A cumulative service state representation for the pickup and delivery problem with transfers

Monirehalsadat Mahmoudi
School of Packaging, College of Agriculture and Natural Resources, Michigan State University, East Lansing, Michigan, 48824, mahmou18@msu.edu

Junhua Chen
School of Traffic and Transportation, Beijing Jiaotong University, Beijing, P. R. China 100044, cjh@bjtu.edu.cn

Tie Shi
School of Transportation and Logistics
Southwest Jiaotong University, Chengdu, P. R. China 610031, tshi2005@my.swjtu.edu.cn

Yongxiang Zhang
School of Transportation and Logistics, Southwest Jiaotong University, Chengdu, P. R. China 610031, bk20100249@my.swjtu.edu.cn

Xuesong Zhou
School of Sustainable Engineering and the Built Environment, Arizona State University, Tempe, Arizona 85281, xzhou74@asu.edu

Abstract

The pickup and delivery problem with transfers is a challenging version of the vehicle routing problem. In order to tackle this problem, we add a time dimension to physical transportation networks to not only track the location of vehicles at any time but also impose parcels' pickup/delivery time windows, synchronization time points, and precedence constraints to the problem. We also add another dimension, described as the "cumulative service state" to the constructed space-time network to track the service status of parcels at any time. The constructed network not only handles real-life transportation networks but also is well-suited for connecting microscopic cumulative service states to macroscopic cumulative flow count diagrams. We develop a continuous time approximation approach using cumulative arrival, departure, and on-board count diagrams to effectively assess the performance of the system and dynamically constrict the search space. To handle a large-scale set of parcels, we develop the traditional cluster-first, route-second approach. We reach optimality for the clusters derived from the original set of parcels. We also propose an integer programming model to improve the vehicles' efficiency. We perform extensive numerical experiments over the standard data set used by Ropke and Pisinger (2006) and real-world large-scale data set proposed by Cainiao Network (with about 10,000 delivery orders) to examine the computational efficiency of our developed algorithm.

Keywords: pickup and delivery problems with transfers, dynamic programming, cumulative service states, cumulative flow count diagrams, Lagrangian heuristics.

1. Introduction

Coordinated transportation services consist of three different levels of service: ride-hailing, ridesharing without transfer, and ridesharing with at least one transfer. Ride-hailing is a level of coordinated service in which a passenger hires a driver to get a transportation service for a fee, and the driver is supposed to deliver the passenger to exactly where he needs to go. Traditional taxi companies offer this form of transport. The way by which a passenger hails a car can be listed as follows: a passenger can hail a taxi from the street, call up a transport service on the phone or hail a car from an app by his cellphone.

Ridesharing without transfer is another level of coordinated transportation service, which is slightly different from ride-hailing. In this mode of transportation, similar to the ride-hailing, a passenger hires a driver to take him where he needs to go, but the passenger may share his ride with other passengers. Recently, a broad range of transportation network companies, such as Uber, Lyft and Sidecar have begun to offer this type of transport service by the aid of three recent technological advances: (1) Global Positioning System (GPS) navigation devices, (2) smartphones, and (3) social networks. Cross-docking, a warehousing strategy of moving of goods directly from the receiving dock to the shipping dock, is an example of this type of transportation service which reduces the handling and storage steps in between to a minimum.

The third level of coordinated transportation service is ridesharing with transfers. In general, transfers are used to provide more efficient transportation networks by reducing the operational costs, as well as making more flexible routes available for passengers. A large number of daily trips are classified in this category. An example of this type of transportation service is the first mile/last mile transport of the commuters who go from an origin to a transit station or from a station to a final destination. Ridesharing between households or fellow workers is another example of transfers. In this case, members of a family or any other social group arrange their trips informally and share their travel information, such as departure time, stops, and transfer points, among themselves. In Figure 1, we show different levels of coordinated transportation service via an example in which six passengers with different origins, destinations, and departure time windows have called for service.

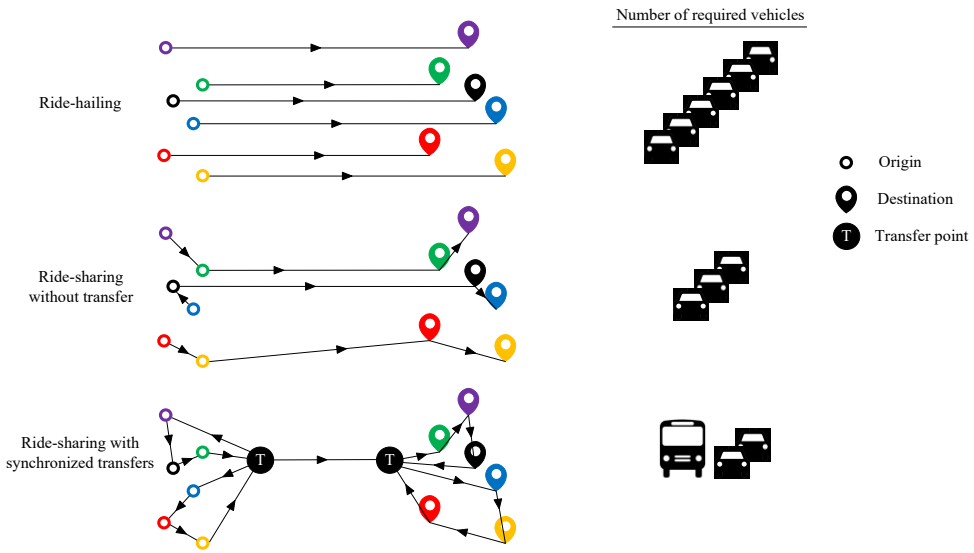


Fig. 1. Different levels of coordinated transportation service.

As shown in Figure 1, in the case of ridesharing with transfers, the vehicles' capacity can be utilized more in comparison to other types of coordinated transportation service. In this case, the first vehicle picks up passengers from different origins and delivers them to a transfer point; the bus picks them up from the first transfer point and delivers them to the second transfer point; and finally, the second vehicle picks them up from the second transfer point and delivers them to their final destinations. In this example, we assume that departure time windows are wide enough such that one vehicle handles all trips from origins to the first transfer point, and the other serves all trips from the second transfer point to the destinations.

The concept of pickup and delivery with transfers is not only used in passenger transit but also applied frequently in freight transportation in what is called "freight consolidation." Freight consolidation is when small shipments are bundled with other small shipments for some legs of their journey. It is also known as consolidation service, assembly service, and cargo consolidation. Different terms are used for transfer centers, depending on the

transportation mode and the type of good being transported, such as rail yards in the railway industry, hub airports in air cargo transportation, transshipment ports in sea cargo transportation, and terminals for transporting goods by trucks.

In the last decade, ridesharing companies have introduced a new mode of transportation that is much more convenient than public transit and less expensive than taxi services. By introducing connected and autonomous vehicles to this mode of transportation and eliminating the cost of hiring drivers, ridesharing can be a good complement for the public transportation in the future. In August 2018, Valley Metro (Metro Phoenix's transit agency) and Waymo (Google's self-driving car project) launched a new partnership to experiment with how new technology can improve traditional public transit options. This arrangement may be a huge relief for those passengers who want to take the bus or light rail to work but live too far from the nearest station.

In general, ride-hailing and ridesharing with or without transfers can be mathematically modeled by the vehicle routing problem with pickup and delivery with time windows (VRPPDTW). The VRPPDTW is a combinatorial optimization problem that searches for an optimal set of routes for a fleet of vehicles to serve a set of requests. Each request is a combination of pickup from the origin and drop-off at the destination within particular time windows. The objective of the VRPPDTW varies from one study to another based on the main focus of the problem. For example, some studies aim to satisfy all demands with fewer vehicles, while others attempt to maximize the number of parcels that can be served by a fixed number of vehicles. Therefore, the objective of the former problem is to minimize costs subject to full demand satisfaction, while the objective of the latter one is to maximize the total number of served passengers subject to vehicle availability. In fact, the latter case is more practical. Other objective functions observed in the literature include but are not limited to: minimization of difference between actual and desired delivery times (see, e.g., Bodin and Sexton, 1986), minimization of differences between actual and shortest possible ride times (see, e.g., Bodin and Sexton, 1986), minimization of total route duration (see, e.g., Dumas et al., 1989; Desrosiers et al., 1991; Ioachim et al., 1995), minimization of total route length (see, e.g., Cordeau and Laporte, 2003), minimization of vehicles' idle time (see, e.g., Diana and Dessouky, 2004), minimization of passengers' inconvenience (see, e.g., Coslovich et al., 2006; Melachrinoudis et al., 2007), or a weighted combination of those mentioned above.

In order to solve the VRPPDTW, several algorithms have been suggested by the extant literature. Dumas et al. (1991) used a set-partitioning model to minimize the total travel cost, considering tight vehicle capacity constraints, as well as time windows and precedence constraints. They proposed a column generation scheme with a constrained shortest path as a sub-problem to construct admissible routes. Savelsbergh and Sol (1998) developed a branch-and-price algorithm to minimize the total number of vehicles needed to serve all passengers as the primary objective, and minimize the total distance traveled as the secondary objective. In addition, Lu and Dessouky (2004), Cordeau (2006), and Ropke et al. (2007) proposed branch-and-cut algorithms to minimize the total routing cost. Ropke and Cordeau (2009) also presented a branch-and-cut-and-price algorithm in which the lower bounds are controlled by a column generation scheme and strengthened by introducing several valid inequalities to the problem. Baldacci et al. (2011) proposed a new exact algorithm based on a set-partitioning formulation improved by additional cuts to minimize total routing costs. In a recent clustering algorithm proposed by Häme and Hakula (2015), the multi-vehicle routing solution is obtained by calling a recursive single-vehicle algorithm based on the passenger-to-vehicle assignment from the first clustering stage.

In terms of algorithmic development, a number of studies have focused on solving the VRPPDTW by the dynamic programming (DP) approach. For instance, the classical work by Psaraftis (1980) presented an exact backward DP for the single-vehicle routing problem with pickup and delivery with time windows. The objective of the problem was to minimize a weighted combination of the total service and waiting time for passengers with $O(n^2 3^n)$ complexity, where n denotes the total number of passengers in the system. Psaraftis (1980) proposed a passengers' service state representation that was adopted from the path representation for the traveling salesman problem proposed by Bellman (1962) and Held and Karp (1962). Psaraftis (1983) further modified the algorithm to a forward DP algorithm with the same space complexity. Desrosiers et al. (1986) proposed a forward DP algorithm for the single-vehicle routing problem with pickup and delivery with time windows to minimize the total distance traveled to serve all passengers. Recently, by the aid of a Lagrangian relaxation solution framework, Mahmoudi and Zhou (2016) have proposed a forward DP solution-based algorithm to minimize the total routing costs of the single vehicle sub-problems on a three-dimensional space-time-state network. Their time-dependent single-vehicle state is jointly defined by the passengers' carrying state, the current node being visited, and the time to reflect time windows and vehicles' capacity constraints in a well-structured network. Furthermore, their special three-dimensional network representation for the multi-vehicle routing problem with pickup and delivery with time windows reduces the space complexity of the DP algorithm from the exponential order of 3^n (n is the total number of passengers) to the much

smaller space requirement of $\sum_{k=0}^{Q} C_k^n$, where Q is vehicles' capacity and C_k^n is the number of k-combinations from n passengers. We note that Q is not a large number in practice (e.g., 2 or 3 for taxi).

There are also a number of studies addressing the issue of synchronization in various vehicle routing and scheduling problems. Sharypova et al. (2012) designed a continuous-time mixed integer programming (MIP) model to minimize the total system operational costs in the intermodal service network design problem, in which the arrival and departure times of vehicles are synchronized to facilitate the transshipment of containers in the terminals. Li et al. (2014) assumed that passengers and parcels can share the same taxi transport services and further proposed an MIP model to optimize the taxi routes and schedules, such that the total system profit can be maximized. They set higher priority for the passengers, such that the transportation of parcels will be synchronized as much as possible. Dellaert et al. (2016) addressed the two-echelon vehicle routing problem with time windows (2E-VRPTW), where the shipment process of the freight is divided into two separate parts and each of the parts is transported by an echelon. They developed two different groups of integer programming (IP) models to minimize the total transportation cost of two echelons. Extra connectivity constraints are introduced into one of the groups to ensure the interconnectivity of the first and second echelon paths.

Despite the extensive prior research on ride-hailing and ridesharing without transfers, few studies have focused on ridesharing with transfers. This is observed in the literature as the pickup and delivery problem with transshipments/transfers (PDPT). A number of studies have focused on showing the usefulness of transfer and schedule coordination in the pickup and delivery problem, such as Mitrović-Minić and Laporte (2006), Qu and Bard (2012), Masson et al. (2013), Kim and Schonfeld (2014), Sun and Schonfeld (2016), and Ghilas et al. (2016). Another stream in the literature has focused on solving the PDPT by heuristic/meta-heuristic algorithms. For example, motivated by the practice in a large San Francisco-based courier company, Mitrović-Minić and Laporte (2006) conducted an empirical study on the effectiveness of transfer points in the pickup and delivery problem. Since the company was serving a large area covering several neighboring cities, they were allowing transshipment of loads between vehicles to keep drivers in their home area and found circumstances under which such transfers may be useful. They applied a two-phase heuristic (a construction phase followed by an improvement phase) to solve the problem. In another practice-driven study, Qu and Bard (2012) examined the usefulness of transshipment in finding daily routes for a regional air carrier. In their study, they developed a greedy randomized adaptive search procedure to handle this complex problem. Furthermore, Masson et al. (2013, 2014) proposed an adaptive large neighborhood search for solving the PDPT. They tested their algorithm on real-world instances related to the transportation of mentally or physically disabled people. They showed that adding the concept of transfer to the pickup and delivery problem can make significant improvements in the objective function. Recently, Ghilas et al. (2016) have proposed an adaptive large neighborhood search heuristic algorithm for the PDPT. They showed the merits of using transfers in the pickup and delivery problem by testing their algorithm on sets of generated instances.

A number of research articles have also focused on finding exact solutions for this problem. For example, Mues and Pickl (2005) developed a path-based MIP model for the PDPT and applied a column generation to solve the model. Cortés et al. (2010) proposed an MIP model for the PDPT in which passengers have different options for transfer from one vehicle to another at particular transfer nodes. They used a branch-and-cut algorithm based on Benders decomposition to solve the model. In three major papers on this topic by Drexl (2012a, 2012b, 2013), different types of synchronization (i.e., task synchronization, operation synchronization, movement synchronization, load synchronization, and resource synchronization) have been extensively discussed. Recently, Rais et al. (2014) proposed an MIP formulation for the PDPT with/without time windows for services in which heterogeneous vehicles and flexible fleet size are allowed. They used the commercial solver GUROBI powered by the simplex method on linear-programming relaxations combined with branch-and-cut and branch-and-bound techniques to solve the MIP model. The MIP model proposed by Rais et al. (2014) solves the problem on passengers' origin/destination-based network. Their model does not work directly with transportation networks in which change in travel time is subject to the time of the day (e.g., high-occupancy vehicles (HOV) lanes) or the load of the vehicle (high-occupancy toll (HOT) lanes). Finally, dealing with several constraints, especially those related to the validity of the time and load variables has prompted us to look at this challenging problem from a different angle.

In this research, we add time dimension to the space graph to not only track the location of vehicles at any time but also impose parcels' pickup and delivery time windows, synchronization time points, and precedence constraints to the problem. Defining time as an explicit dimension and physical transportation networks as the base of our proposed networks help us to handle the foregoing instances of HOV and HOT lanes. We also add another dimension, called the "parcels' cumulative service state" to the constructed space-time graph to track the service status of parcels at any time and impose the coupling and precedence constraints to the model. Based on this premise, our contribution is three-fold.

(i) In terms of methodology, we propose a new mathematical model for the PDPT, in which heterogeneous vehicles and flexible fleet size are allowed. Based on our space-time-state network representation, we apply a DP to include the vehicle-to-task assignment constraints and provide exact solutions for small-scale problems. Meanwhile, our space-time-state network representation prevents sub-tours which is a common issue in the family of vehicle routing problems.

(ii) More importantly, to address the curse of dimensionality, we demonstrate a consistent transition from the cumulative service states to cumulative flow count diagrams to effectively estimate the overall dynamic system performance. It should be noted that the concept of cumulative flow count diagram is widely applied as a representative of dynamic activities in traffic science literature. The concept of cumulative flow count diagram accounts for the cumulative flow count of vehicles passing through a transportation system, in which vehicle concentrations, queue sizes, travel times, and delays are the main measures of the system performance evaluation (see the papers by Newell, 1982 and Hall, 1991 for a summary of corresponding methodologies).

(iii) With the consistent microscopic and macroscopic system representation, our model and algorithm can effectively handle large-scale real-world instances and generate a good initial solution to be applied for our Lagrangian heuristic method. Then, by our Lagrangian heuristic, we evaluate the marginal cost of each transfer and guide a fast search for real-world test cases with about 10,000 delivery orders.

2. Problem statement for the PDPT

Unlike the case for parcels, modeling of the PDPT can be complex due to various preferences of passengers (e.g., to allow different number of transfers in a trip). In order to streamline our discussion around the PDPT, we have conducted the current study based upon parcel transportation, where one does not face the foregoing complexities. The pickup and delivery problem with time windows searches for an optimal set of routes for a fleet of vehicles to serve a set of parcels. Each parcel must be picked up from the origin and dropped off at the destination in given departure and arrival time windows, respectively.

Because of the introduction of transfers, each parcel is picked up and dropped off more than once, and hence, the parcel's trip contains more than one origin-destination (OD) pair. In this case, the OD pair can be a trip from the parcel's origin to a transfer point, from a transfer point to another transfer point, or from a transfer point to the parcel's final destination. In our model, the parcel's origin/destination are called *main origin*/*destination*, and a pickup/drop-off location at a transfer point is called *intermediate origin/destination*.

In this paper, we assign a given set of transfer points to each parcel. The set of transfer points for a parcel is defined based on the historical data related to the previous parcels whose origin and destination were in the same geographical zones of the parcel's origin and destination, respectively. These transfer points can be located anywhere in the given transportation network. Of note, the set of transfer points for a parcel can be empty. A "*way*" is defined as a sequence of landmarks (i.e., the parcel's origin, predefined transfer points (if existing), and the parcel's destination). The set of ways by which a parcel can be served is given. All given ways for a parcel are time-feasible. We provide an example explaining the concept of ways for parcels in Section 3.1. For any feasible solution in the PDPT, the following conditions must hold:

- Every vehicle starts its route from its origin depot at the time when its time horizon starts and ends the route at its destination depot at the ending time of its time horizon.
- Every parcel is served at most once (i.e., not at all or just once).
- For every OD pair, the origin is visited before the destination (precedence constraint).
- A parcel arrives at a transfer point before departing from it (transfer synchronization).
- For every OD pair, the pickup/drop-off occurs within the corresponding departure/arrival time windows.
- For every OD pair, the trip from origin to destination is done by a single vehicle (coupling constraint).
- The time to pickup/drop-off multiple parcels from/at a location is the sum of the times needed to pickup/drop-off each parcel.
- Every vehicle does not exceed its capacity.

To solve the PDPT for a large number of parcels, we initially cluster the OD pairs. By doing this, the large-scale primary problem is broken down to several sub-problems. The objective function of the clustering algorithm is to minimize the weighted combination of mismatches between each OD pair assigned to the cluster and the OD pair used to "seed" the cluster and the total number of clusters. We discuss the clustering algorithm in Section 3.3 in more details.

After clustering OD pairs, we propose a least-cost path model to route vehicles inside each cluster (Section 3.4). We note that, in this problem, the total number of vehicles is given. The objective of this problem is to minimize the routing cost of vehicles while enforcing vehicles to serve OD pairs within their route. To implement this goal, we provide some incentives for vehicles such that they select detours and serve OD pairs instead of selecting the direct route (least-cost path) from their origin depot to their destination depot.

After finding the optimal route for each vehicle in each cluster, we return to the original problem and look at the clusters as an integrated system. In this case, vehicles can be utilized more such that they can serve more OD pairs during their time horizon. To meet this purpose, we present an IP model to find the optimal chains of tasks that can be performed by each vehicle (Section 3.5). A task is defined as a number of OD pairs that have been already served by a vehicle through the algorithm mentioned in Sections 3.4. The objective function of this problem is to minimize the total routing cost for vehicles, provided that each task is taken by only one vehicle.

Section 4 provides computational results over the data set used by Ropke and Pisinger (2006) and the real-world data set proposed by Cainiao Network (the logistics service provider to the Alibaba Group in China). In Section 4, we demonstrate the computational efficiency of our developed algorithm coded in C++. We conclude the paper in Section 5.

3. Our proposed model for the PDPT

The existing MIP model for the PDPT (Rais et al., 2014) contains several constraints related to the validity of the time and load variables, which make the problem difficult to solve for a large number of parcels. The main thrust of this paper is how to construct a multi-dimensional network such that the concept of assignment of vehicles to OD pairs and their routing in the PDPT for a large-scale set of parcels and vehicles are integrated together. In the next section, we explain how to construct a multi-dimensional network for the PDPT. We provide a table of notations for sets, indices, parameters and decision variables used in this paper in Appendix A.

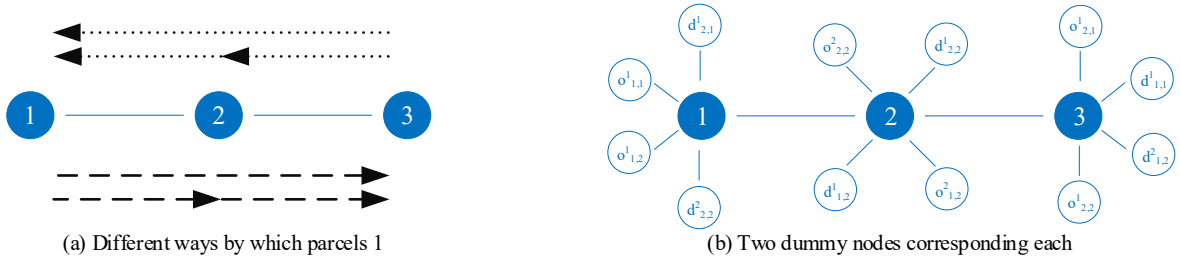
3.1. Space-time network

We define the problem on a physical transportation network, which includes a set of physical nodes (e.g., intersections or freeway merge points) and a set of directed physical links with different types (e.g., freeway segments, arterial streets, or ramps). Generally, data sets given by metropolitan planning organizations (MPOs) define different scenarios to account for the variation of speed over different times of day. By having a fixed length and variant speed for each link, we can obtain the travel time for each link at each time of day. As a result, each link has a time-dependent travel time.

As mentioned before, depending on the number of transfers, each parcel may be picked up/dropped off several times. It is necessary to distinguish pickup/drop-off locations and vehicles' origin/destination depots from physical transportation nodes because, unlike physical transportation nodes, pickup/drop-off locations and vehicles' depots have time window restrictions. Moreover, as another distinguishing feature (compared to physical transportation nodes), our definition for OD pairs' cumulative service states prompts pickup and drop-off actions (state transitions) to only occur from/at particular nodes. To address this, dummy nodes corresponding to these locations are added to the transportation network (see the paper by Mahmoudi and Zhou, 2016 for more details).

Suppose $OD_{j,n}^{m}$ denotes the m^{th} OD pair in n^{th} way by which parcel j is served. Let $o_{j,n}^{m}$ and $d_{j,n}^{m}$ denote dummy nodes corresponding to parcel j's origin and destination in the m^{th} OD pair of n^{th} way, respectively. Each dummy node is only connected to its corresponding physical transportation node by a link. We call this link a service link or, more precisely, a pickup/drop-off link. The travel time of this link is interpreted as the service time, which is the time needed to pick up/drop off a parcel.

Figure 2(a) illustrates a three-node transportation network with bi-directional links. Here, we provide an example on the small-scale three-node transportation network to illustrate the concept of dummy node and parcel's way. Suppose two parcels needed to be transported: parcel 1 should be transported from node 1 to node 3, while parcel 2 from node 3 to node 1. We also assume that both parcels can have a one-stop trip stopping at node 2. Therefore, their set of transfer points is {2}. Figure 2(a) illustrates the two ways by which parcels 1 and 2 can be transported (shown by dashed and dotted lines, respectively). Figure 2(b) demonstrates a network in which dummy nodes corresponding to each OD pair have been added to the network. In this example, there are six OD pairs: $OD_{j,n}^{m}: o_{j,n}^{m} \rightarrow d_{j,n}^{m}$ for $j \in (1,2), n = m = 1$ and $j, m \in \{1,2\}, n = 2$.



(a) Different ways by which parcels 1 and 2 can be transported.

(b) Two dummy nodes corresponding each OD pair have been added to the network.

Fig. 2. (a) Two ways, i.e., 1 → 3 and 1 → 2 → 3, by which parcel 1 can be transported, and two ways, i.e., 3 → 1 and 3 → 2 → 1, by which parcel 2 can be transported; (b) dummy nodes corresponding to each OD pair have been added to the physical transportation network.

As mentioned before, each parcel must be picked up from its main origin and dropped off at its main destination in particular departure and arrival time windows, respectively. Suppose $[\underline{a}_j, \underline{b}_j]$ and $[\overline{a}_j, \overline{b}_j]$ denote parcel j's departure and arrival time windows, respectively. We note that $\underline{a}_j$ and $\underline{b}_j$ are the earliest and latest departure times from parcel j's main origin, and $\overline{a}_j$ and $\overline{b}_j$ are the earliest and latest arrival times to its main destination. Therefore, $[\underline{a}_j, \underline{b}_j]$ is the departure time window for all dummy nodes connected to parcel j's main origin, and $[\overline{a}_j, \overline{b}_j]$ is the arrival time window for all dummy nodes connected to its main destination. For all other dummy nodes connected to an intermediate origin or destination, a wide time window $[\underline{a}_j, \overline{b}_j]$ is considered.

In addition, let o_v and d_v denote dummy nodes corresponding to the origin and destination depots for vehicle v, respectively. Again, each dummy node is only connected to its corresponding physical node by a link. The travel time for this link is interpreted as the preparation time (i.e., we call these links preparation links). Moreover, vehicle v's time horizon is denoted by $[t_{v,start}, t_{v,end}]$, where $t_{v,start}$ and $t_{v,end}$ are the time stamps at which the time horizon (work shift) of vehicle v begins and ends, respectively.

To illustrate space-time networks, we initially map the two-dimensional space graph (physical transportation network with added dummy nodes) to a one-dimensional space at which all nodes are positioned in a row. Then, we add time dimension to the space graph, where the time horizon is discretized into a series of time intervals with the same time length. Although the solution optimality may be generally compromised by time discretization, we have attempted to alleviate this effect by considering smaller time units (e.g., one minute) compared to OD pairs' departure/arrival time windows and ride time. To reflect on this, we use the term "pseudo-optimal" for the solution we obtain by solving the time-dependent least-cost path problem. Nevertheless, we note that the discretization schemes have been already used in the literature to capture the time-related or capacity-related dynamics in transportation system modeling: cell transmission model for traffic flow dynamics (see the paper by Daganzo, 1994), space-time expanded network for departure time choices (see the paper by Yang and Meng, 1998), and early work on space-time network flow models (see the paper by Zawack and Thompson, 1987).

We note that if vehicle v arrives early at a dummy node corresponding to a parcel's pickup/drop-off location, it should wait until the time window starts, while arriving late to these nodes is not permitted (hard time windows). In addition, if vehicle v arrives at d_v earlier than $t_{v,end}$, it should wait until its time horizon ends, and arriving later than $t_{v,end}$ is not allowed. To address this, we add another dimension in the next section, called "parcels' cumulative service state", to the constructed space-time graph to track the service status of OD pairs at any time.

3.2. Multi-dimensional network construction

In the classical study by Psaraftis (1980), an exact backward DP was proposed for the single-VRPPDTW to minimize a weighted combination of the total service and waiting time for passengers. Psaraftis (1983) further developed a forward recursion scheme in this DP to deal with passengers' time windows. However, this approach is not easily used for the VRPPDTW. In that study, the state representation consists of the location currently being visited and the service status of parcels. The service status of parcels is chosen from set {1,2,3}, where 3 means parcel j is still waiting to be picked up, 2 means parcel j has been picked up but the service has not been completed, and 1 means parcel j has been successfully delivered.

In our study, we adapt the Bellman-Held-Karp path representation scheme (Bellman,1962; Held and Karp, 1962) in the traveling salesman problem to define passengers' service patterns. In the foregoing studies, the passengers' service patterns consist of two terms: the node currently being visited and the cumulative service state of passengers. We extend the first term to the node currently being visited at time t, and the second term to the more complicated

cumulative service state of OD pairs, i.e., "pickup" and "drop-off". In our scheme, state s is a vector whose elements represent the cumulative service state of OD pairs. The state of $OD_{j,n}^m$ is an element chosen from set {0,1,2}. The state of $OD_{j,n}^m$ is 0 if parcel j is still waiting to be picked up from $o_{j,n}^m$, 1 means it has been picked up but the trip from $o_{j,n}^m$ to $d_{j,n}^m$ has not been completed yet, and 2 means it has been successfully delivered to $d_{j,n}^m$.

In the example mentioned in Section 3.1, since we have 6 different OD pairs (i.e., 3 OD pairs each for parcel 1 and parcel 2), 3^6 states may exist, which are [0,0,0,0,0,0], [1,0,0,0,0,0], [0,1,0,0,0,0], …, [2,2,2,2,2,2]. We note that the elements of each state are associated with the cumulative service state of OD pairs $OD_{1,1}^1$, $OD_{1,2}^1$, $OD_{1,2}^2$, $OD_{2,1}^1$, $OD_{2,2}^1$, and $OD_{2,2}^2$, respectively. Some of these states are obviously infeasible. For example, at state [2,2,2,2,2,2], both parcels are served by both ways 1 and 2, while a parcel cannot be served by more than one way.

After defining the states, the next step is defining all feasible state transitions. According to our definition for the cumulative service state, there are a limited number of feasible state transitions from state s to s', since several transitions from state s to s' violate the activity precedence constraints (e.g., if drop-off occurs before pickup) or the vehicles' capacity constraints. We note that in the state transition from state s to s', the states s and s' are only different in one element. Figure 3 illustrates a number of feasible and infeasible state transitions for the example mentioned in Section 3.1. State transitions illustrated in Figure 3(a) and 3(b) are feasible. Figure 3(a) shows that both parcels are waiting to be picked up at state s, while at state s', parcel 1 is picked up by way 1 but the trip has not been completed yet. In Figure 3(b), at state s, parcel 2 has been already delivered by way 2 successfully, and parcel 1 has been picked up by way 1, but the service has not been completed yet. At state s', both parcels are served successfully. The state transition shown in Figure 3(c) is infeasible due to the violation of parcel 1's precedence constraint.

[0,0,0,0,0,0] → [1,0,0,0,0,0] (a) [1,0,0,0,2,2] → [2,0,0,0,2,2] (b) [1,0,0,0,2,2] → [0,0,0,0,2,2] (c)

Fig. 3. (a) Feasible state transition in which parcel 1 is picked up, while parcel 2's service has not started yet; (b) feasible state transition in which parcel 1 is dropped off while parcel 2 has already been delivered; (c) infeasible state transition due to the violation of parcel 1's precedence constraint.

A vertex in our multi-dimensional network is an object of the form (i, t, s, v), where i is a node index (it can be either a physical transportation node or a dummy node), t is a time index, s is an index related to the OD pairs' cumulative service state, and v is a vehicle index. Clearly, not all (i, t, s, v)-tuples will form feasible vertexes. Vertex (i, t, s, v) is feasible if the combination of indices i, t, and s is feasible for vehicle v's state-space-time network. Of note, $t \in [t_{v,start}, t_{v,end}]$. We define a set of rules determining when such a tuple is feasible in the following paragraphs.

Figure 4 illustrates a number of important network constructs by the example mentioned in Section 3.1. Suppose there are two vehicles available in the system: vehicle 1 has the same origin and destination depots located at node 1, while vehicle 2's origin and destination depots are at node 3. First, dummy nodes corresponding to OD pairs' pickup and drop-off locations, as well as vehicles' origin and destination depots, are added to the physical transportation network (Figure 4(b)). Then, the two-dimensional space graph (XY plane) is mapped to a one-dimensional space network in which all nodes are positioned in a row (Figure 4(c)). In the third step, the cumulative service state and vehicle-time are added as new dimensions to the one-dimensional space network (Figure 4(c)).

To add the vehicle-time dimension to the space network, a unique block is generated for each vehicle. Several interior layers comprising OD pairs' cumulative service states are added along the vehicle's time horizon. Each block consists of two exterior layers, so-called opening and ending layers, whose job is transmitting the information related to the cumulative service state of OD pairs from the ending layer of the current block/vehicle to the opening layer of the next block/vehicle. In the example shown in Figure 4, similar to a runner in a relay race, vehicle 1 transmits this information from vertex $(d_1, t_{1,end}, s, 1)$ to vertex $(o_2, t_{2,start}, s, 2)$. If a vehicle serves an OD pair, it should complete the trip from the OD pair's origin to its destination. Therefore, we do not see any state with element "1" at the opening and ending layers of blocks. Here, we define a set of rules determining when (i, t, s, v)-tuples are feasible.

Rule 1. The total number of vehicles is given. Index v should not exceed the total number of vehicles.

Rule 2. Node i is considered in vehicle v's network, if and only if it is accessible to nodes o_v and d_v within vehicle v's time horizon. In this paper, we have written a separate module, called a "space-time prism", to determine the set of *accessible* nodes for each vehicle. In this module, we use a forward and backward DP to solve a time-dependent

minimum spanning tree problem. Interested readers can find more information about space-time prism calculation in the paper by Mahmoudi et al. (2019).

Rule 3. Node i is only feasible within its own particular time window. The time window for all physical transportation nodes in vehicle v's network is $[t_{v,start}, t_{v,end}]$. The dummy nodes' time window has been already discussed in Section 3.1.

Rule 4. At the opening and ending layers of a block, states with element "1" must not exist.

Rule 5. State s should not violate vehicle v's capacity constraint. This means that the total number of element "1" at state s must not exceed vehicle v's capacity at any time and location.

Arc (i, i', t, t', s, s') is defined from vertex (i, t, s, v) to vertex (i', t', s', v') in a multi-dimensional network and is feasible if all the following conditions are met:

Condition 1. $v' \in \{v, v+1\}$. $v' = v+1$, if and only if $i = d_v$, $i' = o_v$, $t = t_{v,end}$, $t' = t_{v',start}$, and $s = s'$. $v' = v$, otherwise.

Condition 2. If $v = 1$ (i.e., the first vehicle in the system), $i = o_1$, and $t = t_{1,start}$, then $s = \phi$. State ϕ is the null state at which the service of no OD pair has started yet. In other words, all elements of vector ϕ are 0.

Condition 3. The link between adjacent nodes i and i', $(i \neq i')$, must exist. The link can be a physical transportation link, service link, or preparation link.

Condition 4. $t' = t + TT_{i,i',t}$, where $TT_{i,i',t}$ is the travel time/service time/preparation time from node i to i' starting at time t.

Condition 5. $s \neq s'$ if the link connecting node i to i' $(i \neq i')$ is one of the following links: (1) pickup link that connects dummy node i corresponding to a parcel's origin (main or intermediate origin) to the corresponding transportation node i'; or (2) drop-off link that connects transportation node i to dummy node i' associated with a parcel's destination (main or intermediate destination). In other cases, $s = s'$, where the state transition from state s to s$'$ must be feasible.

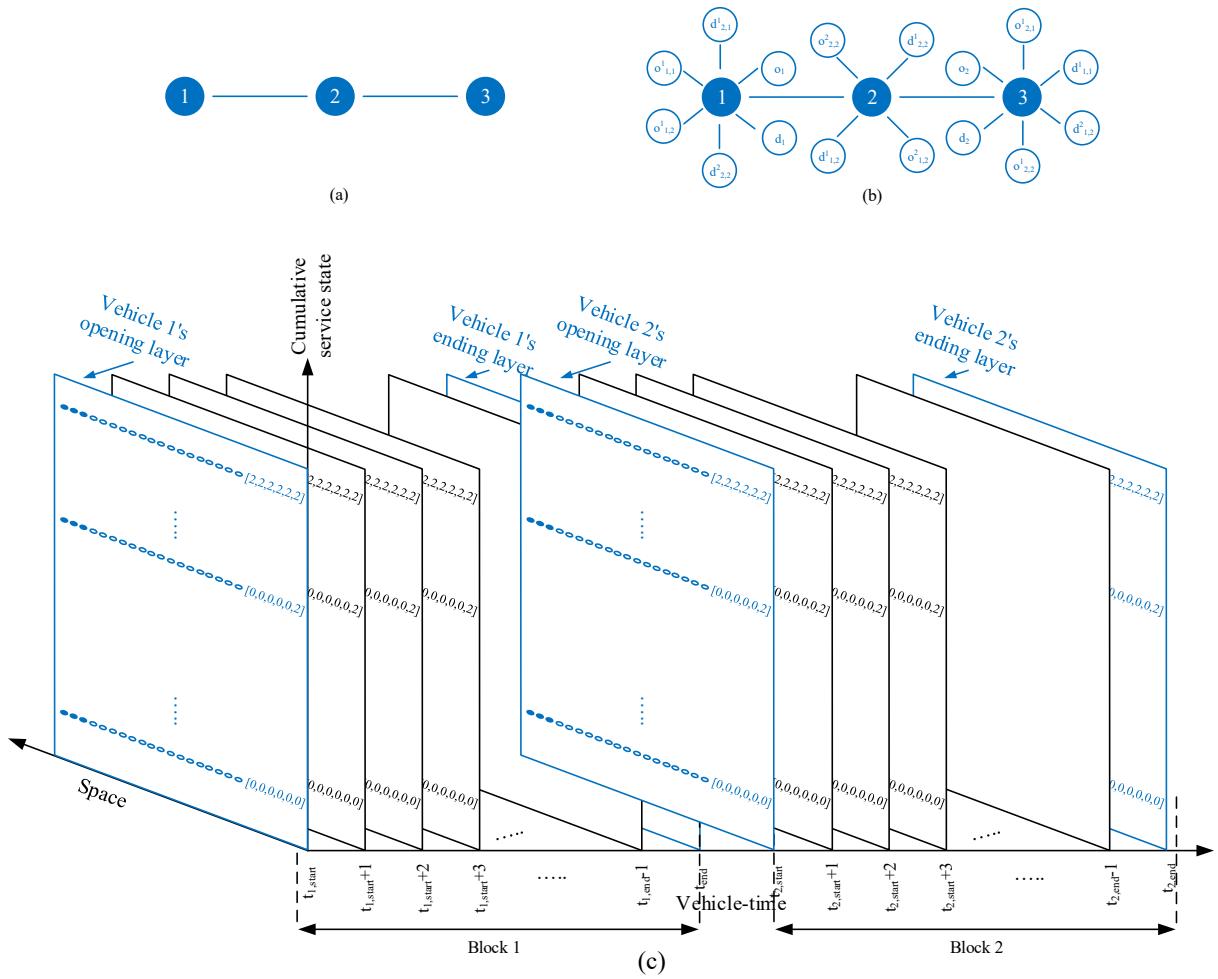


Fig. 4. (a) A three-node transportation network; (b) modified network with dummy nodes for two vehicles and two parcels; (c) three-dimensional space-time-state network.

Due to the complexity of the constructed network, our model is able to solve the PDPT for a limited number of OD pairs and vehicles. To handle a large-scale set of OD pairs, we suggest the traditional cluster-first, route-second approach in which the large-scale primary problem is broken down to several sub-problems. In the next section, we describe the details of the clustering phase, and then explain how to conduct the PDPT on the constructed network.

3.3. Clustering OD pairs

Finding high-quality clusters without having some levels of routing information is a difficult task. Some methods cluster OD pairs based on the proximity of geographical zones of their respective origin and destination nodes. Dumas et al. (1989) introduced the concept of mini-clusters, where parcels with spatio-temporal proximity are clustered together. In their algorithm, a heuristic algorithm provided a set of mini clusters. Desrosiers et al. (1991) proposed another way of constructing mini-clusters by a parallel insertion method based on spatio-temporal proximity of the parcels. Ioachim et al. (1995) later applied an optimization-based technique instead of a heuristic for constructing mini-clusters. Zhu and Guo (2014) presented a hierarchical clustering algorithm in which nearby OD pairs had a higher probability of belonging to the same cluster than to a cluster containing OD pairs that are farther away. Wan et al. (2015) and Kumar (2016) used a density-based hierarchical clustering method, in which clusters were defined as areas of higher density than the remainder of the data set. Ball and Hall (1965) and Lloyd (1982) presented a centroid-based clustering algorithm (i.e., *k*-means) operating based on optimizing an objective function, which typically measures inter-cluster separation, within-cluster variance or both. Pankratz (2005), Bard and Jarrah (2009), Qu and Bard (2012), and Masson et al. (2013) are some of many examples of studies that applied a clustering algorithm for solving the pickup and delivery problem.

In our research, since we examine the pickup and delivery problem with time windows, the time should also be considered. Therefore, we utilize the space-time network and spatiotemporal zones to do the clustering procedure. Since each OD pair has a pickup/drop-off time window and not a unique time stamp for pickup/drop-off action, we assume that the pickup/drop-off action occurs in the middle of each departure/arrival time window. This assumption also helps us to approximately calculate the space-time distance between two space-time vertexes. We calculate the space-time distance between each OD pair's origin vertex and other OD pairs' origin vertexes or each OD pair's destination vertex and other OD pairs' destination vertexes by calculating the weighted combination of geographical and temporal distance between two vertexes. We calculate the geographical distance between two vertexes by the Euclidian distance and calculate the temporal distance between two vertexes by the absolute value of the difference between two corresponding time stamps.

First, to find these space-time distances, we calculate the middle time of $o_{j,n}^m$'s departure time window. The middle time of $o_{j,n}^m$'s departure time window is $\frac{\underline{a}_j + \underline{b}_j}{2}$ if $o_{j,n}^m$ is connected to the main origin, and it is $\frac{\underline{a}_j + \overline{b}_j}{2}$ if $o_{j,n}^m$ is connected to an intermediate origin. We call the middle time of $o_{j,n}^m$'s departure time window as $o_{j,n}^m$'s departure time stamp, and denote it by $t_{o_{j,n}^m}$. Similarly, we calculate the middle time of $d_{j,n}^m$'s arrival time window, i.e., $\frac{\overline{a}_j + \overline{b}_j}{2}$ if $o_{j,n}^m$ is connected to the main destination and $\frac{\underline{a}_j + \overline{b}_j}{2}$ if $o_{j,n}^m$ is connected to an intermediate destination, call it as $d_{j,n}^m$'s arrival time stamp, and denote it by $t_{d_{j,n}^m}$. We also set β_1 as the weight of geographical distance (per mile) and β_2 as the weight of temporal distance (per minute) to weight the space and time dissimilarities in our space-time distance calculation, respectively. Different values of β_1 and β_2 result in different clusters. Suppose $f_{o_{j,n}^m, o_{j',n'}^{m'}}$ is the space-time distance between $o_{j,n}^m$ and $o_{j',n'}^{m'}$, and $f_{d_{j,n}^m, d_{j',n'}^{m'}}$ is the space-time distance between $d_{j,n}^m$ and $d_{j',n'}^{m'}$. Then, we propose equations (1)-(2) to calculate $f_{o_{j,n}^m, o_{j',n'}^{m'}}$ and $f_{d_{j,n}^m, d_{j',n'}^{m'}}$.

$$f_{o_{j,n}^m, o_{j',n'}^{m'}} = \beta_1 \times \sqrt{(x_{o_{j,n}^m} - x_{o_{j',n'}^{m'}})^2 + (y_{o_{j,n}^m} - y_{o_{j',n'}^{m'}})^2} + \beta_2 \times \left| t_{o_{j,n}^m} - t_{o_{j',n'}^{m'}} \right|, \tag{1}$$

$$f_{d_{j,n}^m, d_{j',n'}^{m'}} = \beta_1 \times \sqrt{(x_{d_{j,n}^m} - x_{d_{j',n'}^{m'}})^2 + (y_{d_{j,n}^m} - y_{d_{j',n'}^{m'}})^2} + \beta_2 \times \left| t_{d_{j,n}^m} - t_{d_{j',n'}^{m'}} \right|. \tag{2}$$

where $x_{\cdot}$ and $y_{\cdot}$ are the x-coordinate and y-coordinate of the corresponding dummy node. We note that these coordinates are exactly the same as the x/y-coordinate of the physical node corresponding to the dummy node. The first term of the right-hand side of equations (1)-(2) calculates the geographical distance, while the second term computes the temporal distance between two nodes.

The relationship between β_1 and β_2 is directly related to the average speed limit of transportation links in transportation networks. Let us assume that the average speed limit of transportation links is 25 miles per hour (mph). Suppose one mile as the unit of distance and one minute as the unit of time. Then, by driving with 25 mph speed, we can travel 0.417 miles in one minute. It means that 0.417 miles travel is equivalent to one-minute travel by a vehicle with the average speed of 25 mph; therefore, with 25 mph speed, β_1 is equal to 1 per mile and β_2 is equal to 0.417 per minute. Similarly, by driving with 60 mph speed, we can travel one mile in one minute. In other words, one-mile travel is equivalent to one-minute travel by a vehicle with the average speed of 60 mph; therefore, with 60 mph speed, β_1 is equal to 1 per mile and β_2 is equal to 1 per minute. Suppose the average speed varies between 25 mph and 60 mph, then $\beta_1 = 1$ and $0.417 \leq \beta_2 \leq 1$. Finally, let $r_{OD_{j,n}^m, OD_{j',n'}^{m'}}$ denote the measure of dissimilarity between $OD_{j,n}^m$ and $OD_{j',n'}^{m'}$ and is calculated by $max\{f_{o_{j,n}^m, o_{j',n'}^{m'}}, f_{d_{j,n}^m, d_{j',n'}^{m'}}\}$. We consider the maximum value of $f_{o_{j,n}^m, o_{j',n'}^{m'}}$ and $f_{d_{j,n}^m, d_{j',n'}^{m'}}$ for the calculation of $r_{OD_{j,n}^m, OD_{j',n'}^{m'}}$ to reflect the highest space-time dissimilarity between $OD_{j,n}^m$ and $OD_{j',n'}^{m'}$. Figure 5 illustrates an example in which three OD pairs have been defined in the same cluster.

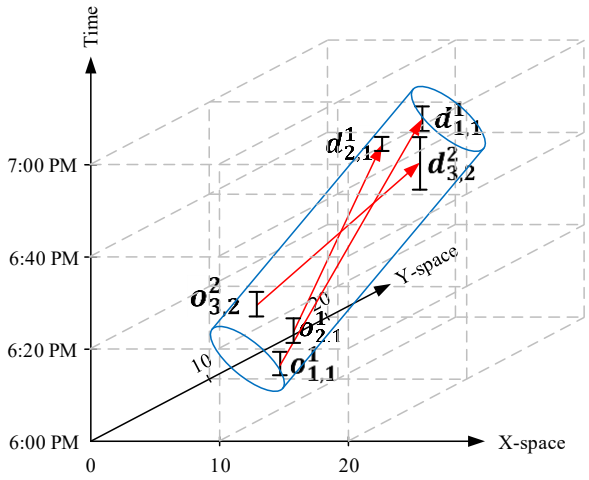


Fig. 5. A cluster of three OD pairs.

Thus, we have calculated the maximum dissimilarity between each OD pair and other OD pairs. In the next step, each OD pair is considered as an individual cluster. As a result, we can obtain the dissimilarity between $OD^m_{j,n}$ and cluster q by calculating the value of $r_{OD^m_{j,n},q}$. Of note, the dissimilarity between $OD^m_{j,n}$ and its corresponding cluster is equal to 0. Defining the pickup and delivery problem on multi-dimensional networks increases the computational efforts for solving the DP algorithm that will be discussed in Section 3.5. Our proposed DP can solve the pickup and delivery problem for at most α number of OD pairs per cluster. The value of α is 35 in our practice. We propose an IP model to cluster OD pairs:

$$Min\left\{\zeta_1 \times \sum_q \sum_j \sum_n \sum_m \left\{r_{OD^m_{j,n},q} . z_{OD^m_{j,n},q}\right\} + \zeta_2 \times \sum_q y_q\right\} \quad (3)$$

s.t.

$$\sum_j \sum_n \sum_m z_{OD^m_{j,n},q} \le \alpha y_q \qquad \forall q, \quad (4)$$

$$\sum_q z_{OD^m_{j,n},q} = 1 \qquad \forall j, \forall n, \forall m, \quad (5)$$

$$z_{OD^m_{j,n},q} \in \{0,1\};\ y_q \in \{0,1\} \qquad \forall q, \forall j, \forall n, \forall m. \quad (6)$$

Variable y_q equals 1 if cluster q exists, and 0 otherwise. Variable $z_{OD^m_{j,n},q}$ is also equal to 1 if $OD^m_{j,n}$ is assigned to cluster q, and 0 otherwise. Objective function in (3) consists of two terms: the first term captures the dissimilarity between each OD pair assigned to the cluster and the OD pair used to "seed" the cluster, and the second term reveals the total number of clusters. Without having term $\sum_q y_q$ in the objective function, each OD pair is assigned to its own cluster. We set ζ_1 as the weight of the former term and ζ_2 as the weight of the latter term in the objective function. Since different values of ζ_1 and ζ_2 result in different clusters, we adjust these values to generate variant clusters.

In this IP model, constraint (4) expresses that each cluster must contain up to α number of OD pairs. Moreover, each OD pair must be assigned to exactly one cluster (constraint (5)). This problem can be solved by any commercial solver, such as ILOG Cplex, Xpress-MP or GUROBI. In our experiments, we use GAMS Distribution 23.00.

The first term of objective function (3), i.e., $\sum_q \sum_j \sum_n \sum_m \left\{r_{OD^m_{j,n},q} . z_{OD^m_{j,n},q}\right\}$, could be much more accurate if it was formulated by the quadratic function $\sum_q \sum_{(j,n,m)} \sum_{j',n',m'} r_{OD^m_{j,n},OD^{m'}_{j',n'}} z_{OD^m_{j,n},q} z_{OD^{m'}_{j',n',q}}$. This quadratic function considers the pairwise dissimilarity between all pairs of OD pairs included in the same cluster. However, commercial solvers, such as CPLEX can handle the quadratic objective but may not perform well for larger problems and will not guarantee an optimal solution.

We note that our data (set of OD pairs) consists of space-time vectors. There are plenty of instances in the literature that have attempted to compare clustering algorithms in high dimensions. But there are too many factors involved: what does our data look like, how do we preprocess it, do we have a well-chosen and appropriate distance measure, how good is our implementation, do we have index acceleration to speed up some algorithms, etc. Our

proposed clustering algorithm is just a preprocessing step to break the main large-scale problem to several sub-problems. Nevertheless, we do not claim that our method performs better than other machine learning methods.

3.4. Routing inside the clusters

Routing cost of arc $(i, i', t, t', s, s', v, v)$, denoted by $c_{i,i',t,t',s,s',v}$, is the real physical movement cost. In this paper, we aim to minimize the routing cost of vehicles while enforcing vehicles to serve OD pairs within their route. To implement this goal, we provide some incentives for vehicles such that they select detours and serve OD pairs instead of selecting the direct route (least-cost path) from their origin depot to their destination depot. This can happen by considering a negative cost associated with the pickup arc of each OD pair in each cluster. Let $\overline{c}_{j,n}^{m}$ denote the service cost of m^{th} OD pair for way n for parcel j. Then, the routing cost for the pickup arc of $OD_{j,n}^{m}$ is the sum of its real physical movement cost and $\overline{c}_{j,n}^{m}$. Let $TC_{j,n}^{m}$ denote the total transportation cost of a vehicle when it leaves its origin depot to exclusively serve $OD_{j,n}^{m}$ and return to its destination depot. We initialize $\overline{c}_{j,n}^{m}$ by $(-TC_{j,n}^{m})$. We noticed that this method of initialization can reduce the convergence time of the Lagrangian heuristic algorithm that will be explained in Section 3.6; however, the quality of final solution is not significantly affected by the initial value of $\overline{c}_{j,n}^{m}$.

At this stage, we assume that the total number of vehicles in cluster q is known, that is equal to the total number of OD pairs in cluster q, and the goal is to minimize the routing cost of vehicles while enforcing vehicles to serve OD pairs within their route. We note that the vehicles serving OD pairs at this stage are not real and have been defined just for the sake of OD pairs' routing inside each cluster. We assume that all these hypothetical vehicles are homogenous in terms of capacity, origin and destination depots, and time horizon. Let us find the average x-coordinate of all OD pairs' origin in cluster q and denote it by $\overline{x}_{o,q}$. Similarly, $\overline{y}_{o,q}$ denotes the average y-coordinate of all OD pairs' origin in cluster q. Then, the closest physical transportation node to $[\overline{x}_{o,q}, \overline{y}_{o,q}]$ is assumed as the origin depot of all vehicles in cluster q, and the dummy node corresponding to this origin depot is added to the network and symbolized by o_q. We also find the x- and y-coordinates of all OD pairs' destination in cluster q and denote them by $\overline{x}_{d,q}$ and $\overline{y}_{d,q}$, respectively. Then, the closest physical transportation node to $[\overline{x}_{d,q}, \overline{y}_{d,q}]$ is assumed as the destination depot of all vehicles in cluster q, and the dummy node corresponding to this destination depot is added to the network and symbolized by d_q.

We also guarantee that the vehicles in cluster q have enough time to reach all OD pairs' origin and destination. In other words, if an OD pair is not served in cluster q, it should not be due to the inaccessibility of the OD pair's origin/destination to the vehicles' depots. Therefore, we assume that all vehicles in cluster q start their routes from the least earliest departure time of all OD pairs in cluster q minus a sufficient tolerant time interval, denoted by $\underline{t}_q$, and end them at the utmost latest arrival time of all OD pairs in cluster q plus a sufficient tolerant time interval, denoted by $\overline{t}_q$. The sufficient tolerant time interval has been set as 20 minutes in our computational experiments.

Thus far, we clustered OD pairs and assigned a number of hypothetical vehicles to serve the OD pairs inside each cluster. Then, we construct a network for each cluster, in which each vehicle starts its route from o_q at time $\underline{t}_q$, choose the least-cost path and may or may not serve OD pairs within its route, and ends its route to d_q at time $\overline{t}_q$. Finally, the vehicle transmits the information related to the cumulative service state of OD pairs to the next vehicle. This is exactly a time-dependent state-dependent least-cost path problem whose mathematical model is provided in the next section.

In the model proposed by Mahmoudi and Zhou (2016) for the VRPPDTW, if an OD pair has a high negative cost in one iteration, it may be served several times by multiple vehicles. To prevent from this issue, their algorithm considers a penalty for that OD pair in the next iteration to prevent vehicles from serving that OD pair. Then, after some iterations, the marginal cost of each OD pair converges to a particular value. However, if OD pairs are densely distributed in the transportation network, it is quite possible that the marginal cost of some OD pairs never converges to a specific value. In this paper, the serial structure of our network is such that the vehicles are routed one by one, and the information related to the parcels' service state is transferred from the current vehicle to the next one. Thanks to this feature in our network, an OD pair is never served by multiple vehicles, and therefore, we never face this issue.

Based on the constructed network that can capture vehicles' capacity constraints, as well as desired departure and arrival time windows and precedence constraints, we now start constructing a least-cost path model for the local clusters derived from the original large-scale real-world instances. The model uses binary variables $x_{i,i',t,t',s,s',v,v'}$ which is equal to 1 if arc $(i, i', t, t', s, s', v, v')$ is used, and 0 otherwise. The objective is to find a set of minimum-cost vehicle routes. Then, the problem can be mathematically modeled as follows:

$$Min \sum_{i,i',t,t',s,s',v}\{c_{i,i',t,t',s,s',v} \cdot x_{i,i',t,t',s,s',v,v}\} \quad (7)$$

s.t.

$$\sum_{i',t',s'} x_{o_q,i',\underline{t}_q,t',\phi,s',v_1,v_1} = 1, \tag{8}$$

$$\sum_{i,t,s,s'} x_{i,d_q,t,\bar{t}_q,s,s',v_{|Vq|},v_{|Vq|}} = 1, \tag{9}$$

$$\sum_{i',t',s',v'} x_{i,i',t,t',s,s',v,v'} - \sum_{i',t',s',v'} x_{i',i,t',t,s',s,v',v} = 0 \qquad \forall (i,t,s,v), \tag{10}$$

$$x_{i,i',t,t',s,s',v,v'} \in \{0,1\} \qquad \forall (i,i',t,t',s,s',v,v'). \tag{11}$$

The objective function (7) minimizes total routing costs, including all real physical movement costs, waiting costs, preparation costs, and all $\bar{c}_{j,n}^{m}$ values for any OD pair (j,n,m) selected. Constraints (8)-(10) ensure flow balance on every vertex in the network. Constraint (11) defines the binary decision variables.

Cumulative service states at the ending layer of the last vehicle of a cluster ensure that, for each parcel, OD pairs from at most one way are selected. To guarantee the sequencing of OD pairs in way n, once a service arc corresponding to parcel j's pickup/drop-off is traversed, the time corresponding to this traverse is recorded. Then, the time windows corresponding to the dummy nodes related to the former and latter OD pairs of way n for parcel j that have not been traversed yet are modified. If more than one OD pair from a way exists in a cluster, by our definition for feasible state transition, the sequencing of OD pairs in that way is guaranteed. Moreover, by our definition for feasible cumulative service states at the ending layer of the last vehicle of the cluster, if in a cluster, any OD pair from way n for parcel j is selected, all OD pairs of way n for parcel j existing in the cluster are selected. To sum up, the least-cost path problem solved in this step uses the cumulative service states and expanded network structure to guarantee that:

(i) OD pairs from at most one way for each parcel are selected,
(ii) any subset of OD pairs from one way for a parcel that are selected occur at times that are feasible (their time sequence corresponds to that of their way and there is enough time to complete any intermediate OD pairs from the way that are not in this cluster), and
(iii) if any OD pair from a way for a parcel in the cluster is selected, then every OD pair for that way and for that parcel that occurs in the cluster is selected.

3.5. Routing outside the clusters

Our clustering procedure may raise the question about whether or not a vehicle can serve in more than one cluster. The answer would be "yes". In fact, if the space-time vertex at which vehicle v' from cluster q' picks up its first parcel is accessible for the space-time vertex at which vehicle v from cluster q drops off its last parcel, then the tasks of vehicles v and v' can be performed by one vehicle. A task/work piece is a number of OD pairs already served by one vehicle. To sum up, vehicles can be utilized more during their time horizon if they could serve in more than one cluster. Figure 6 illustrates this procedure by an example in which seven OD pairs (grouped in three clusters) are served by a single vehicle.

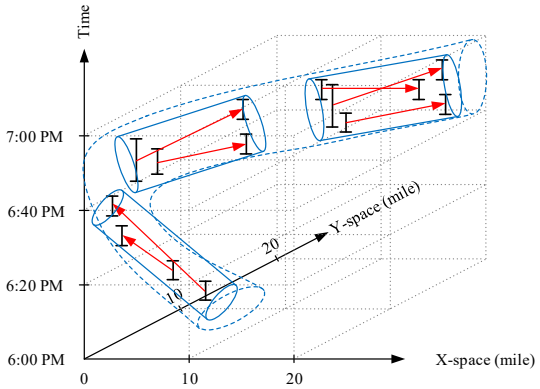


Fig. 6. An illustration of three clusters performed by one vehicle.

At this stage, we work with the real heterogonous vehicles distributed in the transportation network. We find the optimal chains of tasks that can be performed by each real vehicle. We note that each task/work piece has already been completed by a hypothetical vehicle from Section 3.4. The total number of real vehicles is unknown; however,

potential origin/destination depots for these vehicles are known beforehand. Let E denote the set of potential origin depots and E' the set of potential destination depots for real vehicles. Let $\mathcal{W}$ denote the set of work pieces generated in Section 3.4 (so, $\mathcal{W}$ represents a set of hypothetical vehicles' activities, each consisting of a sequence of pickup, travel and drop-off operations occurring at specified times). We define a network to have nodes $E \cup \mathcal{W} \cup E'$ and arcs given by a subset $\mathcal{A} \subseteq (E \times \mathcal{W}) \cup (\mathcal{W} \times \mathcal{W}) \cup (\mathcal{W} \times E')$, so that:

- $(e, \omega) \in \mathcal{A}$ for $e \in E$ and $\omega \in \mathcal{W}$ if and only if a real vehicle starting at origin depot e can perform work piece ω,
- $(\omega, \omega') \in \mathcal{A}$ for $\omega, \omega' \in \mathcal{W}$ if and only if $\omega \neq \omega'$and a real vehicle can perform work piece ω' immediately after completing work piece ω, and
- $(\omega, e') \in \mathcal{A}$ for $\omega \in \mathcal{W}$ and $e' \in E'$ if and only if a real vehicle can perform work piece ω before ending at destination depot e'.

For each $a \in \mathcal{A}$, the cost c_a is

- the transport cost of getting from e to the start of ω, plus M, a cost per real vehicle used, if $a = (e, \omega) \in E \times \mathcal{W}$,
- the cost of performing work piece ω (calculated in Section 3.4) plus the cost of transport from the end of ω to the start of ω' if $a = (\omega, \omega') \in \mathcal{W} \times \mathcal{W}$, and is
- the cost of performing work piece ω (calculated in Section 3.4) plus the cost of transport from the end of ω to depot e' if $a = (\omega, e') \in \mathcal{W} \times E'$.

The problem to be solved in this section is to construct a set of paths in the network, each starting at a node in E and ending at a node in E', so that the paths induce a partition of $\mathcal{W}$, and total cost of arcs used in paths is minimized. This problem can be modeled as an IP model as follows: Define $\delta^+(\omega)$ to be the set of arcs in $\mathcal{A}$ leaving $\omega \in \mathcal{W}$ and $\delta^-(\omega)$ to be the set of arcs in $\mathcal{A}$ entering $\omega \in \mathcal{W}$. We also define decision variable $y_a = 1$ if arc $a \in \mathcal{A}$ appears in a path for a real vehicle; $y_a = 0$ otherwise. Now, the IP model is as follows:

$$min \sum_{a \in \mathcal{A}} c_a y_a \tag{12}$$

s.t.

$$\sum_{a \in \delta^+(\omega)} y_a = 1 \qquad \forall \omega \in \mathcal{W}, \tag{13}$$

$$\sum_{a \in \delta^-(\omega)} y_a = 1 \qquad \forall \omega \in \mathcal{W}, \tag{14}$$

$$y \in \{0,1\}^{\mathcal{A}}. \tag{15}$$

We note that due to the start and end times associated with each work piece $\omega \in \mathcal{W}$, the network with arcs $\mathcal{A}$ will be acyclic. Hence, there is no need for constraints to remove sub-tours in this model. This problem can be viewed as a special, simpler, case of a well-known problem, named the "vehicle scheduling problem" (see Bunte and Kliewer, 2009 for more details).

From a column generation standpoint (Dumas et al., 1991), our approach only examines the chains containing *optimal* routes of multiple vehicles, so-called "*long*" columns, in comparison with the commonly used single vehicle "*short*" columns in a generic column generation scheme. Thus, our algorithms can be viewed as an adaption of column generation algorithm that operates on a small set of long columns with parcel-to-vehicle assignment-routing solutions.

3.6. Lagrangian heuristic for OD pairs' marginal cost adjustment

In the solution obtained from Section 3.5, it is possible that OD pairs from multiple ways for a parcel are selected (the ways were in different clusters). To address this issue, we use constraints that we call leg-covering constraints. These exploit the observation that if a parcel is served at most one way (or partially served by OD-pairs from no more than one way), then at most one OD pair in any set consisting of no more than one OD pair from each of the parcel's ways can be used. Let $O_{j,n}$ denote the set of origin nodes, i.e., the set of nodes of the form $o_{j,n}^m$, for each parcel j and each n indexing a way for that parcel. Then, the leg-covering constraints are as follows:

$$\sum_{(i,i',t,t',s,s',v,v): i \in S, i' \notin S} x_{i,i',t,t',s,s',v,v} \leq 1 \qquad \forall j, \forall S \subseteq \bigcup_n O_{j,n} \ s.t. \left|S \cap O_{j,n}\right| = 1, \forall n. \tag{16}$$

Since the number of ways for each parcel is expected to be small, enumerating these constraints is not prohibitive. We use the example illustrated in Figure 2 to better explain the concept of leg-covering cuts (Figure 7). In this example, $O_{1,1} = \{o^1_{1,1}\}$, $O_{1,2} = \{o^1_{1,2}, o^2_{1,2}\}$, and $S \in \left\{\{o^1_{1,1}, o^1_{1,2}\}, \{o^1_{1,1}, o^2_{1,2}\}\right\}$.

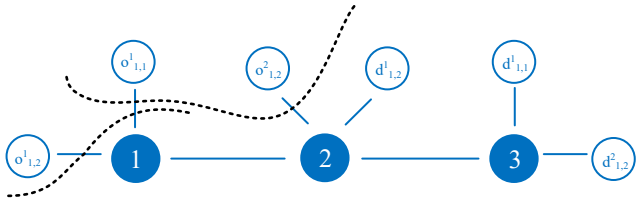


Fig. 7. Two leg-covering cuts to guarantee that parcel 1 is served by exactly one way.

In a half-finished way for a parcel, some but not all OD pairs for the way have been selected (where the OD pairs for one way are split between different clusters and the parcel is not picked-up in all these clusters). In order to prevent half-finished ways, we add another constraint presented as follows:

$$\sum_{i,t,t',s,s',v} x_{o^m_{j,n},i,t,t',s,s',v,v} \leq \sum_{i,t,t',s,s',v} x_{o^{m+1}_{j,n},i,t,t',s,s',v,v}, \qquad \forall j, \forall n, \forall m. \tag{17}$$

We tune the marginal cost of every leg-covering cut and half-finished way for each parcel and subsequently adjust the marginal cost of each OD pair by using the sub-gradient method. Let k denote the iteration number and $\theta_{k,j,n,S}$ and $\lambda_{k,j,n,S}$ denote the step size and Lagrangian multiplier corresponding to cut (j,n,S) at iteration k, respectively. First, we initialize $\theta_{0,j,n,S}$ to a base cost, where base cost has a negative value and is defined as the total negative costs obtained by selecting every link (i,i'), where $i \in S, i' \notin S$. We also set $\lambda_{0,j,n,S} = 0$. Second, we calculate the sub-gradient of cut (j,n,S), denoted by $\nabla L_{\lambda_{k,j,n,S}}$, using equation (18). Third, we update the Lagrangian multiplier corresponding to cut (j,n,S) for the next iteration by (19). Finally, we update the step size of the next iteration by equation (20). After calculating $\lambda_{k+1,j,n,S}$, we distribute the updated marginal cost of cut (j,n,S) among all links, whose origin node belongs to set S, based on their current marginal cost.

$$\nabla L_{\lambda_{k,j,n,S}} = 1 - \sum_{(i,i',t,t',s,s',v,v): i \in S, i' \notin S} x_{i,i',t,t',s,s',v,v}, \tag{18}$$

$$\lambda_{k+1,j,n,S} = min\,(0, \lambda_{k,j,n,S} + \theta_{k,j,n,S}.\nabla L_{\lambda_{k,j,n,S}}), \tag{19}$$

$$\theta_{k,j,n,S} = \frac{\theta_{0,j,n,S}}{k+1}. \tag{20}$$

We do the same procedure for constraint (17); $\theta'_{k,j,n,m}$ and $\lambda'_{k,j,n,m}$ denote the step size and Lagrangian multiplier corresponding to constraint (17) at iteration k, respectively. First, we initialize $\theta'_{0,j,n,m}$ to a base cost, where base cost has a positive value and is defined as the absolute value of the difference between the negative costs of links originated from $o^m_{j,n}$ and $o^{m+1}_{j,n}$. We also set $\lambda'_{0,j,n,m} = 0$. Second, we calculate the sub-gradient corresponding to constraint (17) at iteration k, denoted by $\nabla L'_{\lambda'_{k,j,n,m}}$, by equation (21). Third, we update the Lagrangian multiplier corresponding to the constraint for the next iteration with equation (22). Finally, we update the step size of the next iteration by equation (23). After calculating $\lambda'_{k+1,j,n,m}$, we distribute the updated marginal cost of constraint (17) among the links originated from $o^m_{j,n}$ and $o^{m+1}_{j,n}$ based on their current marginal cost. Our proposed Lagrangian heuristic stops if the total number of iterations becomes greater than a predetermined maximum iteration number (i.e., 30 iterations in our experiments).

$$\nabla L'_{\lambda'_{k,j,n,m}} = \sum_{i,t,t',s,s',v} x_{o^m_{j,n},i,t,t',s,s',v,v} - \sum_{i,t,t',s,s',v} x_{o^{m+1}_{j,n},i,t,t',s,s',v,v}, \tag{21}$$

$$\lambda'_{k+1,j,n,m} = min\,(0, \lambda'_{k,j,n,m} + \theta'_{k,j,n,m}.\nabla L'_{\lambda'_{k,j,n,m}}), \tag{22}$$

$$\theta'_{k,j,n,m} = \frac{\theta'_{0,j,n,m}}{k+1}. \tag{23}$$

To sum up, at each iteration, we solve the least-cost path problem presented in Section 3.4. Then, we solve the IP model presented in Section 3.5. Finally, we adjust the marginal cost of each OD pair by using the sub-gradient method to alleviate the issues related to selecting multiple and/or half-finished ways for a parcel. These steps provide

a lower bound solution for the primal problem. Every parcel in the lower bound solution has one of the following situations:

(i) No OD pair for the parcel has been selected;
(ii) Some but not all OD pairs of a way for the parcel have been selected. This results in a half-finished way;
(iii) OD pairs of multiple ways for the parcel have been selected. These ways can be complete or half-finished.
(iv) All OD pairs of a way for the parcel have been selected. This results in a complete way;

Among all these four situations, only situation (iv) results in feasibility. Situations (ii) and (iii) both result in infeasibility. Situation (i) also causes infeasibility if the objective is to serve all parcels. At the end of each iteration, we develop a heuristic to generate an upper bound based on the solution we obtain from the lower bound.

Step 1:

- If situation (i) occurs for a parcel and causes infeasibility, assign its non-stop way to the parcel. In our experiments, each parcel can be served by its non-stop way. Consider the OD pair corresponding to this non-stop way as a new work piece.
- If situation (ii) occurs for a parcel, assign the half-finished way to the parcel and consider the unserved OD pairs of this way as new separate work pieces.
- If situation (iii) occurs for a parcel and at least one complete way exists among the selected ways, maintain one of the complete ways for the parcel in the upper bound solution and ignore all others.
- If situation (iii) occurs for a parcel and no complete way exists among the selected ways, assign the half-finished way with the least number of unserved OD pairs to the parcel, consider the unserved OD pairs corresponding to this way as new separate work pieces, and ignore all other ways.
- If situation (iv) occurs for a parcel, maintain the way for the parcel in the upper bound solution.

Of note, by ignoring the selected ways, we simply assume that the dummy nodes associated with the OD pairs of these ways were traversed as physical transportation nodes in the least-cost path algorithm presented in Section 3.4.

Step 2: Solve the IP model presented in Section 3.5 again when new work pieces from Step 1 are added to the set of work pieces obtained from Section 3.4. Update the global upper bound (i.e., a solution with the least total cost among all upper bound solutions thus far).

Here, we provide a summary of the steps we take to perform the cluster-first, route-second approach:

Step 1(a): Heuristically cluster all OD pairs in all possible ways, so that there are at most α (set to 35 in practice) OD pairs per cluster (Section 3.3).

Step 1(b): Initialize the cost of each OD pair for each way of each parcel to be the negative of the total transportation costs of a vehicle when it leaves its origin depot, exclusively serves $OD_{j,n}^{m}$, and returns to its destination depot (Section 3.4).

Step 1(c): Use a rule of thumb to construct a hypothetical vehicle origin, destination and time window for each cluster (Section 3.4).

Step 2: For each cluster, solve a least-cost path problem in an expanded network to select the subset of OD pairs to serve by hypothetical vehicles, and to decide the time of each pick-up and drop-off operation in each selected OD pair. Each hypothetical vehicle performs a sequence of parcel pick-up and drop-off operations that is feasible in time and space that does not violate its capacity. The objective is to minimize "total routing costs," while enforcing vehicles to serve OD pairs within their route (Section 3.4).

Step 3: The activities of each hypothetical vehicle in each cluster forms a "work piece." These are partitioned (and sequenced, implied by the timing of the work pieces) by finding a partitioning of an acyclic network into paths. Each

path is a sequence of activities to be performed by a real vehicle. The objective is to minimize "routing cost" (Section 3.5). This may result in (i) no OD pairs for a parcel being selected, (ii) some but not all OD pairs for one way of a parcel being selected, or (iii) OD pairs from multiple ways for a parcel being selected. Issue (i) is addressed implicitly by the initialization of $\bar{c}_{j,n}^{m}$ to be negative values.

Step 4. Adjust $\bar{c}_{j,n}^{m}$ to address issues (ii) and (iii) in Step 3 by Lagrangian relaxation of constraints (16)-(17) to enforce them with sub-gradient optimization used to adjust the Lagrangian multipliers (Section 3.6). A heuristic is proposed to generate the upper bound from the solution obtained from the lower bound. The global upper bound is also updated. Go to Step 2.

3.7. On the computational complexity of the DP algorithm used for each cluster

Several efficient algorithms have been suggested to solve the time-dependent least-cost path problem on a network with time-dependent arc costs (see the papers by Ziliaskopoulos and Mahmassani, 1993 and Chabini, 1998 in the context of deterministic networks; Miller-Hooks and Mahmassani, 1998 and 2000 in the context of stochastic networks). We have used a time-dependent state-dependent DP algorithm to solve the least-cost path problem obtained from Section 3.4. Assume that the unit of time is one minute. Let $L_{i,t,s,v}$ denote the label of vertex (i,t,s,v) and the term "pred" stands for the predecessor. Algorithm 1, described below, presents the proposed time-dependent forward DP algorithm.

```
// Algorithm 1: Time-dependent forward DP algorithm for each cluster q
  for each vehicle v, v ∈ V_q do
  begin
    // initialization
    L_.,.,.,. := +∞;
    node pred of vertex (.,.,.,.) := −1;
    time pred of vertex (.,.,.,.) := −1;
    state pred of vertex (.,.,.,.) := −1;
    vehicle pred of vertex (.,.,.,.) := −1;
    for each time t ∈ [t_q, t̄_q] do
    begin
      for each link (i, i') do
      begin
        for each state s do
        begin
          derive downstream state s' based on the feasible state transition on link (i, i')
          derive arrival time t' = t + TT_{i,i',t};
          derive v'; // Section 3.2, Condition 1
          if (L_{i,t,s,v} + c_{i,i',t,t',s,s',v} < L_{i',t',s',v'})  // The condition corresponds to the Bellman optimality condition.
          begin
            L_{i',t',s',v'} := L_{i,t,s,v} + c_{i,i',t,t',s,s',v} ; // label update
            node pred of vertex (i', t', s', v') := i;
            time pred of vertex (i', t', s', v') := t;
            state pred of vertex (i', t', s', v') := s;
            vehicle pred of vertex (i', t', s', v') := v;
          end;
        end; // for each state
      end; // for each link
    end; // for each time
  end; // for each vehicle
```

Let L_q and T_q denote the set of links and time stamps in cluster q, respectively. According to our definition for cumulative service states, the total number of cumulative service states for α number of OD pairs in cluster q is 3^{α}, because each OD pair's cumulative service state can be 0, 1, or 2. Therefore, the space complexity of our proposed DP algorithm for each cluster is $O(\alpha 3^{\alpha}|T_q||L_q|)$. Our experiments were performed on an Intel Workstation running two Xeon E5-2680 processors clocked at 2.80 GHz with 20 cores and 192GB RAM running Windows Server 2008 x64 Edition. Let us assume that $|T_q| = 10^3$ and $|L_q| = 10^3$. In the innermost loop of Algorithm 1, we record five

data for vertex (i', t', s', v'): the label, node predecessor, time predecessor, state predecessor, and vehicle predecessor. As a result, $5\alpha 3^{\alpha} \times 10^{6}$ bytes of memory are required. To find the maximum value of α for any machine that runs our algorithm, it is sufficient to find the solution for this inequality: $5\alpha 3^{\alpha} \times 10^{6} \leq$ *total available memory*. For the machine we are running our experiments on, 192GB RAM is available; therefore, the maximum theoretical value of α is 7. We note that due to the existence of time and state dimensions in our model, the calculated α is not very large. In practice, the actual value of α solvable by our exact DP on our machine is larger than 7 (i.e., $\alpha \approx 35$), as explained below.

According to Algorithm 1, the network is created before DP implementation. This means that, before DP starts scanning vertexes, for each time t and each node i, all states corresponding to the OD pairs of cluster q must be generated. Sometimes the total number of feasible states for (i, t) is quite large such that the size of the network becomes enormous. In this case, we suggest creating the network dynamically within the scanning process (which could considerably reduce the search space). We also suggest using a beam search algorithm to keep a limited number of states for vertexes built from node i and time t at each level when the number of candidate states is large.

The beam search algorithm reduces the search space by setting dominance rules and solves practical problems within reasonable time and memory. The beam search algorithm is a greedy search algorithm that uses breadth-first search to explore the search tree. In addition, only a limited number of promising nodes (states in our case) are kept at each level of the search tree, and other nodes are pruned off. In this algorithm, beam width refers to the number of promising nodes left at each level. In general, beam width restricts the memory required to perform the tree search. We note that fewer nodes are cut off with greater beam width. The beam search algorithm finally generates a representative set of non-dominated or promising solutions. In order to explain our beam search algorithm, we elaborate the concept of cumulative flow count diagram first.

Edie and Foote (1960) first designed cumulative curves to describe the cumulative flow count of vehicles, and Gazis and Potts (1963) used a cumulative diagram as a predictive tool for the first time. Newell (1993) further presented a three-dimensional version of a cumulative diagram regarding space, time, and cumulative flow, to merge the concepts of cumulative diagrams with wave theory (Makigami et al., 1971). Daganzo (2001) presented a seminal study for extending N curves to study the dynamics and stability of supply chain systems. Our cumulative flow count approach sheds more light on a recent development direction in the field of discrete-time IP (Boland et al., 2017), which aims to iteratively refine and find optimal continuous-time solutions without explicitly modeling the microscopic state changes along the discrete time dimension.

Cumulative flow count diagrams are effective in describing the service process in the queueing system. The service process represents the required time and resources to serve a parcel. The performance of the queueing system is defined by the arrival process, service process, and queue discipline. We use time-dependent cumulative flow counts for each agent (i.e., parcel) to capture the arrival and departure of traveling objects. Figure 8 illustrates the graphs corresponding to parcels' cumulative arrival, on-board, and departure count diagrams. The region bounded by the parcels' cumulative arrival and on-board count diagrams represents the total waiting time for parcels to be picked up. The area bounded by the parcels' cumulative on-board and departure count diagrams represents the total service time.

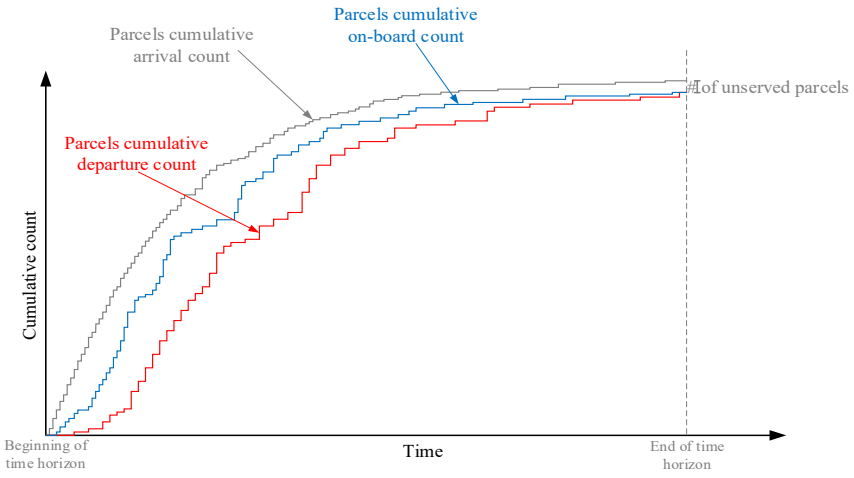


Fig. 8. Cumulative arrival, departure, and on-board count diagrams.

Regarding the performance of a vehicle routing system, we propose four criteria as evaluation rules during the node selection procedure to be used in the beam search process. Let $C_A^p(t)$, $C_O^p(t)$, $C_D^p(t)$, $C_A^v(t)$, and $C_D^v(t)$ denote parcels' cumulative arrival count to the system at time t, parcels' cumulative on-board count at time t, parcels' cumulative departure count from the system at time t, vehicles' cumulative arrival count to the system at time t, and vehicles' cumulative departure count from the system at time t, respectively. Then, our four criteria are presented as follows:

- The total number of unserved parcels at time t are minimized. This criterion is calculated by $C_A^p(t) - C_D^p(t)$.
- To improve service quality, the total waiting time of parcels who are receiving service until time t are minimized. This criterion is calculated by $\sum_{t'=\underline{t}_q}^{t}[C_A^p(t') - C_O^p(t')]$.
- The total serving time of parcels until time t should also be minimized to improve parcels' service quality. This criterion is calculated by $\sum_{t'=\underline{t}_q}^{t}[C_O^p(t') - C_D^p(t')]$.
- The total processing time for vehicles until time t indicates the efficiency of the system to some degree. This criterion is represented by $\sum_{t'=\underline{t}_q}^{t}[C_A^v(t') - C_D^v(t')]$.

In our experiments, we consider two different approaches to define these dominance rules. The first approach is to define a utility function, which is a weighted combination of all four criteria. Then, the top twenty states with the highest utility will be selected for further exploration, while other states will be pruned off (twenty is set after parameter tuning). In the second approach, we keep the top twenty states with the highest collected profit (negative cost) obtained from picking up OD pairs. It may be interpreted that states at which more OD pairs are picked up have higher priority to be selected.

Here, we use the example mentioned in Section 3.1 to show vehicles' processing time as a measure of dynamic system performance. Vehicles' processing time are checked during the node selection procedure in the beam search process. Figure 9 shows the space-vehicle-time path by which each parcel is served by a non-stop trip. This path is as follow: $o_1 \to 1 \to o_{1,1}^1 \to 1 \to 2 \to 3 \to d_{1,1}^1 \to 3 \to 2 \to 1 \to d_1 \to o_2 \to 3 \to o_{2,1}^1 \to 3 \to 2 \to 1 \to d_{2,1}^1 \to 1 \to 2 \to 3 \to d_2$. The state corresponding to the vertex at which the state transition occurs is shown in Figure 9 as well. As can be seen, if parcels are served by non-stop trips, deadheading for vehicles occurs. In other words, each vehicle must travel for 1 hour without carrying any parcel to return to its destination depot. In this case, the total processing time for each vehicle is equal to 2 hours. Figure 10 shows the space-vehicle-time path by which each parcel is served by a one-stop trip. This path is $o_1 \to 1 \to o_{1,2}^1 \to 1 \to 2 \to d_{1,2}^1 \to 2 \to o_{2,2}^2 \to 2 \to 1 \to d_{2,2}^2 \to 1 \to d_1 \to o_2 \to 3 \to o_{2,2}^1 \to 3 \to 2 \to d_{2,2}^1 \to 2 \to o_{1,2}^2 \to 2 \to 3 \to d_{1,2}^2 \to 3 \to d_2$. In this case, since parcels are delivered by one-stop trips, deadheading does not occur, and the total processing time for each vehicle is equal to 1 hour.

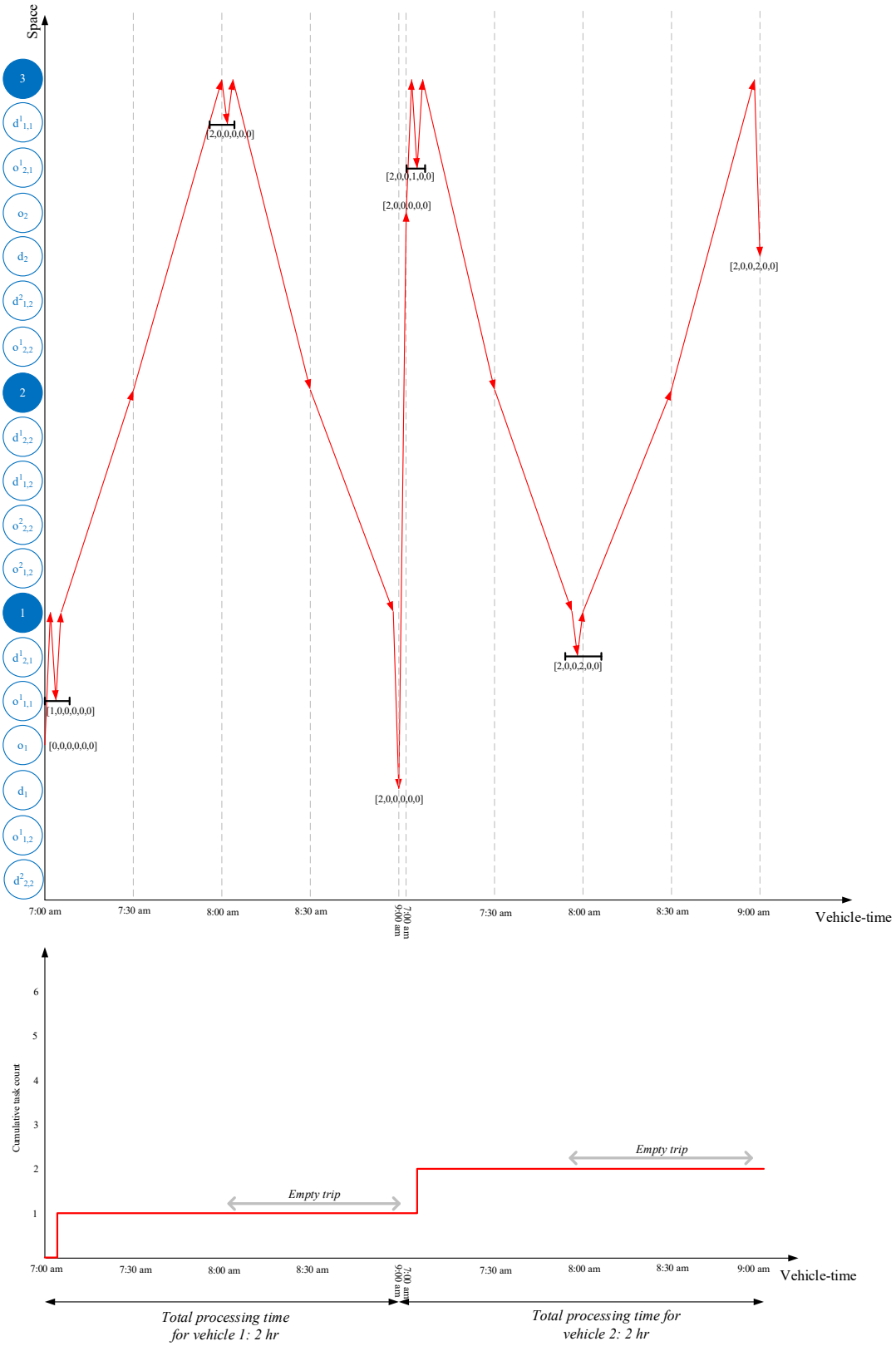


Fig. 9. Measuring the dynamic system performance in a pickup and delivery without any transfer.

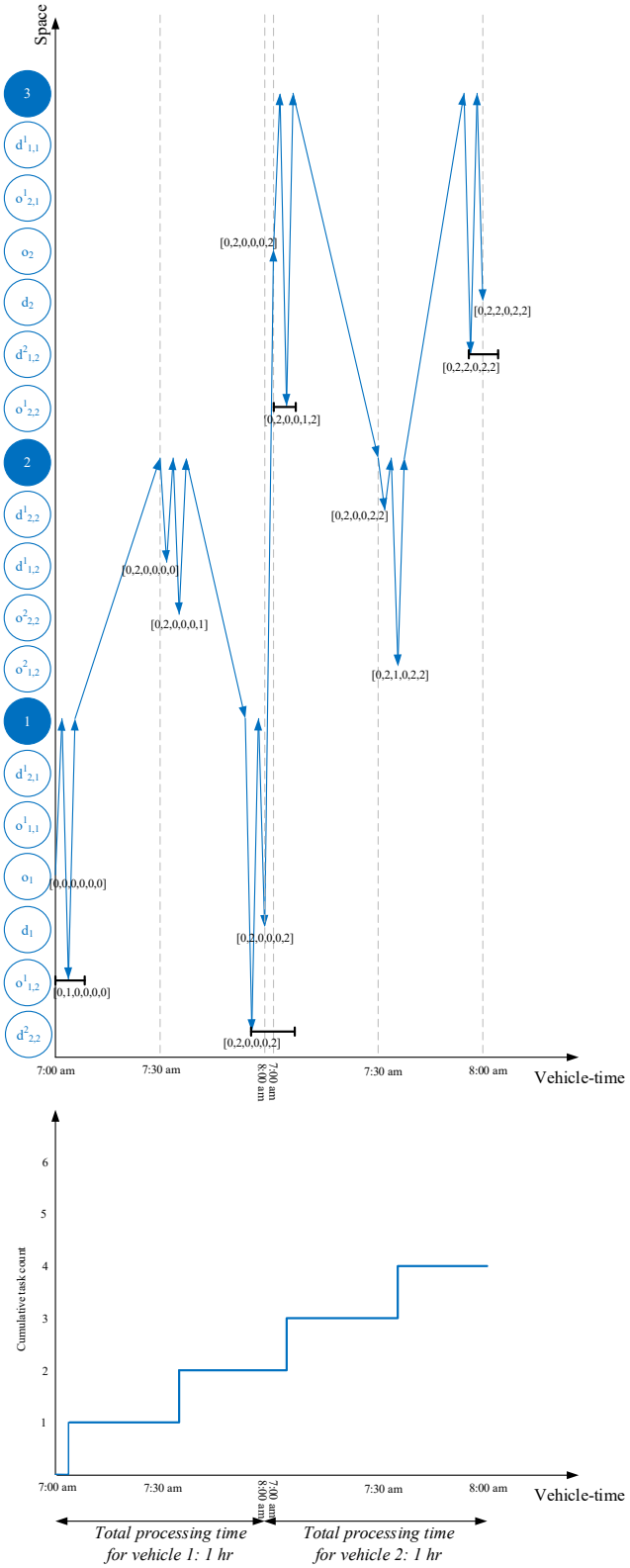


Fig. 10. Measuring the dynamic system performance in a pickup and delivery with a transfer.

4. Computational experiments

The time-dependent state-dependent DP described in this paper was coded in C++ platforms, while OD pairs' clustering and work pieces routing were solved by GAMS Distribution 23.00. The experiments were performed on an Intel Workstation running two Xeon E5-2680 processors clocked at 2.80 GHz with 20 cores and 192GB RAM running Windows Server 2008 x64 Edition. In this section, we examine our proposed model on data set used by Ropke and Pisinger (2006) which is publicly available at http://www.diku.dk/~sropke/, followed by the data provided by Cainiao Network available at https://tianchi.shuju.aliyun.com/datalab/index.htm to demonstrate the computational efficiency of our developed algorithm. The complete C++ implementation of the proposed DP algorithm and data set for the three-node example are available at https://github.com/xzhou99/Agent-Plus. We note that, in this paper, each parcel has at most three different ways to be served, and there are several parcels in our experiments that are served by non-stop trips. In both data sets, the algorithm terminates after 30 iterations.

4.1. The data set used by Ropke and Pisinger (2006)

The data set used by Ropke and Pisinger (2006) is the modified version of instances initially introduced by Savelsbergh and Sol (1998). In this data set, parcel p's origin and destination are denoted by node p and node $n + p$, respectively, where n is the total number of parcels in the system. In addition, the coordinates (x and y) of parcels' origin and destination are randomly generated and uniformly distributed over a $[0,50] \times [0,50]$ square. A single depot is located in $[25,25]$. Moreover, $[0,1000]$ is considered as the vehicles' time horizon (vehicles' time horizon is assumed to be identical). Feasible departure/arrival time windows are also randomly generated for each parcel.

Each instance in this data set is named by two alphabetical letters (i.e., AA, BB, CC, DD, XX, or YY) and a double-digit number after the alphabetical letters. The two alphabetical letters are representative of the class of the instance, while the double-digit number demonstrates the total number of parcels in the instance. Each class presents the characteristics of its instances including vehicles' capacity, length of parcels' departure time windows, and load of parcels (Table 1). The load of each parcel is randomly generated from a given set (Table 1). We note that the load of each parcel does not exceed the maximum capacity of the vehicles. The capacity of vehicles for instances of a class is the same.

Table 1
The characteristics of the instances of the data set used by Ropke and Pisinger (2006)

Class	Vehicles' capacity	Length of parcels' departure time windows	The set from which the loads are randomly generated
AA	15	60	[5,15]
BB	20	60	[5,20]
CC	15	120	[5,15]
DD	20	120	[5,20]
XX	15	60	[1,15]
YY	15	120	[1,15]

According to Table 1, we expect more levels of complexity in instances XX and YY in comparison to instances AA, BB, CC, and DD due to the wide range of parcel's load. To the best of our knowledge, very few papers have published the results of instances XX and YY. We note that, in this data set, the objective is to serve all parcels with fewer vehicles. Table 2 presents the results obtained from our algorithm on the instances of the data set used by Ropke and Pisinger (2006). The third and fourth columns of Table 2 from left, denoted by "*A*" and "*B*", present the total number of vehicles used to serve all parcels in our algorithm and the algorithm proposed by Ropke and Pisinger (2006), respectively. To compare the performance of two algorithms in terms of number of vehicles, we define the fifth column of Table 2 from left as the ratio of *A* to *B*. The sixth and seventh columns of Table 2 from left, denoted by "*C*" and "*D*", present the upper bound of total routing cost obtained from solving the problem by our algorithm and the algorithm proposed by Ropke and Pisinger (2006), respectively. To compare the performance of two algorithms in terms of routing cost, we define the eighth column of Table 2 from left as the ratio of *C* to *D*. The ninth column of Table 2 from left, denoted by "*E*", presents the lower bound of total routing cost obtained from our algorithm. The tenth column of Table 2 from left, calculated by $(\frac{C-E}{C} \times 100)$, presents the gap percentage between lower and upper bounds of our solution. According to Table 2, for the four largest AA instances, for one of the medium-scale CC instances and for all of the DD instances, our heuristic performed substantially better than that of Ropke and Pisinger (2006) (in terms of both number of vehicles and routing cost). The situation was reversed for the four largest BB instances and four of the six largest CC instances, with the method of Ropke and Pisinger (2006) performing substantially better in both respects. On all other instances, the results were similar.

Table 2
A comparison of the results obtained from our algorithm and the algorithm proposed by Ropke and Pisinger (2006).

Name	# of clusters	A	B	$\frac{A}{B}$	C	D	$\frac{C}{D}$	E	Gap %	Computation time (sec)	Computation time (sec) (Ropke and Pisinger (2006))
AA30	5	4	5	0.80	41,316	51,317.40	0.81	41288	0.07	25.6	27
AA35	5	5	5	1.00	51,506	51,343.53	1.00	51244	0.51	44.2	33
AA40	6	6	6	1.00	61,759	61,609.44	1.00	61432	0.53	44.67	41.4
AA45	7	6	6	1.00	62,029	61,693.01	1.01	61542	0.79	52.57	49.8
AA50	8	6	7	0.86	62,213	71,932.03	0.86	61687	0.85	54.25	58.8
AA55	8	8	8	1.00	82,405	82,185.31	1.00	81765	0.78	95.63	64.2
AA60	9	8	9	0.89	82,629	92,366.70	0.89	81936	0.84	94.44	76.8
AA65	10	7	8	0.88	72,783	82,331.12	0.88	72245	0.74	98.3	87.6
AA70	10	8	11	0.73	82,950	112,458.28	0.74	82347	0.73	143.7	98.4
AA75	11	8	9	0.89	83,044	92,529.42	0.90	82545	0.60	141.82	112.8
Average of AA	7.9	6.6	7.4	0.90	68263.4	75976.624	0.91	67803.1	0.64	79.518	64.98
BB30	5	4	5	0.80	41,267	51,193.62	0.81	41065	0.49	28.4	28.2
BB35	5	5	6	0.83	51,580	61,400.07	0.84	51206	0.73	38.2	32.4
BB40	6	5	5	1.00	51,785	51,421.35	1.01	51142	1.24	60.83	44.4
BB45	7	6	6	1.00	61,950	61,787.28	1.00	61632	0.51	71.86	49.2
BB50	8	7	7	1.00	72,164	71,889.75	1.00	71593	0.79	89.5	58.8
BB55	8	8	8	1.00	82,483	82,080.73	1.00	81889	0.72	122.75	64.2
BB60	9	11	10	1.10	112,988	102,323.77	1.10	101998	9.73	122.22	73.8
BB65	10	10	8	1.25	103,211	82,623.98	1.25	82454	20.11	122.9	85.2
BB70	10	11	9	1.22	113,534	92,647.75	1.23	92465	18.56	160.4	100.8
BB75	11	11	9	1.22	113,558	92,476.30	1.23	91897	19.07	154.18	112.8
Average of BB	7.9	7.8	7.3	1.04	80452	74984.46	1.05	72734.1	7.20	97.124	64.98
CC30	5	5	5	1.00	51,358	51,145.18	1.00	50987	0.72	44.2	28.2
CC35	5	5	5	1.00	51,578	51,235.64	1.01	51147	0.84	51.8	34.2
CC40	6	5	6	0.83	51,695	61,473.91	0.84	51213	0.93	49.5	43.2
CC45	7	5	8	0.63	51,955	81,408.89	0.64	51473	0.93	53.57	49.8
CC50	8	7	6	1.17	72,154	61,936.27	1.16	61759	14.41	51.38	63.6
CC55	8	10	6	1.67	102,460	61,930.55	1.65	61770	39.71	90.88	71.4
CC60	9	8	7	1.14	82,546	72,104.00	1.14	71876	12.93	84.89	82.8
CC65	10	9	8	1.13	92,803	82,326.62	1.13	81896	11.75	102.9	90
CC70	10	9	9	1.00	92,963	92,613.68	1.00	92229	0.79	149.3	102
CC75	11	9	9	1.00	93,220	92,711.74	1.01	92231	1.06	149	112.8
Average of CC	7.9	7.2	6.9	1.06	74273.2	70888.648	1.06	66658.1	8.41	82.742	67.8
DD30	5	4	6	0.67	41,426	61,040.10	0.68	41137	0.70	41.4	27.6
DD35	5	5	7	0.71	51,614	71,308.04	0.72	51545	0.13	68.2	33.6
DD40	6	5	6	0.83	51,851	61,531.68	0.84	51290	1.08	68	43.2
DD45	7	5	8	0.63	51,960	81,601.63	0.64	51238	1.39	72.86	48
DD50	8	6	7	0.86	62,131	71,761.23	0.87	62065	0.11	70.25	60
DD55	8	6	7	0.86	62,358	72,051.95	0.87	61789	0.91	121.38	69
DD60	9	7	8	0.88	72,521	82,308.08	0.88	72120	0.55	113.78	78.6
DD65	10	7	8	0.88	72,825	82,200.77	0.89	72385	0.60	120.6	90
DD70	10	8	8	1.00	83,034	82,631.56	1.00	82496	0.65	173.6	102
DD75	11	9	9	1.00	93,255	92,970.84	1.00	92748	0.54	165.45	109.8
Average of DD	7.9	6.2	7.4	0.83	64297.5	75940.588	0.84	63881.3	0.67	101.552	66.18
XX30	5	4	-	-	41,093	-	-	40923	0.41	101.4	-
XX35	5	5	-	-	51,313	-	-	51075	0.46	174.6	-
XX40	6	5	-	-	51,540	-	-	51120	0.81	166.33	-
XX45	7	7	-	-	71,719	-	-	61156	14.73	156.29	-
XX50	8	6	-	-	61,707	-	-	61438	0.44	126.88	-
XX55	8	6	-	-	61,839	-	-	61486	0.57	254.25	-
XX60	9	6	-	-	62,033	-	-	61945	0.14	299.44	-
XX65	10	6	-	-	62,531	-	-	62344	0.30	286.4	-
XX70	10	7	-	-	72,775	-	-	72251	0.72	400.1	-
XX75	11	8	-	-	82,960	-	-	82645	0.38	388.73	-
Average of XX	7.9	6	-	-	61951	-	-	60638.3	1.90	235.442	-
YY30	5	4	-	-	41,195	-	-	41023	0.42	76.2	-
YY35	5	5	-	-	51,363	-	-	51045	0.62	187.6	-
YY40	6	6	-	-	61,608	-	-	61242	0.59	161.33	-
YY45	7	6	-	-	61,806	-	-	61502	0.49	176.14	-
YY50	8	6	-	-	61,966	-	-	61706	0.42	203.88	-
YY55	8	7	-	-	72,121	-	-	61840	14.26	274.13	-
YY60	9	6	-	-	62,321	-	-	61972	0.56	276.56	-
YY65	10	7	-	-	72,464	-	-	72132	0.46	327.3	-
YY70	10	7	-	-	72,586	-	-	72211	0.52	419.6	-
YY75	11	9	-	-	92,679	-	-	82325	11.17	397.64	-
Average of YY	7.9	6.3	-	-	65010.9	-	-	62699.8	2.95	250.038	-

Figure 11 illustrates the position of parcels' origin and destination and the routes of vehicles in data set AA30.

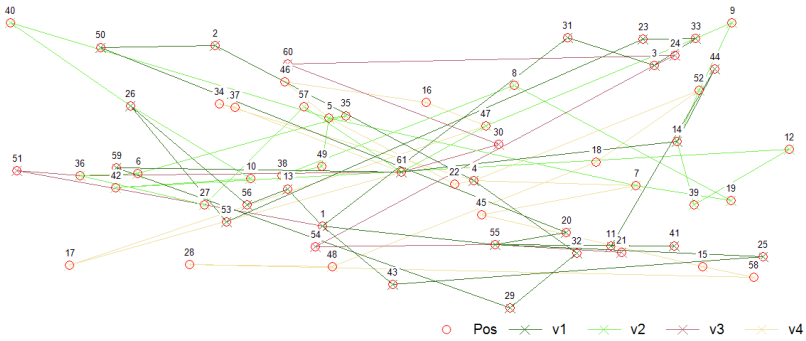


Fig. 11. Position of parcels' origin and destination and route of vehicles in data set AA30.

4.2. Large-scale network and test data set from Cainiao's Last Mile Rush competition

We also tested our algorithm on the data set proposed by Cainiao Network, called Last_Mile_Rush, to address the last mile delivery problem. Cainiao Network is the logistics company launched by Chinese e-commerce giant Alibaba Group in 2013. In this data set, there are two types of parcels: e-commerce parcels and intra-city online-to-offline (O2O) parcels. In the case of e-commerce parcels, couriers pick up parcels from local branches of express companies and deliver them to individuals. In the case of intra-city O2O parcels, couriers pick up parcels from O2O shops and deliver them to the customers, each of whom has particular pickup and delivery time windows. Several assumptions are considered in this data set:

- One express company with 124 local branches serves all e-commerce delivery requests in Shanghai. We note that there is no overlapping between the service ranges of any two local branches.
- All e-commerce parcels arrive at local branches before 8:00am and must be delivered by 8:00pm.
- All couriers start working at 8:00am. They are allowed to deliver both types of parcels at the same time.
- Type "A" demand: There are 9,214 points of interest (POIs), each of which has a particular latitude and longitude. Every POI is representative of the location of an order/customer, which can be either an e-commerce or intra-city O2O order and may contain one or more than one parcel.
- Each POI is served by only one local branch, since there is no overlapping between the service ranges of any two local branches.
- No courier is allowed to carry more than 140 parcels at a time.
- If a customer requests multiple e-commerce parcels, all should be delivered to the corresponding POI at one time.
- Type "B" demand: There are 598 O2O shops in Shanghai. Similar to e-commerce parcels, if a customer requests multiple O2O parcels, all should be delivered to the corresponding POI at one time.
- Each O2O order has particular pickup and delivery time windows. Couriers must pick up O2O orders from O2O shops at the designated pickup time and deliver them to the O2O customers no later than the designated delivery time.
- If a courier arrives at a delivery point of an O2O order earlier than its designated delivery time, it does not wait, and it is actually allowed to leave the parcel(s) there.
- The data set contains the latitude (lat) and longitude (lng) of all local branches, POIs, and O2O shops. The distance between any two locations can be calculated by equation (24).

$$distance = 2R.arcsin(\sqrt{sin^2\left(\frac{\pi}{180}\Delta lat\right) + \cos\left(\frac{\pi}{180}lat_A\right).\cos\left(\frac{\pi}{180}lat_B\right).sin^2(\frac{\pi}{180}\Delta lng)}, \quad (24)$$

where (lat_A, lng_A) and (lat_B, lng_B) are the latitude and longitude of nodes A and B, respectively; $\Delta lat = \frac{lat_A - lat_B}{2}$; $\Delta lng = \frac{lng_A - lng_B}{2}$; and $R = 6{,}378{,}137\ m$.

- The traveling speed of all couriers is assumed to be 15 km per hour. There are 1,000 available couriers in this data set. Each courier may remain idle or serve in one or more than one local branch.
- All couriers start their work shift exactly at 8:00am from local branches and end it at the last POI.
- The process time of a POI's request (in minutes), T, is also calculated by equation (25).

$$T = 3\sqrt{n} + 5, \quad (25)$$

where n is the number of parcels in the POI's order.

This data set contains several attributes related to the location of local branches, POIs, and O2O shops, as well as information related to the e-commerce and intra-city O2O orders. The total number of e-commerce orders (type 'A')

and O2O orders (type 'B') are 9,214 and 598, respectively, while at most 1,000 vehicles are available for service. There is always a trade-off between minimizing the total cost (total travel time) and maximizing the total number of performed orders (both types A and B). In order to find a balance between these two objectives, we offer an evaluation function containing two terms: (1) total travel time (TT) with the weight of 1, and (2) total number of performed orders with the weight of ρ, $\rho \leq 0$. In addition, in order to guide the search, we assign different weights/priorities for orders being picked up and delivered successfully, as well as orders that have been only picked up but not delivered. Therefore, we define the evaluation function as follows:

$$Min\ \{TT + \rho\left(\mu_A^{P\&D} n_A^{P\&D} + \mu_B^{P\&D} n_B^{P\&D} + \mu_A^{P} n_A^{P} + \mu_B^{P} n_B^{P}\right)\}, \tag{26}$$

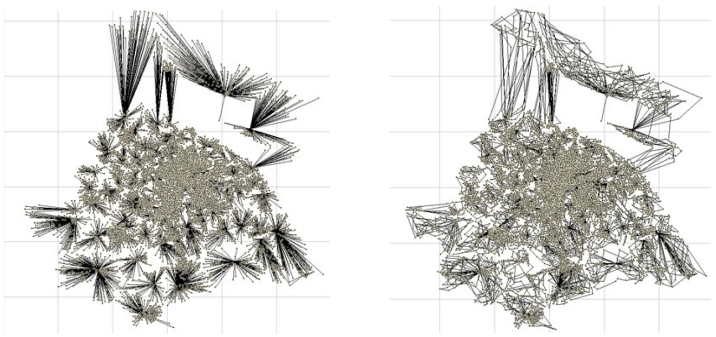
where TT is the total travel time (total travel cost),
ρ is the weight of total number of performed orders (complete pickup and delivery or just pickup),
$\mu_A^{P\&D}$ is the weight/priority of type "A" orders that have been picked up and delivered successfully,
$n_A^{P\&D}$ is the total number of type "A" orders that have been picked up and delivered successfully,
$\mu_B^{P\&D}$ is the weight/priority of type "B" orders that have been picked up and delivered successfully,
$n_B^{P\&D}$ is the total number of type "B" orders that have been picked up and delivered successfully,
μ_A^{P} is the weight/priority of type "A" orders that have been only picked up but not delivered,
n_A^{P} is the total number of type "A" orders that have been picked up but not delivered,
μ_B^{P} is the weight/priority of type "B" orders that have been only picked up but not delivered, and
n_B^{P} is the total number of type "B" orders that have been picked up but not delivered.

Evaluation function (26) is utilized as a selection criterion for the set of states. We note that the results are very sensitive to parameters $\mu_A^{P\&D}$, $\mu_B^{P\&D}$, μ_A^{P}, and μ_B^{P}, whose values are varied by the distribution of OD demands and the characteristics of types "A" and "B" orders. Therefore, the value of these parameters must be determined by a large number of experiments for each group of orders. We have selected group 31 (out of all 111 groups of orders) to explain how we have adjusted these parameters for a group of orders. Our attempts for tuning the parameters for group 31 are presented in Appendix B. We note that we performed the same procedure for all 111 groups of orders. We also have used the Fibonacci sequence (max number of states) as the width of the beam search algorithm to reduce the search region.

All local branches, POIs, O2O shops, and OD demands in the Cainiao Network have been shown in Figure 12(a). As shown in Table 3, we have used five different parameters for the beam width to implement our beam search algorithm. According to Table 3, it can be concluded that case IV provides the best solution among other cases, since in this case all orders (both types "A" and "B") have been performed by 893 vehicles in 3,764.51 seconds. Comparing the results of cases I and II to case IV, types "A" and "B" orders are incomplete (IC) in the prior two cases. Comparing the results of case III to case IV, the total cost and total number of vehicles needed in this case is much larger than those in case IV. Finally, comparing the results from case V to IV, the RAM usage and CPU time taken for case V are roughly twice as those used for case IV, while the total number of vehicles needed and total cost for both cases are almost the same. Therefore, we have selected case IV and shown the delivery routes of all couriers in Figure 12(b). We note that the objective in (26) is a weighted combination of minimizing total travel time and maximizing total number of served orders. Therefore, an unserved order in the final solution does not cause infeasibility.

Table 3
Results obtained from our algorithm on the Cainiao network.

Case No.	Beam width	RAM usage (GB)	CPU time (sec)	Total cost (min)	Total # of complete tasks (A+B)	# of vehicles
I	6,765	19.84	431.78	459337	6908+433 (IC)	755
II	10,946	48.09	888.68	389440	8936+497 (IC)	876
III	17,711	84.93	1829.06	364270	9214+598	954
IV	**28,657**	**140.96**	**3764.51**	**297200**	**9214+598**	**893**
V	46,368	220.23	7748.42	296934	9214+598	892



(a) (b)

Fig. 12. (a) Cainiao network with all local branches, POIs, O2O Shops, and OD demands; (b) Delivery routes of all Couriers for Case IV in Table 3.

Due to the size of this data set, we take some additional steps to reduce the computational time. For this purpose, we reduce the total number of hypothetical vehicles in each cluster. As discussed in Section 3.4, we assume that the total number of hypothetical vehicles in a cluster is equal to the total number of OD pairs in that cluster. After running our algorithm on this data set, we noticed that in most cases, only about one third of the vehicles in each cluster transport parcels, while others run empty. We note that this is just an observation and cannot be generalized for other data sets. Thus, we reduce the total number of vehicles in each cluster by two third, run the algorithm and obtain the results. We print out the cumulative arrival and on-board count diagrams to assess the waiting time of OD pairs along the time horizon manually. If there is a period of time with atypical waiting time, we identify the set of clusters temporally located in that period of time to check the service performance of OD pairs inside these clusters. If an OD pair in the cluster derived from the identified set of clusters was not served at all, it can be either due to the lack of vehicle availability in its cluster or due to the low value of its $\bar{c}_{j,n}^{m}$. Hence, we check the routing of vehicles in the cluster to see if there is a vehicle running empty in that cluster. If so, it indicates that we have already had enough vehicles available at the depot, otherwise we increase the total number of hypothetical vehicles in the cluster by one and run the algorithm again to see if any improvement is observed. If so, we modify the total number of vehicles in the cluster to the new number, otherwise we move to the next cluster from the set of identified clusters.

5. Conclusions

In this research, by extending the work pioneered by Bellman (1962), Held and Karp (1962) and Psaraftis (1980) on using the dynamic programing method to solve the vehicle routing problem, we embed many complex constraints of the pickup and delivery problem with transfers on a three-dimensional space-time-state network. In this network construction process, elements of time and load are explicitly added as new dimensions to the physical transportation network. To address the curse of dimensionality, we demonstrate a consistent transition from the microscopic cumulative service states to macroscopic cumulative flow count diagrams, which can be used to effectively estimate the overall dynamic system performance. We also split the large-scale primary problem into a number of small-scale sub-problems. We use a dynamic programming algorithm to solve a least-cost path problem for the local clusters derived from the original set of parcels. At the end, extensive computational results over the data set used by Ropke and Pisinger (2006) and real-world data set proposed by Cainiao Network were performed to examine the effectiveness of our developed algorithm.

Future work may concentrate on building a computational engine to establish a wrapper for the dynamic programming algorithm with the inputs of a transportation network and possible state transition matrices, and the output of various vehicle-path assignment and routing solutions. Another interesting extension of our space-time-state framework is to build a more practical and robust model in the context of pickup and delivery of passengers including some levels of travelers'/service providers' behavior.

Acknowledgments

This paper is mainly supported by the National Science Foundation – United States under Grant No. CMMI 1538105 "Collaborative Research: Improving Spatial Observability of Dynamic Traffic Systems through Active Mobile Sensor Networks and Crowdsourced Data." The second author is partially supported by National High Technology Research and Development Program of China (No.2015AA016404). The work presented in this paper remains the sole responsibility of the authors.

References

https://www.azcentral.com/story/news/local/phoenix/2018/07/31/valley-metro-experiment-waymo-autonomous-vehicles/853719002/

Baldacci, R., Bartolini, E., Mingozzi, A. (2011) An exact algorithm for the pickup and delivery problem with time windows. *Operations Research*, 59(2): 414-426.

Ball, G. and Hall, D. (1965) ISODATA, a novel method of data analysis and pattern classification, *Technical report*, NTIS AD 699616. Stanford Research Institute, Stanford, CA.

Bard, J.F. and Jarrah, A.I. (2009) Large-scale constrained clustering for rationalizing pickup and delivery operations, *Transportation Research Part B: Methodological*, 43(5), 542-561.

Bellman, R. (1962) Dynamic programming treatment of the traveling salesman problem. *Journal of the Association for Computing Machinery*, 9: 61-63.

Bodin, L.D. and Sexton, T. (1986) The multi-vehicle subscriber dial-a-ride problem. *TIMS Studies in Management Science*, 26, 73-86.

Boland, N., Hewitt, M., Marshall, L., Savelsbergh, M.W.P. (2017) The continuous time service network design problem. *Operations Research*, article in advance, https://doi.org/10.1287/opre.2017.1624.

Bunte, S., Kliewer, N., 2009. An overview on vehicle scheduling models. *Public Transport 1*, 299–317.

Chabini, I. (1998) Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Record: Journal of the Transportation Research Board*, 1645: 170-175.

Cordeau, J-F. (2006) A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3): 573-586.

Cordeau, J.F. and Laporte, G. (2003) A tabu search heuristic for the static multi-vehicle dial-a-ride problem, *Transportation Research Part B: Methodological*, 37, 579-594.

Cortés, C.E., Matamala, M., Contardo, C. (2010) The pickup and delivery problem with transfers: formulation and a branch-and-cut solution method. *European Journal of Operational Research*, 200, 711–724.

Coslovich, L., Pesenti, R., Ukovich, W. (2006) A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem. *European Journal of Operational Research*, 175, 1605–1615.

Daganzo, C. F. (1994) The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4), 269-287.

Daganzo, C.F. (2001) A theory of supply chains. 526 lecture notes in economics and mathematical systems, *Springer.*

Dellaert, N., Saridarq, F., Van Woensel, T., Crainic, T. (2016) Branch & price based algorithms for the two-echelon vehicle routing problem with time windows. Working paper, CIRRELT.

Desrosiers, J., Dumas, Y., Soumis, F. (1986) A dynamic programming solution of the large-scale single-vehicle dial-a-ride problem with time windows. *American Journal of Mathematical and Management Sciences*, 6: 301-325.

Desrosiers, J., Dumas, Y., Soumis, F., Taillefer, S., Villeneuve, D. (1991) An algorithm for mini-clustering in handicapped transport. *Les Cahiers du GERAD*, G-91-02, HEC Montréal.

Diana, M. and Dessouky, M.M. (2004) A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. *Transportation Research Part B: Methodological*, 38, 539–557.

Drexl, M. (2012a) Synchronization in vehicle routing – A survey of VRPs with multiple synchronization constraints. *Transportation Science*, 297–316.

Drexl, M. (2012b) Rich vehicle routing in theory and practice. *Logistics Research*, 5, 47–63.

Drexl, M. (2013) Applications of the vehicle routing problem with trailers and transshipments. *European Journal of Operational Research*, 227, 275–283.

Dumas, Y., Desrosiers, J., Soumis, F. (1989) Large scale multi-vehicle dial-a-ride problems. *Les Cahiers du GERAD*, G-89-30, HEC Montréal.

Dumas, Y., Desrosiers, J., Soumis, F. (1991) The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54: 7–22.

Edie, L.C., Foote, R.S. (1960) Effect of shock waves on tunnel traffic flow. *Proceedings of the Highway Research Board,* 39, 492-505.

Gazis D. C., Potts R.B. (1963) The oversaturated intersection. *Proceedings of the 2nd International Symposium on the Theory of Road Traffic Flow*, 221–237.

Ghilas, V., Demir, E., Van Woensel, T. (2016) An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows and scheduled lines. *Computers and Operations Research*, 72, 12-30.

Hall, R.W. (1991) Queueing Methods for Services and Manufacturing, Prentice Hall, Engelwood Cliffs, New Jersey.

Häme, L., Hakula, H. (2015) A maximum cluster algorithm for checking the feasibility of dial-d-ride instances. *Transportation Science*, 49(2): 295-310.

Held, M., Karp, R.M. (1962) A dynamic-programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1): 196-210.

Ioachim, I., Desrosiers, J., Dumas, Y., Solomon, M.M. (1995) A request clustering algorithm for door-to-door handicapped transportation. *Transportation Science*, 29, 63–78.

Kim, M. and Schonfeld, P. (2014) Integration of conventional and flexible bus services with timed transfers. *Transportation Research Part B*, 68, 76–97.

Kumar, D., Wu, H., Lu, Y., Krishnaswamy, S., Palaniswami, M. (2016) Understanding urban mobility via taxi trip clustering. *IEEE International Conference on Mobile Data Management*, 318-324.

Large-scale network and test data set from Cainiao's Last Mile Rush competition, Retrieved on August 18, 2017 from https://tianchi.shuju.aliyun.com/competition/information.htm?spm=5176.100067.5678.2.LRPZpR&raceId=231581.

Li, B., Krushinsky, D., Reijers, H.A., Woensel, T.V. (2014) The share-a-ride problem: people and parcels sharing taxis. *European Journal of Operational Research*, 238 (1), 31–40.

Lloyd, S. (1982) Least squares quantization in PCM, *IEEE Transactions on Information Theory*, 28, 129–137.

Lu, Q. and Dessouky, M. (2004) An exact algorithm for the multiple vehicle pickup and delivery problem. *Transportation Science*, 38(4): 503–514.

Makigami, Y., Newell, G.F., Rother, R. (1971) Three-dimensional representations of traffic flow. *Transportation Science,* 5, 302-313.

Mahmoudi, M, Song, Y., Miller, H.M., Zhou, X. (2019) Accessibility with time and resource constraints: Computing hyper-prisms for sustainable transportation planning. *Computers, Environment and Urban Systems*, 73: 171-183.

Mahmoudi, M. and Zhou, X. (2016) Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state-space-time network representations. *Transportation Research Part B: Methodological*, 89: 19-42.

Masson, R., Lehuédé, F., Péton, O. (2013) An adaptive large neighborhood search for the pickup and delivery problem with transfers. *Transportation Science*, 47(3): 344-355.

Masson, R., Lehuédé, F., Péton, O. (2014) The dial-a-ride problem with transfers. *Computers and Operations Research*, 41 12–23.

Melachrinoudis, E., Ilhan, A. B., &Min, H. (2007) A dial-a-ride problem for client transportation in a healthcare organization. *Computers & Operations Research*, 34, 742–759.

Miller-Hooks, E.D., Mahmassani, H.S. (1998) Least possible time paths in stochastic, time-varying networks. *Computers and Operations Research*, 25, 1107–1125.

Miller-Hooks, E.D., Mahmassani, H.S. (2000) Least expected time paths in stochastic, time-varying transportation networks. *Transportation Science*, 34 (2), 198–215.

Mitrovic´-Minic´, S., Laporte, G. (2006) The pickup and delivery problem with time windows and transshipment. *INFOR: Information Systems and Operational Research*. 44, 217–227.

Mues, C. and Pickl, S. (2005) Transshipment and time windows in vehicle routing. In *Proceedings of the 8th international symposium on parallel architectures. Algorithms and networks* (pp. 113–119). Piscataway, NJ: IEEE.

Newell, G.F. (1982). *Applications of Queueing Theory,* Chapman and Hall, London.

Newell, G.F. (1993) A simplified theory of kinematic waves in highway traffic, part I: General theory. *Transportation Research Part B*, 27(4), 281–287.

Pankratz, G. (2005). A grouping genetic algorithm for the pickup and delivery problem with time windows, *OR Spectrum*, 27, 21-41.

Psaraftis, H.N. (1980) A dynamic programming approach to the single-vehicle, many-to-many immediate request dial-a-ride problem. *Transportation Science*, 14(2): 130–154.

Psaraftis, H.N. (1983) An exact algorithm for the single-vehicle many-to-many dial-a-ride problem with time windows. *Transportation Science*, 17(3): 351-357.

Rais, A., Alvelos, F., Carvalho, M.S. (2014) New mixed integer programming model for the pickup-and-delivery problem with transshipment. *European Journal of Operational Research*, 235(3):530–539.

Ropke, S., Cordeau J-F., Laporte G. (2007) Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks*, 49(4): 258–272.

Ropke, S. and Cordeau, J-F. (2009) Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science*, 43(3): 267–286.

Ropke, S. and Pisinger, D. (2006) An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4) 455–472.

Qu, Y. and Bard J.F. (2012) A GRASP with adaptive large neighborhood search for pickup and delivery problems with transshipment. *Computers and Operations Research*, 39(10): 2439–2456.

Savelsbergh, M.W.P and Sol, M. (1998) Drive: Dynamic routing of independent vehicles. *Operations Research*, 46(4): 474-490.

Sharypova, K., Crainic, T.G., van Woensel, T., Fransoo, J. C. (2012) Scheduled Service Network Design with Synchronization and Transshipment Constraints for Intermodal Container Transportation Networks. Publication CIRRELT-2012-77, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal.

Sun, Y. and Schonfeld, P. (2016) Holding decisions for correlated vehicle arrivals at intermodal freight transfer terminals. *Transportation Research Part B*, 90, 218–240.

Wan, X., Gao, M., Kang, J., Zhao, J. (2013) Taxi origin-destination areas of interest discovering based on functional region division. *International Conference on Innovative Computing Technology (INTECH)*, 365–370.

Yang, H. and Meng, Q. (1998) Departure time, route choice and congestion toll in a queuing network with elastic demand. *Transportation Research Part B: Methodological*, 32(4), 247-260.

Zawack, D.J. and Thompson, G. L. (1987) A dynamic space-time network flow model for city traffic congestion, *Transportation Science*, 21(3), 153-161.

Zhu X. and Guo, D. (2014) Mapping large spatial flow data with hierarchical clustering. *Transactions in GIS*, 18(3): 421–435.

Ziliaskopoulos, A.K. and Mahmassani, H.S. (1993) Time-dependent, shortest-path algorithm for real-time intelligent vehicle highway system application. *Transportation Research Record: Journal of the Transportation Research Board*, 1408: 94-100.

Appendix A Notations

Table A1

Notations used for the sets, indices, parameters, and decision variables in this paper

Indices and parameters used throughout the paper	
j, j'	Indices for parcels
n, n'	Indices for the ordinal number of a way for a parcel
m, m'	Indices for the ordinal number of an OD pair in a way for a parcel
$OD_{j,n}^{m}$	The m^{th} OD pair in n^{th} way by which parcel j is served
$o_{j,n}^{m}$	The dummy node corresponding to parcel j's origin in the m^{th} OD pair of n^{th} way
$d_{j,n}^{m}$	The dummy node corresponding to parcel j's destination in the m^{th} OD pair of n^{th} way
$\underline{a}_j$	The earliest departure time from parcel j's main origin
$\underline{b}_j$	The latest departure time from parcel j's main origin
$\overline{a}_j$	The earliest arrival time to parcel j's main destination
$\overline{b}_j$	The latest arrival time to parcel j's main destination
$[\,\underline{a}_j, \underline{b}_j\,]$	Parcel j's departure time window
$[\,\overline{a}_j, \overline{b}_j]$	Parcel j's arrival time window
v, v'	Indices for vehicles
o_v	The dummy node corresponding to the origin depot for vehicle v
d_v	The dummy node corresponding to the destination depot for vehicle v
$t_{v,start}$	The time stamp at which vehicle v's time horizon begins
$t_{v,end}$	The time stamp at which vehicle v's time horizon ends
$[t_{v,start}, t_{v,end}]$	Vehicle v's time horizon
s, s'	Indices for states which are vectors whose elements represent the cumulative service state of OD pairs
i, i'	Indices for nodes
t, t'	Indices for time stamps
(i, t, s, v)	The vertex in vehicle v's network
$(i, i', t, t', s, s', v, v')$	The arc connecting vertex (i, t, s, v) to (i', t', s', v')
$TT_{i,i',t}$	The travel time/service time/preparation time from node i to i' starting at time t
Indices, parameters, and decision variables exclusively used in Section 3.3	
$t_{o_{j,n}^{m}}$	The middle time of $o_{j,n}^{m}$'s departure time window
$t_{d_{j,n}^{m}}$	The middle time of $d_{j,n}^{m}$'s arrival time window
β_1	The weight of geographical distance (per mile) in the space-time distance calculation
β_2	The weight of temporal distance (per minute) in the space-time distance calculation
$f_{\circ,*}$	The space-time distance between $\circ$ and $*$
$r_{\circ,*}$	The highest space-time dissimilarity between $\circ$ and $*$
q, q'	Indices for clusters
α	Maximum number of OD pairs in a cluster
y_q	$y_q = 1$ if cluster q exists; $y_q = 0$ otherwise
$z_{OD_{j,n}^{m},q}$	$z_{OD_{j,n}^{m},q} = 1$ if $OD_{j,n}^{m}$ is assigned to cluster q; $z_{OD_{j,n}^{m},q} = 0$ otherwise
ζ_1	The weight of the first term of the objective function in the clustering algorithm
ζ_2	The weight of the second term of the objective function in the clustering algorithm
Sets, indices, parameters, and decision variables exclusively used in Section 3.4	
$c_{i,i',t,t',s,s',v}$	The routing cost of arc $(i, i', t, t', s, s', v, v)$
$\overline{c}_{j,n}^{m}$	The service cost of m^{th} OD pair in n^{th} way for parcel j
$TC_{j,n}^{m}$	The total transportation cost of a vehicle when it leaves its origin depot to exclusively serve $OD_{j,n}^{m}$ and return to its destination depot
$\overline{x}_{o,q}$	The average x-coordinate of all OD pairs' origin in cluster q
$\overline{y}_{o,q}$	The average y-coordinate of all OD pairs' origin in cluster q
o_q	The dummy node corresponding to the origin depot of all hypothetical vehicles in cluster q
$\overline{x}_{d,q}$	The average x-coordinate of all OD pairs' destination in cluster q
$\overline{y}_{d,q}$	The average y-coordinate of all OD pairs' destination in cluster q
d_q	The dummy node corresponding to the destination depot of all hypothetical vehicles in cluster q
$\underline{t}_q$	The time at which hypothetical vehicles' time horizon in cluster q starts
$\overline{t}_q$	The time at which hypothetical vehicles' time horizon in cluster q ends

$x_{i,i',t,t',s,s',v,v'}$	$x_{i,i',t,t',s,s',v,v'} = 1$ if arc (i,i',t,t',s,s',v,v') is selected; $x_{i,i',t,t',s,s',v,v'} = 0$ otherwise
V_q	The set of hypothetical vehicles in cluster q, where $V_q = \{v_1, v_2, \dots, v_{\lvert V_q \rvert}\}$
ϕ	The null state at which the service of no OD pair has started yet
Sets, indices, parameters, and decision variables exclusively used in Section 3.5	
E	The set of potential origin depots for real vehicles
E'	The set of potential destination depots for real vehicles
$\mathcal{W}$	The set of work pieces generated in Section 3.4 and represents a set of hypothetical vehicles' activities, each consisting of a sequence of pickup, travel and drop-off operations occurring at specified times
$\mathcal{A}$	The set of arcs for the algorithm presented in Section 3.5
e	The index of a node in set E
e'	The index of a node in set E'
ω	The index of a node in set $\mathcal{W}$
ω'	The index of a node in set $\mathcal{W}'$
$\delta^+(\omega)$	The set of arcs in $\mathcal{A}$ leaving $\omega \in \mathcal{W}$
$\delta^-(\omega)$	The set of arcs in $\mathcal{A}$ entering $\omega \in \mathcal{W}$
c_a	The cost of arc $a \in \mathcal{A}$ in the algorithm presented in Section 3.5
y_a	$y_a = 1$ if arc $a \in \mathcal{A}$ appears in a path for a real vehicle; $y_a = 0$ otherwise
Sets, indices, and parameters exclusively used in Section 3.6	
$O_{j,n}$	The set of origin nodes, i.e., the set of nodes of the form $o_{j,n}^m$, for each parcel j and each n indexing a way for that parcel
S	The set where $S \subseteq \bigcup_n O_{j,n}\ s.t. \lvert S \cap O_{j,n} \rvert = 1$
k	The iteration number
$\theta_{k,j,n,S}$	The step size corresponding to cut (j,n,S) at iteration k
$\lambda_{k,j,n,S}$	The Lagrangian multiplier corresponding to cut (j,n,S) at iteration k
$\nabla L_{\lambda_{k,j,n,S}}$	The sub-gradient of cut (j,n,S)
$\theta'_{k,j,n,m}$	The step size corresponding to constraint (17) at iteration k
$\lambda'_{k,j,n,m}$	The Lagrangian multiplier corresponding to constraint (17) at iteration k
$\nabla L'_{\lambda'_{k,j,n,m}}$	The sub-gradient corresponding to constraint (17) at iteration k
Sets, indices, and parameters exclusively used in Section 3.7	
$L_{i,t,s,v}$	The label of vertex (i,t,s,v)
L_q	The set of links in cluster q
T_q	The set of time stamps in cluster q
$C_A^p(t)$	Parcels' cumulative arrival count to the system at time t
$C_O^p(t)$	Parcels' cumulative on-board count at time t
$C_D^p(t)$	Parcels' cumulative departure count from the system at time t
$C_A^v(t)$	Hypothetical vehicles' cumulative arrival count to the system at time t
$C_D^v(t)$	Hypothetical vehicles' cumulative departure count from the system at time t

Appendix B Parameters tuning for Cainiao Network

We have taken our group 31 as an example of a group of orders to explain how we have adjusted these parameters for a group of orders by the aid of our evaluation function. In these experiments, we have defined 5 different scenarios as follows:

I. Set $\mu_A^{P\&D} = 2$, $\mu_B^{P\&D} = 1$ and define some sub-scenarios to adjust the values of μ_A^P and μ_B^P;
II. Set $\mu_A^{P\&D} = 2$, $\mu_B^{P\&D} = 0.5$ and define some sub-scenarios to adjust the values of μ_A^P and μ_B^P;
III. Set $\mu_A^{P\&D} = 2$, $\mu_B^{P\&D} = 1.5$ and define some sub-scenarios to adjust the values of μ_A^P and μ_B^P;
IV. Set $\mu_A^{P\&D} = 2$, $\mu_B^{P\&D} = 2$ and define some sub-scenarios to adjust the values of μ_A^P and μ_B^P;
V. Set $\mu_A^{P\&D} = 2$, $\mu_B^{P\&D} = 2.5$ and define some sub-scenarios to adjust the values of μ_A^P and μ_B^P.

We note that in Table B1, the demand completion time is the time that the last vehicle finishes its task. Figure B1 illustrates the values of the objective function for different scenario/sub-scenarios to help us choose the best parameter settings. As can be seen in this figure, scenario I, sub-scenario III has the smallest objective function among other scenarios. Therefore, we set the parameters of group 31 based on the parameters used in scenario I, sub-scenario III.

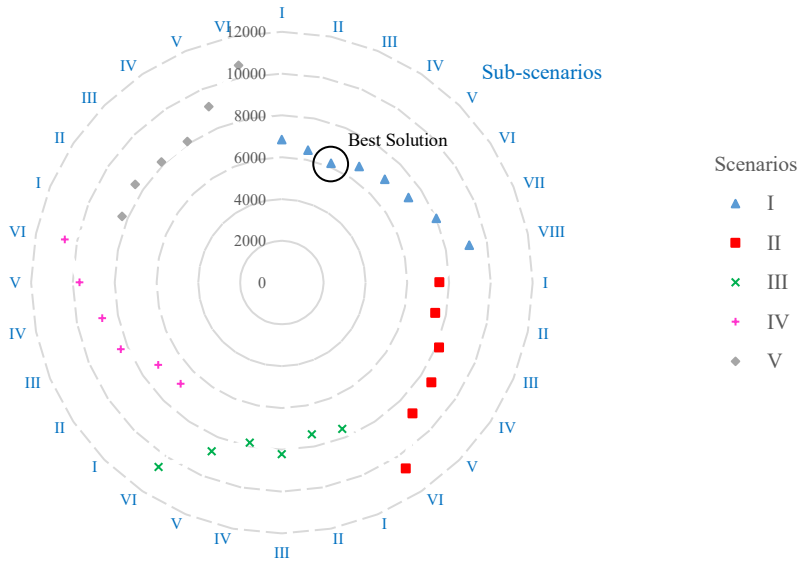


Fig. B1. A comparison between the values of the objective function for different scenarios-sub scenarios for group 31.

Table B1
Parameter setting for group 31.

Scenario No.	Sub-scenario No.	$\mu_A^{P\&D}$	$\mu_B^{P\&D}$	μ_A^P	μ_B^P	Performed tasks: Type "A" Picked up only	Type "A" Picked up & delivered	Type "B" Picked up only	Type "B" Picked up & delivered	Total travel time (min)	Demand completion time (min)	Total # of vehicles needed	Value of objective function
I	I	2	1	0.5	2	0	44	0	18	3769	671	8	6840
	II	2	1	1	2	0	44	0	18	3443	617	8	6460
	III	**2**	**1**	**1.5**	**2**	**0**	**44**	**0**	**18**	**3153**	**612**	**8**	**6165**
	IV	2	1	2	2	0	44	0	18	3586	685	8	6671
	V	2	1	2.5	2	0	42	3	15	2968	659	7	6977
	VI	2	1	3	2	3	41	5	12	2924	659	7	7303
	VII	2	1	3.5	2	5	39	6	12	3061	698	8	8009
	VIII	2	1	4	2	7	37	7	11	3505	571	9	9156
II	I	2	0.5	0.5	2	0	44	0	16	3769	775	8	7544
	II	2	0.5	1	2	0	44	0	16	3706	786	8	7492
	III	2	0.5	1.5	2	4	40	2	16	3797	896	8	8153
	IV	2	0.5	2	2	4	40	2	15	3896	964	8	8620
	V	2	0.5	2.5	2	2	42	6	11	3909	978	8	8867
	VI	2	0.5	3	2	6	38	10	8	4126	1236	9	10702
III	I	2	1.5	0.5	2	1	42	2	16	3669	617	8	7576
	II	2	1.5	1	2	1	43	2	16	3756	764	8	7410
	III	2	1.5	1.5	2	4	40	2	16	3797	964	8	8221
	IV	2	1.5	2	2	4	40	2	16	3443	906	8	7809
	V	2	1.5	2.5	2	6	38	2	16	3606	999	9	8745
	VI	2	1.5	3	2	9	35	4	14	4236	1369	9	10615
IV	I	2	2	0.5	2	1	43	0	18	3569	689	8	6848
	II	2	2	1	2	0	44	1	17	3797	764	8	7111
	III	2	2	1.5	2	4	40	2	16	3909	964	8	8333
	IV	2	2	2	2	4	40	4	14	3896	1096	8	8752
	V	2	2	2.5	2	6	38	6	12	3960	978	9	9678
	VI	2	2	3	2	9	35	6	12	4038	1236	9	10584
V	I	2	2.5	0.5	2	4	38	0	18	3614	697	8	8271
	II	2	2.5	1	2	4	38	0	18	3709	764	8	8433
	III	2	2.5	1.5	2	6	38	0	18	3666	938	8	8144
	IV	2	2.5	2	2	6	38	0	18	3705	869	8	8114
	V	2	2.5	2.5	2	6	38	2	16	3976	986	9	9102
	VI	2	2.5	3	2	9	35	4	14	4236	1345	9	10591

In Figure B2, we intend to show the cumulative arrival/departure and on-board diagrams for orders in case scenario I, sub-scenarios I to VIII. In this figure, the graphs indicated by black color are the cumulative arrival orders

to the system, the graphs indicated by red color are the cumulative on-board orders (pickup only), and the graphs indicated by blue color are the cumulative departure orders from the system. Figure B2(a) to B2(d) are the cumulative flow graphs for scenario I, sub-scenario I to IV, while B2(e) to B2(h) are the cumulative flow graphs for scenario I, sub-scenario V to VIII. As we can see in this figure, there is no incomplete task in sub-scenarios I to IV, while a number of orders have remained incomplete in sub-scenarios V to VIII.

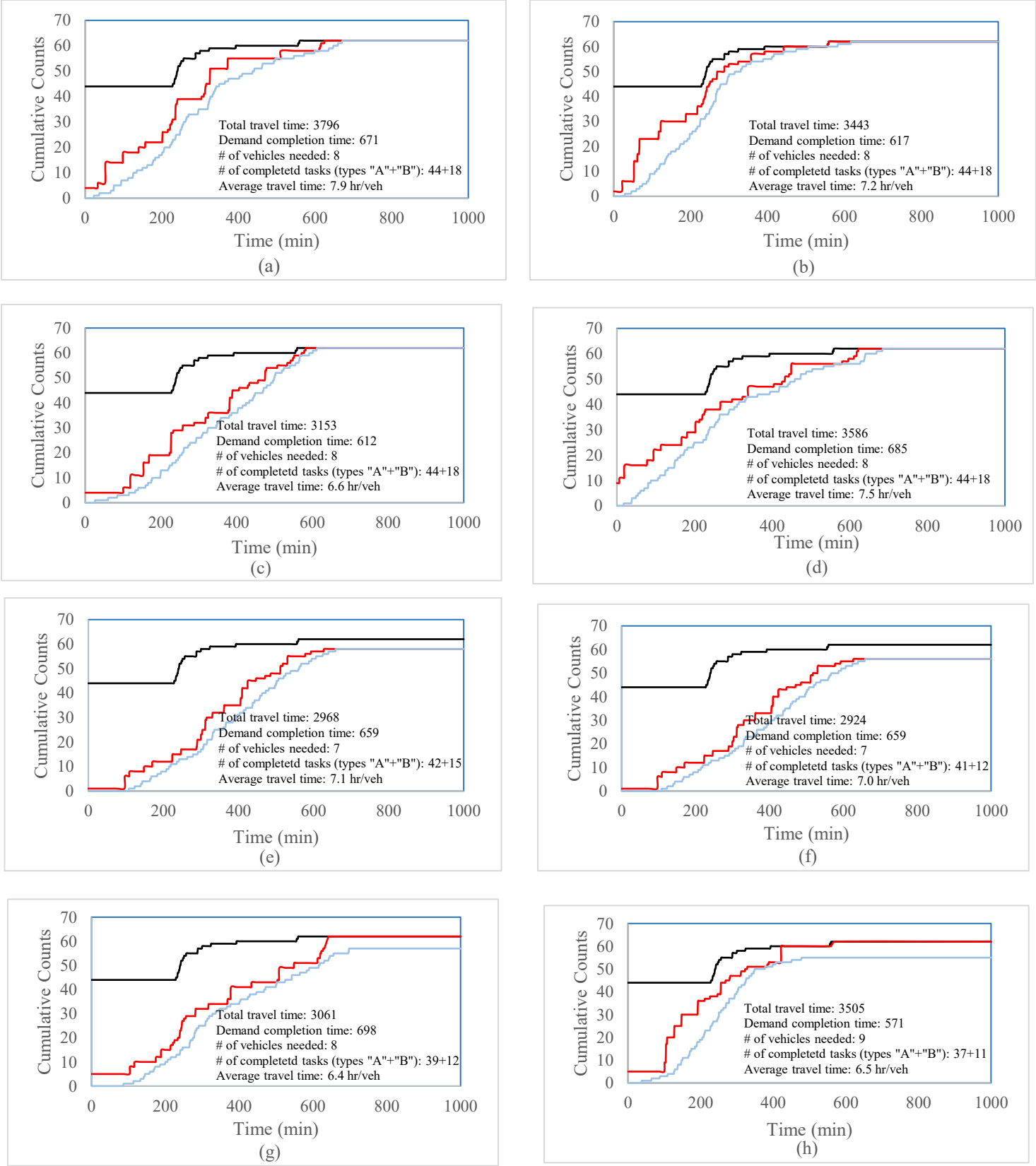


Fig. B2. Cumulative arrival/departure, as well as on-board diagrams for scenario I (a) sub-scenario I, (b) sub-scenario II, (c) sub-scenario III, (d) sub-scenario IV, (e) sub-scenario V, (f) sub-scenario VI, (g) sub-scenario VII, and (h) sub-scenario VIII.