3-2022

Coordinated delivery to shopping malls with limited docking capacity

Ruidian SONG
Tsinghua University

Hoong Chuin LAU
Singapore Management University, hclau@smu.edu.sg

Xue LUO
Tsinghua University

Lei ZHAO
Tsinghua University

Citation

SONG, Ruidian; LAU, Hoong Chuin; LUO, Xue; and ZHAO, Lei. Coordinated delivery to shopping malls with limited docking capacity. (2022). *Transportation Science*. 56, (2), 501-527.
Available at: https://ink.library.smu.edu.sg/sis_research/6233

Submitted to *Transportation Science*
manuscript TS-2019-0384

Coordinated Delivery to Shopping Malls with Limited Docking Capacity

Ruidian Song
Department of Industrial Engineering, Tsinghua University, Beijing, 100084, China, srd13@mails.tsinghua.edu.cn

Hoong Chuin Lau
School of Computing and Information Systems, Singapore Management University, Singapore, Singapore, hclau@smu.edu.sg

Xue Luo
Department of Industrial Engineering, Tsinghua University, Beijing, 100084, China, luoxue16@mails.tsinghua.edu.cn

Lei Zhao
Department of Industrial Engineering, Tsinghua University, Beijing, 100084, China, lzhao@mail.tsinghua.edu.cn

Shopping malls are densely located in major cities such as Singapore and Hong Kong. Tenants in these shopping malls generate a large number of freight orders to their contracted logistics service providers, who independently plan their own delivery schedules. These uncoordinated deliveries and limited docking capacity jointly cause congestion at the shopping malls. A delivery coordination platform centrally plans the vehicle routes for the logistics service providers and simultaneously schedules the dock time slots at the shopping malls for the delivery orders. Vehicle routing and dock scheduling decisions need to be made jointly against the backdrop of travel time and service time uncertainty and subject to practical operations rules. We model this problem as a two-stage stochastic mixed integer program, develop an Adaptive Large Neighborhood Search algorithm that approximates the second stage recourse function using various sample sizes, and examine the associated in-sample and out-of-sample stability. Our numerical study on a testbed of instances based on real data in Singapore demonstrates the value of coordination and the value of stochastic solutions.

Key words: Coordinated delivery; Docking capacity; Endogenous time window; Stochastic travel and service times; Adaptive large neighborhood search

History:

1. Introduction

In major cities such as Singapore and Hong Kong, shopping malls are densely located to cater to the demands of tourists and local shoppers. Typically within a shopping mall, there could be hundreds of tenants including retail shops, restaurants, supermarkets, etc., which generate a large number of freight orders each day, to be delivered by the individually contracted logistics service providers (LSPs) of these tenants.

Unfortunately, many of these deliveries are fragmented in that different LSPs independently plan their delivery schedules without coordination with one another. Since the number of unloading docks in a shopping mall is normally very limited, a vehicle has to wait when all the docks are occupied. Consequently, high volumes of uncoordinated traffic can lead to substantial waiting times and delays, which has been observed in our site investigations of shopping malls (such as Tampines Mall and ION Orchard) in Singapore from December 2017 to February 2018. Clearly, coordinating the schedule of vehicle arrivals across different LSPs can reduce the waiting time and delay, and achieve a better utilization of docks.

Productivity aside, fragmented deliveries will result in negative environmental (noise, pollution) as well as social (traffic congestion and safety) externalities. Motivated by the social, economic, and environmental enhancements, the Collaborative Urban Delivery Optimization (CUDO) project aims to develop a centralized decision making platform where LSPs can submit their orders and obtain recommendations on vehicle routes and dock schedules. This project is supported by the Info-communications Media Development Authority (IMDA, a national government agency responsible for enhancing Singapore's competitiveness through adoption of IT) of Singapore.

The workflow of the CUDO platform is outlined as follows. During the day, the customers (i.e., tenants in shopping malls) place delivery orders to their contracted LSPs. An order indicates the location, the amount of goods to be delivered on the next day, and the time window within which the delivery should be made. Each LSP submits its orders to the CUDO platform by a certain cutoff time. The CUDO platform will then generate the routes of all the delivery orders and schedule the (dock) time slots at the shopping malls, all to be done within one hour. Specifically, this means that the platform will decide which order is assigned to which vehicle and to what time slot (the usage of a dock) at a shopping mall, along with the vehicle routes. The scheduled time slots are then reserved via the interface with the various external Dock Scheduling and Queuing (DSQ) systems. On the next day, a vehicle will serve the orders according to the plan, and a shopping mall will manage the vehicles' entrances based on the time slot reservations.

Note that each reserved time slot is [hard, soft] where the vehicle has to wait outside the shopping mall if it arrives early, and is allowed to enter if it arrives late (with some penalty in the objective function). This time slot can be regarded as an *endogenous* time window, which is to be differentiated from the *exogenous* time window of an order imposed by the customer.

In order to schedule good time slots, the model must deal with the uncertainty in travel times between two locations and the uncertainty in orders' service times. Furthermore, the limited docking capacity introduces more uncertainty in that a vehicle may need to wait (inside the shopping mall) for an available dock, and the waiting time may influence the subsequent orders, which in turn affects the schedule of other vehicles delivering to the same shopping mall.

We refer to the above problem faced by the CUDO platform as *coordinated delivery to shopping malls with limited docking capacity*. The objective is to minimize the total cost that consists of the routing cost of the vehicles and the expected penalty cost that includes the penalties for outside waiting time, inside waiting time, late service start time, and late service end time of the orders and for overtime working of the vehicles.

Our paper makes a number of contributions to the literature as follows.

- We introduce the novel problem of the coordinated delivery to shopping malls with limited docking capacity, which is the first in the literature that integrates the practical features of docking capacity, endogenous time windows (i.e., scheduled dock time slots), practical operations rules such as First-Arrive-First-Enter (FAFE) and First-Enter-First-Serve (FEFS), and stochastic travel times and service times. We formulate the problem as a two-stage stochastic mixed integer program. In the first stage, we plan the vehicle routes and schedule the time slots (endogenous time windows) at the shopping malls for the orders. Then the recourse function of the second stage evaluates the first stage solution on the expected penalty cost of outside & inside waiting time and late service start & end time of the orders and the overtime working of the vehicles, under the stochasticity in both travel times and service times and subject to practical operations rules. Note that the incorporation of the above practical features is important to properly represent the coupling effect of the scheduled dock time slots on vehicle routes in the first stage and evaluate the interaction among vehicle routes and scheduled dock time slots due to the docking capacity and the operations rules such as FAFE and FEFS in the second stage.

- We develop an Adaptive Large Neighborhood Search (ALNS) metaheuristic and approximate the second stage recourse function using sample average. To overcome the computational challenge of evaluating the first stage solutions (of vehicle routing and dock scheduling) in a very large solution space, we employ various sample sizes at different layers of the ALNS algorithm and test the in-sample and out-of-sample stability. We also design two new scheduling operators in our ALNS algorithm that enable the search to consider various time slot decisions. Note also that our ability to formulate the practical features in a two-stage mixed-integer program enables us to solve the mean value problems (MVP) up to certain scales to optimality so that we are able to evaluate the quality of the ALNS in deterministic settings.

- We construct a testbed of instances based on real data in Singapore. We handle the interdependency between the travel times on arcs in the road network using the copula-based scenario generation method (Kaut 2014, Guo, Wallace, and Kaut 2019), and we generate the service time scenarios by bootstrapping from real service time data from 76 stores in a shopping mall in Singapore, with 3470 observations of service time. To show the value of coordination, we compare with the current practice that each LSP plans its own deliveries and schedule the time slots in an uncoordinated manner ("uncoordinated solution"). The average percentage saved in total cost by the coordinated solutions is 9.4%, which is mainly driven by the reduction in penalty cost of late start and end service time of the orders. The uncoordinated solutions tend to underestimate the probability of congestion and schedule time slots optimistically, which potentially results in longer expected waiting time inside the shopping malls and this propagates to significantly longer expected late start and end service time of the orders. Our numerical study also shows the significant value of stochastic solutions (an average increase of 21.1% in total cost if the stochasticity in both travel times and service times is ignored), and further, based on our test instances, the stochasticity in service times seems to have a higher impact on the total cost than the stochasticity in travel time does.

The remainder of the paper is organized as follows. Section 2 reviews the related literature. The problem is formally described in Section 3 and the corresponding two-stage stochastic mixed integer program is formulated in Section 4. The proposed ALNS metaheuristic is introduced in Section 5. The newly generated instances, the tuning process of the algorithm parameters, and the results of numerical experiments are presented in Section 6. Finally, we provide our concluding remarks in Section 7.

2. Literature review

The problem under study is characterized by stochastic travel times and service times, endogenous time windows, and limited docking capacity. We review the related literature in this section.

2.1. VRP with stochastic travel time or service time

In the literature on stochastic vehicle routing problems (SVRP), stochastic demands, stochastic customers, and stochastic travel times and/or service times are the three mainly studied stochastic characteristics (Gendreau, Jabali, and Rei 2016, Oyola, Arntzen, and Woodruff 2017, 2018). There are two modeling paradigms for SVRP (Gendreau, Jabali, and Rei 2016): the a priori paradigm and the re-optimization paradigm. In the a priori paradigm, routes are established before any stochastic information is revealed and are adjusted following a predefined recourse policy based on the revealed information during the execution of the routes. The re-optimization paradigm gradually constructs the routes as the stochastic information is revealed. This paper falls into the a priori paradigm.

Introduced by Laporte, Louveaux, and Mercure (1992), the vehicle routing problem with stochastic travel times or service times (VRPST) that uses the a priori paradigm is usually modeled as a two-stage stochastic integer program (2S-SIP) or a chance constrained program (CCP). The recourse policies in the 2S-SIP include skipping customers (Binart et al. 2016, Errico et al. 2016), ending the tour (Campbell, Gendreau, and Thomas 2011), and penalizing shift violation (Laporte, Louveaux, and Mercure 1992, Lei, Laporte, and Bo 2012) or earliness/tardiness in terms of customers' time windows (Ando and Taniguchi 2006, Russell and Urban 2008, Taş et al. 2013). In the CCP, there could be constraints or objectives on the probability of deadline violation (Adulyasak and Jaillet 2016) or time window violation (Ehmke, Campbell, and Urban 2015, Miranda and Conceição 2016).

In a 2S-SIP, a general objective is $F = \min_x [c^T \boldsymbol{x} + \mathbb{E}_\omega \mathcal{Q}(\boldsymbol{x}, \boldsymbol{\xi}(\omega))]$, where $\boldsymbol{x}$ is the first stage decision, $\boldsymbol{\xi}$ is a random vector, ω is a scenario in sample space Ω, and $\mathcal{Q}(\cdot)$ is the second stage cost function. In different VRPST variants, researchers design solution methods based on different assumptions on the sample space Ω, properties of the second stage cost function $\mathcal{Q}(\cdot)$, etc. When Ω is finite and small, the VRPST can be solved directly by solving its deterministic equivalent problem (Laporte, Louveaux, and Mercure 1992, Kenyon and Morton 2003).

When Ω is infinite (continuous random variable) but the recourse function $\mathbb{E}_\omega \mathcal{Q}(\boldsymbol{x}, \boldsymbol{\xi}(\omega))$ has a closed form expression, the VRPST can be solved by exact methods or metaheuristics for deterministic VRPs. As summarized in Table 1, these papers normally assume that the probability distribution of travel (or service) times adhering to the convolution property and there is no or only [soft, soft] time windows, where the vehicle is allowed to start its service whenever it arrives but is penalized for earliness/tardiness.

Table 1 VRPST literature that models as a 2S-SIP and has a closed form of $\mathcal{Q}(\cdot)$ function

Research	Recourse policy	Probability distribution	Exogenous time window	Solution method[*]
Russell and Urban (2008)	Earliness and tardiness	Erlang	[Soft, soft]	TS
Campbell, Gendreau, and Thomas (2011)	End the tour or not	Gamma	No	DP, VNS
Lei, Laporte, and Bo (2012)	Shift violation	Normal	No	VNS
Taş et al. (2013)	Earliness and tardiness	Gamma	[Soft, soft]	TS
Taş et al. (2014a)	Earliness and tardiness	Gamma	[Soft, soft]	TS, ALNS
Taş et al. (2014b)	Earliness and tardiness	Gamma	[Soft, soft]	B&P
Yuan, Liu, and Jiang (2015)	Tardiness	Normal	[Null, soft]	B&P
Errico et al. (2016)	Skip current or next customer	Discrete triangular	[Hard, hard]	B&C&P

[*] Adaptive Large Neighborhood Search; B&C: Branch-and-Cut; B&P&C: Branch-and-Price-and-Cut; LP: Linear Programming; MIP: Mixed Integer Programming; SD: Scenario Decomposition; TS: Tabu Search.

When the time window is hard, the arrival time distribution will be truncated so that the convolution property will not hold. Errico et al. (2016) consider [hard, hard] time windows, where the vehicle has to wait if it arrives early and is not allowed to serve if it arrives late. The service time is only revealed upon arrival. If the a priori route is infeasible, one of the two recourse actions will be taken: skip the current customer or skip the next customer. The authors formulate it as a set partitioning problem and solve it by exact branch-cut-and-price algorithms, where they impose two practically reasonable conditions to restrict the set of feasible routes. To approximate the distribution of arrival times with no convolution property due to hard time windows, Chang, Wan, and Ooi (2009) use a convolution propagation approach, Ehmke, Campbell, and Urban (2015) resort to the extreme value theory, and Miranda and Conceição (2016) develop a statistical model.

When Ω is infinite or large, and when the recourse function $\mathbb{E}_\omega \mathcal{Q}(\boldsymbol{x}, \boldsymbol{\xi}(\omega))$ has no closed form expression, one common approach is to use a sample average to approximate the recourse function. Li, Tian, and Leung (2010) and Adulyasak and Jaillet (2016) use Monte Carlo sampling to generate a sample $\Omega' \subset \Omega$, and replace the $\mathbb{E}_\omega \mathcal{Q}(\boldsymbol{x}, \boldsymbol{\xi}(\omega))$ with the sample average $|\Omega'|^{-1} \sum_{\omega \in \Omega'} \mathcal{Q}(\boldsymbol{x}, \boldsymbol{\xi}(\omega))$. $|\Omega'|$ often needs to be large. Kenyon and Morton (2003), Verweij et al. (2003), Demir et al. (2016) use the Sample Average Approximation (SAA), which is proposed by Kleywegt, Shapiro, and Homem-De-Mello (2002). SAA also approximates the recourse function $\mathbb{E}_\omega \mathcal{Q}(\boldsymbol{x}, \boldsymbol{\xi}(\omega))$ by the sample average $|\Omega'|^{-1} \sum_{\omega \in \Omega'} \mathcal{Q}(\boldsymbol{x}, \boldsymbol{\xi}(\omega))$, denoted by an "SAA problem." However, it solves M SAA problems and evaluates the solutions by a larger independent sample Ω'' ($\Omega'' \subset \Omega$ and $|\Omega''| \gg |\Omega'|$). If a target statistical optimality gap is achieved, the solution with the best evaluated objective function value is selected; otherwise, the sample size $|\Omega'|$ needs to be increased and the process repeats.

An alternative approach is to use a scenario generation method and select a sample size that exhibits both in-sample and out-of-sample stable. This approach has been applied in stochastic vehicle routing (Guo, Wallace, and Kaut 2019), service network design (Lium, Crainic, and Wallace 2009, Wang, Crainic, and Wallace 2019), and logistics capacity planning (Crainic et al. 2016). In-sample stability measures the stability of the (best) objective function value of the solution obtained from a sample, while out-of-sample stability measures the stability of the real performance of the (best) solution obtained from a sample. Both in-sample and out-of-sample stability require that each sample should lead to a solution with approximately the same objective function value, except that in the in-sample stability test, the objective function value is evaluated by the sample used to find the solution, while in the out-of-sample stability test, the objective function value is evaluated by the true objective function or a different sample than that was used to find the solution.

In this paper, we consider both stochastic travel times and service times and formulate the problem as a two-stage stochastic mixed integer program. We develop an Adaptive Large Neighborhood Search algorithm that approximates the second stage recourse function using various sample sizes, and examine the associated in-sample and out-of-sample stability.

2.2. SVRP with endogenous time windows

Due to the rapid growth of online purchase and home delivery, logistics service providers face fierce competitions and are highly motivated to increase their service level. One approach is to voluntarily promise a narrow delivery time window to the customer. This endogenous time window is the decision made by the LSP, which is different from the exogenous time window imposed by the customer.

As summarized in Table 2, the literature on the SVRP with endogenous time windows is scarce. In Jabali et al. (2015), the endogenous time window ("self-imposed time window") is [hard, soft], where a vehicle has to wait if arrives early and incurs a tardiness penalty if it arrives late. The authors consider the situation where exactly one arc will suffer a disruption from its baseline duration, decompose the problem into the routing phase and scheduling phase, and develop a hybrid algorithm (Tabu Search for the routing phase and solving an LP model for the scheduling phase). Vareias, Repoussis, and Tarantilis (2019) extend the work of Jabali et al. (2015) to allow more than one arc to be disrupted. The authors develop an ALNS algorithm, where an LP/MIP is solved in each iteration to assign endogenous time windows. Hoogeboom et al. (2021) study a robust vehicle routing problem with time window assignments, in which the travel times are uncertain and only some descriptive statistics such as mean, minimum, and maximum travel times are available. The objective is to minimize the risk of violating the assigned time windows, while the expected total travel time is below a certain threshold. The problem is solved in a branch-and-cut fashion, where the subgradient cuts generated from the time window assignment subproblem are iteratively added.

Table 2 SVRP with endogenous time windows

Research	Stochastic characteristics	Probability distribution	Endogenous time window	Solution method[*]
Jabali et al. (2015)	Travel time	Discrete	[Hard, soft]	TS + LP
Spliet and Gabor (2015)	Demand	Discrete	[Hard, hard]	B&P&C
Spliet and Desaulniers (2015)	Demand	Discrete	[Hard, hard]	B&P&C
Dalmeijer and Spliet (2018)	Demand	Discrete	[Hard, hard]	B&C
Neves-Moreira et al. (2018)	Demand	Discrete	[Hard, hard]	Matheuristic
Spliet, Dabia, and Van Woensel (2018)	Demand	Discrete	[Hard, hard]	B&P&C
Subramanyam, Wang, and Gounaris (2018)	Demand	Discrete	[Hard, hard]	SD
Vareias, Repoussis, and Tarantilis (2019)	Travel time	Discrete/ Gamma	[Soft, soft]	ALNS + MIP/ ALNS + LP
Hoogeboom et al. (2021)	Travel time	Descriptive statistics	[Soft, soft]	B&C

[*] Adaptive Large Neighborhood Search; B&C: Branch-and-Cut; B&P&C: Branch-and-Price-and-Cut; LP: Linear Programming; MIP: Mixed Integer Programming; SD: Scenario Decomposition; TS: Tabu Search.

In the studies of Spliet and Gabor (2015), Spliet and Desaulniers (2015), Dalmeijer and Spliet (2018), Neves-Moreira et al. (2018), Spliet, Dabia, and Van Woensel (2018), and Subramanyam, Wang, and Gounaris (2018), an endogenous time window is assigned to each customer in the first stage. After the stochastic demands are revealed, the routes are then decided. Since the endogenous time window is [hard, hard] and cannot be violated, a VRP with time windows needs to be solved in the second stage.

2.3. VRP with docking capacity

The VRP literature considering docking capacity is very scarce. Lam and Van Hentenryck (2016) study a pickup and delivery vehicle routing problem with time windows and limited location resources. The authors consider two types of resources, that is, service resources (e.g., forklifts) and presence resources (e.g., parking bays) at the locations. They solve the underlying vehicle routing problem using a branch-and-price algorithm and check the location resource feasibility using constrained programming. Froger et al. (2017) study an electric vehicle routing problem with capacitated charging stations, where an arriving vehicle has to wait if the charging station is full. The paper considers nonlinear charging functions and allows for partial recharge. The objective is to minimize the total time, which includes driving, service, charging, and waiting time. The authors develop a route-first assemble-second matheuristic as the solution method. Both papers consider problems in the deterministic setting.

The VRP with cross-docking (VRPCD) literature deals with the pickup of products from a set of origins and the delivery of these products to destinations through an intermediate cross-dock, where products are unloaded from inbound vehicles at the inbound dock doors, sorted and consolidated in the intermediate storage area, and reloaded onto outbound vehicles at the outbound dock doors (Wen et al. 2009, Van Belle, Valckenaers, and Cattrysse 2012, Li et al. 2019). Some VRPCD papers consider the docking capacity by jointly determining the vehicle routes, the assignment of inbound and outbound vehicles to a limited number of inbound and outbound dock doors, respectively, and the sequence to process the vehicles at each dock door (Agustina, Lee, and Piplani 2014, Dondo and Cerdá 2014, 2015, Nasiri et al. 2018, Rahbari et al. 2019). Agustina, Lee, and Piplani (2014) study the vehicle routing and scheduling problem (VRSP) that integrates the routing of the outbound vehicles and the scheduling of the inbound vehicles at, the unloading and loading operations in, and the outbound vehicles from the cross-docking facility. Rahbari et al. (2019) extend the problem to consider the uncertainty in the product freshness-life and the travel time using robust optimization approaches. Dondo and Cerdá (2014) propose a mixed linear integer programming formulation that simultaneously consider both the pickup (inbound) and delivery (outbound) vehicle routing, performed by the same homogeneous fleet, and the assignment of inbound/outbound vehicles to dock doors. Nasiri et al. (2018) incorporate supplier selection and order allocation in the VRPCD in a multi-cross-dock system.

In this paper, we consider the coordinated delivery to shopping malls that jointly considers limited docking capacity, endogenous time windows in dock scheduling, and stochastic travel times and service times. We formulate the problem as a two-stage stochastic mixed integer program, develop an ALNS algorithm where the second stage recourse function is approximated by the sample average, and test the in-sample and out-of-sample stability.

3. Problem description and illustration

In this section, we formally describe the coordinated delivery problem to shopping malls with limited docking capacity in Section 3.1 and then provide a small illustrative example in Section 3.2.

3.1. Problem description

An urban shopping mall delivery coordination platform (short as "platform") coordinates the deliveries between a set of logistics service providers (LSPs) $\mathcal{L} = \{1, 2, \dots, L\}$ and a set of shopping malls $\mathcal{M} = \{1, 2, \dots, M\}$. Before the end of each day, LSPs submit the delivery orders that they have received from their customers (tenants in the shopping malls) to the platform. The platform determines the next day delivery routes of these orders and schedules, for each order, the time slot of the docks at the corresponding shopping mall.

We denote the set of orders placed by the customers located at shopping mall $m \in \mathcal{M}$ by $\mathcal{N}^M_m$ and the set of orders that LSP $l \in \mathcal{L}$ receives by $\mathcal{N}^L_l$. The complete set of orders is denoted by $\mathcal{N} = \bigcup_{l\in\mathcal{L}} \mathcal{N}^L_l = \bigcup_{m\in\mathcal{M}} \mathcal{N}^M_m = \{1, 2, \dots, N\}$. The customer who places order $i \in \mathcal{N}$ is located in shopping mall $m_i \in \mathcal{M}$. Each order $i \in \mathcal{N}$ also specifies the demand (delivery quantity) d_i and the time window $[e_i, l_i]$ within which the customer requires the delivery service (unloading at a dock and possibly delivering to the store inside the shopping mall) to start. We assume that an LSP will consolidate the orders located at the same shopping mall with overlapping or close time windows before it submits these orders to the CUDO platform ("off-site consolidation" as in Dalla Chiara and Cheah 2017), such that the time windows of the submitted orders do not overlap.

Each LSP $l \in \mathcal{L}$ owns a fleet of vehicles $\mathcal{K}_l$, each with capacity Q. The complete set of vehicles is denoted by $\mathcal{K} = \bigcup_{l\in\mathcal{L}} \mathcal{K}_l = \{1, 2, \dots, K\}$. Each LSP l can only use its own vehicles to serve orders that it receives ($\mathcal{N}^L_l$). We assume that all the vehicles share a common depot 0. Note that our model in Section 4 and the algorithm in Section 5 can be easily extended to the multi-depot situation where each LSP has a different depot, with very minor changes to the model (on the locations of the depots and the associated distance and travel time matrices).

We define the directed network $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \{0\} \cup \mathcal{M}$ and $\mathcal{A} = \{(i, j) : i, j \in \mathcal{V}\}$. A vehicle traveling on arc $(i, j) \in \mathcal{A}$ incurs a *deterministic* routing cost of $c^{\mathcal{A}}_{ij}$, while the travel time is *stochastic*, described by a random variable $t^{\mathcal{A}}_{ij}$.

On each day, a vehicle can travel multiple trips to deliver orders. The regular working time is T. While working overtime is allowed, it incurs a penalty. We denote the unit penalty cost of overtime working by c^o. A *trip* is a single vehicle tour that departs from the depot, visits customers in the predetermined sequence determined by the platform, and returns to the depot. A *route* is the set of consecutive trips that a vehicle travels during the entire day.

For each order, the platform schedules the time slot of a dock in the shopping mall $m_i \in \mathcal{M}$, denoted by $[y_i, z_i]$. y_i specifies the time when the vehicle that serves order i is allowed to enter shopping mall m_i and to utilize a dock, if available, and z_i specifies the time before which the vehicle should finish the delivery service and after which the vehicle should no longer occupy any dock in the shopping mall.

There are a few practical considerations related to the time slots. First, it makes no senses to schedule the time slot earlier than e_i and therefore y_i should lie in $[e_i, l_i]$, i.e., $e_i \le y_i \le l_i$. While relaxing the requirement and allowing $y_i > l_i$ may lead to a better solution theoretically, we stick to this logical requirement in practice. Second, y_i and z_i are usually a multiple of some discretization time unit τ (e.g., 15 minutes, which is a parameter set by the CUDO platform) for operational convenience. Third, the length of a time slot is usually order-specific and depends on the order's expected service time. Therefore, the length of the time slot $(z_i - y_i)$ can be specified as $U_i \tau$, where $U_i \in \mathbb{N}$ is an order-specific parameter set by the CUDO platform, typically based on the service time distribution.

When making the vehicle routing and dock scheduling decisions, the platform faces two sources of uncertainty: travel time uncertainty of the vehicles traveling in the road network and service time uncertainty of the orders in the shopping malls. These two sources of uncertainty stem from two distinct sample spaces. Denote the sample space of travel times by Ω^T and that of service times by Ω^S, the joint sample space Ω of these two sources of uncertainty can then be denoted by $\Omega^T \times \Omega^S$. A joint scenario is denoted by $\omega = (\omega^T, \omega^S) \in \Omega$, in which a travel time scenario ω^T and a service time scenario ω^S can be separately generated from Ω^T and Ω^S, respectively, with a proper treatment of the correlation between these two sources of uncertainty. Note that in the context of this paper, we may reasonably assume that the travel time uncertainty of the vehicles in the road network and the service time uncertainty of the orders in shopping malls are independent. In the remaining of the paper, we only use the joint sample space Ω, for notation simplicity. For each scenario $\omega \in \Omega$, we denote the travel time realization by $t^{\mathcal{A}}_{ij}(\omega), \forall (i,j) \in \mathcal{A}$, and the service time realization by $\mu_i(\omega), \forall i \in \mathcal{N}$.

At the beginning of the working time on the next day (i.e., when the time of the day is 0), all the vehicles leave the depot and follow the scheduled routes. When a vehicle that serves order $i \in \mathcal{N}$ arrives at shopping mall $m_i \in \mathcal{M}$, the shopping mall manages the vehicle's entrance according to the scheduled time slot $[y_i, z_i]$. If the vehicle arrives earlier than y_i, it has to wait outside the shopping mall (e.g., park on-street) until y_i; otherwise, it can directly enter, even if it arrives later than z_i. When two vehicles, one serving order i and another serving order j (with $y_i > y_j$), both arrive at the shopping mall where these two orders are located earlier than y_j, they both have to wait outside the shopping mall. Vehicle serving order j enters the shopping mall at y_j, earlier than the vehicle serving order i that enters at y_i.

Note that entering the mall does not guarantee the immediate start of the (unloading) service due to the limited docking capacity at each shopping mall. We denote the vehicle arrival time by $a_i(\omega)$. We further denote the vehicle's *outside waiting time* by $w_i(\omega)$, which is $w_i(\omega) = \max\{y_i - a_i(\omega), 0\}$, and its unit penalty cost by c^w. When multiple orders are scheduled the same y (i.e., $y_i = \ldots = y_j$) at a shopping mall, the vehicles enter the shopping mall according to the First-Arrive-First-Enter (FAFE) rule.

Each shopping mall $m \in \mathcal{M}$ has a limited docking capacity, denoted by C_m. If there is an empty dock, the vehicle that enters the shopping mall can start its service immediately (since $y_i \geq e_i$); otherwise, it has to wait in line, where the First-Enter-First-Serve (FEFS) rule applies. We denote the vehicle's *inside waiting time* by $\varpi_i(\omega)$ and its unit penalty cost by c^{ϖ}. Then the service start time is $s_i(\omega) = a_i(\omega) + w_i(\omega) + \varpi_i(\omega)$. If the service starts later than l_i, a penalty cost of c^s is incurred per unit of *late service start time* $\delta_i^s(\omega) = \max\{s_i(\omega) - l_i, 0\}$. Similarly, the service of order i (i.e., the occupation of a dock in the shopping mall) is expected to finish before z_i. If the service finishes after z_i, a penalty cost c^e is incurred per unit of *late service end time* $\delta_i^e(\omega) = \max\{s_i(\omega) + \mu_i(\omega) - z_i, 0\}$.

After serving order i at $s_i(\omega) + \mu_i(\omega)$, the vehicle leaves the shopping mall and travels to the next location (either a shopping mall or the depot). If the vehicle returns to the depot and loads for the next delivery trip, the loading time is assumed to be a constant, μ_0, and we assume that there is no congestion and thus no waiting time at the depot.

Figure 1 is an illustration of time window $[e_i, l_i]$, time slot $[y_i, z_i]$, and the sequence of events for order i under scenario $\omega \in \Omega$. The vehicle arrives early (i.e., $a_i < y_i$) so there is an outside waiting time $w_i(\omega)$. The vehicle starts ($s_i(\omega) > l_i$) and finishes ($s_i(\omega) + \mu_i(\omega) > z_i$) the service late and therefore the corresponding penalties incur.

Figure 2 is an illustration of the First-Arrive-First-Enter (FAFE) rule for vehicle entrance into the shopping mall and the First-Enter-First-Serve (FEFS) rule for dock utilization. In this example, the shopping mall has only one dock. As shown in the upper-right table, orders 5 to 8 (from the same shopping mall) are served by vehicles 1 to 4, respectively. Vehicle 1 arrives after y_5 and the dock is empty upon its arrival, so it can enter the shopping mall and start the service immediately. Both vehicles 2 and 3 arrive earlier than $y_6 = y_7$ so they have to wait outside the shopping mall. At

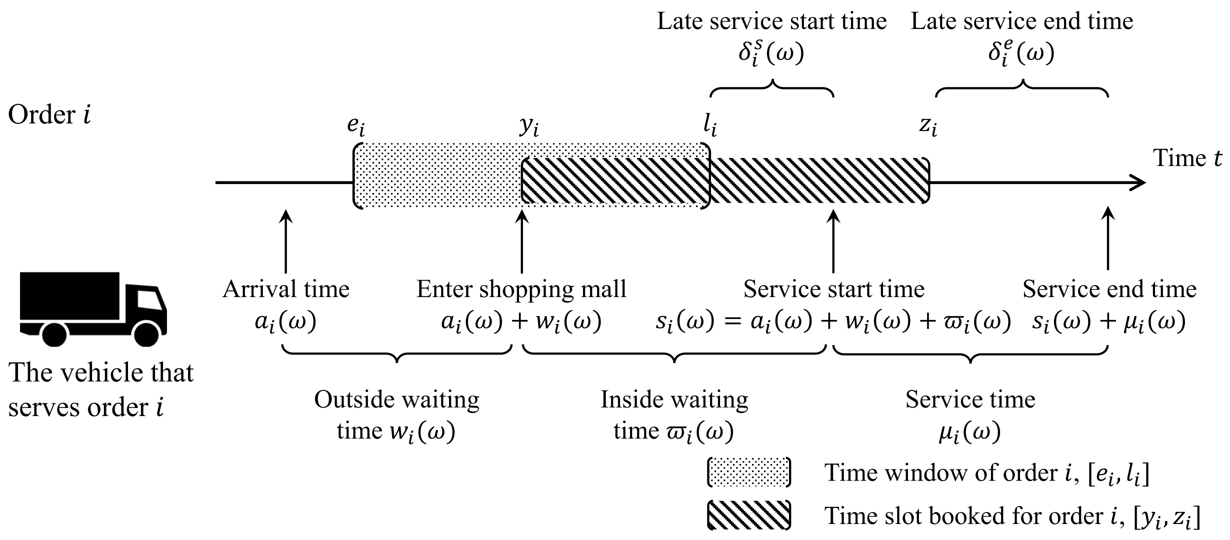
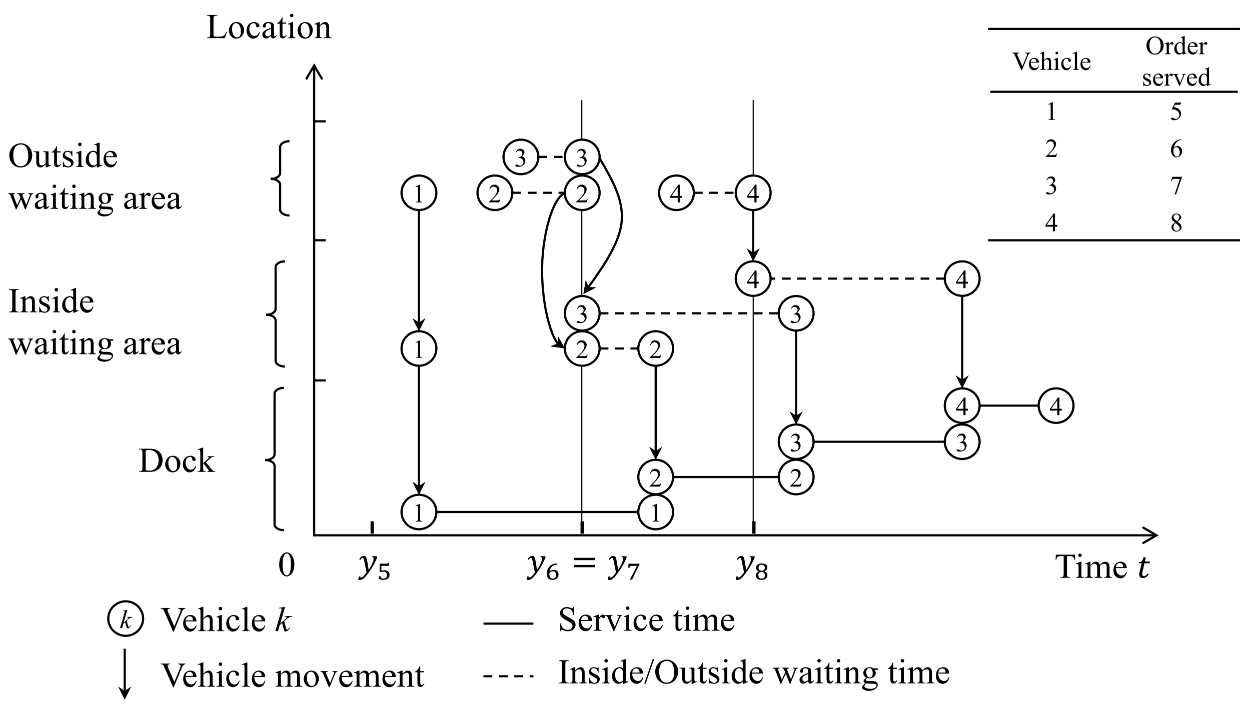

Late service start time $\delta_i^s(\omega)$
Late service end time $\delta_i^e(\omega)$
Order i
e_i
y_i
l_i
z_i
Time t
Arrival time $a_i(\omega)$
Enter shopping mall $a_i(\omega)+w_i(\omega)$
Service start time $s_i(\omega)=a_i(\omega)+w_i(\omega)+\varpi_i(\omega)$
Service end time $s_i(\omega)+\mu_i(\omega)$
The vehicle that serves order i
Outside waiting time $w_i(\omega)$
Inside waiting time $\varpi_i(\omega)$
Service time $\mu_i(\omega)$
Time window of order i, $[e_i, l_i]$
Time slot booked for order i, $[y_i, z_i]$

Figure 1 **An illustration of time window, time slot, and sequence of events under scenario** $\omega \in \Omega$

$y_6 = y_7$, vehicle 2 enters the shopping mall earlier than vehicle 3, as it arrives earlier than vehicle 3 (the FAFE rule). However, vehicle 2 has to wait inside the shopping mall, as the dock is occupied by vehicle 1. When Vehicle 1 finishes the service and the dock is available again, Vehicle 2 starts the service earlier than vehicle 3, as it enters the shopping mall earlier than Vehicle 3 (the FEFS rule).



Vehicle	Order served
1	5
2	6
3	7
4	8

Figure 2 **An illustration of FAFE and FEFS rule in a shopping mall with only one dock**

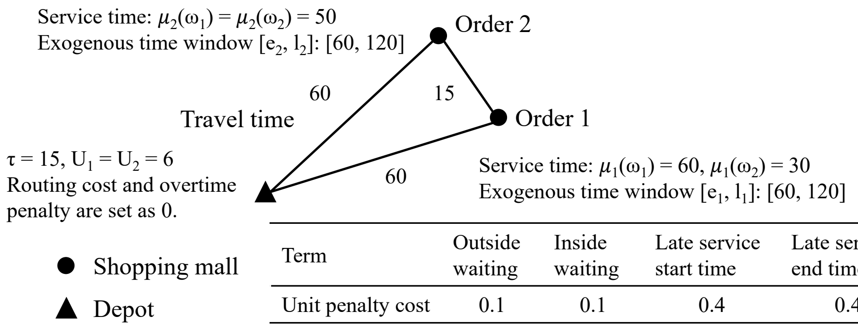
The objective of the platform is to minimize the total cost that consists of the routing cost of the vehicles and the expected penalty cost that includes the outside & inside waiting time and late service start & end time of the orders and the overtime working of the vehicles.

We summarize the notation of sets, data, parameters, and decision variables in Online Appendix A. We also provide a description of the discrete-event simulation to evaluate the total cost, given a solution $\mathcal{S} = \{\boldsymbol{x}, (\boldsymbol{y}, \boldsymbol{z})\}$ and a scenario ω, in Algorithm 5 in Online Appendix B.1.

3.2. An illustrative example

Next, we provide a small example to illustrate that considering uncertainty can make a difference to the visiting sequence of orders (routing) and the dock reservation time slots. We further elaborate the value of considering uncertainty (stochastic solutions) in Section 6.6.

As shown in Figure 3, let us consider two orders served by one LSP with one vehicle. The symmetric travel times between nodes are shown along the arcs. The routing cost is not affected by the visiting sequence of the two orders (shopping malls) and thus can be ignored for simplicity. We also ignore the overtime penalty for illustration simplicity. The exogenous time windows $[e_i, l_i]$ and service time μ_i are shown beside each order (shopping mall). The only source of uncertainty is the service time of Order 1, characterized by two equally likely scenarios ω_1 and ω_2, $\mu_1(\omega_1) = 60$ minutes and $\mu_1(\omega_2) = 30$ minutes. The unit penalty costs of outside & inside waiting and late service start & end are shown in the table.


Service time: $\mu_2(\omega_1) = \mu_2(\omega_2) = 50$
Exogenous time window $[e_2, l_2]$: [60, 120]
Order 2
60
15
Travel time
Order 1
$\tau = 15$, $U_1 = U_2 = 6$
Routing cost and overtime penalty are set as 0.
60
Service time: $\mu_1(\omega_1) = 60$, $\mu_1(\omega_2) = 30$
Exogenous time window $[e_1, l_1]$: [60, 120]
● Shopping mall
▲ Depot

Term	Outside waiting	Inside waiting	Late service start time	Late service end time
Unit penalty cost	0.1	0.1	0.4	0.4

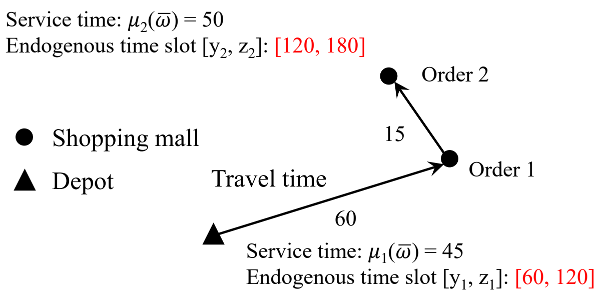
Figure 3 An illustration: Two orders served by one LSP with one vehicle

If we ignore the uncertainty in service times and replace the stochastic service times of Order 1 with the mean value (denoted by the *deterministic* "average" scenario $\bar{\omega}$), i.e., $\mu_1(\bar{\omega}) = 45$ minutes, the solution of the Mean Value Problem (MVP) $\mathcal{S}^{\text{MVP}}$ is shown in Figure 4. In the MVP solution, the vehicle first serves Order 1 and then serves Order 2, and the dock reservation time slots of Order 1 and Order 2 are $[60, 120]$ and $[120, 180]$, respectively. Evaluated under the average scenario $\bar{\omega}$, the total cost of $\mathcal{S}^{\text{MVP}}$ is zero (Figure 4(a)). However, evaluated by the "reality," that is, the scenarios ω_1 and ω_2, the actual expected total cost of $\mathcal{S}^{\text{MVP}}$ is 4.75 (Figure 4(b)).

MVP solution ($\mathcal{S}^{\text{MVP}}$) in the average scenario

Order 2

Scenario	Arrive	Enter	Start service	End service
ϖ	120	120	120	170
Scenario	**Outside waiting**	**Inside waiting**	**Late service start time**	**Late service end time**
ϖ	0	0	0	0


Service time: $\mu_2(\varpi) = 50$
Endogenous time slot $[y_2, z_2]$: [120, 180]
Order 2
15
● Shopping mall
▲ Depot
Travel time
Order 1
60
Service time: $\mu_1(\varpi) = 45$
Endogenous time slot $[y_1, z_1]$: [60, 120]

Order 1

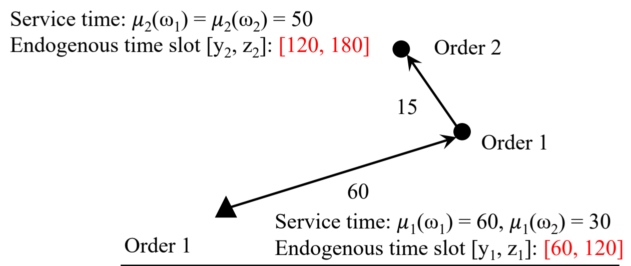
Scenario	Arrive	Enter	Start service	End service
ϖ	60	60	60	105
Scenario	**Outside waiting**	**Inside waiting**	**Late service start time**	**Late service end time**
ϖ	0	0	0	0

(a) Total cost of $\mathcal{S}^{\text{MVP}}$ under average scenario ϖ is 0.

MVP solution ($\mathcal{S}^{\text{MVP}}$) in two scenarios

Order 2

Scenario	Arrive	Enter	Start service	End service
ω_1	135	135	135	185
ω_2	105	120	120	170
Scenario	**Outside waiting**	**Inside waiting**	**Late service start time**	**Late service end time**
ω_1	0	0	15	5
ω_2	15	0	0	0


Service time: $\mu_2(\omega_1) = \mu_2(\omega_2) = 50$
Endogenous time slot $[y_2, z_2]$: [120, 180]
Order 2
15
Order 1
60
Service time: $\mu_1(\omega_1) = 60$, $\mu_1(\omega_2) = 30$
Endogenous time slot $[y_1, z_1]$: [60, 120]

Order 1

Scenario	Arrive	Enter	Start service	End service
ω_1	60	60	60	120
ω_2	60	60	60	90
Scenario	**Outside waiting**	**Inside waiting**	**Late service start time**	**Late service end time**
ω_1	0	0	0	0
ω_2	0	0	0	0

(b) Expected total cost of $\mathcal{S}^{\text{MVP}}$ is 4.75.

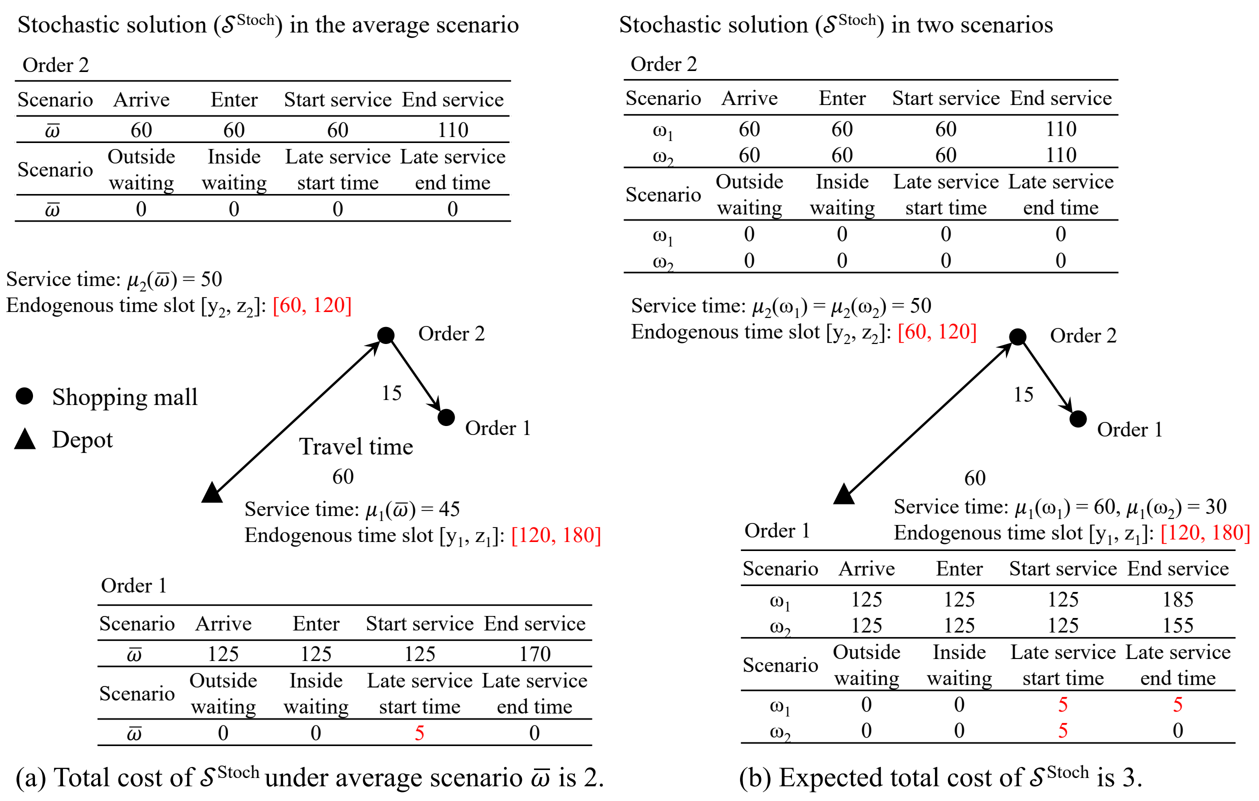
Figure 4 **MVP solution and its performances**

If we consider the uncertainty in the service times of Order 1, the vehicle first serves Order 2 and then serves Order 1, and the dock reservation time slots of Order 1 and Order 2 are $[120, 180]$ and $[60, 120]$, respectively. Evaluated by the average scenario $\bar{\omega}$, the total cost associated with the stochastic solution $\mathcal{S}^{\text{Stoch}}$ is 2 (Figure 5(a)), which is (naturally) worse than that of the MVP solution. However, evaluated by the "reality," that is, the two scenarios ω_1 and ω_2, the actual expected total cost of $\mathcal{S}^{\text{Stoch}}$ is 3 (Figure 5(b)), better than that of the MVP solution (i.e., 4.75).

4. Model formulation

In this section, we model the problem as a two-stage stochastic mixed integer program. In the first stage (1)–(17), we plan the vehicle routes for all the LSPs and schedule the dock time slots at the shopping malls for all the orders. In each second stage subproblem (18)–(44), the first stage solution is evaluated under a realized joint scenario of stochastic travel times and service times and subject to the FAFE and FEFS rules.

To model the multi-trip setting, we extend the network $\mathcal{G}$ to $\bar{\mathcal{G}} = (\bar{\mathcal{V}}, \bar{\mathcal{A}})$. Denote the maximum number of trips that one vehicle can travel during the day by R. Note that R can be either set by experience, or conservatively set as the maximum number of orders that an LSP needs to serve during the day (that is, the number of orders that the LSP receives from its customers). We replicate


Stochastic solution ($\mathcal{S}^{\text{Stoch}}$) in the average scenario
Service time: $\mu_2(\varpi) = 50$
Endogenous time slot $[y_2, z_2]$: [60, 120]
Order 2
● Shopping mall
▲ Depot
15
Order 1
Travel time
60
Service time: $\mu_1(\varpi) = 45$
Endogenous time slot $[y_1, z_1]$: [120, 180]
Stochastic solution ($\mathcal{S}^{\text{Stoch}}$) in two scenarios
Service time: $\mu_2(\omega_1) = \mu_2(\omega_2) = 50$
Endogenous time slot $[y_2, z_2]$: [60, 120]
Order 2
15
Order 1
60
Service time: $\mu_1(\omega_1) = 60$, $\mu_1(\omega_2) = 30$
Endogenous time slot $[y_1, z_1]$: [120, 180]

Stochastic solution ($\mathcal{S}^{\text{Stoch}}$) in the average scenario

Order 2

Scenario	Arrive	Enter	Start service	End service
ϖ	60	60	60	110
Scenario	Outside waiting	Inside waiting	Late service start time	Late service end time
ϖ	0	0	0	0

Order 1

Scenario	Arrive	Enter	Start service	End service
ϖ	125	125	125	170
Scenario	Outside waiting	Inside waiting	Late service start time	Late service end time
ϖ	0	0	5	0

(a) Total cost of $\mathcal{S}^{\text{Stoch}}$ under average scenario ϖ is 2.

Stochastic solution ($\mathcal{S}^{\text{Stoch}}$) in two scenarios

Order 2

Scenario	Arrive	Enter	Start service	End service
ω_1	60	60	60	110
ω_2	60	60	60	110
Scenario	Outside waiting	Inside waiting	Late service start time	Late service end time
ω_1	0	0	0	0
ω_2	0	0	0	0

Order 1

Scenario	Arrive	Enter	Start service	End service
ω_1	125	125	125	185
ω_2	125	125	125	155
Scenario	Outside waiting	Inside waiting	Late service start time	Late service end time
ω_1	0	0	5	5
ω_2	0	0	5	0

(b) Expected total cost of $\mathcal{S}^{\text{Stoch}}$ is 3.

Figure 5 Stochastic solution and its performances

$K(R+1)$ dummy depots. For each vehicle $k \in \mathcal{K}$, its associated dummy depot set is denoted by $\mathcal{N}_k^0 = \{n_{k0}^0, n_{k1}^0, \ldots, n_{kR}^0\}$. Define $\mathcal{N}^0 = \bigcup_{k\in\mathcal{K}} \mathcal{N}_k^0$. Vehicle $k \in \mathcal{K}$ leaves n_{k0}^0 at the beginning of the day and returns to n_{kR}^0 after serving all of its orders. If multiple trips have to be made, vehicle k returns to and leaves from the dummy depots in $\{n_{k1}^0, \ldots, n_{k,R-1}^0\}$ in between n_{k0}^0 and n_{kR}^0. We define the node set $\bar{\mathcal{V}} = \mathcal{N}^0 \cup \mathcal{N}$, where the location of order $i \in \mathcal{N}$ is the same as $m_i \in \mathcal{M}$. In this paper, a "node" can refer to either (the located shopping mall of) an order or a dummy depot. We extend the definition of $\mu_i(\omega)$ to node $i \in \mathcal{N}^0$, such that for any $k \in \mathcal{K}$, $\mu_i(\omega) = 0$ if $i \in \{n_{k0}^0, n_{kR}^0\}$, and $\mu_i(\omega) = \mu_0$ if $i \in \mathcal{N}_k^0 \setminus \{n_{k0}^0, n_{kR}^0\}$. We further define $\bar{\mu}_i$ as the expected service time at node $i \in \bar{\mathcal{V}}$. The arc set is defined as $\bar{\mathcal{A}} = \bigcup_{l\in\mathcal{L}} \bar{\mathcal{A}}_l$, where $\bar{\mathcal{A}}_l = \{(i,j) : i,j \in \mathcal{N}_l^L\} \cup \{(i,j),(j,i) : i \in \mathcal{N}_l^L, j \in \mathcal{N}_k^0, k \in \mathcal{K}_l\}$, containing all the arcs that connect the orders contacted to LSP l ($\mathcal{N}_l^L$) and its (dummy) depots ($\mathcal{N}_k^0, \forall k \in \mathcal{K}_l$). Note that there is no arc between any two dummy depots and any two orders from different LSPs. A vehicle traveling on arc $(i,j) \in \bar{\mathcal{A}}$ incurs a *deterministic* routing cost of $c_{ij}^{\bar{\mathcal{A}}}$, which equals to $c_{ij}^{\mathcal{A}}$ of the corresponding arc $(i,j) \in \mathcal{A}$. Further, on arc $(i,j) \in \bar{\mathcal{A}}$, we define $t_{ij}^{\bar{\mathcal{A}}}(\omega)$ as the travel time under scenario ω, and $\bar{t}_{ij}^{\bar{\mathcal{A}}}$ as the expected travel time.

The routing decision x_{ijk} equals to 1 if vehicle k travels on arc $(i,j) \in \bar{\mathcal{A}}$, and 0 otherwise. v_{ijk} denotes the total load on vehicle k while traveling on arc $(i,j) \in \bar{\mathcal{A}}$. As described in Section 3, the time slot scheduled for order i is $[y_i, z_i]$. We use n_i^Y and U_i to ensure that y_i and z_i are multiples of the time unit τ (a parameter set by the CUDO platform). The first stage problem can be formulated as in (1)–(17).

$$\min_{\boldsymbol{x},(\boldsymbol{y},\boldsymbol{z})} \quad \sum_{(i,j)\in\bar{\mathcal{A}}}\sum_{k\in\mathcal{K}} c_{ij}^{\bar{\mathcal{A}}} x_{ijk} + \mathbb{E}_{\boldsymbol{\xi}}\mathcal{Q}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\boldsymbol{\xi}(\omega)), \tag{1}$$

s.t.

$$\sum_{i\in\mathcal{N}_l^L} x_{n_{k0}^0,i,k} = \sum_{i\in\mathcal{N}_l^L} x_{i,n_{kR}^0,k} = 1, \forall l\in\mathcal{L}, \forall k\in\mathcal{K}_l, \tag{2}$$

$$\sum_{j\in\mathcal{N}_l^L} x_{ijk} = \sum_{j\in\mathcal{N}_l^L} x_{jik} \leq 1, \forall l\in\mathcal{L}, \forall k\in\mathcal{K}_l, \forall i\in\mathcal{N}_k^0\backslash\{n_{k0}^0, n_{kR}^0\}, \tag{3}$$

$$\sum_{k\in\mathcal{K}_l}\sum_{j\in\mathcal{N}_k^0\cup\mathcal{N}_l^L} x_{ijk} = \sum_{k\in\mathcal{K}_l}\sum_{j\in\mathcal{N}_k^0\cup\mathcal{N}_l^L} x_{jik} = 1, \forall l\in\mathcal{L}, \forall i\in\mathcal{N}_l^L, \tag{4}$$

$$\sum_{j\in\mathcal{N}_k^0\cup\mathcal{N}_l^L} (x_{ijk} - x_{jik}) = 0, \forall l\in\mathcal{L}, \forall k\in\mathcal{K}_l, \forall i\in\mathcal{N}_l^L, \tag{5}$$

$$\sum_{k\in\mathcal{K}_l}\sum_{j\in\mathcal{N}_k^0\cup\mathcal{N}_l^L} (v_{jik} - v_{ijk}) = d_i, \forall l\in\mathcal{L}, \forall i\in\mathcal{N}_l^L, \tag{6}$$

$$v_{ijk} \leq Q x_{ijk}, \forall l\in\mathcal{L}, \forall k\in\mathcal{K}_l, \forall (i,j)\in\bar{\mathcal{A}}_l, \tag{7}$$

$$e_i \leq y_i \leq l_i, \forall i\in\mathcal{N}, \tag{8}$$

$$y_i = n_i^Y \tau, \forall i\in\mathcal{N}, \tag{9}$$

$$z_i - y_i = \tau U_i, \forall i\in\mathcal{N}, \tag{10}$$

$$\bar{a}_i + \bar{\mu}_i + \bar{t}_{ij}^{\bar{\mathcal{A}}} \leq \bar{a}_j + M_0(1 - \sum_{k\in\mathcal{K}_l} x_{ijk}), \forall l\in\mathcal{L}, \forall (i,j)\in\bar{\mathcal{A}}_l, \tag{11}$$

$$\bar{a}_{n_{k0}^0} = 0, \forall k\in\mathcal{K}, \tag{12}$$

$$\bar{a}_i \geq 0, \forall i\in\bar{\mathcal{V}}, \tag{13}$$

$$x_{ijk}\in\{0,1\}, \forall l\in\mathcal{L}, \forall (i,j)\in\bar{\mathcal{A}}_l, \forall k\in\mathcal{K}_l, \tag{14}$$

$$y_i, z_i \geq 0, \forall i\in\mathcal{N}, \tag{15}$$

$$n_i^Y \in\mathbb{N}, \forall i\in\mathcal{N}, \tag{16}$$

$$0 \leq v_{ijk} \leq Q, \forall l\in\mathcal{L}, \forall (i,j)\in\bar{\mathcal{A}}_l, \forall k\in\mathcal{K}_l, \tag{17}$$

The objective function (1) includes the total routing cost and the expected second stage cost, including the summation of the penalty cost of the outside & inside waiting time and late service start & end time of each order, as well as the working overtime of each vehicle as calculated in the second-stage problem ((18)–(44)). Note that under the *min* operator in (1), we only include the decisions on the vehicle route $\boldsymbol{x} = \{x_{ijk}\}_{(i,j)\in\bar{\mathcal{A}},k\in\mathcal{K}}$ and the dock time slots $(\boldsymbol{y},\boldsymbol{z}) = \{(y_i,z_i)\}_{i\in\mathcal{N}}$, but omit the induced and auxiliary variables. Note also that we use $[y_i,z_i]$ to represent the *time* *interval* of scheduled dock time slot for order i and use (y_i, z_i) to represent the *decisions* on dock time slot for order i.

Constraints (2) enforce that the route of vehicle $k \in \mathcal{K}$ starts from dummy depot n^0_{k0} and ends at dummy depot n^0_{kR}. Constraints (3) ensure that if vehicle $k \in \mathcal{K}$ returns to a dummy depot (in the multi-trip setting) in $\mathcal{N}^0_k \backslash \{n^0_{k0}, n^0_{kR}\}$, it will leave from the same dummy depot. Constraints (4) enforce that orders contracted to LSP $l \in \mathcal{L}$ ($\mathcal{N}^L_l$) must be served and can only be served by the fleet of LSP l (i.e., $k \in \mathcal{K}_l$). Further, constraints (5) ensure the balance of vehicle flow for each vehicle at each node $i \in \mathcal{N}^L_l$. Constraints (6) ensure the net commodity inflow at node $i \in \mathcal{N}^L_l$ is exactly d_i. Constraints (7) ensure that the vehicle capacity cannot be violated. Constraints (8)–(10) are related to time slot decisions, i.e., $[y_i, z_i], i \in \mathcal{N}$. Constraints (8) enforce that y_i must lie in the time window of order i, i.e., $[e_i, l_i]$. Constraints (9) ensure y_i is multiples of the discrete time unit τ. Constraints (10) enforce that the length of the time slot equals to the predefined order-specific length τU_i (determined by the platform parameter τ and order specific parameter U_i). Constraints (11) are the subtour elimination constraints, where $\bar{a}_j$ is the *expected* arrival time of the vehicle at node j based on the expected arrival time of the vehicle at node i, $\bar{a}_i$, the expected service time at node i, $\bar{\mu}_i$, and the expected travel time on arc (i,j), $\bar{t}^{\bar{\mathcal{A}}}_{ij}$, $\forall (i,j) \in \bar{\mathcal{A}}_l$. Note that the subtour elimination constraints (11) can be replaced by another tighter formulation, that is,

$$\sum_{i,j \in \mathscr{S}, (i,j) \in \bar{\mathcal{A}}_l} x_{ijk} \leq |\mathscr{S}| - 1, \forall l \in \mathcal{L}, \forall k \in \mathcal{K}_l, \forall \mathscr{S} \subseteq (\bigcup_{k \in \mathcal{K}_l} \mathcal{N}^0_k) \cup \mathcal{N}^L_l, |\mathscr{S}| \geq 2.$$

We choose to use constraints (11) to obtain a more compact (polynomial-sized) formulation. Constraints (12) initialize the departure times at $n^0_{k0}, k \in \mathcal{K}$, as the beginning of the day. The variables are defined in (13)–(17).

Given the first stage decisions on the vehicle routes ($\boldsymbol{x}$) and scheduled dock time slots ($[\boldsymbol{y}, \boldsymbol{z}]$), a second stage subproblem ((18)–(44)) tracks the vehicle delivery activities and evaluates the cost $\mathcal{Q}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\xi}(\omega))$, under the realized joint scenario ω of travel times and service times, represented by vector $\boldsymbol{\xi}(\omega)$.

$$\mathcal{Q}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\xi}(\omega)) = \min_{(\boldsymbol{w}, \boldsymbol{\varpi}, \boldsymbol{\delta}^s, \boldsymbol{\delta}^e), \boldsymbol{\delta}^o} \quad \sum_{i \in \mathcal{N}} [c^w w_i(\omega) + c^{\varpi} \varpi_i(\omega) + c^s \delta^s_i(\omega) + c^e \delta^e_i(\omega)] + \sum_{k \in \mathcal{K}} [c^o \delta^o_k(\omega)] \tag{18}$$

s.t.

$$s_i(\omega) = a_i(\omega) + w_i(\omega) + \varpi_i(\omega), \forall i \in \bar{\mathcal{V}}, \tag{19}$$

$$\begin{aligned} M_0(\sum_{k \in \mathcal{K}_l} x_{ijk} - 1) &\leq s_i(\omega) + \mu_i(\omega) + t^{\bar{\mathcal{A}}}_{ij}(\omega) - a_j(\omega) \\ &\leq M_0(1 - \sum_{k \in \mathcal{K}_l} x_{ijk}), \forall l \in \mathcal{L}, \forall (i,j) \in \bar{\mathcal{A}}_l, \end{aligned} \tag{20}$$

$$w_i(\omega) \leq M_0[1-\varphi_i(\omega)], \forall i \in \mathcal{N}, \tag{21}$$

$$0 \leq a_i(\omega) + w_i(\omega) - y_i \leq M_0\varphi_i(\omega), \forall i \in \mathcal{N}, \tag{22}$$

$$-M_0\varsigma_{ij}(\omega) \leq a_i(\omega) - a_j(\omega) \leq M_0[1-\varsigma_{ij}(\omega)], \forall m \in \mathcal{M}, \forall i,j \in \mathcal{N}_m^M, i<j, \tag{23}$$

$$\begin{aligned}-M_0\zeta_{ij}(\omega) &\leq [a_i(\omega)+w_i(\omega)] - [a_j(\omega)+w_j(\omega)]\\ &\leq M_0[1-\zeta_{ij}(\omega)], \forall m \in \mathcal{M}, \forall i,j \in \mathcal{N}_m^M, i<j,\end{aligned} \tag{24}$$

$$-M_0\chi_{ij} + \epsilon_0 \leq y_i - y_j \leq M_0(1-\chi_{ij}), \forall m \in \mathcal{M}, \forall i,j \in \mathcal{N}_m^M, i<j, \tag{25}$$

$$-M_0\kappa_{ij} + \epsilon_0 \leq y_j - y_i \leq M_0(1-\kappa_{ij}), \forall m \in \mathcal{M}, \forall i,j \in \mathcal{N}_m^M, i<j, \tag{26}$$

$$\chi_{ij} + \kappa_{ij} - 2 \leq \varsigma_{ij}(\omega) - \zeta_{ij}(\omega) \leq 2 - \chi_{ij} - \kappa_{ij}, \forall i,j \in \mathcal{N}_m^M, i<j, \tag{27}$$

$$-M_0\varrho_{ij}(\omega) \leq s_i(\omega) - s_j(\omega) \leq M_0[1-\varrho_{ij}(\omega)], \forall m \in \mathcal{M}, \forall i,j \in \mathcal{N}_m^M, i<j, \tag{28}$$

$$\zeta_{ij}(\omega) = \varrho_{ij}(\omega), \forall m \in \mathcal{M}, \forall i,j \in \mathcal{N}_m^M, i<j, \tag{29}$$

$$\sum_{j \in \mathcal{N}_m^M \cup \{n_m^s\}} \sigma_{ji}(\omega) = \sum_{j \in \mathcal{N}_m^M \cup \{n_m^d\}} \sigma_{ij}(\omega) = 1, \forall m \in \mathcal{M}, \forall i \in \mathcal{N}_m^M, \tag{30}$$

$$\sum_{j \in \mathcal{N}_m^M} \sigma_{n_m^s,j}(\omega) \leq C_m, \forall m \in \mathcal{M}, \tag{31}$$

$$\varpi_i(\omega) \leq M_0(1-\sigma_{n_m^s,i}), \forall m \in \mathcal{M}, \forall i \in \mathcal{N}_m^M, \tag{32}$$

$$\varpi_i(\omega) \leq M_0\phi_i(\omega), \forall i \in \mathcal{N}, \tag{33}$$

$$\begin{aligned}&M_0[\sigma_{ij}(\omega)-1] \leq s_j(\omega) - [s_i(\omega)+\mu_i(\omega)] \leq M_0(2-\phi_j(\omega)-\sigma_{ij}(\omega)),\\ &\qquad \forall m \in \mathcal{M}, \forall i,j \in \mathcal{N}_m^M,\end{aligned} \tag{34}$$

$$s_i(\omega) - \delta_i^s(\omega) \leq l_i, \forall i \in \mathcal{N}, \tag{35}$$

$$s_i(\omega) + \mu_i(\omega) - \delta_i^e(\omega) \leq z_i, \forall i \in \mathcal{N}, \tag{36}$$

$$a_{n_{kR}^0}(\omega) - \delta_k^o(\omega) \leq T, \forall k \in \mathcal{K}, \tag{37}$$

$$a_{n_{k0}^0}(\omega) = 0, \forall k \in \mathcal{K}, \tag{38}$$

$$w_i(\omega) = \varpi_i(\omega) = 0, \forall i \in \mathcal{N}^0, \forall \omega \in \Omega, \tag{39}$$

$$a_i(\omega), s_i(\omega), w_i(\omega), \varpi_i(\omega) \geq 0, \varphi_i(\omega) \in \{0,1\}, \forall i \in \bar{\mathcal{V}}, \tag{40}$$

$$\sigma_{ij}(\omega) \in \{0,1\}, \forall m \in \mathcal{M}, \forall i \in \{n_m^s\} \cup \mathcal{N}_m^M, \forall j \in \{n_m^d\} \cup \mathcal{N}_m^M, \tag{41}$$

$$\chi_{ij}, \kappa_{ij}, \zeta_{ij}(\omega), \varrho_{ij}(\omega), \phi_{ij}(\omega) \in \{0,1\}, \forall m \in \mathcal{M}, \forall i,j \in \mathcal{N}_m^M, i<j, \tag{42}$$

$$\delta_i^e(\omega), \delta_i^s(\omega) \geq 0, \forall i \in \mathcal{N}, \tag{43}$$

$$\delta_k^o(\omega) \geq 0, \forall k \in \mathcal{K}. \tag{44}$$

The objective function (18) calculates the total penalty cost of outside & inside waiting time and late service start & end time of each order $i \in \mathcal{N}$, as well as the working overtime of each vehicle $k \in \mathcal{K}$. Note that under the *min* operator in (18), we only include the main second stage variables $(\boldsymbol{w}, \boldsymbol{\varpi}, \boldsymbol{\delta}^s, \boldsymbol{\delta}^e) = \{(w_i, \varpi_i, \delta_i^s, \delta_i^e)\}_{i\in\mathcal{N}}$ and $\boldsymbol{\delta}^o = \{\delta_k^o\}_{k\in\mathcal{K}}$, but omit the induced and auxiliary variables.

Constraints (19) define the service start time of the vehicle to serve node $i \in \bar{\mathcal{V}}$ as a summation of arrival time, outside waiting time, and inside waiting time. Note that if i is a dummy depot (i.e., $i \in \mathcal{N}^0$), both inside and outside waiting time is 0 as specified in constraints (39). Constraints (20) calculate the arrival times. If a vehicle k serves node j right after node i (that is, $x_{ijk} = 1$), then the vehicle's arrival time (outside the shopping mall m_j to serve order j or at the dummy depot j) is the summation of service start time of node i, service time of node i, and the travel time between nodes i and j. That is, $a_j(\omega) = s_i(\omega) + \mu_i(\omega) + t_{ij}^{\bar{\mathcal{A}}}(\omega)$.

Constraints (21)–(22) calculate the outside waiting time $w_i(\omega)$ for an order $i \in \mathcal{N}$, that is, $w_i(\omega) = \max\{y_i - a_i(\omega), 0\}$. The auxiliary binary variable $\varphi_i(\omega)$ helps linearize this maximization function. When $y_i - a_i(\omega) > 0$, $w_i(\omega) > 0$ because of the first inequalities in constraints (22). Then $\varphi_i(\omega) = 0$ because of constraints (21). We have $w_i(\omega) = y_i - a_i(\omega)$ as a result of the first and second inequalities in constraints (22). When $y_i - a_i(\omega) < 0$, $\varphi_i(\omega) = 1$ because of the second inequalities in constraints (22). We have $w_i(\omega) = 0$ as a result of constraints (21). When $y_i - a_i(\omega) = 0$, $w_i(\omega) = 0$ because of the second inequalities in Constraints (21) and (22).

Constraints (23)–(27) enforce the First-Arrive-First-Enter (FAFE) rule, when multiple orders in the same shopping mall m share the same y (e.g., $i, j \in \mathcal{N}_m^M$ and $y_i = y_j$). In Constraints (23), the binary variable $\varsigma_{ij}(\omega) = 1$ indicates that the vehicle to serve order i arrives at the shopping mall no later than the vehicle to serve order j (i.e., $a_i(\omega) \le a_j(\omega)$). In constraints (24), the binary variable $\zeta_{ij}(\omega) = 1$ indicates that the vehicle to serve order i enters the shopping mall no later than the vehicle to serve order j (i.e., $a_i(\omega) + w_i(\omega) \le a_j(\omega) + w_j(\omega)$). Constraints (25) ensure that $\chi_{ij} = 1$ if $y_i \le y_j$, and $\chi_{ij} = 0$ otherwise. Similarly, constraints (26) ensure that $\kappa_{ij} = 1$ if $y_i \ge y_j$ and $\kappa_{ij} = 0$ otherwise. Note that we use a small positive number ϵ_0 to ensure that $\chi_{ij} = 1$ if $y_i = y_j$ in Constraints (25) and $\kappa_{ij} = 1$ if $y_i = y_j$ in Constraints (26). Constraints (25) and (26) jointly ensure that $\chi_{ij} = \kappa_{ij} = 1$ if and only if $y_i = y_j$, which in turn leads to $\varsigma_{ij}(\omega) = \zeta_{ij}(\omega)$ in constraints (27).

Constraints (28)–(29) enforce the First-Enter-First-Serve (FEFS) rule. In constraints (28), the binary variable $\varrho_{ij}(\omega) = 1$ indicates that the service start time of order i is no later than the service start time of order j (i.e., $s_i(\omega) \le s_j(\omega)$). Constraints (29) ensure that the vehicle enters the shopping mall earlier starts the unloading service earlier.

To model the docking capacity constraints, for each shopping mall $m \in \mathcal{M}$, we introduce a dummy source order n_m^s and a dummy destination order n_m^d. There can be at most C_m paths connecting n_m^s and n_m^d. Each path represents a sequence of orders in $\mathcal{N}_m^M$ to be served by the same dock. We define the binary variable $\sigma_{ij}(\omega) = 1$ if order i is followed by order j on a path. As an illustration, in Figure 6, orders i and j use dock 1. We have $\sigma_{n_m^s,i}(\omega) = \sigma_{ij}(\omega) = \sigma_{j,n_m^d}(\omega) = 1$ in the path. Constraints (30) ensure that the in-degree and out-degree of each order $i \in \mathcal{N}$ equal to one. Constraints (31) are the docking capacity constraints, which enforce the number of the paths that start from n_m^s in shopping mall $m \in \mathcal{M}$ is no more than C_m. Note that we do not enforce the docking capacity constraints in the first stage model, which means we allow "overbooking" of the docks, considering the uncertainty in travel and service times. However, the docking capacity constraints are respected by constraints (30)–(31) and are handled together with the FAFE and FEFS rules in the second stage model. The first stage vehicle routing and dock scheduling solution will be evaluated by the second stage recourse function on the expected penalty costs. Therefore, a poorly performed first stage solution is very unlikely (subject to statistical evaluation errors) to be selected as the optimal solution.

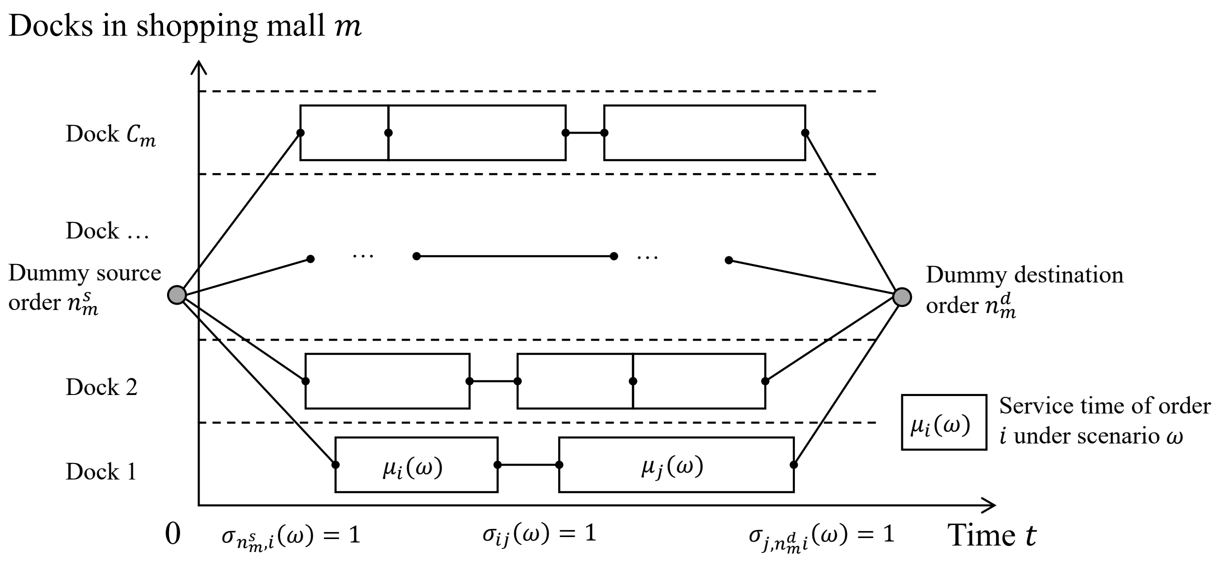

Docks in shopping mall m
Dock C_m
Dock …
Dummy source order n_m^s
Dock 2
Dock 1
Dummy destination order n_m^d
$\mu_i(\omega)$
Service time of order i under scenario ω
$\mu_j(\omega)$
0
$\sigma_{n_m^s,i}(\omega) = 1$
$\sigma_{ij}(\omega) = 1$
$\sigma_{j,n_m^d i}(\omega) = 1$
Time t

Figure 6 An illustration of dummy source/destination order, path, and $\sigma_{ij}(\omega)$ under scenario $\omega \in \Omega$

Constraints (32)–(34) calculate the inside waiting time $\varpi_j(\omega)$ for order $j \in \mathcal{N}$. Constraints (32) specify that for the first order served at each dock, there is no inside waiting time. The binary variable $\phi_j(\omega) = 1$ if $\varpi_j(\omega) > 0$, as in constraints (33). When the vehicle to serve order j follows the vehicle to serve order i at the same dock (i.e., $\sigma_{ij}(\omega) = 1$), $s_j(\omega) - [s_i(\omega) + \mu_i(\omega)] > 0$ and $\varpi_j(\omega) > 0$ can not hold simultaneously. This can be enforced by constraints (33) and the second inequalities of constraints (34). The first inequalities of constraints (34) ensure that $s_j(\omega) \geq s_i(\omega) + \mu_i(\omega)$ when $\sigma_{ij}(\omega) = 1$.

Constraints (35) calculate the late service start time, $\delta_i^s(\omega) = \max\{s_i(\omega) - l_i, 0\}, \forall i \in \mathcal{N}$. Constraints (36) calculate the late service end time, $\delta_i^e(\omega) = \max\{s_i(\omega) + \mu_i(\omega) - z_i, 0\}, \forall i \in \mathcal{N}$. Constraints (37) calculate the overtime working, $\delta_k^o(\omega) = \max\{a_{n_{kR}^0}(\omega) - T, 0\}, \forall k \in \mathcal{K}$. Constraints (38) initialize the arrival time for each vehicle as time 0. Constraints (39) set both inside and outside waiting time as 0 at dummy depots. Decision and auxiliary variables are defined in (40)–(44).

When a vehicle serves two orders located at the same shopping mall, the service end time of the first order may be within or close to the scheduled dock time slot of the second order, due to travel and service time stochasticity. In practice, the vehicle may very likely stay inside the shopping mall and continue to serve the second order after serving the first order. However, the vehicle has to "exit and re-enter" the shopping mall to serve the second order, based on the formulation of the second stage subproblem ((18)–(44)). The non-overlapping time window assumption (see Section 3.1) does not eliminate all possible "exit and re-enter" cases. Essentially, the second stage subproblem in a two-stage stochastic program is to (approximately) evaluate the effect of the first stage solution (i.e., vehicle routing and dock scheduling decisions in this paper). Thus, our current treatment serves as a reasonable representation (or approximation) of the key characteristics of the problem under study.

Furthermore, we introduce the symmetry breaking constraints (45)–(47) to strengthen the model.

$$\sum_{(i,j)\in\bar{\mathcal{A}}} c_{ij}^{\bar{\mathcal{A}}} x_{ijk} \le \sum_{(i,j)\in\bar{\mathcal{A}}} c_{ij}^{\bar{\mathcal{A}}} x_{ij,k+1}, \forall l \in \mathcal{L}, \forall k \in \{k_l, k_l+1, \dots, k_l+|\mathcal{K}_l|-2\}, \tag{45}$$

$$\sum_{i\in\mathcal{N}_l^L} x_{n_{k,r+1}^0,i,k} \le \sum_{i\in\mathcal{N}_l^L} x_{n_{kr}^0,i,k}, \forall k\in\mathcal{K}, r\in\{0,1,\dots,R-2\}, \tag{46}$$

$$\bar{a}_{n_{kr}^0} \le \bar{a}_{n_{k,r+1}^0}, \forall k\in\mathcal{K}, r\in\{0,1,\dots,R-1\}. \tag{47}$$

Specifically, constraints (45) ensure that the route with a smaller routing cost is assigned to the vehicle with a smaller index, where k_l is the smallest index in $\mathcal{K}_l$. Constraints (46) ensure that dummy depot $r+1$ is used only if dummy depot r is used (except when $r = R-1$). Constraints (47) ensure that dummy depot r is visited before $r+1$.

In Online Appendix A, we summarize the notation of all the decision variables in the two-stage stochastic model.

5. Solution method

The two-stage stochastic mixed integer program in Section 4 has a large number of binary integer variables in both stages ((1)–(17) and (18)–(44)). Solving this model to optimality is computationally intractable. In this section, we present an Adaptive Large Neighborhood Search (ALNS) algorithm that can be utilized to solve realistic size instances. The framework of the ALNS is described in Section 5.1, and its components are explained in Sections 5.3 and 5.4.

5.1. ALNS framework

First proposed by Ropke and Pisinger (2006), the ALNS is a metaheuristic that extends the Large Neighborhood Search (LNS) algorithm. The basic idea of the ALNS is that, in each iteration, a removal (destruction) and an insertion (construction) operator are selected from a set of operators by a roulette wheel mechanism with adaptive weights, and these two selected operators apply on the current solution to find a new solution. The new solution is then evaluated by the objective function and the weights of the two selected operators are adjusted based on its quality. Thus, operators that result in solutions of better quality are assigned with more weights and are more likely to be selected in the subsequent iterations. Our proposed ALNS algorithm is in Algorithm 1.

Algorithm 1: The ALNS algorithm

```
   Input: an instance, algorithm parameter settings, and the sample $\Omega^{eval}$.
   Output: a solution $\mathcal{S}^*$ and $F(\mathcal{S}^*, \Omega^{eval})$.
 1 Randomly generate a sample $\Omega^{alns}$.
 2 Initialize the weights and scores of operators.
 3 Generate an initial solution $\mathcal{S}^0$, $\mathcal{S}^* \Leftarrow \mathcal{S}^0$, $n \Leftarrow 0$. Evaluate solution $\mathcal{S}^0$ by $\Omega^{alns}$ (see Section 5.3), i.e.,
     evaluate $F(\mathcal{S}^0, \Omega^{alns})$.
 4 while the stopping criterion is not reached do
 5     Use a roulette wheel selection principle (based on operators' weights, see Section 5.2) to select a removal
         operator, an insertion operator, and a scheduling operator.
 6     Randomly generate a sample $\Omega^{oper}$.
 7     Apply the selected removal operator, insertion operator, and scheduling operator (see Section 5.4) in
         sequence on solution $\mathcal{S}^n$ to get $\mathcal{S}'$ ($\Omega^{oper}$ is used to evaluate the neighborhood solutions).
 8     Evaluate $F(\mathcal{S}', \Omega^{alns})$ (see Section 5.3).
 9     if $F(\mathcal{S}', \Omega^{alns}) < F(\mathcal{S}^*, \Omega^{alns})$ then $\mathcal{S}^* \Leftarrow \mathcal{S}'$, $\mathcal{S}^{n+1} \Leftarrow \mathcal{S}'$.
10     else if the accepting criteria (see Section 5.2) are reached then $\mathcal{S}^{n+1} \Leftarrow \mathcal{S}'$.
11     else $\mathcal{S}^{n+1} \Leftarrow \mathcal{S}^n$.
12     Update the scores of selected operators.
13     if $mod(n, N^{seg}) = 0$ then update the weights of all the operators (see Section 5.2).
14     $n \Leftarrow n + 1$.
15 Evaluate $F(\mathcal{S}^*, \Omega^{eval})$.
16 return $\mathcal{S}^*$ and $F(\mathcal{S}^*, \Omega^{eval})$.
```

Given a first stage solution $\mathcal{S} = \{\boldsymbol{x}, (\boldsymbol{y}, \boldsymbol{z})\}$, we approximate the expectation in $\mathcal{F}(\mathcal{S}, \Omega) = f_1(\mathcal{S}) + \mathbb{E}_{\omega\in\Omega}[f_2(\mathcal{S}, \omega)]$ by the sample average under Ω' ($\Omega' \subset \Omega$), i.e., $F(\mathcal{S}, \Omega') = f_1(\mathcal{S}) + 1/|\Omega'| \cdot \sum_{\omega\in\Omega'} f_2(\mathcal{S}, \omega)$. $f_1(\mathcal{S})$ is the first stage cost of solution $\mathcal{S}$ (equivalent to the first item in equation (1)) and $f_2(\mathcal{S}, \omega)$ is the second stage cost of solution $\mathcal{S}$ under scenario $\omega \in \Omega'$ (equivalent to $\mathcal{Q}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\xi}(\omega))$ as in equation (18)). There are three sizes of Ω', i.e., Ω^{eval}, Ω^{alns}, and Ω^{oper}, which we will discuss in more details in Section 5.3.

In the *initialization phase* (Lines 1–3), we first randomly generate a sample $\Omega^{alns} \subset \Omega$ (Line 1). Then for each operator, we initialize its weight as one and score as zero (Line 2). To construct an initial solution $\mathcal{S}^0$ (Line 3), we create an empty route for each vehicle and set the time slot of order $i \in \mathcal{N}$ as $[y_i, z_i] = [e_i, e_i + U_i\tau]$. Then we apply an insertion operator (greedy insertion) to generate the initial routes, apply a scheduling operator (intra-route scheduling) to adjust time slots. $\mathcal{S}^0$ is then evaluated with sample Ω^{alns} and stored as the incumbent solution $\mathcal{S}^*$. Note that the sample Ω^{alns}, once generated (Line 1), is fixed in the entire Algorithm 1. Our exploratory numerical study suggests that, compared with the ALNS with a random Ω^{alns} in each iteration, the ALNS with a fixed Ω^{alns} spends less time in each iteration and can produce solutions with better objective function values within the same computational time.

The solution $\mathcal{S}^n$ is iteratively improved in the while loop (Lines 4–14) until reaching the *stopping criterion*, when the incumbent solution $\mathcal{S}^*$ is evaluated by Ω^{eval} and returned (Lines 15–16). In the practice of the CUDO platform, the computational time is limited to one hour, which serves as the stopping criterion.

In each iteration within the while loop, we first generate a new solution $\mathcal{S}^{'}$ from the current solution $\mathcal{S}^n$ (Lines 5–7). Specifically, we use the *roulette wheel selection principle* to select a removal, an insertion, and a scheduling operator (Line 5), and apply the selected operators sequentially. A sample Ω^{oper} is generated (Line 6) which is needed by some local search operators to evaluate the neighboring solutions.

In Lines 8–13, we evaluate the newly generated solution $\mathcal{S}'$ and update the scores and weights of the operators. We first evaluate $F(\mathcal{S}', \Omega^{alns})$ (Line 8). If $\mathcal{S}'$ improves from $\mathcal{S}^*$, we update both the incumbent solution $\mathcal{S}^*$ and $\mathcal{S}^{n+1}$ with $\mathcal{S}'$ (Line 9); otherwise, we decide whether accepting $\mathcal{S}'$ to update $\mathcal{S}^{n+1}$ using the *simulated annealing acceptance criteria* (Lines 10–11). We update the scores of the selected operators based on the quality of $\mathcal{S}'$ in each iteration (Line 12) and the weights of all operators at the end of each segment (Line 13). The various conditions and their corresponding scores are listed in Table 3.

Note that we divide the total number of iterations into a series of consecutive *segments*, indexed by j $(j = 1, 2, \ldots)$, each of which consists of N^{seg} iterations.

Table 3 Scores under different conditions

Score	Situation
π_1	Solution $\mathcal{S}'$ is a new best solution.
π_2	Solution $\mathcal{S}'$ is a new solution that has not been accepted before, and is worse than $\mathcal{S}^*$ but better than Solution $\mathcal{S}^n$.
π_3	Solution $\mathcal{S}'$ is a new solution that has not been accepted before, and is worse than $\mathcal{S}^n$ but reaches accepting criteria.

5.2. ALNS details

Some details in Algorithm 1 are described below.

Roulette wheel principle (Line 5). We denote the three sets of operators by O_1 (removal), O_2 (insertion), and O_3 (scheduling) and g_{ij} by the *weight* of operator i in segment j. In each iteration in segment j, we select an operator $i \in O_k$ $(k = 1, 2, 3)$ with the probability of $g_{ij} / \sum_{i \in O_k} g_{ij}$.

Simulated annealing acceptance criteria (Lines 10–11). Given $\mathcal{S}^n$, we accept the new generated solution $\mathcal{S}'$ with the probability of $\min\{1, e^{-[F(\mathcal{S}', \Omega^{alns}) - F(\mathcal{S}^n, \Omega^{alns})]/\mathcal{T}^n}\}$, where $\mathcal{T}^n$ is the *temperature*. The temperature is initialized as $\mathcal{T}^0$ and decreases after every iteration by multiplying a *cooling rate* θ $(\theta < 1)$, i.e., $\mathcal{T}^n = \mathcal{T}^{n-1} \cdot \theta$. We set the initial temperature $\mathcal{T}_0$ such that a solution $\mathcal{S}$ with $F(\mathcal{S}, \Omega^{alns}) = (1 + \epsilon) F(\mathcal{S}^0, \Omega^{alns})$ has a 50% of probability to be accepted, and set the cooling rate θ such that after γ iterations, as the temperature decreases, this solution $\mathcal{S}$ has a 0.001% of probability to be accepted. In this way, $\mathcal{T}_0$ and θ can be decided by ϵ and γ (Hof, Schneider, and Goeke 2017). More specifically, $\mathcal{T}_0 = \epsilon \cdot F(\mathcal{S}^0, \Omega^{alns}) / \ln(2)$, $\theta = [\ln(2)/\ln(100{,}000)]^{1/\gamma}$.

Update the weights of operators (Line 13). We update g_{ij} at the end of segment $j - 1$ as

$$g_{ij} = \begin{cases} g_{i,j-1}, & \text{if } q_{i,j-1} = 0, \\ (1-\rho) g_{i,j-1} + \rho \cdot \kappa_{i,j-1}/q_{i,j-1}, & \text{o.w.}, \end{cases} \tag{48}$$

where $q_{i,j-1}$ is the number of iterations that operator i is selected in segment $j - 1$. If operator i is not selected in segment $j - 1$ (i.e., $q_{i,j-1} = 0$), g_{ij} remains the same as $g_{i,j-1}$; otherwise, we update g_{ij} based on the *reaction factor* $\rho \in [0, 1]$ and the accumulated scores earned by operator i in segment $j - 1$, denoted by $\kappa_{i,j-1}$.

5.3. Solution evaluation

Given a first stage solution $\mathcal{S}$ and a sample $\Omega' \subset \Omega$, the evaluation of $F(\mathcal{S}, \Omega') = f_1(\mathcal{S}) + 1/|\Omega'| \cdot \sum_{\omega \in \Omega'} f_2(\mathcal{S}, \omega)$ requires solving a second stage mixed integer program ((18)–(44)) under each scenario $\omega \in \Omega'$, which is often computationally challenging. We overcome this challenge by using different sample sizes at different levels of the ALNS, evaluating $f_2(\mathcal{S}, \omega)$ by a discrete-event simulation, and approximating the change of $f_2(\mathcal{S}', \omega) - f_2(\mathcal{S}, \omega)$ by only considering the direct impact of the altered order in each step of a local search operator.

Given the budget on the total computational time, we face the trade-off between the accuracy in evaluation of a certain solution and the number of solutions we are able to explore (and evaluate). Note that in the ALNS, Ω^{eval} is only used once to evaluate the "true" objective of the final solution (Line 15), whereas Ω^{alns} is used to evaluate the newly generated solution in each iteration (Line 8), and Ω^{oper} is used to evaluate the solutions explored by the neighborhood search operators in each iteration (Line 7). Intuitively, we set $|\Omega^{eval}| \gg |\Omega^{alns}| > |\Omega^{oper}|$, where an appropriate size of $|\Omega^{alns}|$ is chosen by considering its in-sample and out-of-sample stability (Kaut and Wallace 2007).

Given a first stage solution $\mathcal{S}$, in the second stage, each vehicle serves the orders in the pre-determined sequence and each shopping mall manages the vehicles and docks according to the vehicles' arrival time and the scheduled time slots, based on the First-Arrive-First-Enter and First-Enter-First-Serve rules. Hence, instead of solving the mixed integer program as in (18)–(44), we can calculate $f_2(\mathcal{S},\omega)$ by simulating all the events and bookkeeping the resulted arrival times, inside/outside waiting times, late service start/end times, etc. The details of this discrete-event simulation algorithm for calculating $f_2(\mathcal{S},\omega)$ are presented in Algorithm 5 in Online Appendix B.1. The worst case time complexity of Algorithm 5 is $O(N \log K)$, where N is the number of orders and K is the number of vehicles.

When we apply some operators (worst removal, greedy insertion, regret insertion, intra-route scheduling, and intra-shopping-mall scheduling), we need to extensively search and evaluate a large number of solutions in the neighborhood of the current solution $\mathcal{S}$, denoted by $\delta(\mathcal{S})$. Note that only one order i is altered in each step of a removal, insertion, or scheduling operator. Thus, we may use $\Delta G(\mathcal{S},\mathcal{S}',i,\omega,B(\mathcal{S},\omega))$ to approximate the change in the objective, i.e., $f_2(\mathcal{S}',\omega) - f_2(\mathcal{S},\omega)$, rather than calculating $f_2(\mathcal{S}',\omega)$ from scratch. For a removal operator, i is the successor of the order that has been removed from $\mathcal{S}$. For an insertion operator, i is the order that has been inserted to $\mathcal{S}$. For a scheduling operator, i is the order whose time slot (y_i and z_i) has been rescheduled. $B(\mathcal{S},\omega) = \{\boldsymbol{a}(\omega), \boldsymbol{w}(\omega), \boldsymbol{\varpi}(\omega), \boldsymbol{H}(\omega)\}$ is the output of the discrete-event simulation algorithm when evaluating solution $\mathcal{S}$ under scenario ω, where $\boldsymbol{a}(\omega)$, $\boldsymbol{w}(\omega)$, and $\boldsymbol{\varpi}(\omega)$ are the arrival, outside waiting, and inside waiting times of all the orders, and $\boldsymbol{H}(\omega)$ is the sorted array indicating the entering sequence of all the orders.

The details of the algorithm to calculate $\Delta G(\mathcal{S},\mathcal{S}',i,\omega,B(\mathcal{S},\omega))$ are presented in Algorithm 6 in Online Appendix B.2. While the calculation of $f_2(\mathcal{S},\omega)$ in Algorithm 5 in Online Appendix B.1 requires simulating all the events, the approximation in Algorithm 6 only considers the impact on the orders 1) that are located in the same shopping mall where order i is located and are served by the vehicles that enter the mall before $l_i + U_i\tau$ (the largest possible value that z_i can be) and 2) that are served after order i on the same vehicle. The worst case time complexity of Algorithm 6 is $O(N)$.

5.4. Local search operators

In this section, we describe ten removal operators, two insertion operators, and two scheduling operators.

Removal operators: (1) random, (2) Shaw, (3) worst, (4) shopping mall, (5) vehicle, (6) trip, (7) outside waiting time, (8) inside waiting time, (9) time window violation, and (10) time slot violation. The first six removal operators are adapted from common removal operators in the ALNS literature (Ropke and Pisinger 2006, Azi, Gendreau, and Potvin 2014), while the last four removal operators remove orders based on a certain cost term, as suggested by the names of these operators.

A general outline of the removal operators is presented in Algorithm 2. A removal operator takes parameters $\overline{\psi}$ and $\underline{\psi}$ as inputs. In Line 1, the target number of orders to be removed (denoted by q) is generated uniformly in $[\lfloor\underline{\psi}N\rfloor, \lceil\overline{\psi}N\rceil]$. The preprocessing procedure (Line 3) and the method to select a subset of orders to remove (Line 5) are specified in each removal operator. Note that in some removal operators, the orders are removed in a batch (e.g., in trip removal, the orders on a trip must be removed simultaneously). The algorithm stops once the number of orders to be removed is greater than or equal to q (Line 7). We describe the details of removal operators in Online Appendix C.

Algorithm 2: A general outline of removal operator

```
Input: A solution $\mathcal{S}$; parameter $\overline{\psi}$ and $\underline{\psi}$.
Output: A subset of orders to remove, $\mathcal{N}^{remove} \subset \mathcal{N}$.
1 Generate a random integer number $q \sim U[\lfloor\underline{\psi}N\rfloor, \lceil\overline{\psi}N\rceil]$.
2 $n \Leftarrow 0, \mathcal{N}_0^{remove} \Leftarrow \emptyset$, stop $\Leftarrow$ false.
3 Preprocessing.                                  /* specified in each removal operator */
4 while not stop do
5     Select the orders to remove, $\mathcal{N}^{temp} \subset \mathcal{N} \setminus \mathcal{N}_n^{remove}$.   /* specified in each removal operator */
6     $\mathcal{N}_{n+1}^{remove} \Leftarrow \mathcal{N}_n^{remove} \cup \mathcal{N}^{temp}$.
7     if $|\mathcal{N}_{n+1}^{remove}| \geq q$ then stop $\Leftarrow$ true.
8     $n \Leftarrow n+1$.
9 return $\mathcal{N}_n^{remove}$.
```

Insertion operators: The two insertion operators (greedy insertion and regret insertion) are both adapted from Ropke and Pisinger (2006). We describe the details in Online Appendix C.

Scheduling operators: Here, the purpose is to adjust time slot decisions after a removal and an insertion operator have been performed on the routes. We introduce two scheduling operators: intra-route scheduling (Algorithm 3) and intra-shopping-mall scheduling (Algorithm 4). The two operators share the same idea that, given a current solution $\mathcal{S} = \{\boldsymbol{x}, (\boldsymbol{y}, \boldsymbol{z})\}$, we pick an order $i \in \mathcal{N}$ and adjust the time slot $[y_i, z_i]$, while the time slots of other orders remain unchanged. We try every possible $[y_i, z_i]$ in its feasible region and evaluate the resulted solutions using Ω^{oper}. After figuring out the best $[y_i, z_i]$, we keep it fixed and continue to search the next order until all the orders in $\mathcal{N}$ have been searched. The two operators differ in the selection of which order to search next. The intra-route scheduling operator follows the sequence of each route, while the intra-shopping-mall scheduling operator follows the sequence of average arrival time in each shopping mall.

Algorithm 3: Intra-route local search scheduling operator

```
    Input: a solution $\mathcal{S}$, a sample $\Omega^{oper}$.
    Output: an improved solution $\mathcal{S}^*$.
1   $\mathcal{K}' \Leftarrow \mathcal{K}, \mathcal{S}^* \Leftarrow \mathcal{S}$                                   /* initialize */
2   while $|\mathcal{K}'| > 0$ do                                          /* enumerate all vehicles */
3       Randomly select a vehicle $k \in \mathcal{K}$, with route $r_k = (0, \ldots, i, j, \ldots, 0)$.
4       $\Delta G^* \Leftarrow +\infty$.
5       foreach order $i$ on route $r_k$ do                     /* enumerate all orders on vehicle $k$ */
6           Evaluate solution $\mathcal{S}$ by $\Omega^{oper}$, i.e., evaluate $F(\mathcal{S}, \Omega^{oper})$.
7           foreach feasible $[y_i, z_i]$ do              /* enumerate all possible time slots of order $i$ */
8               $\mathcal{S}' \Leftarrow$ change the time slot of order $i$ in solution $\mathcal{S}$ to $[y_i, z_i]$.
9               Calculate $\Delta G = \sum_{\omega \in \Omega^{oper}} \Delta G(\mathcal{S}, \mathcal{S}', i, \omega, B(\mathcal{S}, \omega))$.
10              if $\Delta G < \Delta G^*$ then $\Delta G^* \Leftarrow \Delta G, \mathcal{S}^* \Leftarrow \mathcal{S}'$.
11          $\mathcal{S} \Leftarrow \mathcal{S}^*$.
12      $\mathcal{K}' \Leftarrow \mathcal{K}' \backslash \{k\}$.
13  return $\mathcal{S}^*$.
```

Algorithm 4: Intra-shopping-mall local search scheduling operator

```
    Input: a solution $\mathcal{S}$, a sample $\Omega^{oper}$.
    Output: an improved solution $\mathcal{S}^*$.
1   $\mathcal{M}' \Leftarrow \mathcal{M}, \mathcal{S}^* \Leftarrow \mathcal{S}$                                   /* initialize */
2   while $|\mathcal{M}'| > 0$ do                                   /* enumerate all shopping malls */
3       Randomly select a shopping mall $m \in \mathcal{M}'$.
4       Sort order $i \in \mathcal{N}_m^M$ in an increasing order of average arrival time.
5       $\Delta G^* \Leftarrow +\infty$.
6       foreach order $i \in \mathcal{N}_m^M$ do                /* enumerate all orders in shopping mall $m$ */
7           Evaluate solution $\mathcal{S}$ by $\Omega^{oper}$, i.e., evaluate $F(\mathcal{S}, \Omega^{oper})$.
8           foreach feasible $[y_i, z_i]$ do              /* enumerate all possible time slots of order $i$ */
9               $\mathcal{S}' \Leftarrow$ change the time slot of order $i$ in solution $\mathcal{S}$ to $[y_i, z_i]$.
10              Calculate $\Delta G = \sum_{\omega \in \Omega^{oper}} \Delta G(\mathcal{S}, \mathcal{S}', i, \omega, B(\mathcal{S}, \omega))$.
11              if $\Delta G < \Delta G^*$ then $\Delta G^* \Leftarrow \Delta G, \mathcal{S}^* \Leftarrow \mathcal{S}'$.
12          $\mathcal{S} \Leftarrow \mathcal{S}^*$.
13      $\mathcal{M}' \Leftarrow \mathcal{M}' \backslash \{m\}$.
14  return $\mathcal{S}^*$.
```

6. Numerical experiments

In this section, we describe the design of base instances, the algorithm parameter tuning, the performance of the proposed ALNS algorithm, and the numerical study on the value of coordination and value of stochastic solutions. All experiments are performed on a computer with an Intel Core i7-8700 processor with 3.20 GHz CPU and 16 GB RAM. The ALNS algorithm is implemented in Java, and Gurobi 9.0.2 is used as the solver.

6.1. Base instances

We construct our instances based on a subset of ten large shopping malls ($M = 10$) in Singapore, as shown in Figure 7. The LSPs are located in the same logistics park (Jurong in Figure 7). Each LSP assigns a fleet of vehicles dedicated to the deliveries to these shopping malls. For simplicity in the numerical study, we assume that each shopping mall has an equal number of docks, varying from one to four, that is, $C_1 = C_2 = \ldots = C_{10} = 1, 2, 3,$ or 4. At each value of $C_m, m \in \mathcal{M}$, we set three levels on the total number of orders (N), representing three levels of order density. The number of LSPs (L) is proportional to the total number of orders. Each LSP has a fleet of three vehicles and serves the same number of orders in each instance. We summarize the instance settings in Table 4. For each instance setting in Table 4, we run five replications. We denote the instance based on its setting and replication number. For example, "M10-C2-N100-L6-K18-5" denotes an instance with ten shopping malls (M10) each with two docks (C2), a total number of 100 orders (N100), six LSPs (L6), a total number of 18 vehicles (K18), and replication 5. In our instance construction, we also make sure that the endogenous time windows of the orders located at the same shopping mall and to be delivered by the same LSP do not overlap.

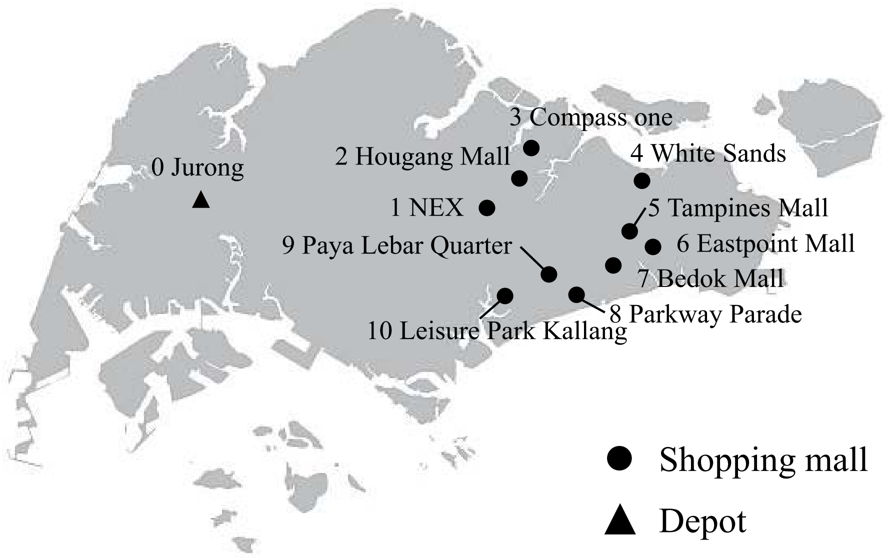


Figure 7 Shopping malls and the depot in the instances

For an order i, the demand d_i is uniformly selected from $\{10, 20, \ldots, 50\}$, and the length of the time window $[e_i, l_i]$ is fixed as two hours. We generate the service time of each order based on the real service time data we collected in the Tampines Mall from January 12, 2017 to November 17, 2017. We collected the service time data of 108 stores, but removed 32 stores with fewer than five orders. For the remaining 76 stores, we calculated each store's average service time and then the

Table 4 Base instances

# Docks per shopping mall (C_m)	# Orders (N)	# LSPs (L)	# Vehicles (K)
1	50	3	9
1	100	6	18
1	150	9	27
2	100	6	18
2	200	12	36
2	300	18	54
3	150	9	27
3	300	18	54
3	450	27	81
4	200	12	36
4	400	24	72
4	600	36	108

median of these average service times, which is 26.84 minutes. We then divided these 76 stores into two sets. The stores that have an average service time larger than 26.84 are denoted by "slow movers", while the rest are denoted by "fast movers." Overall, there are 1140 observations of service time for fast movers, and 2330 observations of service time for slow movers. For each order in an instance, we first decide whether it is placed by a fast mover or a slow mover, with equal probability. Then, for each of its service time realizations in Ω^{eval} and Ω^{alns}, we correspondingly bootstrap (with replacement) from the service time observations of the fast or slow movers, with equal probability. For Ω^{oper}, we bootstrap (with replacement) the service time from those in Ω^{alns}.

The length of a dock reservation time slot ($U_i\tau = z_i - y_i$) is a multiple of the discretization time unit $\tau = 15$ minutes. We set U_i to approximately equal to 95% quantile of the service time distribution, which is 90 minutes for a slow mover ($U_i = 6$) and 60 minutes ($U_i = 4$) for a fast mover.

Each LSP serves either 16 or 17 orders, which are distributed in all the ten shopping malls. So each LSP has one or two orders to serve in each shopping mall. If there is only one order (say order i) in a shopping mall, then its time window $[e_i, l_i]$ is selected from $\{[9:00, 11:00], [9:15, 11:15], \dots, [15:00, 17:00]\}$ with equal probability. If there are two orders (say order i and j), their time windows are also randomly selected from $\{[9:00, 11:00], [9:15, 11:15], \dots, [15:00, 17:00]\}$, but $[e_i, l_i + U_i\tau]$ and $[e_j, l_j + U_j\tau]$ are not allowed to overlap.

We generate the travel time scenarios based on the real travel time data in Singapore that we collected via the Google API. For each arc (between each pair of locations in Figure 7), we collected the travel time via Google API in every 30 minutes from 8:00 to 18:00 on July 16, 2018 (Monday) to July 20, 2018 (Friday). We thus obtained a set of $5 \times 21 = 105$ distinct travel time matrices, which serves as the base for generating a travel time scenario. The average length of an arc is 14.05 KM, and the average speed of the vehicle is 43.37 KM/H. Note that each travel time matrix has the inter-dependent travel times on $11 \times 10 = 110$ arcs in the network with one depot and ten shopping malls. When generating a travel time scenario in Ω^{eval}, we first bootstrap (with replacement) a travel time matrix from the 105 travel time matrices, and then add a perturbation $\epsilon_{ij} \sim N(0, \hat{\sigma}^O_{ij})$ on each arc (i,j), where $\hat{\sigma}^O_{ij}$ is the sample standard deviation of the travel time on arc (i,j) in the 105 travel time matrices. When generating a service time scenario in Ω^{eval}, for each order, we bootstrap (with replacement) from the service time observations of either a fast or a slow mover. In a joint scenario that combines stochastic travel times of the vehicles and stochastic service times of the orders, we can reasonably assume that these two stochastic elements are mutually independent. In our experiments, we set the size of Ω^{eval} as $10,000$.

When generating the travel time scenarios in Ω^{alns} and Ω^{oper}, in order to maintain the interdependency between the travel times on arcs in the road network, we apply the copula-based scenario generation method (Kaut 2014, Guo, Wallace, and Kaut 2019) to the travel time scenarios in Ω^{eval}. The copula-based scenario generation method first generates a copula sample to minimize the average deviation between the copula sample and the target copula (i.e., the empirical copula of travel times in Ω^{eval} in our experiment), and then uses the marginal distributions to transform the copula sample into travel time scenarios. We provide a description of the copula-based scenario generation method in Online Appendix D. More details can be found in Kaut (2014).

Each vehicle has a capacity $Q = 100$. The regular working time is $T = 480$ minutes. The loading time at the depot is set to 0 (for simplicity). The routing cost is estimated as 0.2 SGD/KM, which is calculated based on fuel consumption rate 11.0 L per 100 KM (The Land Transport Authority of Singapore 2018) times fuel price 1.82 SGD/L (Shell 2018)[1], and is equivalent to about 0.14 SGD/minute based on the average speed 43.37 KM/H in the network. The unit penalty costs in the objective function are set as $c^w = c^{\varpi} = 0.1$ SGD/minute, $c^s = c^e = 0.4$ SGD/minute, $c^o = 0.2$ SGD/minute, where the unit penalty cost of overtime (c^o) is derived from the public information available on the The Ministry of Manpower (2018) website, and all other coefficients are set according to the prevailing setting in the CUDO platform. The relative high unit penalty costs of late service start and end time reflect the CUDO platform's importance weighting toward the punctuality of the services (high late service start time) and the release of the docks in time (high late service end time).

6.2. Algorithm parameter tuning

The ALNS introduced in Section 5 has many algorithmic parameters. Following the tuning process and settings provided in Ropke and Pisinger (2006), we begin with a fixed parameter setting and then tune the parameters one by one in the order and within the value ranges as shown in Table 6.

[1] 1 SGD = 0.7357 USD, as of the exchange rate on December 1, 2019.

When one parameter is being tuned, the rest of the parameters were fixed from the previously determined values.

When tuning the parameters, we generate a set of test instances with order size $N = 200, 400, 600$, which are independent of those in the base instances and with different numbers of LSPs (see Table 5). For each instance setting, we generate two instance replications. For each algorithm parameter setting, we run the ALNS on each test instance for five times, where the same sample Ω^{eval} ($|\Omega^{eval}| = 10,000$) is used for variance reduction. The parameter value that produced the best average performance, i.e., $F(\mathcal{S}, \Omega^{eval})$, is selected and fixed. The final choice of the parameter settings is shown in Table 6.

Table 5 Test instances for parameter tuning

# Docks per shopping mall (C_m)	# Orders (N)	# LSPs (L)	# Vehicles (K)
2	200	12	36
4	400	24	72
4	600	36	108

Table 6 Algorithm parameter settings

Parameter	Notation	Range	Value
Relatedness function coefficients	ϑ^c	[1, 3, 5, 7, 9]	7
	ϑ^a	[1, 3, 5, 7, 9]	3
	ϑ^d	[1, 3, 5, 7, 9]	1
	ϑ^l	[1, 3, 5, 7, 9]	3
Regret insertion	β	[2, 3, 4, 5, 6]	3
Removal operator	$\overline{\psi}$	[10%, 15%, 20%, 25%, 30%, 35%, 40%]	15%
	$\underline{\psi}$	[5%, 10%]	5%
Intensity of the bias	p	[4, 6, 8, 10]	8
Segment length	N^{seg}	[25, 50, 100, 200]	50
Simulated annealing acceptance criteria	ϵ	[1%, 3%, 5%, 7%, 9%]	3%
	γ	[1000, 2000, 3000, 4000]	2000
Score	π_1	[1, 5, 9, 13, 17]	5
	π_2	[1, 5, 9, 13, 17]	9
	π_3	[1, 5, 9, 13, 17]	1
Reaction factor	ρ	[0.3, 0.4, 0.5, 0.6, 0.7]	0.5

We further experiment on the comparative performance between the ALNS with ΔG (i.e., Algorithm 6 in Online Appendix B.2) and that without ΔG (i.e., replacing ΔG with an exact evaluation by Algorithm 5 in Online Appendix B.1). As shown in Table 7, on average, the use of ΔG in the ALNS increases the number of iterations by 32 times and reduces the mean objective function value by 6.1%. The results show that the approximation of ΔG in Algorithm 6 does reduce the evaluation time significantly, which enables the ALNS algorithm to search more solution and obtains better solutions, under the budget on total computational time.

Table 7 **Comparisons between the performances of the ALNS with ΔG vs. without ΔG**

Instance	With ΔG Mean (1)	With ΔG # Iterations (2)	Without ΔG Mean (3)	Without ΔG # Iterations (4)	Comparisons Mean $(5)=\frac{(1)-(3)}{(3)}$	Comparisons # Iterations $(6)=\frac{(2)-(4)}{(4)}$
M10-C2-N200-L12-K36-6	2667.42	15263	2786.51	696	-4.3%	21
M10-C2-N200-L12-K36-7	2806.39	15392	2860.67	693	-1.9%	21
M10-C4-N400-L24-K72-6	4876.70	3619	5176.41	104	-5.8%	34
M10-C4-N400-L24-K72-7	4800.62	3661	5152.19	104	-6.8%	34
M10-C4-N600-L36-K108-6	10425.58	1490	11534.51	35	-9.6%	42
M10-C4-N600-L36-K108-7	10570.59	1485	11519.92	35	-8.2%	42
Average	6024.55	6818	6505.03	278	-6.1%	32

6.3. Scenario generation and stability test

In Section 5, we introduce three sizes of samples, Ω^{eval}, Ω^{alns}, and Ω^{oper}. Ω^{eval} is used to evaluate the "true" objective of the final solution, so its size is very large ($|\Omega^{eval}| = 10,000$ in our experiments). Ω^{alns} is used to evaluate the solution in every iteration of the ALNS, while Ω^{oper} is used more frequently to evaluate the solutions generated by the neighborhood search operators.

For either $|\Omega^{alns}|$ or $|\Omega^{oper}|$, a larger size will lead to more accurate evaluation, but will also take more computational time such that less iterations can be done, given the one-hour solution time limit (and vise versa). To study this tradeoff, we test different combinations of $|\Omega^{alns}|$ and $|\Omega^{oper}|$. For each test instance, we run the ALNS with each combination of $|\Omega^{alns}|$ and $|\Omega^{oper}|$ five times, and report the average gap compared to the best $F(\mathcal{S}^*, \Omega^{eval})$ among all the replications of all the combinations. The results are in Table 8. In each block where the $|\Omega^{alns}|$ is fixed, the best combination is in **bold**. We observe that, given a fixed $|\Omega^{alns}|$, the gap initially decreases and then increases with $|\Omega^{oper}|$, forming a U-shape. Too small $|\Omega^{oper}|$ introduces too much noise in the neighborhood search, while too large $|\Omega^{oper}|$ reduces the number of ALNS iterations, given a fixed $|\Omega^{alns}|$. The best setting overall is $|\Omega^{alns}| = 200$ and $|\Omega^{oper}| = 20$, with the gap of 2.85%.

Table 8 **Gaps and number of iterations for different combinations of $|\Omega^{alns}|$ and $|\Omega^{oper}|$**

$\|\Omega^{alns}\|$	$\|\Omega^{oper}\|$	Gap	# Iterations	$\|\Omega^{alns}\|$	$\|\Omega^{oper}\|$	Gap	# Iterations	$\|\Omega^{alns}\|$	$\|\Omega^{oper}\|$	Gap	# Iterations
50	2	13.79%	52,362	100	2	12.84%	41,155	150	2	13.02%	33,798
50	6	5.57%	21,221	100	6	5.53%	19,162	150	6	5.67%	17,324
50	10	4.01%	13,242	100	10	3.89%	12,320	**150**	**10**	**3.57%**	**11,532**
50	**20**	**3.54%**	**6741**	**100**	**20**	**3.69%**	**6521**	150	20	3.92%	6333
50	30	3.95%	4488	100	30	3.97%	4410	150	30	3.80%	4296
200	2	13.47%	28,632	600	2	13.62%	13,144	1000	2	14.21%	8519
200	6	5.10%	15,700	600	6	5.68%	9387	1000	6	6.25%	6723
200	10	3.78%	10,788	600	10	4.14%	7325	1000	10	4.65%	5577
200	**20**	**2.85%**	**6050**	**600**	**20**	**3.62%**	**4775**	**1000**	**20**	**3.88%**	**3956**
200	30	3.54%	4169	600	30	3.83%	3517	1000	30	3.91%	3046

We evaluate the in-sample and out-of-sample stability for a fixed sample size of Ω^{alns} obtained above (Kaut and Wallace 2007). For each test instance, we generate J samples ($\Omega^{alns}_j, j = 1, 2, \ldots, J$)

and denote the corresponding ALNS solutions by $\mathcal{S}_j^*$. In-sample stability requires that different samples should lead to solutions with approximately the same objective function values, i.e., $F(\mathcal{S}_i^*, \Omega_i^{alns}) \approx F(\mathcal{S}_j^*, \Omega_j^{alns}), \forall 1 \leq i \neq j \leq J$. Out-of-sample stability measures the stability of the samples, evaluated by the true objective function or a different sample than that was used to find the solutions, i.e., $F(\mathcal{S}_i^*, \Omega^{eval}) \approx F(\mathcal{S}_j^*, \Omega^{eval}), \forall 1 \leq i \neq j \leq J$.

In Table 9, we show the in-sample and out-of sample stability ($J = 20$) of test instances in Table 5. Columns (1) and (2) are the mean values and standard deviations of $F(\mathcal{S}_j^*, \Omega_j^{alns})$, $1 \leq j \leq J$, respectively. We measure the in-sample stability by the coefficients of variation in Column (3), which range between 1.02%–1.60%, with an average of 1.32%. Columns (4) and (5) are the mean values and standard deviations of $F(\mathcal{S}_j^*, \Omega^{eval})$, $1 \leq j \leq J$, respectively. We measure the out-of-sample stability by the coefficients of variation in Column (6), which range between 0.73%–1.33%, with an average of 1.10%. Column (7) shows that the relative gap

$$\frac{\sum_{1 \leq j \leq J} F(\mathcal{S}_j^*, \Omega^{eval}) - \sum_{1 \leq j \leq J} F(\mathcal{S}_j^*, \Omega_j^{alns})}{\sum_{1 \leq j \leq J} F(\mathcal{S}_j^*, \Omega^{eval})}$$

is 0.85% on average. The above calculations demonstrate that using 200 as $|\Omega^{alns}|$ is in- and out-of-sample stable (Kaut and Wallace 2007, Wang, Crainic, and Wallace 2019).

Table 9 In-sample and out-of-sample stability

Instance	In-sample stability Mean (1)	 Std. dev. (2)	 (2)/(1) (3)	Out-of-sample stability Mean (4)	 Std. dev. (5)	 (5)/(4) (6)	 ((4)-(1))/(4) (7)
M10-C2-N200-L12-K36-6	2634.98	37.84	1.44%	2677.67	32.41	1.21%	1.59%
M10-C2-N200-L12-K36-7	2758.80	43.68	1.58%	2795.70	35.23	1.26%	1.32%
M10-C4-N400-L24-K72-6	4854.23	57.07	1.18%	4861.98	53.40	1.10%	0.16%
M10-C4-N400-L24-K72-7	4784.36	76.72	1.60%	4807.83	64.17	1.33%	0.49%
M10-C4-N600-L36-K108-6	10,347.08	105.92	1.02%	10,420.00	75.59	0.73%	0.70%
M10-C4-N600-L36-K108-7	10,516.22	116.69	1.11%	10,607.87	101.61	0.96%	0.86%
Average			1.32%			1.10%	0.85%

Moreover, the results in Table 9 are consistent with the observations in Kaut and Wallace (2007). In the in-sample stability test, the objective function value of each solution is evaluated by the sample used to find the solution, while in the out-of-sample stability test, the objective function value of each solution is evaluated by the same Ω^{eval} in the paper. We observe that the mean values in Column (1) are consistently lower than those in Column (4), because the in-sample stability test (Column (1)) tends to "overestimate the quality of its own solution." Also, while our solutions obtained by the ALNS (Algorithm 1) are heuristic in nature, the observations on the "over-optimistic" solutions associated with a (smaller) subset of sample and on the monotonicity of the solution quality in sample size are consistent with what are shown in Mak, Morton, and Wood

(1999). We also observe that standard deviations in Column (2) are consistently higher than those in Column (5), because the out-of-sample stability test (Column (5)) uses the common sample Ω^{eval} to evaluate the solutions and thus reduces the variance of the results.

6.4. ALNS performance

The problem and model in Sections 3 and 4 often result in large instances in practice that are very difficult to solve. Take a medium-size instance M10-C2-N300-K54-L18-1 as an example, the first stage includes ~24,000 binary variables, ~600 general integer variables, ~25,000 continuous variables, and ~43,000 constraints; the second stage (for one scenario) includes ~51,000 binary variables, ~2900 continuous variables, and ~112,000 constraints.

We benchmark our proposed ALNS with a commercial solver (Gurobi 9.0.2) on the mean value problem (MVP) of some small instances, in which we replace the stochastic travel times and service times with their mean values, and show the results in Table 10. We generate and name these instances in a similar fashion to the base instances (Section 6.1). Note that in the instances with $N = 15$, there are only two malls (Compass One and Tampines Mall in Figure 7).

In Table 10, Columns (1) to (5) are the results of Gurobi. Specifically, Columns (1) and (2) are the upper bound and lower bound provided by Gurobi, respectively. Column (3) is the optimality gap, calculated by $(3) = [(1) - (2)]/(2)$. Columns (4) and (5) are the time to find the best solution and the total time until Gurobi terminates. Columns (6) to (9) are the results of the ALNS. Specifically, Column (6) is the objective function value of the best solution of the ALNS. Column (7) is the optimality gap, calculated by $(7) = [(6) - (2)]/(2)$. Column (8) is the time to find the best solution of the ALNS. Column (9) is the improvement of the best objective function value of the ALNS compared with that of Gurobi, calculated by $(9) = [(6) - (1)]/(1)$.

For the five instances of $N = 15$, on average, Gurobi solves them to optimality in 2.1 seconds, while the ALNS finds the best solution within 0.06 seconds with an average optimality gap of 0.13%. We further notice that the ALNS finds the optimal solutions in four instances.

For instances of $N = 25$, we set the time limit of 3600 seconds for both algorithms. Gurobi and the ALNS reach similar average optimality gaps of 17.10% and 15.97%, respectively, with the ALNS slightly better than Gurobi. In terms of the average time to find the best solution (column "TB"), it takes Gurobi 2091 seconds, much longer than 17 seconds of the ALNS.

For the smallest instances ($N = 50$) in Table 4, Gurobi fails to find even a feasible solution to the MVP within days of computational time. Therefore, in the remaining experiments, we use the ALNS to solve all the instances.

In Table 11, we show the performance of the ALNS operators, based on solving the instances in Table 4. For each operator, we report the average one-run CPU time, and the percentage of the

Table 10 Mean value problem: Commercial solver (Gurobi 9.0.2) vs. ALNS

Instance	Gurobi					ALNS			
	UB (1)	LB (2)	Gap (3)	TB (sec) (4)	TT (sec) (5)	UB (6)	Gap (7)	TB (sec) (8)	% Improve (9)
M10-C1-N15-L3-K3-1	121.15	121.15	0.00%	2.2	2.2	121.15	0.00%	0.01	0.00%
M10-C1-N15-L3-K3-2	119.83	119.83	0.00%	1.7	1.7	119.83	0.00%	0.01	0.00%
M10-C1-N15-L3-K3-3	121.69	121.69	0.00%	2.7	2.7	121.69	0.00%	0.11	0.00%
M10-C1-N15-L3-K3-4	106.27	106.27	0.00%	1.7	1.7	106.97	0.66%	0.01	-0.66%
M10-C1-N15-L3-K3-5	118.13	118.13	0.00%	2.0	2.0	118.13	0.00%	0.17	0.00%
Average (N = 15)			0.00%	2.1	2.1		0.13%	0.06	-0.13%
M10-C1-N25-L3-K5-1	171.00	150.48	12.00%	2990	3600	169.61	11.28%	1.8	0.81%
M10-C1-N25-L3-K5-2	225.74	169.76	24.80%	1262	3600	218.76	22.40%	4.7	3.09%
M10-C1-N25-L3-K5-3	168.03	150.99	10.14%	3000	3600	166.62	9.38%	1.3	0.84%
M10-C1-N25-L3-K5-4	199.68	154.96	22.39%	355	3600	199.39	22.28%	67.7	0.15%
M10-C1-N25-L3-K5-5	175.99	147.54	16.16%	2848	3600	172.62	14.53%	9.6	1.92%
Average (N = 25)			17.10%	2091	3600		15.97%	17.0	1.36%

Note UB: upper bound; LB: lower bound; Gap in (3) = [(1) − (2)]/(2); Gap in (7) = [(6) − (2)]/(2); TB: time to the best solution; TT: total time; % Improve in (9) = [(6) − (1)]/(1).

Table 11 Performance of the ALNS operators

Operator	Time (sec)	Usage	Score	Operator	Time (sec)	Usage	Score
Removal				*Removal (continued)*			
Random	3.86×10^{-5}	1.9%	4.6%	Time window violation	4.10×10^{-5}	2.1%	4.5%
Shaw	1.59×10^{-3}	1.8%	2.6%	Time slot violation	4.19×10^{-5}	5.2%	4.4%
Worst	6.40×10^{-3}	21.4%	**34.0%**	*Insertion*			
Shopping mall	1.83×10^{-5}	3.7%	1.9%	Greedy	1.17×10^{-2}	53.3%	**53.9%**
Vehicle	2.60×10^{-5}	10.1%	5.5%	Regret	1.19×10^{-2}	46.7%	46.1%
Trip	5.96×10^{-5}	3.1%	1.5%	*Scheduling*			
Outside waiting time	5.50×10^{-5}	2.8%	12.5%	Intra-route	4.26×10^{-1}	45.2%	**55.3%**
Inside waiting time	5.65×10^{-5}	47.9%	**28.5%**	Intra-shopping-mall	4.24×10^{-1}	54.8%	44.7%

usage (measured by the number of iterations the operator is used relative to the total number of iterations) and the score obtained in its (removal/insertion/scheduling) operator set.

In terms of computational time, we note that scheduling operators > insertion operators > removal operators. We also observe that worst removal and inside waiting time removal are the best two removal operators, with a summed score of 62.5%; greedy insertion outperforms regret insertion; and intra-route scheduling outperforms intra-shopping mall scheduling.

6.5. Value of coordination

The platform serves as a centralized decision maker that coordinates the order deliveries of different LSPs and dock reservations in shopping malls (denoted by the coordinated solution). In the current practice, LSPs plan their deliveries and dock reservations in an uncoordinated manner, i.e., each LSP $l \in \mathcal{L}$ independently makes routing and scheduling decisions on its own orders and reserves the docks accordingly (denoted by the uncoordinated solution). In this section, we measure the value of coordination by comparing these two solutions. To ensure a fair comparison, we evaluate both the coordinated and uncoordinated solutions using the same Ω^{eval}, and for each LSP-specific problem, we use the same ALNS parameter setting and computational time (i.e., 3600 seconds) as those in the coordinated problem.

For the base instances described in Section 6.1, we compare the total cost $F(\mathcal{S}^*, \Omega^{eval})$ in Table 12, in which we show the average of five replications for each instance. The average percentage saved by coordination of all instances is 9.4%, which is significant (compared to the coefficient of variance of less than 2% in the stability tests in Section 6.3). For a fixed number of total docks, we observe that the total cost increases significantly as the number of orders (N) increases. This is intuitive as more orders increase the chance of congestion. The value of coordination is higher if the docks are more congested, which is evidenced by the increase of the percentage saved by coordination as N increases. This is also shown below in Figure 8.

We also observe in Table 12 that, when increasing the number of docks while keeping the number of orders fixed, the total costs of both uncoordinated and coordinated solutions reduce dramatically, and further, the % saved by coordination also reduces significantly. For instance, while keeping $N = 300$, increasing the number of docks from 2 to 3 reduces the total costs of both uncoordinated and coordinated solutions by more than 50%, and the % saved by coordination reduces from 20.1% to 2.1%. This is intuitive, when the number of orders is fixed, increasing the number of docks provides more unloading capacity, which naturally reduces the total cost, and further, increasing the number of docks provides more flexibility and reduces the chance of congestion (even for uncoordinated solutions), which results in the diminishing of the value of coordination.

Table 12 Total cost and % saved by coordination: Base instances

# dock		10×1			10×2	
N	50	100	150	100	200	300
Uncoordinated	647.0	2600.1	7334.7	1066.4	3002.8	8659.5
Coordinated	599.9	2085.5	5675.3	1043.2	2753.0	6919.8
% saved	7.3%	19.8%	22.6%	2.2%	8.3%	20.1%
# dock		10×3			10×4	
N	150	300	450	200	400	600
Uncoordinated	1531.4	3594.9	10,126.3	1973.8	4575.6	12,071.1
Coordinated	1531.4	3518.1	8510.4	1973.8	4504.1	10,527.0
% saved	0.0%	2.1%	16.0%	0.0%	1.6%	12.8%

In Figure 8, We plot the values of “% saved” (by coordination) in Table 12. We observe that, when the order-dock ratio is fixed (that is, even when the demand-capacity ratio is fixed), the value of coordination decreases in the number of docks in each shopping mall. Again, increasing the number of docks provides more flexibility and thus reduces the chance of congestion.

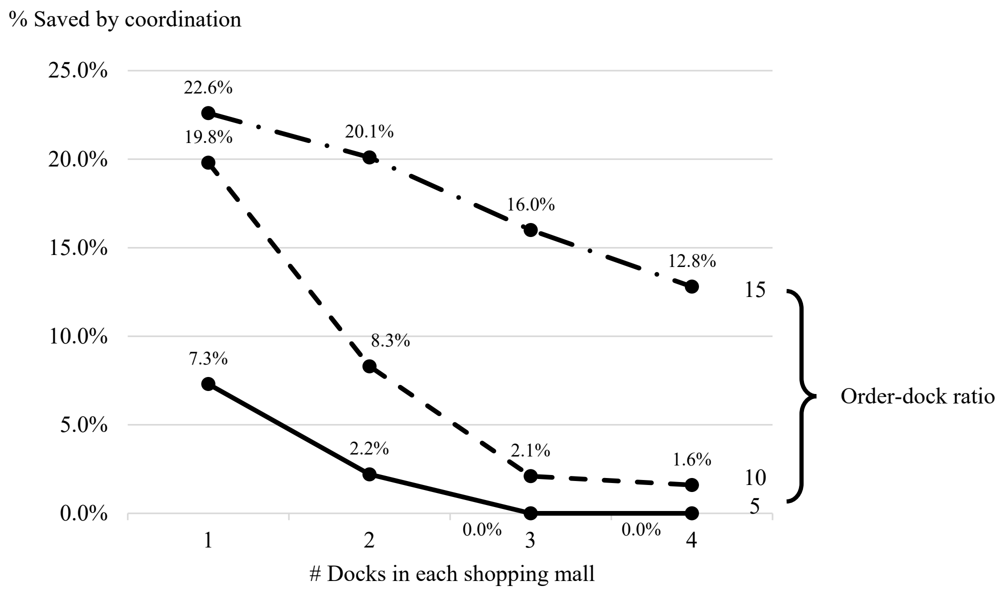


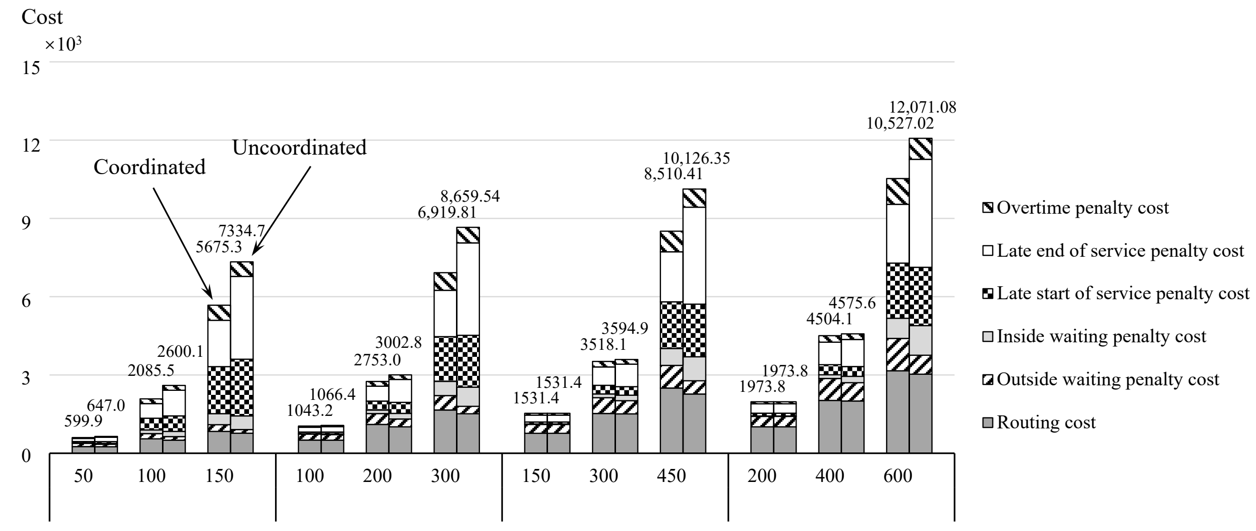
Figure 8 Value of coordination: Fixed order-dock ratio

Figure 9(a) shows the total cost and the decomposed cost components. For each instance setting, the left bar corresponds to the coordinated solution while the right bar corresponds to the uncoordinated solution. Each bar consists of routing cost, outside waiting penalty cost, inside waiting penalty cost, late start of service penalty cost, late end of service penalty cost, and overtime penalty cost (from bottom to top). We observe that the routing cost of the coordinated solution is always larger than that of the uncoordinated solution. In order to alleviate congestion in docks, some vehicles may need to deviate from their distance minimizing routes and visit less congested shopping malls. We also observe that the saving in the total cost of the coordinated solution is mainly driven by the reduction in late start/end of service penalty.

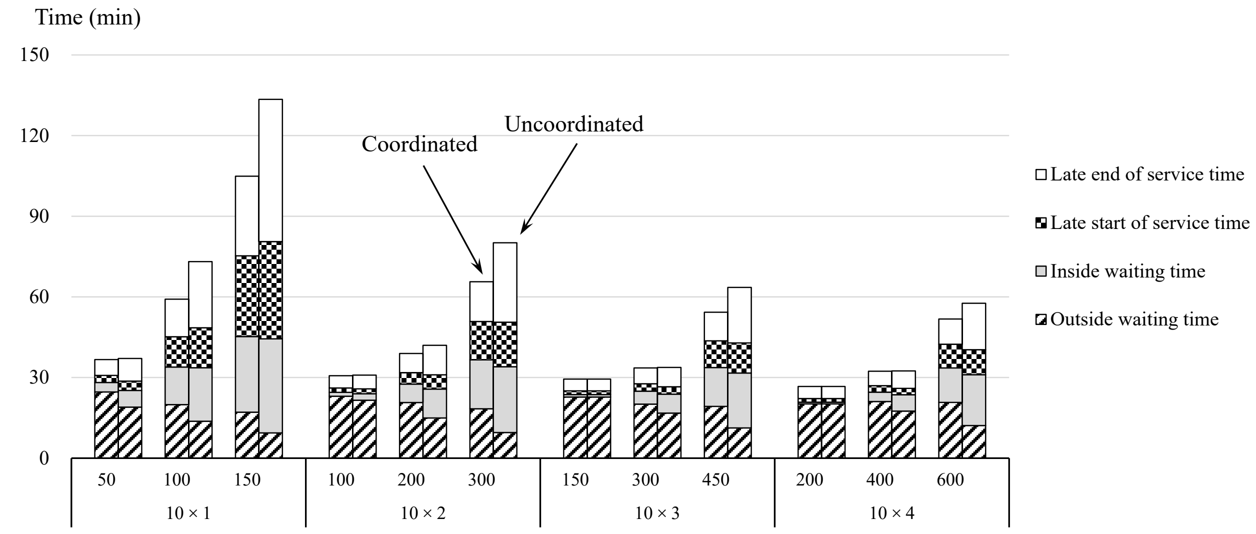
Noting the difference in penalty coefficients (0.4 SGD/minute for late start/end of service penalty and 0.1 SGD/minute for outside/inside waiting time), we compare the decomposed time (without penalty coefficients) in Figure 9(b), where we show the average time per order for each time item. Compared to the coordinated solution, the outside waiting time of the uncoordinated solution is shorter, as each LSP independently schedules the time slots earlier (based on their own routing decisions), which in turn allows the vehicles to enter the mall earlier. However, the underestimation of potential congestion causes longer inside waiting time, which propagates to significantly longer late start and end service time.

6.6. Value of stochastic solutions

Next, we study the value of stochastic solutions. There are two types of stochasticity: travel time (TT) and service time (ST). When ignoring one type of stochasticity, we replace the corresponding



(a) Cost decomposition



(b) Time decomposition (time per order)

Figure 9 Value of coordination: Base instances

distributions with the expected values. When both types of stochasticity are ignored, the problem becomes the mean value problem (MVP). We vary the stochasticity settings from the base instances, solve these instances by the ALNS algorithm, and evaluate the solutions under the same Ω^{eval}.

In Figure 10 (from left to right), S-ST&S-TT (black bar) represents the setting with stochastic service time & stochastic travel time, S-ST&D-TT (dark grey bar) represents the setting with stochastic service time & deterministic travel time, D-ST&S-TT (light grey bar) represents the setting with deterministic service time & stochastic travel time, and D-ST&D-TT (white bar) represents the mean value problem. The gaps of the last three bars relative to the S-ST&S-TT are shown on top of these bars.

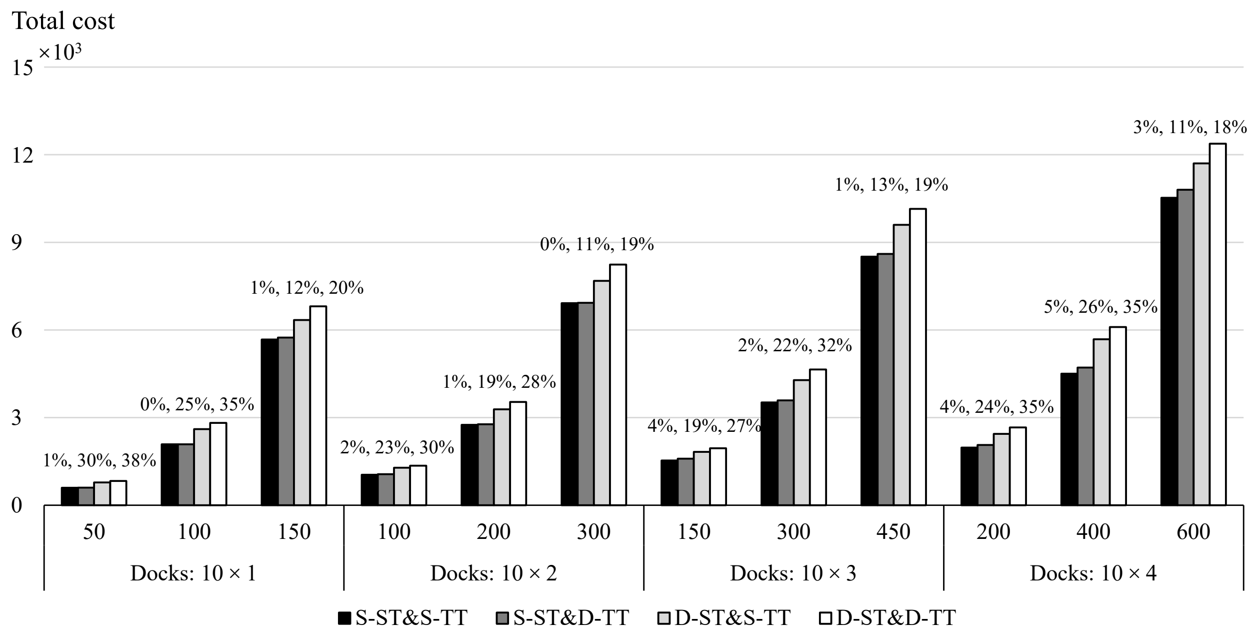


Figure 10 Value of stochastic solutions

We observe that the gaps between the stochastic solutions (S-ST&S-TT) and the MVP solutions (D-ST&D-TT) range from 18% to 38%, with an average of 21.1%, which shows the significant value of stochastic solutions. Further, we observe that the average gap between the S-ST&S-TT and S-ST&D-TT is only 2.0%, while that between S-ST&S-TT and D-ST&S-TT is 19.6%. This indicates that, under our base instances, the stochasticity in service time has a higher impact than the stochasticity in travel time does.

In order to further understand the impact of the stochasticity in travel time on the value of stochastic solutions, we (arbitrarily) select the base instances with 300 orders and 3 docks in each mall and increase the variance of perturbations (from $\hat{\sigma}_{ij}^O$ to $2\hat{\sigma}_{ij}^O$ and $3\hat{\sigma}_{ij}^O$) when generating travel time scenarios based on the 105 travel time matrices, while keeping the stochastic service time settings unchanged. In Table 13, We observe that, when increasing the travel time variances, the average gap between S-ST&S-TT and S-ST&D-TT (Column (5)) increases from 1.98% to 2.21% and then 2.88% but is still much smaller than the average gap (22.13%) between S-ST&S-TT and D-ST&S-TT (Column (6)). One potential explanation on the higher impact of stochasticity in service time than the stochasticity in travel time is that the former has a direct impact on the dock occupation while the latter only has an indirect impact. More specifically, a longer service time will lead to longer occupation of the dock while a longer travel time only delays the arrival of the vehicle at the shopping mall and affects the sequence of services in that shopping mall.

Table 13 Value of stochastic solutions under different travel time variances (base instances with 300 orders and 3 docks in each mall)

Travel time perturbation	S-ST&S-TT (1)	S-ST&D-TT (2)	D-ST&S-TT (3)	D-ST&D-TT (4)	$\frac{(2)-(1)}{(1)}$ (5)	$\frac{(3)-(1)}{(1)}$ (6)	$\frac{(4)-(1)}{(1)}$ (7)
$\epsilon_{ij} \sim N(0, \hat{\sigma}_{ij}^O)$	3518.15	3587.86	4281.39	4649.14	1.98%	21.69%	32.15%
$\epsilon_{ij} \sim N(0, 2\hat{\sigma}_{ij}^O)$	3530.52	3608.57	4283.59	4698.88	2.21%	21.33%	33.09%
$\epsilon_{ij} \sim N(0, 3\hat{\sigma}_{ij}^O)$	3563.15	3665.65	4289.18	4814.88	2.88%	20.38%	35.13%

7. Conclusions

In many cities where shopping malls are densely distributed, the tenants in shopping malls are replenished by logistics service providers (LSPs), which route their own fleet of vehicles without coordination with each other. Uncoordinated arrivals of vehicles and limited docking capacity jointly cause congestion within and outside shopping malls, which highlights the importance to better coordinate the deliveries. One approach as discussed in this paper is to develop a centralized decision making platform that routes and schedules at the system level. The underlying problem is denoted by the coordinated delivery to shopping malls with limited docking capacity, which jointly considers docking capacity, endogenous time windows, and stochastic travel times in the road network and stochastic service times of orders in the shopping malls. We formulate this problem as a two-stage stochastic mixed integer program.

We develop an Adaptive Large Neighborhood Search (ALNS) metaheuristic, where removal and insertion operators adjust routes and newly introduced scheduling operators adjust time slot decisions. Given a first stage solution, the precise evaluation of the second stage recourse function involves calculating the expectation of the second stage cost. To overcome the computational challenge of evaluating the first stage solutions (of vehicle routing and dock scheduling) in a very large solution space and to trade off between the evaluation accuracy and the number of solutions that can be searched, we use various sample sizes to approximate the second stage recourse function at different layers of the ALNS algorithm. Further, we test the in-sample and out-of-sample stability of our approximations. We construct a test bed of instances based on real data in Singapore and demonstrate the performance of the proposed solution method as well as the value of coordination and the value of stochastic solutions in our numerical study.

Notably, our computational study on the impact of stochastic travel times is somehow constrained by our limited collection of raw travel time data, due to various of factors. Further, the variances in travel times in Singapore are relatively low, compared with other cities in the world. One extension to this work is to study the impact of stochastic travel times using rich travel time data of various city profiles. This would also enables a more comprehensive understanding on the relative impact between stochastic travel times and stochastic service times.

Acknowledgments

This research was supported by 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 and the National Natural Science Foundation of China (NSFC) under Project No. 71771137. The authors thank professor Stein W. Wallace at NHH for the valuable suggestions on scenario generation and Dr. Michal Kaut of SINTEF for providing the executive of the copula-based scenario generation method. The authors are also grateful to the associate editor and anonymous referees for their careful reviews and constructive suggestions.

References

Adulyasak Y, Jaillet P, 2016 *Models and algorithms for stochastic and robust vehicle routing with deadlines. Transportation Science* 50(2):608–626.

Agustina D, Lee C, Piplani R, 2014 *Vehicle scheduling and routing at a cross docking center for food supply chains. International Journal of production economics* 152:29–41.

Ando N, Taniguchi A, 2006 *Travel time reliability in vehicle routing and scheduling with time windows. Networks and Spatial Economics* 6:293–311.

Azi N, Gendreau M, Potvin J, 2014 *An adaptive large neighborhood search for a vehicle routing problem with multiple routes. Computers & Operations Research* 41:167–173.

Binart S, Dejax P, Gendreau M, Semet F, 2016 *A 2-stage method for a field service routing problem with stochastic travel and service times. Computers & Operations Research* 65:64–75.

Campbell AM, Gendreau M, Thomas BW, 2011 *The orienteering problem with stochastic travel and service times. Annals of Operations Research* 186(1):61–81.

Chang T, Wan Y, Ooi WT, 2009 *A stochastic dynamic traveling salesman problem with hard time windows. European Journal of Operational Research* 198(3):748–759.

Crainic TG, Gobbato L, Perboli G, Rei W, 2016 *Logistics capacity planning: A stochastic bin packing formulation and a progressive hedging meta-heuristic. European Journal of Operational Research* 253(2):404–417.

Dalla Chiara G, Cheah L, 2017 *Data stories from urban loading bays. European Transport Research Review* 9:50–65.

Dalmeijer K, Spliet R, 2018 *A branch-and-cut algorithm for the time window assignment vehicle routing problem. Computers & Operations Research* 89:140–152.

Demir E, Burgholzer W, Hrusovsku M, Arikan E, Jammernegg W, Van Woensel T, 2016 *A green intermodal service network design problem with travel time uncertainty. Transportation Research Part B: Methodological* 93(7):789–807.

Dondo R, Cerdá J, 2014 *A monolithic approach to vehicle routing and operations scheduling of a cross-dock system with multiple dock doors. Computers & Chemical Engineering* 63:184–205.

Dondo R, Cerdá J, 2015 *The heterogeneous vehicle routing and truck scheduling problem in a multi-door cross-dock system. Computers & Chemical Engineering* 76:42–62.

Ehmke JF, Campbell AM, Urban TL, 2015 *Ensuring service levels in routing problems with time windows and stochastic travel times. European Journal of Operational Research* 240(2):539–550.

Errico F, Desaulniers G, Gendreau M, Rei W, Rousseau L, 2016 *A priori optimization with recourse for the vehicle routing problem with hard time windows and stochastic service times. European Journal of Operational Research* 249(1):55–66.

Froger A, Mendoza JE, Jabali O, Laporte G, 2017 *A matheuristic for the electric vehicle routing problem with capacitated charging stations.* Working Paper 31, CIRRELT.

Gendreau M, Jabali O, Rei W, 2016 *50th anniversary invited article—future research directions in stochastic vehicle routing. Transportation Science* 50(4):1163–1173.

Guo Z, Wallace SW, Kaut M, 2019 *Vehicle routing with space-and time-correlated stochastic travel times: Evaluating the objective function. INFORMS Journal on Computing* 31(4):654–670.

Hof J, Schneider M, Goeke D, 2017 *Solving the battery swap station location-routing problem with capacitated electric vehicles using an avns algorithm for vehicle-routing problems with intermediate stops. Transportation Research Part B: Methodological* 97:102–112.

Hoogeboom M, Adulyasak Y, Dullaert W, Jaillet P, 2021 *The robust vehicle routing problem with time window assignments. Transportation Science* 55(2):395–413.

Jabali O, Leus R, Van Woensel T, De Kok T, 2015 *Self-imposed time windows in vehicle routing problems. OR Spectrum* 37(2):331–352.

Kaut M, 2014 *A copula-based heuristic for scenario generation. Computational Management Science* 11(4):503–516.

Kaut M, Wallace SW, 2007 *Evaluation of scenario generation methods for stochastic programming. Pacific Journal of Optimization* 3(2):257–271.

Kenyon AS, Morton DP, 2003 *Stochastic vehicle routing with random travel times. Transportation Science* 37(1):69–82.

Kleywegt AJ, Shapiro A, Homem-De-Mello T, 2002 *The sample average approximation method for stochastic discrete optimization. SIAM Journal on Optimization* 12(2):479–502.

Lam E, Van Hentenryck P, 2016 *A branch-and-price-and-check model for the vehicle routing problem with location congestion. Constraints* 21(3):394–412.

Laporte G, Louveaux F, Mercure H, 1992 *The vehicle routing problem with stochastic travel times. Transportation Science* 26(3):161–170.

Lei H, Laporte G, Bo G, 2012 *A generalized variable neighborhood search heuristic for the capacitated vehicle routing problem with stochastic service times. TOP* 20(1):99–118.

Li H, Bai M, Zhao Y, Dai C, 2019 *Vehicle flow formulation for two-echelon time-constrained vehicle routing problem. Journal of Management Science and Engineering* 4(2):75–90.

Li X, Tian P, Leung SCH, 2010 *Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm. International Journal of Production Economics* 125(1):137–145.

Lium AG, Crainic TG, Wallace SW, 2009 *A study of demand stochasticity in service network design. Transportation Science* 43(2):144–157.

Mak W, Morton DP, Wood RK, 1999 *Monte Carlo bounding techniques for determining solution quality in stochastic programs. Operations Research Letters* 24(1–2):47–56.

Miranda DM, Conceição SV, 2016 *The vehicle routing problem with hard time windows and stochastic travel and service time. Expert Systems with Applications* 64:104–116.

Nasiri MM, Rahbari A, Werner F, Karimi R, 2018 *Incorporating supplier selection and order allocation into the vehicle routing and multi-cross-dock scheduling problem. International Journal of Production Research* 56(19):6527–6552.

Neves-Moreira F, da Silva DP, Guimarães L, Amorim P, 2018 *The time window assignment vehicle routing problem with product dependent deliveries. Transportation Research Part E Logistics & Transportation Review* 116:163–183.

Oyola J, Arntzen H, Woodruff DL, 2017 *The stochastic vehicle routing problem, a literature review, Part II: solution methods. EURO Journal on Transportation and Logistics* 6(4):349–388.

Oyola J, Arntzen H, Woodruff DL, 2018 *The stochastic vehicle routing problem, a literature review, Part I: models. EURO Journal on Transportation and Logistics* 7(3):193–221.

Rahbari A, Nasiri MM, Werner F, Musavi M, Jolai F, 2019 *The vehicle routing and scheduling problem with cross-docking for perishable products under uncertainty: Two robust bi-objective models. Applied Mathematical Modelling* 70:605–625.

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

Russell RA, Urban TL, 2008 *Vehicle routing with soft time windows and erlang travel times. Journal of the Operational Research Society* 59:1220–1228.

Shell, 2018 *Diesel price.* `https://www.shell.com.sg/motorists/shell-fuels/shell-station-price-board.html`, last accessed on 27 September 2019.

Spliet R, Dabia S, Van Woensel T, 2018 *The time window assignment vehicle routing problem with time-dependent travel times. Transportation Science* 52(2):261–276.

Spliet R, Desaulniers G, 2015 *The discrete time window assignment vehicle routing problem. European Journal of Operational Research* 244(2):379–391.

Spliet R, Gabor AF, 2015 *The time window assignment vehicle routing problem. Transportation Science* 49(4):721–731.

Subramanyam A, Wang A, Gounaris CE, 2018 *A scenario decomposition algorithm for strategic time window assignment vehicle routing problems. Transportation Research Part B: Methodological* 117:296–317.

Taş D, Dellaert N, Van Woensel T, De Kok T, 2013 *Vehicle routing problem with stochastic travel times including soft time windows and service costs. Computers & Operations Research* 40(1):214–224.

Taş D, Dellaert N, Van Woensel T, De Kok T, 2014a *The time-dependent vehicle routing problem with soft time windows and stochastic travel times. Transportation Research Part C: Emerging Technologies* 48:66–83.

Taş D, Gendreau M, Dellaert N, Van Woensel T, De Kok AG, 2014b *Vehicle routing with soft time windows and stochastic travel times: A column generation and branch-and-price solution approach. European Journal of Operational Research* 236(3):789–799.

The Land Transport Authority of Singapore, 2018 *Vehicle fuel economy data.* https://vrl.lta.gov.sg/rpt/vrl/data_arch/VT035W201909212155052699.pdf.zip, last accessed on 27 September 2019.

The Ministry of Manpower, 2018 *Hours of work, overtime and rest day.* https://www.mom.gov.sg/employment-practices/hours-of-work-overtime-and-rest-days, last accessed on 18 August 2018.

Van Belle J, Valckenaers P, Cattrysse D, 2012 *Cross-docking: State of the art. Omega* 40(6):827–846.

Vareias AD, Repoussis PP, Tarantilis CD, 2019 *Assessing customer service reliability in route planning with self-imposed time windows and stochastic travel times. Transportation Science* 53(1):256–281.

Verweij B, Ahmed S, Kleywegt AJ, Nemhauser G, Shapiro A, 2003 *The sample average approximation method applied to stochastic routing problems: A computational study. Computational Optimization & Applications* 24(2–3):289–333.

Wang X, Crainic TG, Wallace SW, 2019 *Stochastic network design for planning scheduled transportation services: The value of deterministic solutions. INFORMS Journal on Computing* 31(1):153–170.

Wen M, Larsen J, Clausen J, Cordeau JF, Laporte G, 2009 *Vehicle routing with cross-docking. Journal of the Operational Research Society* 60(12):1708–1718.

Yuan B, Liu R, Jiang Z, 2015 *A branch-and-price algorithm for the home health care scheduling and routing problem with stochastic service times and skill requirements. International Journal of Production Research* 53(24):7452–7464.