efficient ACO algorithm for the PHVRP, which relies on a novel solution construction scheme to iteratively produce solutions that can maximize the number of served passengers. The proposed algorithm incorporates the following: i) a novel pheromone initialization information that jointly considers the temporal-spatial distance as well as depot similarity among different travel requests, ii) a solution construction scheme, called trip-by-trip (TBT) scheme, to construct a complete solution by iteratively forming a trip for the vehicle with earliest end time until no more trips can be formed for any vehicle, and iii) within the procedure of forming a single trip, efficient techniques are employed to construct CNM (candidate next move) set and choose the next move from the CNM set. These techniques enable each vehicle to intelligently decide whether to visit a depot and start a new trip from that depot or serve as many passengers as possible in a single trip.

[1]https://www.google.com/maps/dir/

II. Literature Review

A. Existing Works on Similar VRP Variants

The VRP problem has been the subject of intensive research ever since it was proposed by Dantzig and Ramser in 1959 [8]. The broad range of applications has spurred numerous VRP variants, such as capacitated VRP (CVRP) [9], [10], VRP with heterogeneous fleet of vehicles [11], VRP with time windows [12], VRP with pickup and delivery [13], dynamic VRP [14], periodic VRP [15], multi-trip VRP (MTVPR) [5], [16], split delivery VRP [17], multi-depot VRP (MDVRP) [18], selective VRP (SVRP) [19], etc. In contrast to general studies in VRP, there are also many existing works that address practical problems in urban mobility [20]. This includes traffic regulations, traffic congestion [21], road condition, parking space, air pollution [22], noise pollution, emergencies [23], etc. The objects to be delivered in these studies include consumer goods, materials, mails/packages, waste products as well as passengers. Recently, research in urban mobility for passenger movements is gaining wide attention. Aim at encouraging the use of public transport, many researchers studied the first (last)-mile transit problem, which is now recognized by urban transport planners as a key barrier to better public-transit utilization. Solutions to the first (last)-mile transit problem include the usage of feeder buses [1], autonomous vehicles [24], bicycle sharing systems [25], cybercars [26], car sharing programs [27] etc. The work in [28] proposed to boost ride-sharing opportunities with alternative destination of passengers to reduce the number of circulating vehicles, by developing an activity-based ride matching (ABRM) algorithm to match ride requests with ride offers.

These existing works related to the PHVRP can be grouped in the following three categories: 1) Multiple Depots Vehicle Routing Problem (MDVRP), which targets multiple depots such that a request can be satisfied by any of the depots, 2) Multiple-Trip Vehicle Routing Problem (MTVRP) which enables each vehicle to perform successive trips during the same work shift, and 3) Selective Vehicle Routing Problem (SVRP) which does not impose that all clients need to be visited by the vehicles.

(1) MDVRP. The MDVRP problem [3] aims to determine a set of vehicle routes with the objective of minimizing the number of vehicles or total travel distance/costs, where there are multiple depots and a solution is feasible if each route satisfies the standard VRP constraints and begins and ends at the same depot. Several variants of the model are studied: time windows [11], [29], split delivery [30], heterogeneous fleet [29], periodic deliveries [31], and pickup and delivery [32]. The PHVRP considered in this paper differs from the existing MDVRP variants in many aspects: 1) the PHVRP is targeted towards transferring human passengers, and hence it can only tolerate very short waiting time for the vehicles (usually few minutes); 2) the PHVRP has insufficient vehicles

to serve all the passengers, thus its objective is to maximize the service capability instead of minimizing the overall cost; 3) each travel request in PHVRP can have several alternative depots (not all) such that a request is regarded as served if the passenger is taken to any of the preferred depots. In the existing works for MDVRP, a customer can be served by any of the depots; 4) each vehicle in PHVRP must be used many times (i.e. multi-trip) in order to serve more people.

(2) MTVRP. MTVRP generalizes the traditional VRP problem such that a vehicle can perform successive trips during the same working shift to serve a set of customers. In this category, many works have been reported for the variant called cumulative capacitated vehicle routing problem (mt-CCVRP) [4], where the goal is to minimize the sum of arrival times at demand nodes. The time constraints, which involve the time windows and time horizons on each customer, has also been taken into consideration [5]. It has also been jointly considered with vehicle scheduling [16], where the objectives are to minimize the number of vehicles and the total travel time/cost while satisfying constraints on vehicle capacity as well as delivery time windows. However, 1) existing MTVRP variants target at delivering goods, and hence they usually have slack (or none) time window for pickups; 2) They have different objectives from PHVRP, e.g. minimizing the number of vehicles and the total travel overhead (time/cost). As such, existing methods for MTVRP problem do not tend to benefit the PHVRP problem; 3) The existing MTVRP methods seldom consider multiple depots, thus they do not consider the fact that a customer can be served by multiple depots.

(3) SVRP. The SVRP problem generalizes the traditional VRP problem to which not all clients need to be visited by the vehicles. Variants such as single vehicle routing problem with deliveries and selective pickups (SVRPDSP) [19], and selective multi-depot VRP problem with pricing [33] have been studied. The selective VRP problem has also been applied to post-disaster needs assessments [6], and mobile blood collection system [7]. The PHVRP considered in this paper has significant differences with the above-mentioned works. First, since it targets at serving passengers during peak hours, there are a large number of passengers with tightly pickup time windows, leading to extremely hard problems on large-scale. The PHVRP problem assumes that individual passengers have equal chance of being picked up and should not be distinguishable simply based on individual request information, thus it does not assign a profit revenue to each individual passenger. Hence it is not possible to achieve the pickup selection simply based on the benefit of visiting each individual passenger. In addition, passengers in PHVRP usually have multiple acceptable depots as alternative choices which helps improve the chances of being picked up during the peak hours.

B. Ant Colony Optimization for VRP Problems

Many efforts have been undertaken to find the optimal solutions for a variety of VRP variants [34], [35]. The major drawback of the exact solutions is that computing the optimal solutions requires intolerable computation time, e.g., few hours or even days for a small case with less than one hundred requests. In addition, they usually consider simplified versions of VRP variants and make ideal assumptions on criteria such as sufficient depot/vehicles, vehicle capacity, etc. As such, these approaches cannot be directly applied to real-world problems.

Many reported works have attempted to solve the classical VRP and its variants using the ACO method, due to its powerful adaptation capabilities for addressing hard combinatorial optimization problems [36], [37]. The general ACO for VRP problems includes three main components: pheromone initialization, solution construction and pheromone updating. Existing algorithms typically initialize the pheromone on links based on the length of the link (or travel cost, travel time), or simply set them to equal pheromone [36]. For pheromone updating, three different strategies have been used in existing works: local pheromone update, global pheromone update, and best solution strengthening [37], [38]. To prevent early convergence, the value of pheromone trails on each solution is confined to a predefined interval [39]. In solution construction, the ants rely on the pheromone information to construct vehicle routes. However, due to the capacity constraint, each ant cannot visit all the customers, thus a feasible solution needs to be constructed based on several ants' paths. To tackle this, researchers generally employ the following steps: 1) each ant travels in the road network and visits some customers (not all) to form a path, 2) construct feasible solutions using the paths of all ants by modeling it as a set cover problem [38]. However, the set cover problem is also NP-Hard and it is possible that no feasible solution can be found. The work in [40] first constructs a "big" trip and then divides it into capacity-feasible vehicle trips. Then the resultant trips are served one by one by multiple vehicles following the order in the "big" trip. However, vehicle capacity has a direct effect on the final solutions, and thus should be considered when constructing the paths by the ants. To overcome this problem, the work in [36] proposes to allow each ant to set out again after returning to the depot, until it has visited all customers. In this way, each ant will return a feasible solution, while the capacity constraint is taken into account during construction of each sub-trip. In addition, the existing methods also differ from each other in terms of design strategies for adjusting adaptive parameters [41], search space reduction [42], multi-type ants-usage [36], utilizing hybrid models (e.g. with genetic algorithm [43], particle swarm optimization [44], Tabu search[45]), etc.

Our approach differs from the existing works in optimization objective, methods for constructing solution, initializing and updating pheromone. We focus on the PHVRP problem which has not been considered in the existing works. To fully capture the characteristics of the PHVRP problem, we develop novel strategies for constructing solution, initializing and updating pheromone. Specifically, we carefully designed our strategies for initializing and updating pheromone, by jointly take into account the temporal-spatial distance as well as depot similarity among different travel requests. Different from existing methods, we propose a solution construction scheme, called trip-by-trip (TBT) scheme, which constructs a complete solution by iteratively forming a trip for the vehicle

with earliest end time until no more trips can be formed for any vehicle. The scheme does not try to maximize the number of request visited by each trip, and is shown to be efficient for VRP variants whose objective is to maximize the rate of service rather than system efficiency (e.g. the total travel cost).

In this paper, we focus on the PHVRP problem which has not been considered in previous works. The PHVRP problem has unique characteristics in comparison to other VRP variants. For example, customers are passengers with a much tighter bound on pickup time and waiting time. In addition, each request may specify multiple acceptable depots while different requests have different depots, etc. To fully capture the characteristics of the PHVRP, our proposed method differs from existing approaches in many aspects, including initializing the pheromone as well as constructing the solution. Specifically, our algorithm initializes the pheromone by jointly taking into account the temporal-spatial distance as well as depot similarity among different travel requests, while the existing works typically rely on only travel distance or travel time between two request nodes. We propose a trip-based constraint to model the maximum riding time constraint which is specially designed for the PHVRP problem (whose objective is maximizing the number of served passengers), while the existing works use request-based model. Based on the trip-based constraint model, we develop a trip-by-trip (TBT) scheme which constructs a solution by iteratively forming a trip for inclusion into current route, while other existing routing algorithms typically add a single node into a travel route each time. In addition, we propose an efficient technique to impose routing constraints (e.g. vehicle capacity, pickup time window) by constructing a CNM set during route planning. This can be easily extended to simultaneously impose more constraints considered in other VRP variants.

III. Problem Description and Formulation

A. Road Network and Request Model

The region served by our system is modeled as an undirected graph $RN = (V, E)$, where V is a set of locations (pickup points or depots) and E refers to the road segments connecting the locations. The set V can be partitioned into two parts: a subset P of pickup points, and a subset D of depots where travellers will continue their journey using bus or train. The link $(i, j) \in E$ connecting locations v_i and v_j ($v_i, v_j \in V$) indicates a shortest path between v_i and v_j. Each edge $(i, j) \in E$ is associated with a distance $d_{i,j}$, calculated as the length of the shortest path between v_i and v_j.

It is worth noting that commuters contribute a major part of travel request, which means that the travel demands exhibit periodical similarity among different days. In this work, the travel requests of all customers are known in advance and denoted by R, where each element $r_i \in R$ is a 5-tuple $< s_i, D_i, [e_i, l_i], q_i >$. $s_i \in P$ represents the passenger pickup point. D_i is the set of alternative depots that can meet r_i's requirement, meaning that r_i is regarded as served if the passenger is taken to any location in D. $[e_i, l_i]$ refers to the pickup time window, where e_i and l_i are the earliest and latest pickup timec. q_i denotes the number of seats needed by r_i.

Fig. 1 shows an PHVRP instance. There are 8 pickup locations for passengers, i.e. $P = \{1, 2, \cdots, 8\}$, and 2 alternative destinations, i.e. $D = \{d_1, d_2\}$, and these locations are connected via a set of road segments. Travel requests continuously appear at those pickup locations, and a pickup time window is associated with each request. To illustrate this, we assume that D=$\{d_1, d_2\}$ is the set of alternative depots of each request, meaning that any request can be satisfied if the passenger is taken to either d_1 or d_2. Two vehicles, ve_1 and ve_2, travel among all the pickup locations as well as the depots with the objective of maximizing the number of passengers served by the limited vehicle capacity during peak-hour. To this end, all the vehicles cooperatively perform pickup tasks, and each vehicle decides whether to end its current trip and start a new one or take more requests in the current trip.

Each location in P may have many requests with different time windows, thus each location should be visited multiple times by different vehicle at different time slots. This increases the complexity of problem formulation as well as the algorithm design. Therefore, we construct a demand graph $DG = (V', L)$ based on the travel request set R and the road network RN to solve the PHVRP problem more efficiently. The node set V' is constructed as: (1) For each request $r_i \in R$, we create a node v_i located at pickup point s_i, and v_i has all the properties of request r_i. (2) For each depot $d_j \in D$, we create a node v_j located at depot d_j. The other properties of v_j will be set to empty. The set of links, L, is constructed as follows: For any node pair v_i and v_j in V' we add a link (i, j) to L and the length of the link $dist(i, j)$ is set to the length of the shortest path between v_i and v_j on network RN.

B. Vehicle Trip Model

The system coordinates a fleet of homogeneous vehicles denoted by $VE = \{ve_1, ve_2, \ldots, ve_\kappa\}$, where κ is the number of vehicles and the vehicle ve_i is associated with capacity ϵ_i. We define a *vehicle trip* as a travel path/route where a vehicle starts from a depot (or its original parking location) to pick up passengers and return to a final depot (may be different from the start depot) after all the onboard passengers have alighted at their acceptable depots. Each trip may contain multiple depots if a single depot cannot meet the requirement of all travel requests assigned to the trip. Each vehicle will stay at the last node (a depot) of a trip after finishing the trip.

Fig. 1 shows an example of the PHVRP problem, where there are 8 pickup points, 2 depots and 29 travel requests. For the simplicity of demonstration, each of the requests can be satisfied by both of the two depots (i.e. d_1 and d_2). The pickup time window of each request is illustrated via the length in x-axis. We omit the information on the number of required seats per request, the travel time and distance of each road in the network, due to the limit on the page length. In this instance, 6 trips are constructed based on the constraints of vehicle capacity, pickup time window, and the bound on the maximum travel time of each trip. Three of them will be served by vehicle 1 in sequential order and the others will be served by vehicle 2. Due to the fact that the total travel requests exceed the service capacity of all the available vehicles during peak hours, some of the travel requests will not be served.

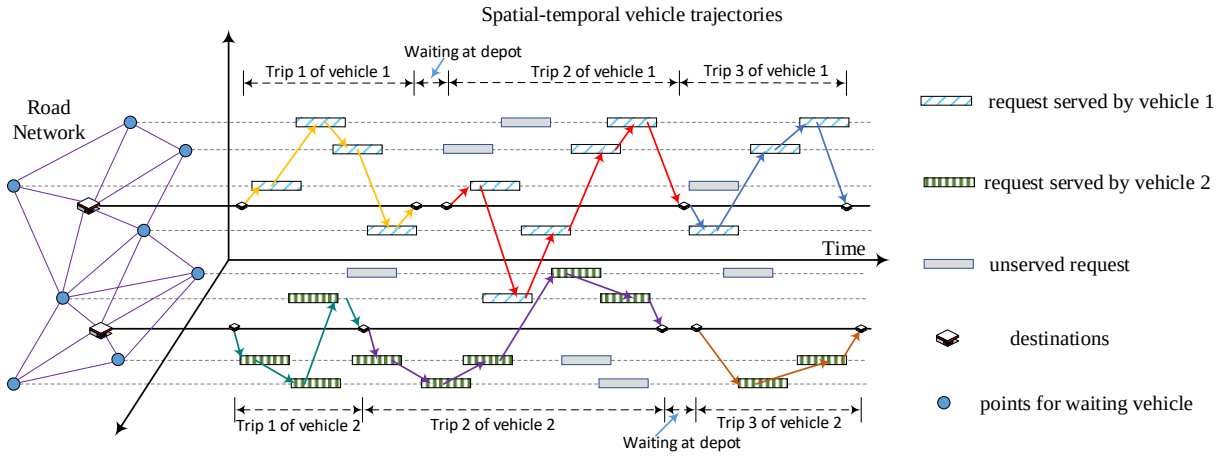


Fig. 1. An instance of PHVRP

We assume that any vehicle will not wait for passengers at a pickup location. This means that a request r can only be served by vehicle ve if ve arrives at r's location within r's pickup time window. The rational is that waiting for passengers will frustrate onboard passengers thus lowering the users' experience. In addition, waiting for passengers will lead to performance loss in terms of the number of served passengers. However, if there are passengers waiting at the pickup location, the vehicle will wait until all the passengers are on board, no matter how long time the boarding process takes. For the case when a vehicle with no onboard passengers stops at a depot, the vehicle can wait for a certain period before starting the next trip. Also, as shown in Fig. 1, vehicles will not wait any passengers *at any pickup point*. However, when vehicle 1 completes its first trip, there is no feasible next move (request node) from the depot d_1, thus it waits *at the depot* d_1 for a certain period before starting its 2nd trip.

We assume that the number of required seats in any request is no larger than the vehicle capacity,

$$q_i \leq \epsilon_j, \quad \forall r_i \in R, ve_j \in VE \tag{1}$$

Let $TR_{i,j}$ be the j-th trip of the i-th vehicle for $1 \leq i \leq \kappa, 1 \leq j \leq n_i$, where κ is the number of vehicles and n_i is the number of trips assigned to the vehicle ve_i. The notation TR_i = $\{TR_{i,1}, TR_{i,2}, \ldots, TR_{i,n_i}\}$ is utilized to denote the set of trips of ve_i. Formally, assume $TR_{i,j} =< v_0, v_1, \cdots, v_{n_{i,j}} >$ ($v_k \in V'$ for $0 \leq k \leq n_{i,j}$), where $n_{i,j}$ is the number of nodes that ve_i needs to visit after leaving depot v_0, and ve_i needs to stop for τ_{v_k} time at node v_k for passengers to alight or board. For $v_k \in TR_{i,j}$,

$$\tau_{v_k} = \tau_0 + \tau_u \cdot q_k + \tau_u \cdot q'_k \tag{2}$$

where τ_0 is the extra time taken due to vehicle acceleration, deceleration, door opening and closing at a pickup point or depot, τ_u is the time required for a passenger to alight or board, q_k and q'_k is the number of passengers at node v_k that need to board and alight respectively. Middle-sized vehicles such as feeder buses with capacity ranging only from 10 to 20 passengers, are used to offer point-to-point services (e.g. deliver passengers to the nearest depots) [1]. These vehicles typically have a single door, and hence boarding and alighting cannot happen simultaneously.

As mentioned earlier, this model allows a vehicle to wait at the depot for a certain period before starting its next trip. Since there are no passengers onboard the bus, this is acceptable. Therefore, the waiting time at the first node, v_0, of the trip can be calculated as

$$\tau_{v_0} = \max\{0,\ e_1 - t(v_0, v_1)\} \tag{3}$$

where $t(v_0, v_1)$ indicates the travel time from node v_0 to node v_1, v_0 is the current node before starting a new trip, v_1 is the next node to visit from node v_0, e_1 is the earliest pickup time of request node v_1. $e_1 - t(v_0, v_1) > 0$ indicates that the vehicle will arrive earlier than the earliest pickup time if it starts the next trip immediately thus needs to wait for passengers at v_1. To avoid that, the vehicle will wait at v_0 for $e_1 - t(v_0, v_1)$ seconds before making a new trip. In this way, the vehicle will arrive at node v_1 at time e_1 and need not to wait the passengers at v_1.

In addition, it is clear that the equation $\sum_{v_k \in TR_{i,j}} q_k = \sum_{v_k \in TR_{i,j}} q'_k$. Thus, the overall riding time $\tau_{i,j}$ of trip $TR_{i,j}$ (for $1 \leq i \leq \kappa, 1 \leq j \leq n_i$), which is defined as the period from the time that an empty vehicle leaves its depot to the time that the vehicle has taken all the visited passengers to their depots and be ready for the next trip (after completing the trip), can be calculated as

$$\begin{aligned}
\tau_{i,j} &= \sum_{k=0}^{n_{i,j}-1} t(v_k, v_{k+1}) + \sum_{k=0}^{n_{i,j}} \tau_{v_k} \\
&= \sum_{k=0}^{n_{i,j}-1} t(v_k, v_{k+1}) + \tau_{v_0} + \sum_{k=1}^{n_{i,j}} (\tau_0 + \tau_1 \cdot q_k + \tau_1 \cdot q'_k) \\
&= \sum_{k=0}^{n_{i,j}-1} t(v_k, v_{k+1}) + \tau_{v_0} + \sum_{k=1}^{n_{i,j}} (\tau_0 + 2\tau_1 \cdot q_k)
\end{aligned} \tag{4}$$

where $t(v_k, v_{k+1})$ indicates the travel time from node v_k to node v_{k+1}, $n_{i,j}$ is the number of nodes the vehicle needs to

visit in trip $TR_{i,j}$, and q_k is the number of required seats corresponding to node v_k.

Since we are not aiming at maximizing the number of passengers for each single trip, we use a stronger constraint by restricting the riding time of the whole trip instead of restricting the ride time of each single request (passengers), in order to reduce the overall computation overhead. It is worth noting that in the formulation of the existing works, it may not be possible to partition a vehicle's travel route into several trips such that any travel request only belong to a single trip. Restricting the ride time of each single request is usually effective for optimizing other objectives, such as minimizing the overall/average travel cost/time of all the requests etc., and it tends to assign much more passengers in a single trip as long as the ride time of each request is not violated. However, the generated vehicle routes produce longer average ride time which is not helpful for maximizing the number of served passengers in the PHVRP problem. In PHVRP, we use a stronger constraint which is not only computationally efficient but it also reduces the average ride time of all travel requests leading to more people served during peak hours. In our formulation, each trip is time-constrained by an upper bound τ_{qos} which guarantees the quality of service for travellers, and exceeding this by τ_{qos} will lead to customer dissatisfaction.

$$\tau_{i,j} \leq \tau_{qos}, \quad 1 \leq i \leq \kappa, 1 \leq j \leq n_i. \tag{5}$$

Similarly, we also adopted a stronger constraint on the vehicle capacity, such that all the passengers assigned to the same trip must not exceed the vehicle capacity ϵ. Formally, for trip $TR_{i,j} =< v_0, v_1, \cdots, v_{n_{i,j}} >$,

$$\sum_{k=0}^{n_{i,j}} q_k \leq \epsilon, \quad 1 \leq i \leq \kappa, 1 \leq j \leq n_i. \tag{6}$$

For the trip $TR_{i,j} =< v_0, v_1, \cdots, v_{n_{i,j}} >$, let $t^a_{v_k}$ be the time by which a vehicle arrives at node v_k ($v_k \in TR$), which can be calculated as

$$\begin{aligned} t^a_{v_k} &= t^a_{v_{k-1}} + \tau_{v_{k-1}} + t(v_{k-1}, v_k) \\ &= t^a_{v_{k-2}} + \tau_{v_{k-2}} + t(v_{k-2}, v_{k-1}) + \tau_{v_{k-1}} + t(v_{k-1}, v_k) \\ &= ... = t^a_{v_0} + \sum_{k=0}^{n_{i,j}-1} (\tau_{v_k} + t(v_k, v_{k+1})) \end{aligned} \tag{7}$$

Since not all requests can be served, a binary vector $A =< a_1, a_2, \ldots, a_n >$ is defined as decision variables, such that $a_k = 1$ if request r_k is selected to be served, otherwise a_k is set to be 0. For a request node v_k whose pickup time is $t^a_{v_k}$, the constraint on pickup time window must be satisfied.

$$e_k \cdot a_k \leq t^a_{v_k} \leq l_k \cdot a_k + (1 - a_k) \cdot \tau_{pk}, \quad 1 \leq k \leq n \tag{8}$$

where $[e_k, l_k]$ indicate the time window of pickup for request r_k, and τ_{pk} indicates the peak hour period (planning period). On one hand, if $a_k = 1$, then $e_k \leq t^a_{v_k} \leq l_k$ must hold, otherwise, $0 \leq t^a_{v_k} \leq \tau_{pk}$ must hold.

For each request node v_k in $TR_{i,j}$, the $TR_{i,j}$ must also contain at least one of r_k's alternative depots,

$$|D_k \cap D(TR_{i,j})| > 0, \qquad v_k \in TR_{i,j} \tag{9}$$

where D_k is the set of alternative depots of request r_k, $D(TR_{i,j})$ is the set of depots that the vehicle will visit during the trip. In addition, each request can only be served by at most one trip,

$$|TR_{i,j} \cap TR_{k,l} \cap R| < 1, \quad i \neq k \text{ or } j \neq l \tag{10}$$

where $TR_{i,j} \cap TR_{k,l} \cap R$ indicates the set of requests contained in both $TR_{i,j}$ and $TR_{k,l}$.

C. Problem Formulation

Problem PHVRP*. Given the set R of n travel requests, the set VE of κ vehicles (with parameters ν, ϵ) and the pickup point set P and depot set D in the form of road network RN, determine a subset of R to serve and produce a set of trips for each vehicle in VE to maximize the serving of requests during the peak hour period τ_{pk}, with consideration of the trip constraints including vehicle capacity, maximum riding time τ_{qos} as well as the pickup time window.* Formally,

$$\max \quad \sum_{1 \leq k \leq n} a_k \cdot q_k \tag{11}$$

$$\text{s.t.} \quad (5), (6), (8), (9), (10)$$

$$t^a_{v_i} \leq \tau_{pk}\,, \qquad v_k \in DG\,, \tag{12}$$

$$a_k \in \{0, 1\}\,, \qquad 1 \leq k \leq n\,, \tag{13}$$

where equations (5), (6), (8) are constraints on each single trip with respect to QoS, vehicle capacity and time window of the travel requests. Constraint (9) guarantees that any selected passenger will be taken to their alternative depots, and constraint (10) indicates that a request will be served by at most one trip. Constraint (12) imposes that any trip cannot exceed peak hour period.

IV. Proposed Approach

This section presents the proposed Ant Colony Optimization (ACO) algorithm based on a trip-by-trip (TBT) routing scheme, denoted as `TBT-ACO`, to solve the PHVRP problem.

A. Preliminary on ACO Method

The ACO method is a population-based approach, which searches the solutions by simulating food-seeking behaviors of ant colonies in nature. Real ants find a short path to a food source in the following way. Initially, ants wander randomly, and upon finding food return to their colony while laying down pheromone trails on the path traveled. If other ants find such a path, they are likely to follow the trail, returning and reinforcing it if they eventually find food. The pheromone trail also evaporates over time if no new pheromone is added. In this way, a short path gets marched over more frequently, and the pheromone density becomes higher on shorter paths than longer ones. This results in many ants following a single path.

The idea of the ant colony algorithm is to mimic this behavior with "simulated ants" walking around the graph that represents the problem to solve. Artificial ants locate optimal solutions by moving through a parameter space representing

all possible solutions, where better solutions correspond to short paths in the natural world. The artificial ants record their positions and the quality of their solutions via simulated pheromone so that in the subsequent simulation iterations, more ants locate better solutions. The ACO algorithm works in an iterative way, and it generally includes three main steps in each iteration, namely, initializing, solution construction and pheromone updating. In our problem, the roadways connecting consecutive pickups are the solution components to construct the travel routes. The first step initializes the amount of pheromone associated with each solution component. In the second step, ants utilize solution components to construct feasible solutions (i.e. the travel routes) based on the pheromone trails such that roadways with more pheromone are more likely to be used by ants in constructing their feasible solutions in the next iteration. In the third step, the ants with good solutions will update the pheromone to the solution components included in their solutions.

B. Main Algorithm

The proposed TBT−ACO starts by constructing a demand graph $DG = (V', L)$ using travel requests and depots. With the demand graph DG, the algorithm TBT−ACO solves the problem in several iterations. α solutions (by α ants) are generated in each generation population based on the pheromone distribution, and each solution consists of κ feasible sets of trips (for κ vehicles). At the end of each iteration, after α solutions are generated, the algorithm chooses some ants with good performance and allows these ants to deposit some pheromone on the corresponding links. Solution components (links) with larger amount of pheromone are more likely to be used by ants in constructing their feasible solutions in the next iteration.

As shown in Algorithm 1, TBT−ACO first builds the demand graph, then initializes the pheromone based on spatial-temporal distance information as well as the destinations information of different travel requests (line 3). Next, it performs λ iterations of operations during which λ generations of ant populations are generated (lines 5-16). In each generation, a complete solution is constructed for each of the α ants, resulting in a population of α solutions that are generated using the TBT scheme based on the pheromone information. Specifically, the entire inner while-loop (lines 10-13) is the process for an ant to construct one complete solution for κ vehicles, using the *trip-by-trip (TBT) scheme*, i.e. iteratively constructs a trip for the vehicle with earliest finish time, as shown in Fig. 2. In Fig. 2(a), vehicle ve_2 has the earliest finish time t_1 ($t_1 < peakRange$ meaning ve_2 can serve more trips during the peak hour period), thus algorithm TBT−ACO forms a trip $TR_{2,2}$ for vehicle ve_2; then ve_3 has the earliest finish time t_2 as in Fig. 2(b), thus TBT−ACO forms a trip $TR_{3,2}$ for vehicle ve_3. At this stage, ve_1 has the earliest finish time t_3 as in Fig. 2(c), thus TBT−ACO forms a new trip $TR_{1,2}$ for vehicle ve_1. The construction of a new trip is achieved by the procedure SubTrip (line 12), which constructs a trip using TBT scheme for the vehicle with earliest finish time. After each of the α ant has obtained their corresponding solutions,

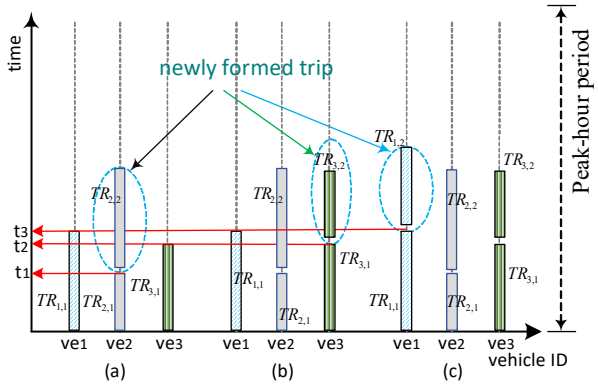


Fig. 2. Illustrative example of the solution construction based on TBT scheme

TBT−ACO evaluates all the solutions and updates the historical best solution if better results are found. Then, the TBT−ACO chooses some ants with good performance and lets these ants deposit more pheromone on the corresponding links which are used for constructing their solutions (trips). The whole process will repeat λ times.

Algorithm 1: TBT−ACO

Input: Travel request set R, road network RN connecting all pickup points and destinations, vehicle set VE, parameters $\tau_{pk}, \tau_{qos}, \nu, q, \lambda$

Output: Decision vector A, set TR_i of travel trips for each $ve_i \in VE$, the arrival time $t^a_{v_k}$ of a vehicle at the node v_k ($v_k \in DG$).

```
1  begin
2  Construct demand graph DG = (V', L);
3  Initialization(); /* Pheromone Initialization */
4  iter ← 1;
5  while (iter < λ) do
6      for (ant ← 1 to α) do
7          TR_i ← ∅, for each vehicle ve_i;
8          /* Construct a solution based on TBT scheme */
9          while (true) do
10             k ← arg min_{1≤i≤κ} {ve_i.ft}; /* vehicle with earliest
                 finish time */
11             if ve_k.ft ≥ τ_pk then break;
12             TR ← SubTrip();
13             TR_k ← TR_k ∪ {TR};
14         Solution quality evaluation;
15         updata historical best solution if applicable;
16         pheromoneUpdate();
17 end
```

C. Pheromone Initialization

In the algorithm, the links in L are components for constructing solutions, and each link is initialized with certain amount of pheromone. A larger amount of pheromone on a link, (i, j), indicates a higher probability that the link will be used (i.e. moving to node v_j from current node v_i) by ants in constructing their respective routes. Existing approaches

usually consider the distance between v_i and v_j. In this paper, we jointly consider the spatial-temporal distance information as well depot similarity information, between the two request nodes. The new initialization method helps to optimize the pheromone distribution in the early stage. The initialization strategy aims to let the link connecting two requests, which are close to each other in spatial-temporal dimension or have similar dest depots, to have more pheromone so that the two requests will have higher chance to be included into the same trip.

① For a link (i, j) connecting *two request nodes* v_i and v_j: Note that requests r_i and r_j have locations s_i and s_j, depot set D_i and D_j, time windows $[e_i, l_i]$ and $[e_j, l_j]$, respectively. We use set CD to denote the common set of the depot set D_i and D_j, i.e. $CD = D_i \cap D_j$. The similarity of destination sets between r_i and r_j is calculated as

$$sim\_dest = (\frac{|CD|}{|D_i|} + \frac{|CD|}{|D_j|})/2. \tag{14}$$

The distance factor, denoted as $close\_fact$, which characterizes the spatial-temporal distance between two requests r_i and r_j, is calculated as follows.

- If r_i and r_j are located at the same position (i.e. $s_i = s_j$) and their time window overlaps with each other ($e_i \leq l_j$ and $e_j \leq l_i$), then

$$close\_fact = \frac{\max\{0, \min\{l_i, l_j\} - \max\{e_i, e_j\}\}}{\delta\tau}$$

  where $\delta\tau = ((l_i - e_i) + (l_j - e_j))/2$ is the average length of the time windows of the two requests.
- If $s_i \neq s_j$ and a vehicle can reach s_j at a time instance in $[e_j, l_j]$ after picking up request r_i or vice versa, then

$$close\_fact = \frac{MAX - dist(s_i, s_j)}{MAX - MIN}$$

  where MAX and MIN are the maximum and minimum length of direct links in the road networks RN.
- If $s_i \neq s_j$ and a vehicle cannot reach s_j without violating time window constraint after picking up request r_i or vice versa, then

$$close\_fact = 0$$

Based on the above sim_dest and $close\_fact$, the amount of initial pheromone for the link connecting two requests is calculated as

$$phe_{i,j} = sim\_dest/2 + close\_fact/2. \tag{15}$$

② For a link (i, j) connecting *two depot nodes* or connecting *a request node and a depot node*: The pheromone is initialized based on the distance between the two nodes, i.e., the sim_dest is calculated as 1, and $close\_fact$ is calculated as $\frac{MAX - dist(s_i, s_j)}{MAX - MIN}$.

D. Trip Construction

This section introduces how an ant intelligently constructs the travel route for κ vehicles based on pheromone information. Since `TBT-ACO` iteratively constructs a trip for the vehicle with earliest finish time, in the following we only introduce how to construct a trip $TR_{i,j}$ for vehicle ve_i.

During the trip construction, the trip $TR_{i,j}$ can be partitioned into two segments: the first part only contains request nodes (request-segment denoted as TR_{req}) and the second part only contains depots (depot-segment denoted as TR_{dep}). Initially, the vehicle ve is located at a depot (end of last trip), and both TR_{req} and TR_{dep} are empty. At any time instance, $TR_{i,j} = TR_{req} \cup TR_{dep}$ and each node $v \in TR_{req}$ has an acceptable depot in TR_{dep}. The procedure `SubTrip` (i.e. Algorithm 2) repeatedly adds a request node (and its depot when necessary) into $TR_{i,j}$, by performing the two procedures, i.e., 1) identify the candidate set, CNM, of nodes for next move, and 2) select a candidate from CNM for inclusion into $TR_{i,j}$ as the next move. The last node in TR_{req}, say u, is called the current node in TR_{req}, and the algorithm repeatedly searches a next move/node for u, by performing the above mentioned two procedures. The first procedure finds all valid nodes that can be inserted after node u without violating the capacity, time-window and QoS constraints. Then the second procedure chooses one node from the candidate set CNM based on pheromone information. We next explain the two procedures in detail, i.e. how to construct the set CNM of candidate next moves for the current node u and how to select a node from CNM as u's subsequent node.

Algorithm 2: `SubTrip()`

```
1  TR_{i,j}, TR_req, TR_dep ← ∅;   /* initialize as empty */
2  while true do
3      CNM ← CNM_Construction();
4      nm ← Next_Move_from_CNM();
5      if nm is a request node then
6          put nm to the end of TR_req;
7          update ve.ft and ve.load;
8          if TR_dep ∩ Depot(nm) = ∅ then
9              nd ← choose a depot from Depot(nm);
10             if nd is the MRT station; then
                   TR_dep ← {nd};
11             else insert nd into TR_dep with minimum
                   trip-length increase.
12     else if nm is a depot node then
13         update ve.ft and ve.load based on TR_dep;
14         return TR_{i,j};
```

1) CNM Construction: Suppose we are constructing a trip $TR_{i,j}$ for the vehicle ve_k. Initially, both TR_{req} and TR_{dep} are empty, and the current node u of TR is the last node of ve_k's previous trip. The TR_{req}, TR_{dep} and current node u will be iteratively updated as the algorithm progresses. Given the current node u, the procedure `CNM_Construction` constructs the set CNM consisting of all valid candidate next moves of u. It checks each of the unserved request nodes, say v_i, to see if capacity constraint, time window constraint and QoS constraint are met. If any of the three constraints is violated, the node v_i will not be considered.

- The capacity constraint (lines 3-4) requires that the cur-

rent load (from requests in TR_{req}) plus the amount of v_i's request should not exceed the vehicle's capacity ϵ.

- The time window constraint (lines 5-9) requires that the vehicle currently located at u should be able to arrive at node v_i within v_i's time window $[e_i, l_i]$. The arrival time can be calculated as $t^a(v_i) \leftarrow t^a(u) + t(u, v_i) + \tau(u)$, where $t^a(u)$ is the arrival time at u, $t(u, v_i)$ is the travel time from node u to node v_i and $\tau(u)$ is the stopping time at node u which have been previously defined. There are two cases: 1). If u is a depot, it only needs to check whether $t^a(v_i)$ exceeds l_i, as the vehicle can wait for a certain period at the depot if $t^a(v_i) < e_i$. 2). Otherwise if u is a request node, then it requires that the $t^a(v_i)$ falls into $[e_i, l_i]$, since the vehicle cannot wait for passengers at the request node.
- The QoS constraint (lines 10-15) requires that the resultant trip travel time which includes node i and its dest depot (if $D_i \cap TR_{dep} = \emptyset$) should not exceed the QoS limit. If $D_i \cap TR_{dep} = \emptyset$, the node that is closest to the last node of TR_{dep}, say ln, will be selected and inserted in TR_{dep} after node ln.

Algorithm 3: `CNM_Construction()`

Input: current node u, request-segment TR_{req}, depot-segment TR_{dep}, vehicle capacity ϵ, trip start time t_s, vehicle ve.
Output: CNM.

1 $CNM \leftarrow \emptyset$; /* initialize CNM as empty set */
2 **for** each unserved request node v_i **do**
3 /* **(1):** Capacity constraint */
4 **If** $ve.load + q_i > \epsilon$ **then** continue;
5 /* **(2):** Time-window constraint */
6 **if** $u \in D$ /*vehicle locates at a depot*/ **then**
7 **If** $t^a(u) + t(u, v_i) + \tau_u > l_i$ **then** continue;
8 **else**
9 **If** $t^a(u) + t^a(u) + t(u, v_i) + \tau_u > l_i$ or $t^a(u) + t^a(u) + t(u, v_i) + \tau_u < e_i$ **then** continue;
10 /* **(3):** QoS constraint */
11 $d \leftarrow null$;
12 **if** $D_i \cap TR_{dep} = \emptyset$ **then**
13 $ln \leftarrow$ last node of TR_{dep};
14 $d \leftarrow \arg\min_{v_j \in D_i}\{\ dist(v_j, ln)\ \}$;
15 **If** $\tau(\{v_i, d\} \cup TR_{req} \cup TR_{dep}) > \tau_{qos}$ **then** continue;
16 $CNM \leftarrow CNM \cup \{v_i\}$;

2) Next Move Selection: As mentioned before, the algorithm is capable of allowing a vehicle to end a trip $TR_{i,j}$ without including more nodes, even when there are many valid candidate nodes to visit from u without violating any constraints. To achieve this, we develop an algorithm `Next_Move_from_CNM`, which regards the set TR_{dep} as a special candidate node of the set CNM, the $|CNM|+1$-th node. Different candidates have different probabilities of being chosen based on the pheromone distribution. The probability that the i-th node in CNM, denoted as $CNM(i)$, will be chosen is calculated as follows.

$$p_i \leftarrow \frac{ph_{u,CNM(i)}}{tp + tp\_dest} \tag{16}$$

where $CNM(i)$ is the i-th node in CNM, $ph_{u,CNM(i)}$ is the pheromone on the link $(u, CNM(i))$, $tp = \sum_{v \in CNM} ph_{u,v}$ is the total pheromone of links connecting u and nodes in CNM, and $tp\_dest = \sum_{v \in TR_{dep}} ph_{u,v}$ is the total pheromone of links connecting u and nodes in TR_{dep}. The probability that the special node (i.e. TR_{dep}) will be chosen, meaning no more request node will be added into current trip, is calculated as $p_{dep} \leftarrow \frac{tp\_dest}{tp + tp\_dest}$. The algorithm selects the next move using algorithm 4.

Algorithm 4: `Next_Move_from_CNM`

Input: candidate set CNM, current node u, depot-segment TR_{dep} of current trip, pheromone matrix $ph_{i,j}$.
Output: node for next move nm.

1 $tp \leftarrow \sum_{v \in CNM} ph_{u,v}$; /* total pheromone of all links connecting u and nodes in CNM */
2 $tp\_dest \leftarrow \sum_{v \in TR_{dep}} ph_{u,v}$; /* total pheromone of all links connecting u and nodes in $TR_{|CNM|+1}$ */
3 $p_i \leftarrow \frac{ph_{u,CNM(i)}}{tp + tp\_dest}$, $1 \le i \le |CNM|$; /* chance of choosing v_i */
4 $P_i \leftarrow \sum_{1 \le j \le i} p_j$, $1 \le i \le |CNM|$; /* accumulated probability */
5 $p_{|CNM|+1} \leftarrow \frac{tp\_dest}{tp + tp\_dest}$; $P_{|CNM|+1} \leftarrow 1$; /* set TR_{dep} will be treated as a special node */
6 $r_a \leftarrow random(0, 1)$;
7 $v_i \leftarrow \arg\min_{1 \le j \le |CNM|+1}\{P_j | P_j \ge ra\}$;
8 **if** $i \le |CNM|$ **then** $nm \leftarrow CNM(i)$;
9 **else** $nm \leftarrow$ the first node in TR_{dep};

E. Pheromone Update

At the end of each iteration, the pheromone trail will be updated after all ants obtained their respective solutions. Let $ph_{avg} = (\sum_{(u,v) \in TR} ph_{u,v})/(|TR| - \kappa)$ be the average pheromone of trajectory TR, where κ is the number of vehicles and $|TR| - \kappa$ is the number of links in TR. To simulate the pheromone evaporation, the pheromone on each link is multiplied by a residual concentration factor $(1 - \rho_0)$, where ρ_0 is the evaporation coefficient $(0 \leqslant \rho_0 \leqslant 1)$.

Pheromone update consists of two parts: 1) best solution strengthening and 2) historical update. In part 1), for the best solution in the current iteration, the pheromone along the trajectory will be strengthened by ph_{avg}. In part 2), any solution that is better than the historical best solution, the historical updating rule is applied, with the increase of ph_{avg} for each link along the trajectory. At the beginning of the algorithm, the historical best solution is not good enough, thus many ants can produce solutions better than the historical best solution, leading to pheromone updating over a large number of links (by many ants). As the algorithm continues, the historical best solution improves. Hence, fewer ants can produce better solutions, leading to pheromone updating over relatively fewer links. In this way, the convergence speed is relatively faster at the early stage and then gradually slow down as the algorithm advances.

Runtime analysis: ① We can easily derive that both pheromone initialization and updating run in $O((n + |D|)^2)$, where $n + |D|$ is the number of nodes in the demand graph. ② Algorithm 3 runs in $O(n)$ time, and Algorithm 4 runs

in $O(|CNM|) < O(n)$ time. ③ During the procedure `Solution_by_TBT`, the number of CNM constructions (Algorithm 3) and next move selection (Algorithm 4) will repeat at most $n + |D|$ times. ④ Based on ③ and ④, each ant requires $O(n \cdot (n + |D|))$ time to produce a complete solution, thus each generation requires $O(\alpha \cdot (n^2 + n \cdot |D|)) = O(\alpha \cdot n^2)$ time, as $|D|$ is much smaller than n. ⑤ The overall time running time consists of pheromone initialization ($O((n + |D|)^2)$) at the beginning, producing λ generations ($O(\lambda \cdot (\alpha \cdot n^2 + (n + |D|)^2))$), and update the pheromone trail after each generation ($O(\lambda \cdot ((n+|D|)^2))$). Therefore, the time complexity is $O((n+|D|)^2)$ + $O(\lambda \cdot (\alpha \cdot n^2 + (n+|D|)^2))$ + $O(\lambda \cdot ((n+|D|)^2))$ = $O(\lambda \cdot (\alpha \cdot n^2 + (n+|D|)^2))$ = $O(\lambda \cdot \alpha \cdot n^2)$, where λ is the iteration bound of algorithm `TBT-ACO`, α is the number of ants, and n is the number of travel requests.

V. Performance Evaluation

A. Datasets

Road Network: We construct the road network covering the area as shown in Fig. 3, which is exported from OpenStreetMap[2]. Singapore has relatively dense deployment of bus stops, with an average distance of 300 meters between two consecutive stops. Thus, the bus demand information can well reflect the distribution of travel requests. As such, in the experiments, we use bus stop location to simulate pickup points to test the performance of our vehicle routing algorithms. To distinguish the pickup points from depots, we select 37 bus stops to simulate the pickup points and choose 10 different bus stops to be the candidates of passengers' acceptable alternative depots.

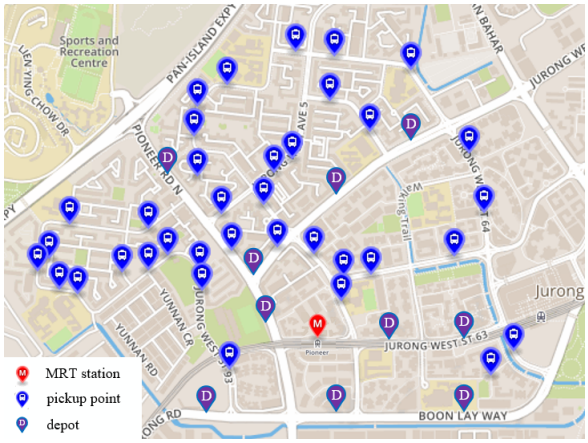


Fig. 3. Road network used for experiments, which contains 37 pickup points (set P), 10 alternative depots (set D) and one MRT station (i.e. pioneer station).

Travel requests: The travel requests are generated based on real-world bus travel demand information, which is publicly available at DataMall, LTA, Singapore[3]. With the bus demand data (the total passenger volume at each bus stop in each hour), we are able to generate reasonable request dataset to evaluate our proposed vehicle routing algorithm. Fig. 4 shows the passenger volumes (departure) of different bus stops at different hours of the day. Each curve illustrates the passenger volume of an individual bus stop. The figure shows that different bus stops have significant different patterns of passenger volumes. During different hours of the day, the average passenger volume of all the bus stops is 22.49 with variation of 374.06. For each individual bus stop, the average passenger volume ranges from 1.61 to 98.88, with the variation ranging from 1.81 to 9892.51.

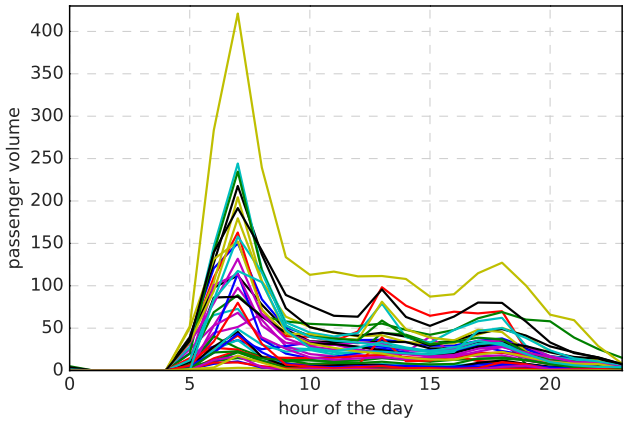


Fig. 4. Passenger volumes at different bus stops

Given the total number of passengers at the pickup point p_j, i.e. m_j, we generate $\rho \cdot m_j/SPR$ travel requests, where $\rho \in [0, 1]$ (e.g. demand ratio $\rho = 0.5$ means that the generated requests equal to 50% of the real bus demand), SPR is the average seats per request. Thus for pickup point p_j, we generate a set of travel requests $\{r_i\}$ such that $\sum q_i = \rho \cdot m_j/SPR$. Specifically, $r_i =< s_i, D_i, [e_i, l_i], q_i >$ is generated in the following way: 1) origin stop s_i is set as p_j; 2) q_i is randomly generated in $[1, 2 * SPR]$; 3) the earliest pickup time e_i is randomly generated in the time period following uniform distribution, while l_i is set to be $e_i + \theta$ where θ (size of pickup time-window) is set to 10 minutes; 4) Each request r_i has a set D_i consisting of multiple alternative depots. In real-world scenario, the alternative depots will be provided by the passengers. In the experiment, the Pioneer metro station is included in each r_i's D_i. In addition, each request has $|D_i| - 1$ extra alternative depots, which are selected from set D (as shown in Fig. 3) as those have minimum distance for connecting the origin s_i and the metro station. 5) If there are multiple requests of the same e_i and D_i, we merge them into a single request.

All the methods use the same inputs, i.e. road network, request set, vehicle configurations and constraint parameters. If not specified, the experiment parameters are set as follows: 1) In request generation, $\rho = 0.5$, the number of alternative depots of each request is randomly generated in range [1,3], the number of required seats of each request is randomly selected from [1, 5]. 2) In algorithm `TBT-ACO`, the population size α is set to 20, iteration bound λ is set to 50, evaporation coefficient $\rho_0 = 0.2$. 3) Without loss of generality, we assume that each vehicle is with capacity of 20 seats in the

[2]https://www.openstreetmap.org/export
[3]https://www.mytransport.sg/content/mytransport/home/dataMall.html

experiments. The travel time between node v_i and v_j, i.e. $t(v_i, v_j)$, is calculated $\frac{dist(v_i,v_j)}{\nu}$ where travel speed ν is set to 5 m/s. Note that our approach can be easily extended to the scenario with heterogeneous traffic situations by replacing the method for calculating $t(v_i, v_j)$. Existing machine learning based methods (e.g. deep learning based method) showed promising performance for travel time prediction and thus can be used to provide accurate estimation of $t(v_i, v_j)$ based on historical vehicle travel data. In comparison with optimal solution on small-scale instances, $\kappa = 2$ vehicles are used for experiment; and there are $\kappa = 20$ vehicles in total when testing on large-scale instances. 4) The QoS constraint of trip riding time τ_{qos} is set to 15 minutes. 5) In equation (2) for calculating vehicle dwell time at a pickup point, $\tau_0 = 5$ seconds, and $\tau_u = 1.5$ seconds [46].

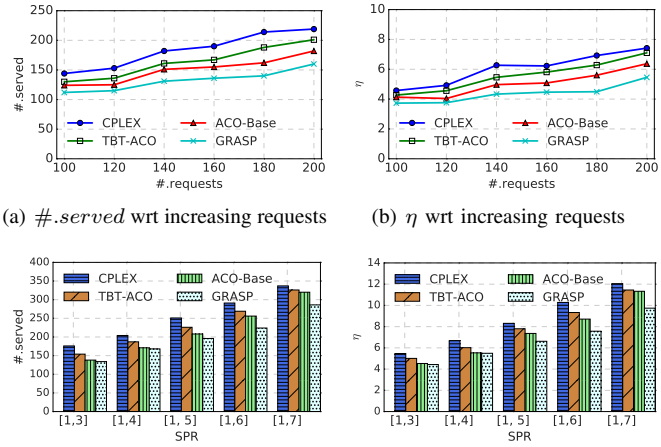
B. Baseline Methods and Evaluation Metrics

Since there is no existing work on the same problem, we compare the proposed method with optimal solutions for small size instances, and with heuristic solutions for large-scale instances. First, the optimal solutions are obtained using CPLEX [47]. For large-scale instances, it is not possible to obtain optimal solution in short time (e.g. 24 hours). Thus we develop the following baseline methods to evaluate the proposed ant colony algorithm `TBT-ACO` as well as the TBT routing scheme. The baseline method is also an ACO based algorithm, denoted as `ACO-Base`, which simultaneously constructs κ travel route for the κ vehicles. Similar to `TBT-ACO`, the `ACO-Base` repeatedly selects a vehicle that has the earliest finish time of current route and then chooses a next stop to include into the route, until no route can accept more requests. There are two main differences between `TBT-ACO` and `ACO-Base`: 1) When identifying the next candidate stops, `ACO-Base` uses the request-based QoS constraint, i.e. the total in-vehicle riding time for any passenger is bounded by τ_{qos}. In contrast, the `TBT-ACO` uses trip-based constraint (formula (5)). 2) The `ACO-Base` does not use the TBT scheme, such that only a single request is included into a vehicle's route in each iteration, while in `TBT-ACO` a trip is formed for a selected vehicle in each iteration. In addition, we also adapt the GRASP Algorithm [48] as a baseline method, which is designed to solve the VRP with time windows and multiple vehicles. Similar to `ACO-Base`, GRASP also uses the request-based QoS constraint.

The performance metrics include: 1) the number of served passengers $\#.served = \sum_{i=0}^{n} a_i q_i$; 2) service efficiency η, i.e. the served passengers by travelling one kilometer, $\eta = 1000 \cdot \frac{\#.served}{\sum_{(u,v)\in TR} dist(u,v)}$; 3) average in-vehicle riding time t^r_{avg} of passengers; and 4) algorithm running time.

C. Results and Analysis

1) Comparison with optimal solutions on small-size instances. The CPLEX program runs in parallel on a workstation with 56 cores (Intel(R) Xeon(R) CPU E5-2660 v4@2.00GHz) and 128G RAM. The other methods run on the same server but use a single core. We test the instances with request number no more than 200, as the CPLEX method cannot find the optimal solution within 20 hours for larger instances. The running time of CPLEX method ranges from 7.8 to 18.7 hours when the number of travel requests ranges from 100 to 200.



(a) $\#.served$ wrt increasing requests (b) η wrt increasing requests

(c) $\#.served$ wrt increasing SPR (d) η wrt increasing SPR

Fig. 5. Comparison of CPLEX, `ACO-Base` and the proposed `TBT-ACO` in terms of number of served passengers ($\#.served$) and service efficiency (η). Each record is averaged over 2 random instances.

Fig. 5 compares the results obtained by the CPLEX, the `TBT-ACO` and other baselines in terms of the number of served passengers ($\#.served$) and service efficiency (η). On relatively smaller problem instances, `ACO-Base` obtains good results that are close to the optimal solutions, with deviation ranging from 5.87% to 13.97% for all the cases evaluated. The results of `TBT-ACO` are better than `ACO-Base` as `TBT-ACO` adopts trip constrained problem formulation and TBT scheme to intelligently decide whether to end current trip or take more passengers. We also observe from Fig. 5 (a) and (b) that all methods tend to obtain higher $\#.served$ and η with the increasing number of travel requests. This is because in the PHVRP problem, the vehicles do not have sufficient capacity to serve all requests. As such, they can only selectively serve as much as possible. More requests provide more opportunities to enable the algorithms to strategically select better pickup points to visit, which helps to improve the number of served passengers and system efficiency. Fig. 5 (c) and (d) evaluate the results with respect to increasing seats per request (SPR). All the methods tend to obtain higher $\#.served$ and η with increasing SPR. This is because higher SPR increases the travel demands at each pickup point, which indirectly reduces the number of stops a trip need to visit. Hence, the vehicles have more freedom to select beneficial pickup points to visit.

Fig. 6 illustrates the algorithm running time of algorithm `TBT-ACO` on relatively smaller instances, in comparison with `ACO-Base`, `GRASP`, and the optimal method CPLEX. In this figure, the experiment setting is the same with that in Fig. 5. From the figure, we observe that the CPLEX method needs nearly 19 hours to obtain the optimal solution when the number of travel requests is 200. The runtime exceeds 24 hours when the requests exceed 200. The other heuristic methods, i.e. our `TBT-ACO`, `ACO-Base`, and GRASP, use much less time. Specifically, the runtimes of `TBT-ACO`, `ACO-Base` and GRASP range from 21 to 72 seconds, from 22 to 76 seconds,

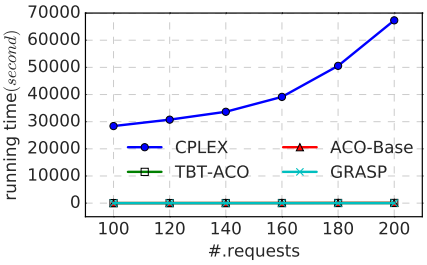


Fig. 6. Comparison on algorithm running time

and from 0.7 to 5.2 seconds, respectively.

2) Comparison on Large-size Instances. Fig. 7 compares TBT−ACO with the baseline methods in terms of the number of served passengers $\#.served$, efficiency η, the average riding time t^r_{avg} and the algorithm running time. All the algorithms tend to achieve higher $\#.served$ and η with increasing ρ (corresponding to increasing travel requests), as shown in Fig. 7 (a) and (b), due to the same reasons discussed in the previous section. Fig. 7 (c) shows that the average passenger riding time t^r_{avg} slightly decreases with the increasing ρ, because larger ρ enables the vehicles to visit lesser stops to pickup passengers. This reduces the trip distance leading to the reduction in t^r_{avg}. The comparison between TBT−ACO and ACO−Based also indicates that the trip based constraint does not necessarily lead to short passenger trip. Fig. 7 (d) compares the algorithm runtime. All the algorithms spend more time when the number of requests increases. Algorithm GRASP uses the least time but the resultant results are not good. TBT−ACO runs slightly faster than ACO−Base while achieving better $\#.served$ and η at the same time. This indicates that, for the PHVRP problem, it is necessary to utilize the proposed trip-based route constraint and the TBT scheme to search for solutions. Note that the algorithm runtime of TBT−ACO and the baselines is obtained using a single processor, and it can be significantly accelerated as each stage of ACO based algorithm (e.g. pheromone initialization and update, solution construction for all ants, etc.) can be implemented in parallel.

3) Impacts of System Parameters. Fig. 8 illustrates the impacts of parameters on the algorithms. The investigated parameters include the Size of pickup Time Window (STW), the maximum ride time τ_{qos} of a single trip, the seats per request (SPR) and the number of alternative depots of the requests (ADR). Fig. 8 (a) and (e) indicate that larger window of pickup time leads to higher $\#.served$ and η, because it increases the chance for a passenger to be picked up. On the other hand, from the vehicle's perspective, it increases the choice of next stops from the current pickup point. Fig. 8 (b) and (f) show that the $\#.served$ and η first decrease with the increasing τ_{qos}, and then becomes relatively stable when τ_{qos} exceeds 20 minutes. This is because when τ_{qos} is small, increasing τ_{qos} results in taking more passengers in a single trip, which reduces the number of trips each vehicle can make. After a certain point, τ_{qos} cannot significantly affect $\#.served$ and η due to the limited capacity of each vehicle. Fig. 8 (c) and (g) show that larger SPR results in higher $\#.served$ and η, due to the same reasons as we discussed in previous sections. Fig. 8 (d) and (h) demonstrate that the increase in ADR can increase $\#.served$ and η, as this potentially reduces the number of depots to visit in order to satisfy all the onboard passengers. This saves time for the vehicle and enables it to make more trips.

VI. Conclusions

This paper studied the PHVRP problem to plan the travel routes for a vehicle fleet to deliver passengers from their doorstep to the depots, from where the passengers can continue their journeys using fixed-route buses or trains. We formulated the PHVRP problem with the objective of maximizing the number of served passengers under constraints on vehicle capacity, pickup time windows, and QoS in terms of riding time. In particular, we characterized the riding time constraint for each vehicle trip instead of for each passenger request, which led to significantly lower computation cost. We proposed a TBT scheme which constructs a vehicle travel route by repeatedly forming a single trip for the vehicle until it cannot accept any more trips. Based on the TBT routing scheme, we proposed a novel ant-colony optimization algorithm which can intelligently decide whether to end its current trip or take more passengers. We demonstrated the effectiveness of the proposed method through extensive experiments by comparing with optimal solutions on small size instances and with heuristic solutions on large size instances, using road network in Singapore and synthetic travel requests that are generated based on real bus travel demands.

References

[1] Zhang, D., Zhao, J., Zhang, F., Jiang, R., He, T. and Papanikolopoulos, N., "Last-Mile Transit Service with Urban Infrastructure Data," *ACM Trans. Cyber-Phys. Syst.*, vol. 1, no. 2, p. 6, 2016.

[2] H. Wang, "Routing and scheduling for a last-mile transportation system," *Transport. Sci.*, vol. 53, no. 1, pp. 131-147, 2017.

[3] Montoya-Torres, J.R., Franco, J.L., Isaza, S.N., Jimnez, H.F. and Herazo-Padilla, N., "A literature review on the vehicle routing problem with multiple depots," *Comput. Ind. Eng.*, vol. 79, pp. 115-129, 2015.

[4] Ke, L., Feng, Z., "A two-phase metaheuristic for the cumulative capacitated vehicle routing problem," *Comput. Oper. Res.*, vol. 40, no. 2, pp. 633-638, 2013.

[5] Despaux, F., Basterrech, S., "A study of the Multi-Trip vehicle routing problem with time windows and heterogeneous fleet," in *Proc. IEEE 14th Int. Conf. Intell. Syst. Des. Appl. (ISDA)*, pp. 7-12, Okinawa, Japan, 2014.

[6] Balcik, B., "Selective routing for post-disaster needs assessments," *Dyn. Disaster.-Key Concepts, Models, Algorithms, and Insights, Springer Proc. Math. Stat.*, vol. 185, pp. 15-36, 2016.

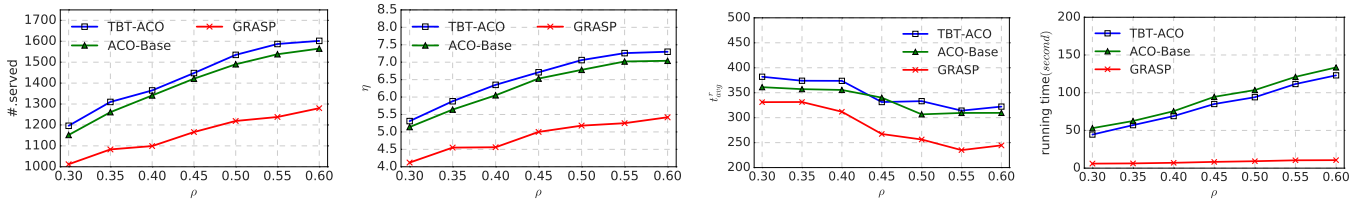
[7] ahinyazan, F.G., Kara, B.Y. and Taner, M.R., "Selective vehicle routing for a mobile blood donation system," *Eur. J. Oper. Res.*, vol. 245, no. 1, pp. 22-34, 2015.

[8] Dantzig, George B., John H. Ramser, "The truck dispatching problem," *Manag. Sci.*, vol. 6, no. 1, pp. 80-91, 1959.

[9] Shahab, M.L., Utomo, D.B. and Irawan, M.I., "Decomposing and solving capacitated vehicle routing problem (CVRP) using two-step genetic algorithm (TSGA)," *J. Theoretical Applied Inf. Technol.*, vol. 87, no. 3, pp. 461-468, 2016.

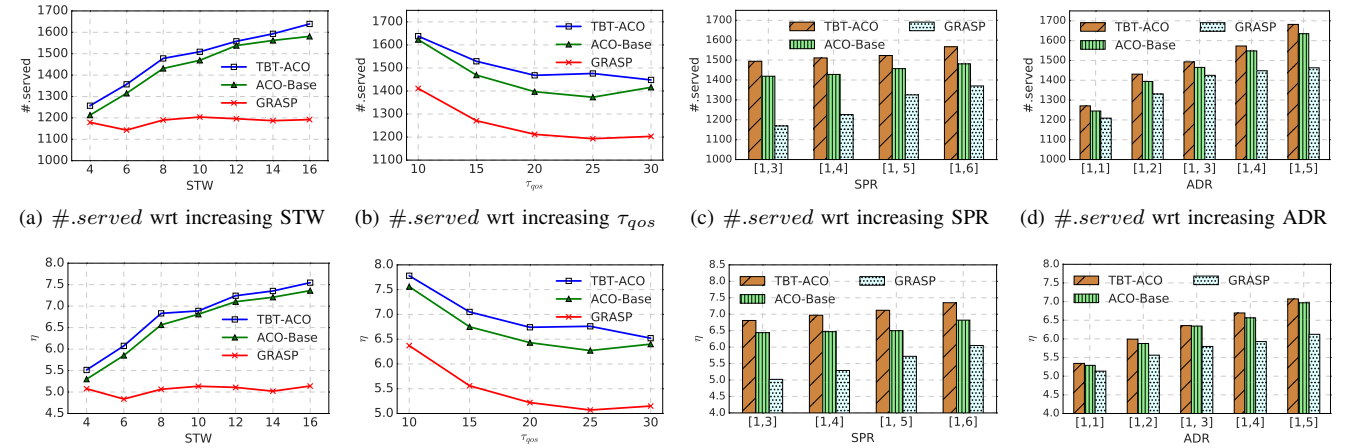
[10] Golden, B., Wasil, E., Kelly, J., & Chao, I., "The impact of metaheuristics on solving the vehicle routing problem: Algorithms, problem sets, and computational results," *Fleet management and logistics, Centre for Res. on Transp. book series (CRT)*, pp. 33-56, 1998.

[11] Bettinelli, A., Ceselli, A., & Righini, G., "A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows," *Transport. Res. C-Emer*, vol. 19, no. 5, pp. 723-740, 2011.



(a) $\#.served$ wrt increasing requests (b) η wrt increasing requests (c) t^r_{avg} wrt increasing requests (d) t_{alg} wrt increasing requests

Fig. 7. Comparison of TBT−ACO, ACO−Base and GRASP on large-scale instances, in $\#.served$, η, t^r_{avg} and algorithm running time t_{alg}. $\rho = 0.3$ indicates that the generated demand equals to 30% of the real travel demand. $\rho = 0.3$ corresponds to about 520 requests and $\rho = 0.6$ corresponds to about 1040 requests. Each result in the figure is averaged over 10 random instances.



(a) $\#.served$ wrt increasing STW (b) $\#.served$ wrt increasing τ_{qos} (c) $\#.served$ wrt increasing SPR (d) $\#.served$ wrt increasing ADR

(e) η wrt increasing STW (f) η wrt increasing τ_{qos} (g) η wrt increasing SPR (h) η wrt increasing ADR

Fig. 8. Impacts of STW (size of pickup time window), τ_{qos} (maximum riding time of a single trip), SPR (seats per request) and ADR (number of alternative depots of the requests) to the results in terms of $\#.served$ and efficiency η. Each result in the figure is averaged over 10 random instances.

[12] Yassen, E. T., Ayob, M., Nazri, M. Z. A., & Sabar, N. R., "Meta-harmony search algorithm for the vehicle routing problem with time windows," *Inf. Sci.*, vol. 325, pp. 140-158, 2015.

[13] Bianchessi, N., & Righini, G., "Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery," *Comput. Oper. Res.*, vol. 34, no. 2, pp. 578-594, 2007.

[14] Ritzinger, U., Puchinger, J., & Hartl, R. F., "A survey on dynamic and stochastic vehicle routing problems," *Int. J. Production Res.*, vol. 54, no. 1, pp. 215-231, 2016.

[15] D. Gulczynski, B. Golden, and E. Wasil, "The period vehicle routing problem: New heuristics and real-world variants," *Transp. Res. Part E: Logistics and Transp. Rev.*, vol. 47, no. 5, pp. 648-668, 2011.

[16] Y. Yang, L. Tang, "A filter-and-fan approach to the multi-trip vehicle routing problem," in *Proc. IEEE Int. Conf. Logist. Syst. Intell. Manag.(ICLSIM)*, pp. 1713-1717, Harbin, China, Jan., 2010.

[17] Archetti, C., & Speranza, M. G., "The split delivery vehicle routing problem: a survey," *The vehicle routing problem: Latest advances and new challenges,Part of the Oper Res./Comput. Sci. Interfaces book series*, vol. 43, pp. 103-122, 2008.

[18] T. T. Liu, Z. B. Jiang, R. Liu, and S. J. Liu, "A review of the multi-depot vehicle routing problem," *Energy Procedia*, vol. 13, pp. 3381-3389, 2011.

[19] Gribkovskaia, I., Laporte, G. and Shyshou, A., "The single vehicle routing problem with deliveries and selective pickups," *Comput. Oper. Res.*, vol. 35, no. 9, pp. 2908-2924, 2008.

[20] Kim, G., Ong, Y.S., Heng, C.K., Tan, P.S. and Zhang, N.A., "City vehicle routing problem (city VRP): A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 1654-1666, 2015.

[21] Kim, G., Ong, Y.S., Cheong, T. and Tan, P.S., "Solving the dynamic vehicle routing problem under traffic congestion," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2367-2380, 2016.

[22] Ćirović, G., Pamućar, D. and Božanić, D., "Green logistic vehicle routing problem: Routing light delivery vehicles in urban areas using a neuro-fuzzy model," *Expert Syst. Appl.*, vol. 41, no. 9, pp. 4245-4258, 2014.

[23] Shen, Z., Dessouky, M.M. and Ordez, F., "A twostage vehicle routing model for largescale bioterrorism emergencies," *Networks*, vol. 54, no. 4, pp. 255-269, 2009.

[24] Lam, A.Y., Leung, Y.W. and Chu, X., "Autonomous-vehicle public transportation system: scheduling and admission control," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 5, pp. 1210-1226, 2016.

[25] Hrn, J., ileck, P., Song, Q. and Jakob, M., "Practical Multicriteria Urban Bicycle Routing," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 3, pp. 493-504, 2017.

[26] Luo, R., van den Boom, T. J., De Schutter, B., "Multi-agent dynamic routing of a fleet of cybercars," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 5, pp. 1340-1352, 2018.

[27] Huang, S.C., Jiau, M.K. and Lin, C.H., "A genetic-algorithm-based approach to solve carpool service problems in cloud computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 352-364, 2015.

[28] de Lira, V. M., Perego, R., Renso, C., Rinzivillo, S., Times, V. C., "Boosting Ride Sharing With Alternative Destinations," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2290-2300, 2018.

[29] Dondo, R., & Cerd, J., "A cluster-based optimization approach for the multidepot heterogeneous fleet vehicle routing problem with time windows," *Eur. J. Oper. Res.*, vol. 176, no. 3, pp. 1478-1507, 2007.

[30] Ray, S., Soeanu, A., Berger, J. and Debbabi, M., "The multi-depot split-delivery vehicle routing problem: Model and solution algorithm," *Knowl-Based Syst.*, vol. 71, pp. 238-265, 2014.

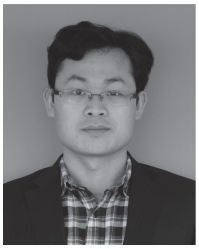
[31] Rahimi-Vahed, A., Crainic, T. G., Gendreau, M., & Rei, W., "A path relinking algorithm for a multi-depot periodic vehicle routing problem," *J. Heuristics*, vol. 19, pp. 497-524, 2013.

[32] Gajpal, Y., & Abad, P. L., "Saving based algorithm for multi-depot version of vehicle routing problem with simultaneous pickup and delivery," *Int. J. Enterprise Netw. Manage.*, vol. 3, no. 3, pp. 201-222, 2010.

[33] Aras, N., Aksen, D. and Tekin, M.T., "Selective multi-depot vehicle routing problem with pricing," *Transport. Res. C-Emer*, vol. 19, no. 5, pp. 866-884, 2011.

[34] Contardo, C. and Martinelli, R., "A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints," *Discrete Optimiz.*, vol. 12, pp. 129-146, 2014.

[35] Rivera, J.C., Afsar, H.M. and Prins, C., "Mathematical formulations and exact algorithm for the multitrip cumulative capacitated single-vehicle routing problem," *Eur. J. Oper. Res.*, vol. 249, no. 1, pp. 93-104, 2016.
[36] Wang, X., Choi, T.M., Liu, H. and Yue, X., "Novel ant colony optimization methods for simplifying solution construction in vehicle routing problems," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3132-3141, 2016.
[37] Mavrovouniotis, M. and Yang, S., "Ant algorithms with immigrants schemes for the dynamic vehicle routing problem," *Inf. Sci.*, vol. 294, pp. 456-477, 2015.
[38] B. Yu, Z. Z. Yang, and J. X. Xie, "A parallel improved ant colony optimization for multi-depot vehicle routing problem," *J. Oper. Res. Soc.*, vol. 62, no. 1, pp. 183-188, 2011.
[39] Stützle, T., & Hoos, H. H., "MAXMIN ant system," *Future Gener. Comput. Syst.*, vol. 16, no. 8, pp. 889-914, 2000.
[40] Vidal, T., Crainic, T.G., Gendreau, M. and Prins, C., "Heuristics for multi-attribute vehicle routing problems: a survey and synthesis," *Eur. J. Oper. Res.*, vol. 231, no. 1, pp. 121, 2013.
[41] Pellegrini, P., Stützle, T., & Birattari, M, "A critical analysis of parameter adaptation in ant colony optimization," *Swarm Intell.*, vol. 6, no. 1, pp. 23-48, 2012.
[42] O. Baskan, S. Haldenbilen, H. Ceylan, H. Ceylan, "A new solution algorithm for improving performance of ant colony optimization," *Appl. Math. Comput.*, vol. 211, pp. 75-84, 2009.
[43] Ciornei, I., Kyriakides, E., "Hybrid ant colony-genetic algorithm (GAAPI) for global continuous optimization," *IEEE Trans. Syst., Man, Cybern., Part B*, vol. 42, no. 1, pp. 234-245, 2012.
[44] B. Shuang, J. Chen, Z. Li, "Study on hybrid PS-ACO algorithm," *Appl. Intell.*, vol. 34, no. 1, pp. 64-73, 2011.
[45] T. Zhang, W. Chaovalitwongse, Y. Zhang, "Integrated ant colony and tabu search approach for time dependent vehicle routing problems with simultaneous pickup and delivery," *J. Comb. Optim.*, vol. 28, no. 1, pp. 288-309, 2014.
[46] Ceder, A., "Public transit planning and operation: Modeling, practice and behavior," CRC press, 2007.
[47] CPLEX package, https://www.ibm.com/products/ilog-cplex-optimization-studio. Last accessed Apr. 2019.
[48] De Grancy, G. S., Reimann, M., "Vehicle routing problems with time windows and multiple service workers: a systematic comparison between ACO and GRASP," *Cent. Eur. J. Oper. Res.*, vol. 24, no. 1, pp. 29-48, 2016.



Guiyuan Jiang received the B.S. degree from Northwest University for Nationalities, China, the M.Eng. degree from Tianjin Polytechnic University, China, and the doctoral degree from Tianjin University (TJU), all in Computer Science and Technology. Currently, he works as a research fellow in the Hardware & Embedded Systems Lab (HESL), School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore. His research interests include citywide traffic data analytics, data driven optimization in intelligent transportation system, methodologies for designing secure and energy-efficient MPSoCs.

Lam Siew Kei received his BASc, MEng and PhD from School of Computer Science and Engineering (SCSE), NTU. He was a Visiting Research Fellow in the Imperial College of London, University of Warwick, and RWTH Aachen, Germany. He is currently an Assistant Professor in SCSE and his research focuses on devising custom computing solutions to meet the challenging demands of energy-efficiency, reliability and security in embedded systems. His current projects include architecture-aware algorithms for vision-enabled sensing, design methodologies for secure and reliable embedded systems, and transportation analytics.

Fangxin Ning received the B.S. degree in Navigation Technology, and M.Eng. degree in Transportation Information Engineering and Control, both from Wuhan University of Technology, China. Currently, he works as a research associate at School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include data driven optimization for intelligent transportation systems, first/last-mile transportation problem.

Peilan He received the Bachelor degree of Management from School of Economics and Management, Hainan University, Haikou, China, and the Master degree of Engineering from School of Computer Science and Technology, Tianjin University, China. She is currently working toward the PhD degree in the School of Computer Science and Engineering, Nanyang Technological University, Singapore. Her research interests include big data analytics in transportation, urban computing, multi-modal journey planning.

Jidong Xie received the B.S. degree and the M.Eng. degree in Computer Engineering from Guangdong University of Technology, China. He was an intern student at the School of Computer Science and Engineering, Nanyang Technological University, Singapore. Currently, he currently with the Huawei Technologies Co., Ltd, China. His research interests include vehicle anomaly detection, road condition detection, vehicle trajectory planning.