Transportation Research Part E 150 (2021) 102324

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

# Solving the winner determination problem for online B2B transportation matching platforms☆

Hoong Chuin Lau *, Baoxiang Li

*Singapore Management University, School of Computing and Information Systems, 80 Stamford Road, 178902 Singapore, Singapore*

ARTICLE INFO

*Keywords:*
B2B Last Mile Logistics
Iterative auction
Online platform
Pickup-and-delivery problem
Winner determination problem

ABSTRACT

We consider the problem of matching multiple shippers and transporters participating in an online B2B last-mile logistics platform in an emerging market. Each shipper places a bid that is made up of multiple jobs, where each job comprises key information like the weight, volume, pickup and delivery locations, and time windows. Each transporter specifies its vehicle capacity, available time periods, and a cost structure. We formulate the mathematical model and provide a Branch-and-Cut approach to solve small-scale problem instances exactly and larger scale instances heuristically using an Adaptive Large Neighbourhood Search approach. To increase the win percentage of both shippers and transporters, we propose an extension of the single round auction to a second-round for failed bids. Using our models and the results obtained, we present interesting managerial insights that are helpful to platform participants. For example, due to the cost structure, there is little impact on profitability with time varying speeds, but time windows play a significant role in the profitability of the system. And there are tangible benefits that all platform participants (shippers, transporters and operator) can gain by extending single-round auction to two rounds.

## 1. Introduction

Rising standards of living and earning powers in emerging economies (such as first and second-tier cities in China and India) have contributed to consumption, which increases the strain on logistics, in particular last-mile logistics. Without proper coordination and collaboration, logistics activities (in particular B2B deliveries) within these cities will tend to be fragmented, as different logistics providers large or small seek to fulfill customer demands with their own resources.

Technology and the Sharing Economy provide new paradigms to meet the challenges of sustainable logistics. Increasingly, online platforms and apps have been developed to help transporters (especially small and medium-sized transporters) to find delivery jobs to increase their revenue, as well as shipper companies, to find new ways to reduce their delivery cost. Such platforms are rapidly emerging with the advent of "Uber"-like business models. According to Techinasia (2017) for example, about half of the citys 70,000 commercial vans in Hong Kong are registered with specific companies, however, the remaining 35,000 are driven by freelancers. Freelancers might form contracts with a number of small-to-medium enterprises, but easily face difficulties to receive orders from clients. On the other end, shippers face the challenge of finding appropriate transporters that will reduce their delivery costs. Hence,

☆ This paper is an expanded version of the conference paper "A Combinatorial Auction for Transportation Matching Service: Formulation and Adaptive Large Neighborhood Search Heuristic" that appeared in Li and Lau (2017)

* Corresponding author.
*E-mail addresses:* hclau@smu.edu.sg (H.C. Lau), bxli@smu.edu.sg (B. Li).

https://doi.org/10.1016/j.tre.2021.102324
Received 31 August 2020; Received in revised form 21 March 2021; Accepted 28 March 2021

matching demand and supply in an efficient way becomes crucial.

Recently, researchers and practitioners proposed the concept of collaboration to balance the supply and demand through a market platform. As illustrated in Fig. 1, the concept is to establish a platform which pools the shippers' delivery requirements (demand) and transporters' vehicle resources (supply) together. Transporters differ with regards to their cost structure, fleet availability and capacities, while shippers have different requirement in terms of their pricing, shipment volume, pickup/delivery time windows; and the platform will assign delivery requests to transporters as efficiently as possible, while respecting the regulatory restrictions. For such platforms to be feasible, they must be financially sustainable, which implies that it must be profit-maximizing while guaranteeing the level of service.

The use of auction enables the buying and selling of transport services with a market-based approach; in particular combinatorial auctions are generally seen as a highly effective mechanism to allocate transportation requests in practice (Sheffi, 2004). Some recent commercial examples are the likes of FreightEMart, CargoBid and ShipNext.

In this paper, we focus on the Winner Determination Problem that matches shippers with transporters participating in an online auction platform for B2B less-than-truckload pickup and delivery. The tripartite stakeholders include the multiple shippers that have bundles of pick up or delivery jobs; the transporters that operate a heterogeneous fleet of vehicles with varying availability, capacity and cost structures; and the platform operator (or auctioneer) whose role is to match jobs with vehicles that maximizes profits.

Note that it is not the goal of this paper to propose price bidding strategies for auctions; rather, both shipper bid prices and transporter cost structures are given as input to our problem. Furthermore, the shipper bid price associated with a job bundle could either be computed based on standard rates (that depends on the distance between pickup and drop off locations and package size), or provided by the shipper taking factors such as cargo type, cargo mix and economies of scale into consideration. Neither is the goal of this paper on profit sharing schemes among the stakeholders after the matching has been accomplished.

This work is a result of a deployed application with a large logistics market platform operator in Asia. Even though the underlying problem is a generalization of the Pickup-and-Delivery Problem with Time Windows, the focus is largely financial in nature with several major problem characteristics. First, we need to handle additional dependencies among jobs plus a number of side constraints. Second, the objective function is a profit function of revenue minus cost. Each shipper specifies a delivery request that comprises one or more pickup-and-delivery jobs, and these are bundled into what we call a "shipper bid", which must either be accepted or rejected together. Grouping delivery jobs into bids offer multiple benefits. Firstly, some jobs taken in isolation may have low volume and hence very difficult to find a matching transporter; if such jobs would be bundled with high volume jobs, then the resulting bid could find matching transporters with higher probability. Secondly, some jobs may themselves be naturally bundled based on their operational need (such as in reverse logistics when a pickup dovetails a delivery within the same premise). And thirdly, for convenience, shippers may prefer a one-stop solution rather than having to manage separate delivery requests.

This problem is further complicated by the requirement that grouped jobs in a shipper bid may themselves be split operationally, that is, different jobs within the same bid can be served by one or more transporters, so long as they are all fulfilled. For simplicity and without loss of generality in this paper, we assume that one transporter operates one single vehicle, hence the delivery assignment in our model is vehicle-based rather than transporter-based. Such an assumption can be easily relaxed by introducing one more index in the mathematical model representing the transporter for instance. Our goal is to maximize the resulting profit, which is calculated as

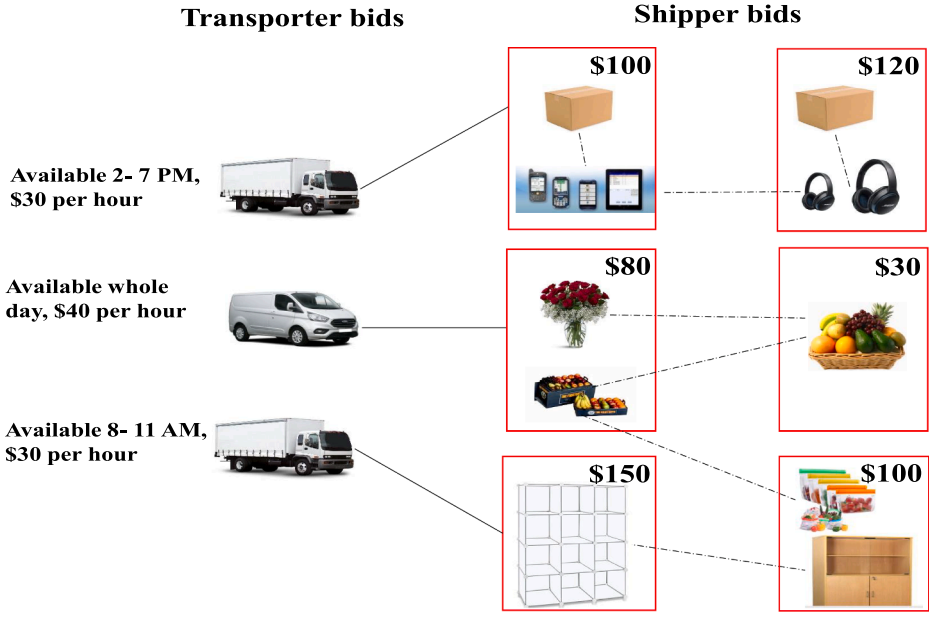



**Fig. 1.** Winner Determination Problem for Transportation Matching Platforms.

the total revenue of the served (or winning) bids minus the total transportation cost incurred to serve those bids.

Moreover, in the actual deployed version, the optimization algorithm needs to cope with a host of additional operational constraints relating to shipper-transporter compatibility, cargo-vehicle compatibility, vehicle-location restriction constraints, etc, which we have chosen to omit from our model in this paper for simplicity. They may be easily reincorporated as additional constraints in the model and the algorithm.

This paper is an expanded version of our earlier work Li and Lau (2017). The key contributions are as follows. First, we study the core Winner Determination Problem for an online matching platform, which we term "Group Pickup and Delivery Problem with Time Windows" (G-PDPTW). This problem is very different from the operational routing problem PDPTW. First, from the perspective of optimality, the concern here is profitability from matching shippers with transporters. Operationally speaking, the problem model caters to job bundling constraints whereas in PDPTW, jobs are independent and all jobs must be served. In G-PDPTW, dependency exists between the bundled shipper bids: all jobs within a bid must either be accepted or rejected together, although jobs within a single bid may be assigned to different vehicles. Clearly, the dependency between jobs increases the computational complexity (in the same way as PDPTW is much more complex than the pure VRP). The dependency among job bundles will inevitably add more complexity, as we need to ensure that the bundling constraint must be satisfied, while bearing in mind that jobs in bundles may be served by different vehicles which themselves may be carrying other bundles (i.e. M-to-M relationship exists between jobs and vehicles).

Second, we propose a Branch-and-Cut algorithm for the G-PDPTW. This is not a trivial addition to the literature for PDPTW, as careful attention is needed to choose proper cuts and neighborhood operators to handle the particular features of this problem. One must apply substantial changes to the standard PDPTW cuts to make it work, in particular, to handle the group constraints and hourly cost function. Further, we summarize our earlier work Li and Lau (2017) on an Adaptive Large Neighborhood Search (ALNS) algorithm for solving larger problem instances and demonstrate the effects of different parameters on solution quality. We also show that the ALNS can be extended to solve the time-dependent extension of G-PDPTW.

Third and quite importantly, this work has been applied in the real-world setting. It was motivated by a large logistics platform operator in an Asian city, and our engine has been deployed and running in a B2B marketplace at the time of writing this paper. Our experimental results are in fact derived from test-bedding our approach with real-world datasets synthesized from the data we obtained from this logistics platform company. Furthermore, from the results collected from running the platform, we observe that having a second round of auction for failed shipper bids will improve the successful match rates significantly thereby improving profitability that will benefit all parties (transporter, shipper, and platform operator).

In terms of differences with our earlier work Li and Lau (2017), firstly, this paper contains a branch and cut model. Secondly, we present an ALNS approach for the time-dependent version of the problem expanded from the non time-dependent version in Li and Lau, 2017. To improve efficiency, we introduced a new path-based selection operator (R4) and show experimentally that the selection operator R1 is the best performing one but combining the four selection operators yield the best solution quality. Thirdly, we propose a new iterative auction mechanism and present experimental results comparing with the single shot auction mechanism.

This paper proceeds as follows. In Section 2, we present a literature review. Section 3 introduces our base mathematical model formulation and proposed exact approach based on Branch-and-Cut. In Section 4, we extend our model to time-dependent G-PDPTW and study the impact of time-dependent speeds. Section 5 extends the auction mechanism from single-round to multi-round. Computational experiments for our exact approach, time-dependent single round auction, and comparison of single vs multi-round auctions are presented in Section 6 and 7 respectively. Finally, Section 8 concludes the paper.

## 2. Literature review

In the age of sharing economy, we see the rising use of transportation auctions that allow shippers compete with one another to procure transportation services at the lowest cost from carriers offering services at the highest price. Generally speaking, most research papers considered "full truckload" or FTL, where bids consume entire vehicle loads, e.g.,Kuyzu et al. (2015) and Olcaytu and Kuyzu (2020). In the context of last-mile logistics, not all pickup/delivery jobs could make up FTL bids, implying that FTL auction will not fully meet shipper nor transporter needs. Rather, practitioners turn to the more challenging settings of "less-than-truckload" or LTL auction platforms. Berger and Bierwirth (2010) for example proposed an LTL transportation auction where bundles of shipper requests are generated and offered to the carriers, and transporters place their bids for the offered bundles.

In terms of research, LTL auctions mainly focus on two aspects: the bundle generation problem and the winner determination problem. For example, Xu and Huang (2013) presented a double auction model for transportation service procurement in a spot market with stochastic demand and supply. Zhang et al. (2015) proposed a tractable two-stage robust optimization approach to solve the winner deterministic problem under shipment volume uncertainty.

More recently, we see more research on combinatorial auctions that consider the supply chain, for example multi-unit supply auctions. Zhang et al. (2019a) proposed an optimal and efficient multi-attribute multi-unit auction for transportation procurement in the setting where the carriers have multi-unit supplies and their capacity information is private. Sen et al. (2020) developed a decision support tool for bidders to provide information feedback for multi-item multi-unit combinatorial auctions, which improve the auction efficiency.

Combining auction and vehicle routing is specially interesting as it is related to the techniques used in this paper. We list several recently published papers as follows. Zhang et al. (2019b) proposed a branch-and-price-and-cut algorithm and a branch-and-bound algorithm to solve the vehicle routing problem with time windows and combinatorial auction. Hammami et al. (2019) addressed a bid construction problem with a heterogeneous fleet, and solved by exact and heuristic approaches based on adaptive large neighborhood search. They showed that an exact solver (CPLEX) could not obtain solutions better than their best heuristic solution provided

as an initial solution within certain time limits. This shows the efficacy of ALNS compared with many known heuristic approaches for Winner Determination on transportation auctions.

In this paper, we focus on a major logistics challenge in last-mile B2B fulfillment in emerging markets where the delivery locations are many while the delivery volumes are usually low per location. It is important to ensure that the underlying mechanism must be nimble enough to allow split deliveries where multiple vehicles can serve a single bundle (consisting of multiple jobs), and each vehicle may in turn serve jobs from different bundles, i.e. "many-to-many matching". Such literature is almost non-existent to our knowledge.

Furthermore, the profit margins for urban logistics are low typically, hence auction mechanism designers must deal with the specific challenge on the economic sustainability of the auction platform. Handoko et al. (2016) for instance discussed a bi-criteria combinatorial auction mechanism to achieve environmental sustainability while ensuring economic viability. Xu et al. (2017) constructed two "bundle double auction" for the one-unit demand case where the lane offered by each transportation service purchaser is required to be covered with one truckload once. Dai et al. (2014) proposed an iterative combinatorial auction mechanism for pickup and delivery collaboration among carriers. Karels et al. (2020) investigated an auction mechanism to facilitate collaboration amongst carriers while maintaining autonomy for the individual carriers.

## 3. Mathematical Model and Branch-and-Cut Approach

We are interested in the winner determination problem for single-shot double auctions where a shipper submits a bid (comprising one or more jobs) along with the bid price, each transporter submits the truck availability and cost structure, and the auctioneer decides the winning shippers and transporters. The Winner Determination Problem G-PDPTW aims to select and match a subset of shipper bids with the transporter availability, and maximize the profit (which is the sum of winning shipper bid prices minus total transporter costs) while respecting shipper and transporter constraints. G-PDPTW brings together the "Pickup and Delivery Problem (PDP)" and the group-bundle constraints. Each job is made up of a pickup and a delivery order. With slight abuse of terminology, we will use the term "job" and "order" almost interchangeably in this paper, since it is a one-to-one relationship.

The notations and our proposed Mixed Integer Programming (MIP) model have appeared in Li and Lau (2017) and for ease of reference, they are reproduced below. The notations are given in Table 1.

**Table 1**
Notations for G-PDPTW Model.

| | |
|---|---|
| $n$ | Number of jobs, each job having one pickup and one delivery order |
| $K$ | Set of vehicles, $K = \{1,2,\ldots,\lvert K\rvert\}$, and $k \in K$ |
| $V^p$ | Set of pickup orders $V^p = \{1,2,\ldots,n\}$ |
| $V^d$ | Set of delivery orders $V^d = \{n+1, n+2,\ldots,2n\}$ |
| $V$ | Set of orders, $V = V^p \cup V^d \cup \{0\} \cup \{2n+1\}$, 0 and $2n+1$ represent the dummy origin and destination points, and $i \in V$ |
| $O$ | Set of bids, $O = \{1,2,\ldots,\lvert O\rvert\}$, and $o \in O$ |
| $m_o$ | Number of jobs in bid $o$ |
| $r_o$ | Revenue obtained from serving bid $o$ |
| $w_i$ | Weight of order $i$ |
| $c_i$ | Volume of order $i$ |
| $[e_i, l_i]$ | Time window for order $i$ |
| $\lambda_i$ | Service time for order $i$ |
| $t_{ij}$ | Travel time between orders $i$ and $j$, with minute unit as measurement |
| $Q_k$ | Weight capacity of vehicle $k$ |
| $H_k$ | Volume capacity of vehicle $k$ |
| $[l_k, h_k]$ | Time window associated with dummy depot for vehicle $k$ |
| $p_k$ | Hourly cost of vehicle $k$ |
| $x_{ij}^k$ | Binary decision variables indicating if vehicle $k$ travels directly from order $i$ to order $j$ |
| $y_k$ | Integer variables for the total number of hours incurred by vehicle $k$ to serve its orders |
| $z_o$ | Binary variables, indicate whether bid $o$ is served |
| $\tau_i^k$ | Start time of service for order $i$ on vehicle $k$ |
| $W_i^k$ | Cumulative weight of vehicle $k$ after visiting order $i$ |
| $C_i^k$ | Cumulative volume of vehicle $k$ after visiting order $i$ |

### 3.1. Mathematical model

Our Mixed Integer Programming model formulation of G-PDPTW is presented as follows:

$$\max \sum_{o\in O} r_o z_o - \sum_{k\in K} p_k y_k \tag{1}$$

Subject to:

$$\sum_{i\in O_o}\sum_{j\in V}\sum_{k\in K} x_{ij}^k = m_o z_o,\ \forall o \in O \tag{2}$$

$$\sum_{j\in V}\sum_{k\in K} x_{ij}^k \leqslant 1,\ \forall i \in V^P \tag{3}$$

$$\sum_{i\in V} x_{0,i}^k = \sum_{i\in V} x_{i,2n+1}^k = 1,\ \forall k \in K \tag{4}$$

$$\sum_{i\in V} x_{i,0}^k = \sum_{i\in V} x_{2n+1,i}^k = 0,\ \forall k \in K \tag{5}$$

$$\sum_{j\in V} x_{ij}^k = \sum_{j\in V} x_{ji}^k,\ \forall i \in V^p \cup V^d, k \in K \tag{6}$$

$$\sum_{i\in V} x_{i,j+n}^k = \sum_{i\in V} x_{ij}^k,\ \forall j \in V^P, k \in K \tag{7}$$

$$\tau_j^k + t_{j,j+n} + \lambda_j \leqslant \tau_{j+n}^k,\ \forall j \in V^P, k \in K \tag{8}$$

$$t_{ij} + \lambda_i + M_{ij}^1\left(x_{ij}^k - 1\right) \leqslant \tau_j^k - \tau_i^k,\ \forall i,j \in V, k \in K \tag{9}$$

$$w_i + M_{ij}^2\left(x_{ij}^k - 1\right) \leqslant W_j^k - W_i^k, \forall i,j \in V, k \in K \tag{10}$$

$$c_i + M_{ij}^3\left(x_{ij}^k - 1\right) \leqslant C_j^k - C_i^k, \forall i,j \in V, k \in K \tag{11}$$

$$e_i \leqslant \tau_i^k \leqslant l_i,\ \forall i \in V^p \cup V^d, k \in K \tag{12}$$

$$0 \leqslant W_i^k \leqslant Q_k,\ \forall i \in V^p \cup V^d, k \in K \tag{13}$$

$$0 \leqslant C_i^k \leqslant H_k,\ \forall i \in V^p \cup V^d, k \in K \tag{14}$$

$$\tau_0^k \geqslant \iota_k,\ \forall k \in K \tag{15}$$

$$\tau_{2n+1}^k \leqslant \hbar_k,\ \forall k \in K \tag{16}$$

$$\left(\tau_{2n+1}^k - \tau_0^k\right)/60 \leqslant y_k,\ \forall k \in K \tag{17}$$

$$x_{ij}^k, \tau_i^k, W_i^k \in \mathbb{R}_+, \forall i,j \in V, k \in K \tag{18}$$

$$z_o \in \{0,1\}, \forall o \in O \tag{19}$$

$$y_k \in \mathbb{Z}_+, \forall k \in K \tag{20}$$

This model can be seen as a complete undirected graph $\mathscr{G} = (V,E)$ where $V = V^p \cup V^d \cup \{0\} \cup \{2n+1\}$. $V^p$ and $V^d$ correspond to the sets of pickup and delivery orders respectively, while orders 0 and $2n+1$ represent the dummy depots. Each arc $(i,j) \in E$ has a travel time $t_{ij}$. The distance from dummy depots to other orders, as well as their service times, weights and volumes are all initialized to 0. For ease of reference, the orders in $V$ are sorted so that all pickups must precede all deliveries, hence each delivery order can be obtained from its pickup counterpart index by offset from a fixed constant value $n$.

Let $K$ denote the set of vehicles. Note again that for simplicity our model assigns orders to vehicles and not transporters, so that it does not matter how many vehicles are owned by each transporter. Each vehicle $k \in K$ has a weight capacity denoted $Q_k$ and volume capacity denoted $H_k$. The hourly cost of vehicle $k$ is represented by $p_k$.

Let $O$ be the set of shipper bids. Each bid contains one or more jobs, and each job comprises two orders (a pickup order and a

delivery order). The revenue (i.e. bid price) for delivering a shipper bid $o$ is represented by $r_o$, while the binary decision variable $z_o$ indicates whether the bid is successful (served). For each order $i \in V$, we let $[e_i, l_i]$ denote the time window where $e_i$ and $l_i$ represent the earliest and latest time, respectively. In addition, let $\lambda_i$ be the service (i.e. loading/unloading) time, $w_i$ and $c_i$ be the weight and volume (for a given pickup and delivery pair, $w_i = -w_{i+n}$, and $c_i = -c_{i+n}$).

For each arc $(i,j) \in A$ and each vehicle $k \in K, x_{ij}^k$ is a binary variable which specifies if vehicle $k$ travels from order $i$ directly to order $j$. Moreover, $y_k$ is an integer variable representing the number of hours traveled by vehicle $k$. For each order $i \in V$ and each vehicle $k \in K$, let $\tau_i^k$ be the time that vehicle $k$ begins to serve order $i$, $W_i^k$ and $C_i^k$ define the load (both weight and volume dimension) of vehicle $k$ after visiting order $i$. The objective function (1) maximizes the total profit i.e. the revenue obtained from winning shipper bids minus the transporter costs incurred to serve those bids. The latter is calculated based on the total travel time (with hourly units).

Constraints (2) constrain that the orders belonging to the same bid $o$ is considered to be a bundle, i.e. they must be served or rejected together. Constraints (3) indicate that every order can be served at most once by one vehicle. Constraints (4) and (5) are imposed to fix the origin and destination points (which are dummy orders, with distance to all orders equal to 0) of vehicles. Note that an empty route will be represented by a path with 2 stops, which starts at 0 and ends at (2 n + 1).

Every order except the origin and the destination of a vehicle must have the same number of preceding and succeeding orders, which is defined in constraints (6). Constraints (7) and (8) ensure that the job origin is visited before the destination. Constraints ()()() (9)–(11) are large M constraints that compute the start times as well as cumulative weights and volumes of vehicles at each visit, where:

$M_{ij}^1 = \max(0, l_i + t_{ij} - e_j)$, $M_{ij}^2 = \min(2Q_k, 2Q_k + w_i)$, and.

$M_{ij}^3 = \min(2H_k, 2H_k + p_i)$.

The order time window constraints are defined in constraints (12). Constraints (13) and (14) represent the vehicle capacity constraints in the weight and volume dimension respectively. The time window associated with dummy depot for each vehicle is defined in constraints (15) and (16). Constraints (17) define the vehicle travel time in hours (translate from minutes to hourly based). Finally, constraints (18)-(20) specify the domains of the variables.

While such model can be solved directly by a state-of-the-art solver such as Gurobi or CPLEX, the computational bottleneck is related to constraints (9)–(11), which is very time-consuming due to the large number combinatorial possibilities and affected by the values of $M_{ij}^1, M_{ij}^2$, and $M_{ij}^3$. In the following, we introduce a branch-and-cut approach to address this issue.

### 3.2. Branch-and-Cut Approach

In this subsection, we present our Branch and Cut approach. We first present a Linear Programming (LP) model according to the MIP model presented in subSection 3.1. All the constraints listed in subSection 3.1 are implemented except nonlinear constraints (9)–(11). Note that by removing constraints (9)–(11), the relaxed linear model may introduce subtours and other forms infeasibility. In our approach, we introduce cuts to ensure feasibility is maintained, similar to the approach taken by Ropke et al. (2007). More precisely, we implemented five types of separation inequalities, namely, strengthened travel time constraints, subtour elimination constraints, strengthened capacity constraints, generalized order constraints, and strengthened infeasible path constraints. Besides the first inequalities, the other four inequalities are inspired by Ropke et al. (2007). The identified inequalities are added to the current LP model and re-optimized. The iteration continues until no more violated inequalities can be found.

Moreover, additional constraints (22)-(25) are added from the onset to strengthen the model. Hence, the branch and cut formulation is given as:

$$(1) - -(8) and (12) - -(17) \tag{21}$$

$$y_k \geqslant \sum_{i \in V} \sum_{j \in V} \left( t_{ij} + \lambda_i + \max\left(0, e_j - l_i - t_{ij} - \lambda_i\right)\right) x_{ij}^k \Big/ 60, \forall k \in K \tag{22}$$

$$y_k \leqslant 1 + \sum_{i \in V} \sum_{j \in V} \left( t_{ij} + \lambda_i + \max\left(0, e_j - e_i - t_{ij} - \lambda_i\right)\right) x_{ij}^k \Big/ 60, \forall k \in K \tag{23}$$

$$\tau_j^k \geqslant e_j + \sum_{i \in V \setminus \{j\}} \max\left(0, e_i + t_{ij} + \lambda_i - e_j\right) x_{ij}^k, \forall i \in V^p \cup V^d, k \in K \tag{24}$$

$$\tau_j^k \leqslant l_j - \sum_{i \in V \setminus \{j\}} \max\left(0, l_i + t_{ij} + \lambda_i - l_j\right) x_{ij}^k, \forall i \in V^p \cup V^d, k \in K \tag{25}$$

We can bound the total amount of time incurred by a vehicle $k$ to serve all its orders by a simple linear sum. Suppose a vehicle $k$ is to serve $i$ followed by $j$ consecutively (i.e. $x_{ij}^k$ is set to 1). Let us consider the following two extreme cases: it starts to serve order $i$ at the earliest and latest possible times (i.e. $e_i$ and $l_i$). For the first case, the wait time at $j$ at most equals $\max(0, e_j - e_i - t_{ij} - \lambda_i)$. Hence, the total time needed to serve $i$, travel to $j$ and wait at $j$ is at most $t_{ij} + \lambda_i + \max(0, e_j - e_i - t_{ij} - \lambda_i)$. For the second case, the wait time at least equals

$\max(0, e_j - l_i - t_{ij} - \lambda_i)$, hence the total time needed equals $t_{ij} + \lambda_i + \max(0, e_j - l_i - t_{ij} - \lambda_i)$. Hence, we impose constraints (22) and (23) to define the lower and upper bounds for the total times in hours of each vehicle. They strengthen the model, especially under the case that most order time windows overlap. Similarly, the start time of each order can be strengthened by constraints (24) and (25).

In the following subsections, we will provide details on our proposed separation procedures. Let $\{\bar{x}\}$ be the current LP optimal solution at a given iteration, the weighted graph $G(\{\bar{x}\})$ induced by edges $(i,j)$ is called the support graph. The weight of edge $(i,j)$ is given by $\sum_{k\in K} x_{ij}^k$. The separation procedures work on the support graph. Let $S$ denote a set of orders, and $W(S)$ and $C(S)$ stand for the total weight and volume of $S$, respectively. Let $x(S)$ be $\sum_{i\in S}\sum_{j\in S}\sum_{k\in K} x_{ij}^k$, and $x(\delta(S))$ be $\sum_{i\in S}\sum_{j\in V\setminus S}\sum_{k\in K} x_{ij}^k + \sum_{i\in V\setminus S}\sum_{j\in S}\sum_{k\in K} x_{ij}^k$.

### 3.3. Strengthened Travel Time Inequalities

To strengthen the travel time variable $y_k$ in the support graph, we use a simple greedy heuristic to identify the feasible path. For a given feasible path $R$ for vehicle $k$, where $R = (r_1, r_2, \ldots, r_\rho)$, we continue to improve the sequence in the feasible path using a simple tabu search. Then, we calculate the route duration $T_k$. Finally, inequalities (26) are added, where $M = \hbar_k - \iota_k$. The rationale is that $\rho - 1 - x(R) = 0$ ensures the orders have all been used in $R$ to form a path, and $y_k$ must be equal or shorter than $\lceil T_k \rceil$.

$$y_k - \lceil T_k \rceil \leqslant M(\rho - 1 - x(R)), \forall k \in K \tag{26}$$

### 3.4. Subtour Elimination Inequalities

To prevent multiple cycles (subtours) for vehicles, we use the following subtour elimination separation procedure to identify subtours:

We search all cycles in the support graph using Johnson's algorithm (Johnson, 1977). For every distinct cycle set $S$ found, Constraint (27) is added.

$$x(S) \leqslant |S| - 1 \tag{27}$$

According to Ropke et al. (2007), the inequality can be lifted because a delivery order $i+n$ should be visited after the corresponding pickup order $i$. Any subset $S \subseteq P \cup D$, let $\pi(S) = \{i \in P | n+i \in S\}$ and $\sigma(S) = \{n+i \in D | i \in S\}$. The lifted inequalities are as shown in constraints (28) and (29).

$$x\left(S\right) + \sum_{i\in S}\sum_{j\in \bar{S}\cap\pi(S)}\sum_{k\in K} x_{ij}^k + \sum_{i\in S\cap\pi(S)}\sum_{j\in\bar{S}\setminus\pi(S)}\sum_{k\in K} x_{ij}^k \leqslant \left|S\right| - 1 \tag{28}$$

$$x\left(S\right) + \sum_{i\in \bar{S}\cap\sigma(S)}\sum_{j\in S}\sum_{k\in K} x_{ij}^k + \sum_{i\in\bar{S}\setminus\sigma(S)}\sum_{j\in S\cap\sigma(S)\setminus\pi(S)}\sum_{k\in K} x_{ij}^k \leqslant \left|S\right| - 1 \tag{29}$$

Note that in our separation procedure, we consider not only integer solution but also fractional solutions of $x_{ij}^k$, as this would speedup our branch-and-cut procedure (it is capable of pruning a greater number of violated inequalities).

### 3.5. Strengthened Capacity Inequalities

For any subset $S \subseteq P \cup D$, we need the following two weight and volume related inequalities:

$$x(\delta(S)) \geqslant 2\lceil C(S)/H_k \rceil \tag{30}$$

$$x(\delta(S)) \geqslant 2\lceil W(S)/Q_k \rceil \tag{31}$$

Similar to Ropke et al. (2007), we apply a tabu search heuristic. For each iteration, we identify a set $S$ such that $x(\delta(S)) \leqslant 4, W(S) > Q_k$ and $C(S) > H_k$. Then we choose 3 pickup orders with the largest weight values, and one of these orders is either removed or inserted to the set $S$ that minimizes the value of $x(\delta(S))$ while satisfying the weight and volume constraints. For our implementation, this heuristic will stop after 25 iterations.

### 3.6. Generalized Order Inequalities

Let $U_1, \ldots U_m \subset N$ be mutually disjoint subsets, and $i_1, \ldots, i_m \in P$ be jobs such that $0, 2n+1 \notin U_l$ and $i_l, n+i_{l+1} \in U_l$ for $l = 1, \ldots, m$. The following precedence cycle breaking inequalities introduced by Ropke et al. (2007) are valid for the G-PDPTW:

$$\sum_{l=1}^{m} x\left(U_l\right) + \sum_{l=2}^{m-1}\sum_{k\in K} x^k_{i_1,i_l} + \sum_{l=3}^{m}\sum_{k\in K} x^k_{i_1,n+i_l} \leqslant \sum_{l=1}^{m}\left|U_l\right| - m - 1 \tag{32}$$

$$\sum_{l=1}^{m} x\left(U_l\right) + \sum_{l=2}^{m-2}\sum_{k\in K} x^k_{n+i_1,i_l} + \sum_{l=2}^{m-1}\sum_{k\in K} x^k_{n+i_1,n+i_l} \leqslant \sum_{l=1}^{m}\left|U_l\right| - m - 1 \tag{33}$$

Similar to Ropke et al. (2007), a greedy heuristic is used to identify the sets that violate generalized order inequalities. We first identify the pair of orders ($i$ and $j$) that maximize $\sum_{k\in K}(x^k_{i,j+n} + x^k_{j+n,i} + x^k_{i+n,j+n})$, due to the higher this value, the higher it potentially leads to a violation of inequalities (32) and (33). Then, we search the order $h$ that may violate inequalities (32) and (33). If such an order $h$ been found, we will set $U_1 = (i, n+i), U_2 = (j, n+j), U_3 = (h, n+h)$, and the generalized order inequalities will be added.

### 3.7. Strengthened Infeasible Path Inequalities

To recognize the inequality violations in the support graph, we use a simple breadth-first search algorithm. Every order $i \in V^p \cup V^d$ is in turn considered to be a start order, the path is extended following breadth-first search, and we evaluate the violation for the order time window, vehicle available time period, weight, and volume inequalities. We define $A(R)$ as the arcs set of $R$, and when $R = (p_1, \ldots, p_\rho)$ is an infeasible path, the following inequality is defensible:

$$\sum_{i=1}^{\rho-1}\sum_{j=i+1}^{\rho}\sum_{k\in K} x^k_{p_i,p_j} \leqslant \left|A\left(R\right)\right| - 1 \tag{34}$$

## 4. Time Dependent G-PDPTW

In this section, we extend our model and ALNS algorithm presented in our earlier work Li and Lau (2017) to solve the time-dependent variant of G-PDPTW (TD-G-PDPTW). For the sake of completeness, we present the expanded ALNS pseudo-code in this section (shown as Algorithm 1).

In essence, the ALNS framework maintains a set of operators for removals and insertions of jobs, and in each iteration, it adaptively chooses the selection and perturbation operators by a roulette wheel mechanism, whose probabilities are derived according to the prevailing performance scores of the operators, which are in turn updated dynamically as the search proceeds.

In our earlier work, we proposed 3 selection operators (R1, R2 and R3) and 5 perturbation operators (I1, I2, I3, I4 and I5). For details of these operators, we refer the reader to Li and Lau (2017). In this paper, we add one more operator R4 which is path-based, i.e. it randomly chooses one non-empty path from the current solution, and shorter paths have higher chosen possibility compared to longer paths. The new operator leads to solutions with each truck serving more orders than without this operator.

More formally, we define $P^t_d$ as the probability of choosing operator $d$ at iteration $t$. It will be updated as follows: $P^{t+1}_d := \rho P^t_d + (1-\rho)\chi_i/\zeta_i$, where $\rho$ represents the weightage on historical performance, $\chi_i$ is the performance score of operator $i$, and $\zeta_i$ is the number of times it has been used during the last 200 iterations. To bootstrap the process, we define $P^0_{d_R}$ and $P^0_{d_I}$ to be the probabilities of choosing selection and perturbation operators $d_R$ and $d_I$ respectively in the initial 200 iterations. For our experiments, we conveniently set these probabilities to be uniformly equal for all operators. See Table 5 in Section 6.3 for all parameter values used in our

**Table 5**
Parameter Settings for ALNS Algorithm.

| Description | Values |
|---|---|
| Job selection parameter, $u_1$ | 15% −25% |
| Bid selection parameter, $u_2$ | 15% −25% |
| Job selection from partial bids parameter, $u_3$ | 40% −60% |
| Score accrued to finding globally better solution, $\pi_1$ | 6.00 |
| Score accrued to finding a better solution, $\pi_2$ | 4.00 |
| Score accrued to finding a worse but accepting solution $\pi_3$ | 2.00 |
| Historical performance weightage, $\rho$ | 0.40 |
| Initial probabilities for selection operators, $P^0_{d_R}$ | 0.25 |
| Initial probabilities for perturbation operators, $P^0_{d_I}$ | 0.20 |

experiments. The value of $\chi_i$ is updated according to its performance of applying it - if the solution improves, the value will be increased by $\pi_1, \pi_2$ and $\pi_3$ respectively, as shown in the pseudo-code. After every 200 iterations, the probabilities $P_d^t$ are updated with the new score values as shown above, and the values of $\chi_i$ will be reset to zeros.

Experimentally, we aim to verify the impact of dynamic changes of travel speeds on the objective value for moderate-size synthesized instances as described in Section 6.

Algorithm 1: Pseudo Code of ALNS

**Input:** Initial solution $s$, initial probabilities $P^0_{d_R}$ and $P^0_{d_I}$, all performance scores are set to 0 initially
**while** *"stopping criteria not reached"* **do**
  $s' := s$
  Choose a selection operator using probabilities $P_d^t$
  Apply selection operator which selects either $u_1$ jobs or $u_2$ bids for removal
  Choose a perturbation operator using probabilities $P_d^t$
  Apply perturbation operator to remove selected jobs from $s'$ and reinsert as many unserved jobs as possible into $s'$
  Remove $u_3$ jobs from every partially served bid in $s'$
  **if** $s'$ *is feasible and* $f(s') > f(s_{best})$ **then**
    $s_{best} := s'$
    increase performance score of chosen operators by $\pi_1$
  **if** $f(s') > f(s)$ **then**
    $s := s'$
    increase performance score of chosen operators by $\pi_2$
  **else**
    **if** *s' is accepted by Simulated Annealing with probability* $p(s', s)$ **then**
      $s := s'$
      increase performance score of chosen operators by $\pi_3$
    **end if**
  **end if**
  Update probabilities $P_d^t$ every 200 iterations
**end while**
**Output:** $s_{best}$;

In order to model time-varying speeds, we derived from GoogleMaps the travel times between 15 locations for 24 h, and the average travel speed distribution is as shown in Fig. 2,3. From Fig. 2, we can see the traffic condition between 22:00–6:00 is quite different from 7:00–21:00. The travel time between 10:00–17:00 is quite stable, where the ratio of highest and lowest speeds between 10:00–17:00 is 1.07. There are two peak periods occurring at 8:00 and 18:00, each lasting only around 1 h. To approximate time-varying speeds, we divide the planning horizon into 24 time epochs, and treat the hourly travel speeds according to the distribution, i.e. for each time

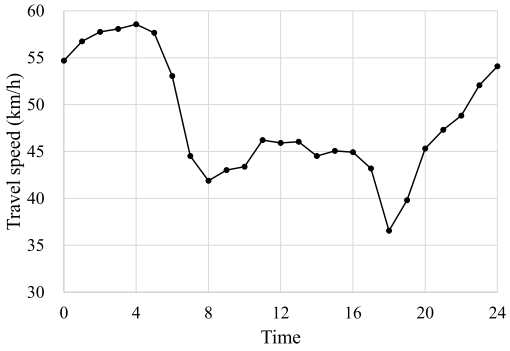



**Fig. 2.** Travel Speed Distribution between 00:00–24:00.

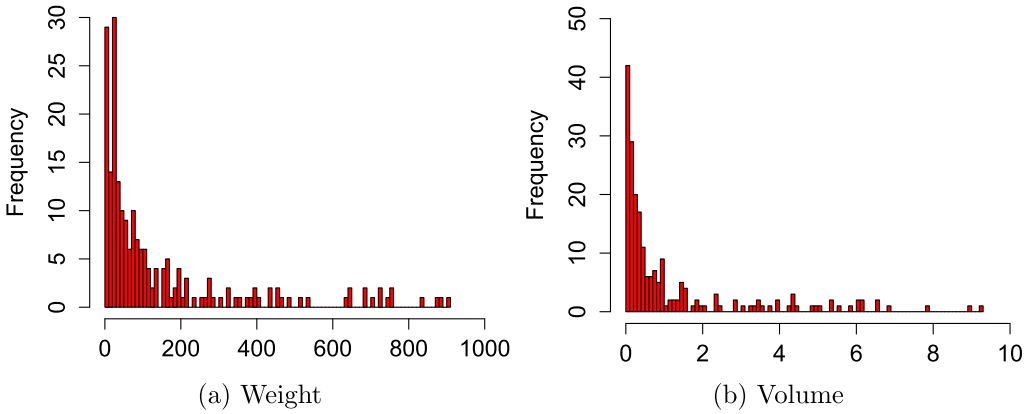



**Fig. 3.** Histogram for Weight ($kg$) and Volume ($m^3$).

epoch, we apply a multiplier to the average speed of 10:00–17:00, and denote the multiplier at time epoch $i$ as $\rho_i$. The speed function is translated to travel time that satisfies the First-in-First-out principle, which means the earlier the start time, the earlier the arrival time.

Note that while the travel time between two locations within a given time epoch is easy to calculate, if travel transcends multiple epochs, then we need to consider how long the vehicle takes in each time epoch.

In Li and Lau (2017), we presented an algorithm to calculate the timespan for a given vehicle route when the travel time between two points is a constant. Now to incorporate time dependency, the computational complexity will increase significantly. Hence, for the overall algorithm of TD-G-PDPTW to be efficient, we propose an improved algorithm for timespan calculation. This modified algorithm is given as Algorithm 2, which is based on the following Fact.

**Fact:** Consider two multipliers $(\rho_1)$ and $(\rho_2)$ (where $\rho_1 \leqslant \rho_2$).

(i) If the vehicle's time span under $\rho_1$ is $t_1$, and vehicle's time span under $\rho_2$ is $t_2$, then, $t_1 \geqslant t_2$.
(ii) If the vehicle's latest departure time (postponement will lead to time window violation) of $\rho_2$ is $d_2$, and vehicle's latest departure time of $\rho_1$ is $d_1$, then, $d_1 \leqslant d_2$.

Overall, the algorithm consists of 4 steps:

1. sort the multipliers in ascending order as $\rho_0, \ldots, \rho_i, \ldots, \rho_N$, if timespan under $\rho_0$ is equal to the timespan under $\rho_N$, then, stop (lines 1–4);
2. calculate the upper bound for depot departure time $t_{max}$ (lines 5–12);
3. calculate the lower bound for depot departure time $t_{min}$ (lines 13–22);
4. increase the lower bound of depot departure time with a 60-min time step (i.e. since anything below 60 min will lead to no difference for the hourly based cost) and repeat until no decrease of timespan can be made (line 23–30).

Algorithm 2: Timespan Calculation for Time-Dependent G-PDPTW

**Input:** sort multipliers in ascending order: $\rho_0, \ldots, \rho_i, \ldots, \rho_N$, timespan value $F$, $F'$, depot start time $t_0$, intermediate parameters $t_{min}$, $t_{max}$, $i$, $N'$

calculate the timespan $F(\rho_0)$ and $F(\rho_N)$ under multiplier $\rho_0$ and $\rho_N$
**if** $F(\rho_0) = F(\rho_N)$ **then**
  $F := F(\rho_0)$, go to **Output**
**end if**
$F' = F(\rho_N)$, $i = N - 1$
**while** $i > 0$ **do**
  calculate the timespan $F$ of vehicle under multiplier $\rho_i$
  **if** $\rho_i$ *leads to a feasible solution for a vehicle route and* $F = F'$ **then**
    $N' := i$, record start time as $t_{max}$, $i := i - 1$
  **else**
    **break**;
**end while**
$i = 0$
**while** $i < N'$ **do**
  calculate the timespan of vehicle under multiplier $\rho_i$
  **if** $\rho_i$ *leads to a feasible solution for a vehicle route and* $F(\rho_i) = F(\rho_{N'})$ **then**
    $F := F(\rho_i)$; go to **Output**
  **if** $\rho_i$ *leads to a feasible solution and* $F(\rho_i) > F(\rho_{N'})$ **then**
    record the current start time as $t_{min}$; **break**;
  $i := i + 1$
**end while**
$t_0 := t_{min}$
**while** $t_0 \leq t_{max} - 60$ **do**
  calculate $F$ associated with $t_0$
  **if** $F > F'$ **then**
    $F := F'$; go to **Output**
  **else**
    $F' := F$, $t_0 := t_0 + 60$
**end while**

**Output:** $F$

## 5. Iterative Auction Mechanism

The profit margins for urban logistics are low typically, hence auction designers must deal with the specific challenge on the economic sustainability of the auction platform. To reduce cost and increase profitability, the auction mechanism should seek to improve the rate of successfully matching shippers and transporters.

In single round auctions, a shipper could fail in its bid, as the shipper may underestimate the market prices. Intuitively, if the platform can provide a second round for a shipper to increase its bid price, the matching rate and hence profitability may increase as well.

In this section, we consider a simple two-round auction mechanism, where the second round is aimed at only the failed bids, i.e. the successful bids for the first round are committed and cannot be changed.

We define the term "win percentage" as the percentage of bids (both shippers and transporters) that are awarded after the auction. If we achieve a win percentage of 100%, it means that all bids are awarded implying that demand and supply match perfectly. However, often times, the win percentage is much lower, due to two reasons: operational constraints (such as time windows and vehicle capacities) prohibit perfect match, and unprofitability (too low shipper bids and/or too high transporter bids). While one cannot do anything with operational constraints to improve profitability, we can improve the win percentage through the following three proposed action policies:

- Increase the shipper bid price by some $x$%.
- Increase the order time window by some $y$ hours.
- Augment the Winner Determination objective function to include a new penalty term for unserved shipper bids:

$$\max \sum_{o \in O} r_o z_o - \sum_{k \in K} p_k y_k - \sum_{o \in O} e_o \left(1 - z_o\right) \tag{35}$$

where $e_o$ is the penalty coefficient for shipper bid $i$ failing the second round auction.

Naturally, this approach may be extended to a multi-round auction setting. However, we show experimentally that two rounds is indeed sufficient for the instances at hand.

Operationally, we first identify the failed shipper bids in the first round auction. Then, we single out the failed transporter bids, as well as trucks that still have some hours of availability upon fulfilling the successful bids in the first round. These bids are collected and sent to the second around auction.

## 6. Experimental Results

We present the computational experiments results to evaluate the performance of our Branch-and-Cut algorithm and ALNS. Our code was implemented on a general-purpose MILP solver (CPLEX 12.6.1), and Java, executed on an Intel Xeon E5-2667v4 8 C/16 T (3.2 GHz) 16 core CPU 32 GB RAM computer. This section proceeds as follows. We first present our test instances are generated. Next, we evaluate the performance of Branch and Cut (exact approach) on small instances, followed by the performance of ALNS (heuristic approach) on medium instances. Then, we present findings and insights on running ALNS on the Time Dependent G-PDPTW problem. Finally, we turn our attention on comparing the benefits of single and two-round auctions.

### 6.1. Test Instances

We design our test instances based on the following factors: number of jobs and vehicles, the spatial distribution of the jobs, time window width, and bid size. These instances are synthesized from data obtained from a large logistics operator from an Asian city. All test instances can be found at https://unicen.smu.edu.sg/pickup-and-delivery-problem-time-window-g-pdptw.

Unlike conventional PDP instances, our instances need to include the bid prices for each job bundle. Such values are determined using a linear regression model that was derived from a real-life historical dataset obtained from our partner logistics provider which contains past delivery transactions and their transacted prices. In essence, the bid price is determined by Eq. (36), which is a linear model based on the number of jobs adjusted according to the weight/volume of the cargo. Let $\zeta$ and $\alpha$ be the per job price and the number of jobs within a shipper bid respectively; $\beta, \chi$, and $\delta$ denote the number of small sizes (with cargo lighter than 10 kg), medium size (with cargo lighter than 100 kg but heavier than 10 kg), and large size (with cargo heavier than 100 kg) jobs within the bid. In addition, the total weight of medium size cargo and large size cargo are represented as $\phi$ and $\varphi$.

$$price = \zeta\alpha - 1.71\beta - 4\chi + 4.7\delta + 0.09\phi + 0.007\varphi \tag{36}$$

#### 6.1.1. Instances of Small Size

Two groups of instances (with 20 and 40 orders) have been generated to test our Branch-and-Cut algorithm. The individual instances with names B1-B12 have 20 orders (excluding the depot), and the instances in the second group (B13-B24) have 40 orders. The number of jobs inside a bid is randomly generated with 1–11 jobs for each bid. Besides that, each job in instances B1-B4 and B13-B16 is

**Table 2**
Design of Experimental Instances.

| Shipper | |
|---|---|
| Spatial distribution | Scatter, Cluster, Mixed Cluster&Scatter |
| Number of orders | 50, 100, 150, 200 |
| Order weight and volume | Based on histograms in Figs. 3(a) and 3(b)<br>Weight is randomly sampled from (0–1000) *kg*,<br>then the corresponding volume is computed |
| Order service time | 15 min |
| Per job price value $\zeta$ | Randomly chosen from \$15, \$30, \$45, \$60 |
| Transporter | |
| Number of vehicles | 7–14 |
| Capacity | 2500 kg, $7m^3$ |
| Vehicles available time period | 9:00–18:00 |
| Vehicles unit cost | charge \$41.25 per hour |

associated with 1-h time windows, B5-B8 and B17-B20 with 3-h time windows, B9-B12 and B21-B24 with 9-h time windows, respectively.

#### 6.1.2. *Instances of Moderate Size*

Three different spatial distributed sets with 50, 100, 150, and 200 orders have been generated: the scattered set, the clustered set, and the mixture of scatter and cluster set. The instances also differ on the width of the time windows. Furthermore, we extend the traditional vehicle routing problems related settings with bid number (3 jobs for each bid). Overall, 48 instances have been generated. Though all bids and jobs used in our experiments are randomly generated, the location, weight, volume, and price are based on real-life data consistent with the logistic operator we worked with. The details of our settings are given in Table 2.

### 6.2. *Branch-and-Cut on G-PDPTW Small Size Instances*

To verify the Branch-and-Cut approach, we experiment on small scale instances and compare the results in terms of objective values with those obtained by applying the MIP model on Cplex. Both approaches are given a wall clock time limit of 2 h. Furthermore, we include the results to demonstrate the efficacy of ALNS to solve small instances. The results are as shown in Table 3 and Table 4. Note that since we are maximizing, the actual solution values are given by lower bound values.

For 20 orders, we observe that both Branch-and-Cut can achieve optimal solutions for all but one instance (B11), while ALNS can achieve optimality for all instances. Interestingly we could prove optimality for Branch-and-Cut solutions for all but one 20-order instances within a wall-clock time 1 to 3 min. Unfortunately, with a larger number of 40 orders, both BC and ALNS give the same solution values (coincidentally), but there is a gap with the upper bounds within the wall-clock time limit of 2 h.

On the other hand, if we apply the MIP model on Cplex, we observe that the results obtained are inferior. More precisely, the gap between the lower and upper bounds remain relatively large after 2 h wall-clock running time. And if we increase the number of orders to 40, Cplex cannot even find a feasible solution after 2 h wall-clock time limit.

**Table 3**
Small Instances (20 orders): Branch-and-Cut vs MIP vs ALNS.

| Ins. | BCLB | BCUB | MIPLB | MIPUB | ALNS (LB) | %Gap |
|---|---|---|---|---|---|---|
| B1 | 295.13 | 295.13 | 295.13 | 666.37 | 295.13 | 0.00% |
| B2 | 295.11 | 295.11 | 295.11 | 295.11 | 295.11 | 0.00% |
| B3 | 295.13 | 295.13 | 295.13 | 295.13 | 295.13 | 0.00% |
| B4 | 243.54 | 243.54 | 243.54 | 243.54 | 243.54 | 0.00% |
| B5 | 295.13 | 295.13 | 295.13 | 384.20 | 295.13 | 0.00% |
| B6 | 336.38 | 336.38 | 295.13 | 398.63 | 336.38 | 13.98% |
| B7 | 336.38 | 336.38 | 336.38 | 397.58 | 336.38 | 0.00% |
| B8 | 326.04 | 326.04 | 326.04 | 381.94 | 326.04 | 0.00% |
| B9 | 336.38 | 336.38 | 336.38 | 389.20 | 336.38 | 0.00% |
| B10 | 336.38 | 336.38 | 295.11 | 394.21 | 336.38 | 13.98% |
| B11 | 336.38 | 366.58 | 336.38 | 389.55 | 366.58 | 0.00% |
| B12 | 326.04 | 326.04 | 326.04 | 382.46 | 326.04 | 0.00% |

Ins.: Instance name.
BCLB: Lower bound of branch and cut approach.
BCUB: Upper bound of branch and cut approach.
MIPLB: Lower bound of applying CPlex MIP model.
MIPUB: Upper bound of applying CPlex MIP model.
%Gap: Percentage Gap between objective values obtained from branch and cut approach and MIP model, computed as (BCLB-MIPLB)/MIPLB*100%

**Table 4**
Small Instances (40 orders): Branch-and-Cut vs MIP vs ALNS.

| Ins. | BCLB | BCUB | MIPLB | MIPUB | ALNS (LB) | %Gap |
|---|---|---|---|---|---|---|
| B13 | 599.16 | 820.16 | – | 886.53 | 599.16 | N.A. |
| B14 | 557.91 | 815.18 | – | 883.78 | 557.91 | N.A. |
| B15 | 557.91 | 804.36 | – | 881.38 | 557.91 | N.A. |
| B16 | 565.30 | 850.69 | – | 879.66 | 565.30 | N.A. |
| B17 | 764.16 | 883.58 | – | 886.53 | 764.16 | N.A. |
| B18 | 681.66 | 883.78 | – | 883.78 | 681.66 | N.A. |
| B19 | 722.91 | 880.35 | – | 881.38 | 722.91 | N.A. |
| B20 | 722.91 | 879.66 | – | 879.66 | 722.91 | N.A. |
| B21 | 764.16 | 884.47 | – | 886.53 | 764.16 | N.A. |
| B22 | 722.91 | 881.03 | – | 883.78 | 722.91 | N.A. |
| B23 | 764.16 | 877.30 | – | 882.41 | 764.16 | N.A. |
| B24 | 764.16 | 878.28 | – | 879.66 | 764.16 | N.A. |

Ins.: Instance name.
BCLB: Lower bound of branch and cut approach.
BCUB: Upper bound of branch and cut approach.
MIPLB: Lower bound of applying MIP model.
MIPUB: Upper bound of applying MIP model.
–: No feasible solution obtained.
%Gap: Percentage Gap between objective values obtained from branch and cut approach and MIP model, computed as (BCLB-MIPLB)/MIPLB*100%

Referring to the last column on gaps, we observe that for 20 orders, within a time limit of 2 h, our proposed approach outperforms Cplex MIP for two instances, and equals Cplex for the rest.

### 6.3. *ALNS on G-PDPTW Medium Size Instances*

Our ALNS approach yields optimal solutions for the small size instances above. In the following, we will present the performance of ALNS on medium size instances for which Branch-and-Cut approach could not return solutions within reasonable time. For this purpose, we will present the run time performance, the effectiveness of different operators, and the solution quality in comparison with a greedy heuristic.

Before that, we present the ALNS parameter settings used in our experiments in Table 5. The parameter tuning strategy for $u_1$ to $u_3$ follows the work of Ropke and Pisinger (2006), i.e. in each iteration we tune a single parameter while fixing the prevailing values of all other parameters.

**1) Wall clock time**

Fig. 4 plots the wall clock time (in seconds) against the number of orders (nodes). From the median value and max value of the CPU time recorded, one can see that even though the run time does not scale linearly with the number of orders, the absolute run time is still within a few minutes for a sufficiently large instance.

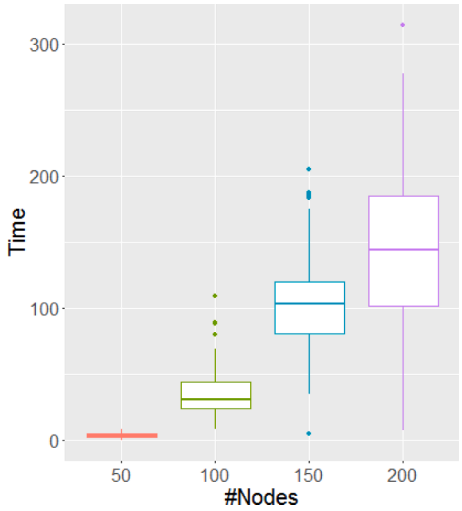
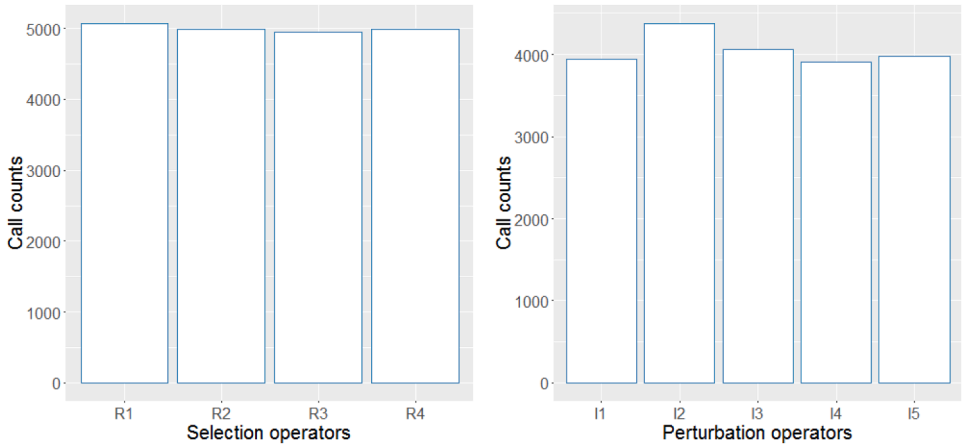



**Fig. 4.** Boxplot of the Wall Clock Time.




(a) Call Counts of Selection Operators (b) Call Counts of Perturbation Operators

**Fig. 5.** Call Counts of Different Operators.

**2) Effectiveness of Different Operators**

Fig. 5 presents the "call counts" (i.e. the number of times the operator was used). Table 6 shows the influence of individual selection operators on solution quality (in terms of gap with solution obtained using all four operators). These values show that we are worse off than the situation that all three operators were used together. Among all the individual operators, **R1** is the best performing operator (with a gap 6.88%). If it were to be excluded, the gap increases to 18.24%. Hence, the contribution of operator **R1** is higher than other operators.

The performance of perturbation operators seem to be indistinguishable from each other. Table 7 shows the influence of individual perturbation operators. Again, these values show that we are worse off than applying all operators together. On the performance of individual operators, operator **I1** outperform the rest (with a gap of 6.88%). If operator **I1** is excluded, the gap increases to 12.16%, while the percentage is highest (20.65%) for the exclusion of **I4**. Overall, one can see that the contribution of operators **I1** and **I4** outperform the other operators.

**3) Solution quality**

Here, we pick a sub-group of instances with 2-h time window and shipper bid sizes of 5 to 20, each bid having 10 orders. The reason is that this sub-group leads to a low bid win percentage, which we will use again in the next section on comparing the results of single versus multi-round auction.

Since Branch-and-Cut could not produce any feasible solution for these instances, we compare our ALNS solutions with those obtained by a greedy heuristic algorithm instead. The greedy algorithm sorts the bids in non-increasing order of revenue as the primary criterion, and the orders within each bid in non-decreasing order of lower bound of the time window as the second criterion. It then inserts the orders into vehicles in the same way as the Next-Fit heuristic for Bin Packing, but ensuring feasibility is preserved.

**Table 6**
Impact of Selection Operators on Solution Quality.

| Gap | R1 | R2 | R3 | R4 |
|---|---|---|---|---|
| use single operator | 6.88% | 27.55% | 27.53% | 27.54% |
| exclude one operator | 18.24% | 7.01% | 6.88% | 6.88% |

**Table 7**
Impact of Perturbation Operators on Solution Quality.

| Gap | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| use single operator | 6.88% | 27.54% | 27.55% | 27.53% | 27.54% |
| exclude one operator | 12.16% | 6.88% | 6.88% | 20.65% | 6.88% |

**Table 8**
Results of Comparison between Greedy Heuristic and ALNS.

| Group | #Orders | #Shipper Bids | Objective of GH | Objective of ALNS |
|---|---|---|---|---|
| S | 50 | 5 | 280.50 | 285.50 |
| S | 100 | 10 | NA. | 475.50 |
| S | 150 | 15 | 277.25 | 689.75 |
| S | 200 | 20 | 480.75 | 1140.75 |
| CS | 50 | 5 | NA. | 50.25 |
| CS | 100 | 10 | 179.00 | 594.50 |
| CS | 150 | 15 | 502.00 | 955.75 |
| CS | 200 | 20 | 220.00 | 838.75 |
| C | 50 | 5 | NA. | 113.75 |
| C | 100 | 10 | 313.75 | 478.75 |
| C | 150 | 15 | 267.50 | 762.50 |
| C | 200 | 20 | 493.50 | 1071.00 |

NA.: No feasible solution can be found (mainly because revenue minus cost is negative with the solution obtained).
C, CS, S: Cluster, mix of cluster scatter, and scatter spatial distribution groups.

As we can see from Table 8, ALNS is able to outperform GH (greedy heuristics) significantly, sometimes by as much as 4 times the greedy solution! This demonstrates the effectiveness of our ALNS approach against a naive algorithm such as greedy.

### *6.4. ALNS on Time Dependent G-PDPTW Instances*

In the following, we compare the objective values obtained by our ALNS approach on G-PDPTW instances versus the corresponding TD-G-PDPTW instances.

The first comparison is on the 10-h time horizon of 09:00–18:00. We define the following settings:

- Initial setting: static speed (i.e. average speed throughout all time epochs)
- First setting: time-varying speeds with multipliers equal to (0.99, 1.03, 1.06, 1.06, 1.04, 1.03, 1.04, 1.01, 0.92, 0.88) (derived from the distribution presented above)

Next, we conduct a sensitivity analysis with different speed variances for a different time horizon of 15:00–23:00:.

- Second setting: speed with multipliers equal to (1.04, 1.01, 0.92, 0.88, 0.98, 1.07, 1.11, 1.15, 1.22, 1.27) (derived from distribution presented above)
- Third setting: speed with multipliers equal to (1.08, 1.02, 0.84, 0.76, 0.96, 1.14, 1.22, 1.30, 1.44, 1.54). Compared to the second setting, this increases the higher and decreases the lower speeds
- Fourth setting: speed with multipliers equal to (1.12, 1.03, 0.78, 0.64, 0.94, 1.21, 1.33, 1.45, 1.66, 1.81). Compared to the third setting, this further increases the higher and decreases the lower speeds

We list the results in Tables 9 and 10 respectively. We group the instances according to three different categories (instance size, spatial distribution and time windows). The bold font number represents the largest value and bold italic font number denotes the smallest value of the corresponding groups. “Obj” and “Obj-TD” denote for the objective values for G-PDPTW and TD-G-PDPTW instances respectively.

**Table 9**
Results of Comparison between G-PDPTW and TD-G-PDPTW.

| | Obj | Obj-TD | |
|---|---|---|---|
| Group | Initial setting | First setting | Gap |
| 50 | 825.07 | 887.84 | 7.61% |
| 100 | 1700.53 | 1607.61 | -5.46% |
| 150 | 2379.85 | 2452.86 | 3.07% |
| 200 | 2847.01 | 2811.36 | -1.25% |
| C | 1919.85 | 1851.66 | -3.55% |
| CS | 1938.88 | 1862.91 | -3.92% |
| S | 1955.61 | 2105.20 | 7.65% |
| 2-TW | 1711.40 | 1640.91 | -4.12% |
| 3-TW | 1738.81 | 1724.35 | -0.83% |
| 4-TW | 2052.18 | 2053.19 | 0.05% |
| 9-TW | 2250.06 | 2341.23 | 4.05% |
| Overall | 1938.11 | 1939.92 | 0.09% |

50, 100, 150, 200: Number of orders.
C, CS, S: Cluster, Mixed Cluster and Scatter, and Scatter spatial distribution groups.
2-TW, 3-TW, 4-TW, 9-TW: 2, 3, 4, and 9-h time window groups.

**Table 10**
Results of Comparison between Different Speed Variances.

| Group | Second setting | Third setting | Fourth setting | L-Gap |
|---|---|---|---|---|
| 50 | **873.15** | *843.13* | 851.53 | 3.44% |
| 100 | **1698.99** | *1645.11* | 1653.01 | 3.17% |
| 150 | *2379.85* | **2445.46** | 2443.47 | 2.68% |
| 200 | 2912.15 | *2832.86* | **2925.72** | 3.17% |
| C | **1918.74** | 1913.14 | *1872.29* | 2.42% |
| CS | **2021.20** | *1869.62* | 1978.59 | 7.50% |
| S | *1958.17* | 2042.16 | **2054.42** | 4.69% |
| 2-TW | **1730.41** | 1681.26 | *1570.65* | 9.23% |
| 3-TW | **1792.74** | *1691.05* | 1792.68 | 5.67% |
| 4-TW | 2087.55 | *2062.08* | **2191.74** | 5.92% |
| 9-TW | *2253.44* | **2332.16** | 2318.66 | 3.38% |

L-Gap: Largest gap between largest value and smallest value.
50, 100, 150, 200: Number of orders.
C, CS, S: Cluster, Mixed Cluster and Scatter, and Scatter spatial distribution groups.
2-TW, 3-TW, 4-TW, 9-TW: 2, 3, 4, and 9-h time window groups.

Table 9 shows the overall gap of the objective values to be 0.09% between the "Obj-TD" and "Obj', which is quite insignificant. One can also see from Table 10 that within each group, as we allow the speed variance to increase, the objective values also do not differ significantly.

One managerial insight is that, given the city's variance in daily traffic (as given in the four settings), our results reveal that time dependent speeds have little effects on G-PDPTW (at least for the test instances used in our experiments). This is not surprising, however, as there are two plausible reasons: 1) the vehicle cost structure of our transportation matching platform is hourly-based, e.g. the cost of vehicle is the same whether we use it for say 2.1 h or 3 h; 2) the peak traffic hour happened around 18:00, but lasts only for 1 h.

Secondly, one can see that the Scatter spatial distribution group performs best under most scenarios, "Cluster" and "Mix of cluster and scatter" show similar behavior, and the differences are negligible. Tables 9 and 10 illustrate that the time window dimension has a significant impact on the objective values. Although the difference is not significant for instances with 2 to 3-h time windows, the objective values of the 2-h time window is much less than those of the 9-h time window.

## 7. Comparing Single and Two-Round Auction

In this section, we quantify experimentally the benefits that the platform participants (shippers, transporters and operator) can gain by extending the standard single-round auction to two rounds. We present our insights on the two-round auction mechanism that improves win percentage from the shippers' perspective.

For this purpose, we run three groups of auctions on instances with order sizes (50, 100, 150, and 200). In all instances, the order time window is set to two hours, and other experimental set up is given in Table 11 (we only list the specific settings in addition to those already listed in Table 2). The price for each shipper bid is calculated according to Eq. (36). We use SBid to represent shipper bid.

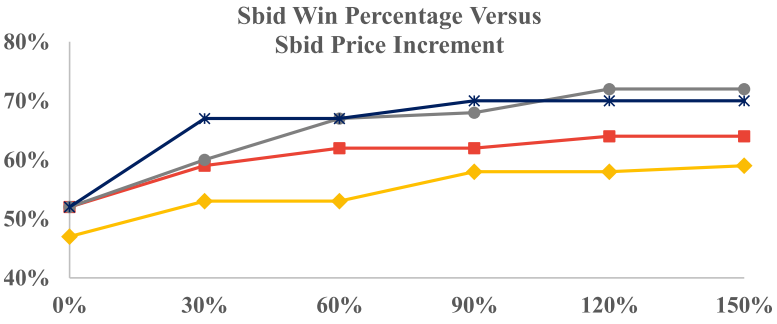
Fig. 6 shows the results obtained. The charts show the win percentage of shipper bids against the three policies (shipper bid price increment, order time window relaxation, and unserved shipper bid penalty, respectively). On average, the percentage increases 18% under the first policy, 38% under the second policy, and 17% under the third policy. In all three policies, while it is not surprising to observe that the win percentage increases with increasing measures, it is interesting to observe that due to capacity and other operational constraints, the win percentage will converge to a certain value beyond a certain threshold.

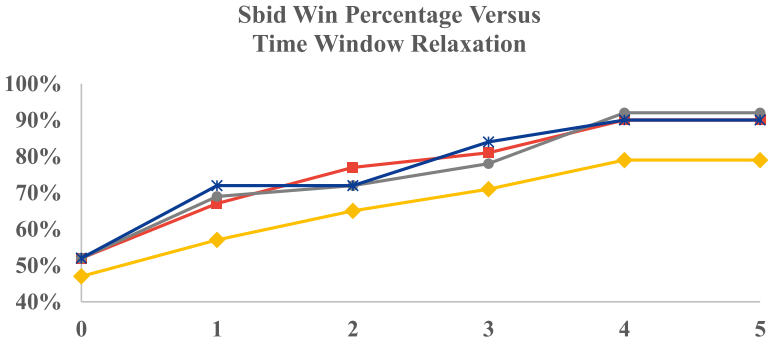
The win percentage under the second policy increases with increasing time window relaxation, until 4 h time window relaxation (i.e. bids with 6 h time window). From 4 h to 5 h nothing changes, the reason is that transporters working duration is set to 9 h, 6 h time window relaxation already take the maximum truck availability in the second round. The first and third policy also show the improvement threshold, for shipper bid price increment, the shipper price increase 120% is a threshold for first policy beyond which the win percentage is only slightly affected for the 200-orders group. For the third policy, the tipping point comes early, at 60%.

Overall, we observe that the win percentage of the second policy (i.e. increase order time windows) performs best among the three policies. It indicates that order time windows is the most effective mechanism for increasing profitability, because even if the shipper bid price were to increase significantly, should no truck be available for the order time windows, the shipper bids will still fail.

**Table 11**
Design of Experimental Instances for Two-Round Auction.

| | |
|---|---|
| First Policy Setting | Increase the shipper bid price by 30%-150% |
| Second Policy Setting | Increase the order time window by 1-5 h |
| Third Policy Setting | Modify the Winner Determination objective function with penalty coefficient (i.e. $e_o$) ranging from 30%-120% of corresponding bid price |



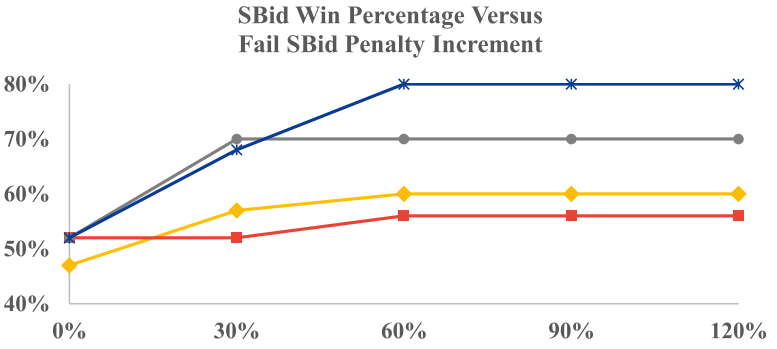




**Fig. 6.** Second Round Auction Shipper Bid win percentage versus Different Factors, *: #Orders = 50, ■: #Orders = 100, ●: #Orders = 150, ◆: #Orders = 250.

Unlike the first two policies which require the shippers to change their bids, in the third policy, the platform changes the objective function. Although the win percentage increment is not as significant as the other two policies, the performance is acceptable, given that it may be challenging to persuade shippers to change their bids. Hence, the platform operator could increase win probabilities and hence profitability using this policy.

Another interesting finding is that the performance varies as the number of orders changes, where the 50 orders group outperforms the 200 orders group, but hard to draw a clear conclusion for other group instances. The reason is that the win percentage not only depends on a particular feature, but also on the interaction between multiple factors, which include the total number of failed shipper bids and available trucks in the second round, etc.

## 8. Conclusion

In this paper, we investigate the Winner Determination Problem associated with a transport marketplace and model it as a variant of the PDPTW with bundled jobs. From the practice point of view, this scheme offers small to medium-sized transporter and shippers an opportunity to achieve a win–win solution for last-mile B2B delivery. SMEs tend to benefit more since the transporters can tap into a

large market (through the platform) while shippers would benefit from economies of scale when their shipments get bundled with other (small, medium) shipments thereby saving delivery cost. That said, large transporters and shippers will also benefit from participating in this platform that provides a channel for dealing with their demand surges. Transporters can enjoy the benefits of resource pooling enabled by this technology, while shippers can find cheaper options by utilizing the platform.

Even though the reported model has been deployed effectively in the platform of a large local logistics platform operator, academically, we do see a number of limitations of this work and potential future extensions to enrich the wider applicability. In addition to those highlighted in Li and Lau (2017), we see the following to be interesting topics that may lead to more profitable business models: 1) reorganizing shipper jobs by splitting the pickup and delivery connection, and allowing different vehicles to serve the pickup and delivery pairs by adding intermediate transfer points; 2) extending the resource-sharing auction concept to other parts of the supply chain, such as warehouse and production facilities; 3) profit sharing with the stakeholders in the form of rebates post-auction, which may incentivize more users to participate in the platform; such profits may be rebated to participants based on varying performance indicators collected over time.

## 9. Acknowledgement:

This research is funded by the National Research Foundation Singapore under its Corp Lab @ University scheme and Fujitsu Limited as part of the A*STAR-Fujitsu-SMU Urban Computing and Engineering Centre of Excellence.

## References

Berger, S., Bierwirth, C., 2010. Solutions to the request reassignment problem in collaborative carrier networks. Transportation Research Part E: Logistics and Transportation Review 46, 627–638.
Dai, B., Chen, H., Yang, G., 2014. Price-setting based combinatorial auction approach for carrier collaboration with pickup and delivery requests. Oper. Res. Int. Journal 14, 361–386.
Hammami, F., Rekik, M., Coelho, L.C., 2019. Exact and heuristic solution approaches for the bid construction problem in transportation procurement auctions with a heterogeneous fleet. Transportation Research Part E: Logistics and Transportation Review 127, 150–177.
Handoko, S.D., Lau, H.C., Cheng, S.F., 2016. Achieving economic and environmental sustainabilities in urban consolidation center with bi–criteria auction. IEEE Trans. Autom. Sci. Eng. 13, 1471–1479.
Johnson, D.B., 1977. Efficient algorithms for shortest paths in sparse networks. Journal of the ACM (JACM) 24, 1–13.
Karels, V.C., Veelenturf, L.P., Woensel, T.V., 2020. An auction for collaborative vehicle routing: Models and algorithms. EURO Journal on Transportation and Logistics 100–109.
Kuyzu, G., Akyol, Ç.G., Ergun, Ö., Savelsbergh, M., 2015. Bid price optimization for truckload carriers in simultaneous transportation procurement auctions. Transportation Research Part B: Methodological 73, 34–58.
Li, B., Lau, H.C., 2017. A combinatorial auction for transportation matching service: Formulation and adaptive large neighborhood search heuristic. International Conference on Computational Logistics, Springer. 127–142.
Olcaytu, E., Kuyzu, G., 2020. An efficient bidding heuristic for simultaneous truckload transportation auctions. Optimization Letters 1–10.
Ropke, S., Cordeau, J.F., Laporte, G., 2007. Models and branch-and-cut algorithms for pickup and delivery problems with time windows. Networks 49, 258–272.
Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation Science 40, 455–472.
Sen, A.K., Bagchi, A., Chakraborty, S., 2020. Designing information feedback for bidders in multi-item multi-unit combinatorial auctions. Decis. Support Syst. 130, 113230.
Sheffi, Y., 2004. Combinatorial auctions in the procurement of transportation services. Interfaces 34, 245–252.
Techinasia, 2017. uber for logistics is already happening in asia, and uber is getting left behind. URL:https://www.techinasia.com/gogovan-easyvan.
Xu, S.X., Huang, G.Q., 2013. Transportation service procurement in periodic sealed double auctions with stochastic demand and supply. Transportation Research Part B: Methodological 56, 136–160.
Xu, S.X., Huang, G.Q., Cheng, M., 2017. Truthful, budget-balanced bundle double auctions for carrier collaboration. Transportation Science 51, 1365–1386.
Zhang, B., Yao, T., Friesz, T.L., Sun, Y., 2015. A tractable two-stage robust winner determination model for truckload service procurement via combinatorial auctions. Transportation Research Part B: Methodological 78, 16–31.
Zhang, J., Xiang, J., Cheng, T.E., Hua, G., Chen, C., 2019a. An optimal efficient multi-attribute auction for transportation procurement with carriers having multi-unit supplies. Omega 83, 249–260.
Zhang, Z., Luo, Z., Qin, H., Lim, A., 2019b. Exact algorithms for the vehicle routing problem with time windows and combinatorial auction. Transportation Science 53, 427–441.