Zhen, L., Wu, J., Wang, S., He, X., & Tian, X. (2024). Courier routing for a new last-mile logistics service. IISE Transactions, 57(8), 957–975. https://doi.org/10.1080/24725854.2024.2366295

Courier routing for a new last-mile logistics service

Abstract: As a new business mode for last-mile logistics services, some instant delivery platforms provide "help me buy" services mainly to satisfy urgent customer demands. Couriers travel to nearby stores, buy commodities requested by the customer, and quickly deliver them to the customer's location. We investigate how to operate this type of platform to maximize profits. A two-stage stochastic programming model is proposed to determine whether to accept a customer's order, how to assign accepted orders to couriers, and how to select stores where the commodities are purchased. The proposed model can be applied using a rolling horizon approach and account for the uncertain arrival of future orders in each epoch. Moreover, because the second-stage subproblem involves an integer programming model, a new decomposition algorithm is proposed to solve the two-stage model. Extensions are also explored so that our proposed methodology can be applied to more general and realistic platform operations. Numerical experiments based on real data are conducted to test the effectiveness of the proposed algorithm and to derive useful managerial insights for operators of help me buy services. This study indicates the necessity of applying the new decomposition to this new delivery model, considering future orders, and allowing platform collaboration, and determines the benefit of the stochastic solution. In addition, this study reveals the influences of the courier-to-order ratio; response time; distribution and radius of order, courier, and store circles; demand density; and decision frequency.

Keywords: Courier routing; last-mile logistics; order assignment; platform revenue; column generation.

1. Introduction

In the era of mobile commerce and online shopping, people are gradually adapting to buying goods from online shopping malls such as Amazon, Tmall, and JD Mall. The companies that operate these online shopping malls are usually mega-companies; they invest heavily in building large-scale, advanced warehouses and high-efficiency logistics networks to realize timely delivery for customers. Delivery time for online shopping orders has drastically decreased. If a customer places an order at an online shopping mall before 23:00, the customer can usually receive their order the next day. However, customers may urgently require certain routine commodities, which are common in some physical stores. Customers hope to obtain these commodities within one or two hours, but they are not willing (or it is inconvenient) to go to nearby stores to buy them in person. They hope that someone can buy the required commodities and deliver them to their home quickly. Such customers are more concerned with timely delivery than with the price of the commodities. To satisfy such demands, a new business mode of last-mile logistics services, that is, instant delivery platforms that provide “help me buy” services, has appeared in recent years. Some instant delivery platforms, such as Instacart, Postmates, Hummingbird, Meituan, and SF Express, have started operating online help me buy services, through which customers can purchase commodities from several stores in a single order and receive their full order on the same day or even within the hour (Zhen et al., 2022).

Different from traditional online shopping malls, instant delivery platforms are third-party platforms. They need not operate their own warehouse or logistics network; they can hire part-time couriers to buy the commodities ordered by customers from groceries and shops and then promptly deliver the commodities to the customers. Figure 1 shows the interface of the instant delivery platform operated by SF Express, a popular Chinese instant delivery platform. Through the app, customers can describe the commodities they want to buy (i.e., Step1 and Step 2). Subsequently, the reference price and purchase location range are shown in the interface (i.e., Step 3). The reference price is pre-set by the instant delivery platform. Potential purchase locations fall within a specific range (i.e., 3 km for SF Express). If a customer accepts the reference price and location range, they place their order (requests) on the platform (i.e., Step 4). The orders enter the platform (the orders that the platform needs to make the decision) are those that customers have already accepted after considering potential price variations and locations. Requests include the order time, location, and commodity category. After receiving a certain number of requests in a given period, the platform runs a decision algorithm to appropriately match customer orders with couriers. A customer order can be accepted or declined in each epoch. Each accepted order is assigned to an available courier. The courier starts at their current location, travels to a store to purchase the commodity, and delivers it by the customer’s deadline.

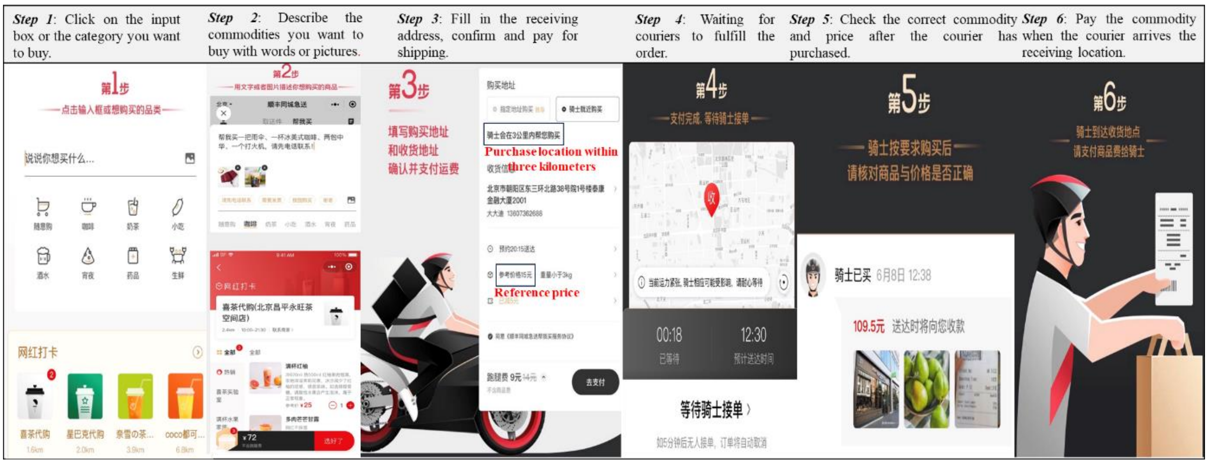


Figure 1: Help me buy delivery service on the SF Express platform

Because help me buy services are an emerging business mode, it is important to investigate how to operate them efficiently so that they can not only satisfy urgent customer demands but also maximize the profits of platform operators. As third-party service providers, instant delivery platforms do not operate their own stores but utilize local brick-and-mortar stores. Revenue can be earned by a platform if a commodity in a store is sold through the platform. Different stores charge different prices, and a platform earns a percentage of a store's sales revenue. A platform can select which store to purchase the requested commodities from to maximize revenue. The key problem with help me buy services is that, unlike traditional online retailers, their customers' requests are urgent. Thus, delivery deadlines are usually a strong constraint on couriers' schedules. Consistent with general courier routing practices in the delivery industry, instant delivery platforms must minimize the lengths of routes that couriers take to fill orders. Therefore, platforms must both choose stores and assign orders to couriers carefully as these decisions affect courier routes (travel costs). Thus, tools driven by mathematical models are needed to support such operational decisions.

Any online algorithm for such a platform should account for the uncertain arrival of orders in the near future in each epoch when it is used to make real-time decisions. Thus, this study adopts a stochastic programming methodology to build a two-stage model featuring an integer recourse function in the second stage. To efficiently solve this complex model, a new decomposition algorithm is designed, which can also pave the way for the design of advanced decision-making tools to assist platform operators. Using real data from an instant delivery platform company based in Shanghai (the largest metropolis in China), numerical experiments are conducted to test the efficiency of the proposed decomposition algorithm, demonstrate the benefits of applying the stochastic programming methodology to our problem, and derive managerial insights through sensitivity analysis, which may be useful for practitioners in the instant delivery industry. In addition, the proposed model is extended through the relaxation of three assumptions and validated by experiments so that our proposed methodology can be applied to more general and realistic contexts in the service industry.

The remainder of this paper is organized as follows. Section 2 reviews the relevant literature. Section 3 describes the background of the problem. Sections 4 and 5 propose the mathematical model and algorithm, respectively. Section 6 presents numerical experiments, and Section 7 extends the model. Section 8 presents the closing remarks.

2. Literature review

Because help me buy services are a new business mode, few studies focus on such platforms or their services. Below, we review related studies from three perspectives.

One key feature of the help me buy service is on-demand delivery. The instant delivery platform is an on-demand service provider that enables customers to buy commodity from local brick-and-mortar stores online. Analytical studies investigate various decision problems faced by on-demand service platforms. For example, Taylor (2018) investigates the influences of delay sensitivity and agent independence on on-demand platforms' optimal service price and wage. Kung and Zhong (2017) analyze service pricing and incentive strategies so that service platforms can achieve win–win solutions for themselves and their couriers. Bai et al. (2019) analyze the relationship between earnings-sensitive independent endogenous service providers and the price-sensitive endogenous customers of on-demand service platforms. He et al. (2020) propose methods for determining the optimal commission that online service platforms should charge retailers and the optimal wages these platforms should pay couriers. Benjaafar et al. (2022) study the government perspective and propose a model for selecting an appropriately sized labor pool and a suitable lower bound (LB) for the nominal wage to maintain agent welfare. Fatehi and Wagner (2022) design a robust optimization model for labor planning and pricing for timely delivery service, which considers customers' required delivery time windows. Aspects such as platform operations (Bahrami et al. 2021), demand management (Yildiz and Savelsbergh 2020) and workforce scheduling (Ulmer and Savelsbergh 2020) have also been investigated in related on-demand delivery problems.

Matching optimization problems are also important to on-demand platforms. Boysen et al. (2019) investigate optimization problems for deterministic matching decisions made by on-demand car-sharing platforms. Li et al. (2019) propose an integrated model for order allocation and routing for an online retailer. Özkan (2020) studies the relationship between pricing and matching decisions for ride-sharing platforms, concluding that it is necessary to jointly consider pricing and matching decisions and that matching decisions may be more important than pricing decisions. As the subject of our study, help me buy service platforms combine commodity pickup and delivery with on-demand services. Few studies research this subject. We consider realistic factors and uncertainty and design mixed-integer programming (MIP) models for order acceptance (Klapp et al., 2020), order assignment, and store selection by an on-demand delivery platform. Moreover, while we do not explicitly consider pricing strategy because platforms usually do not have the right to determine product prices, we consider store selection because different stores may sell their products at different prices. Thus, we implicitly consider price-related decisions and revenue maximization.

The help me buy service is a last-mile delivery service for online retailers. Unlike traditional online retailers (usually mega-retailers), help me buy services use small stores near their customers as "pre-position warehouses" to satisfy urgent customer demand. For example, Qi et al. (2018) propose models for a large-scale last-mile delivery system based on shared (i.e., crowdsourced) mobility, which is also an important feature of help me buy services. Arslan et al. (2019) investigate the potential benefits of crowdsourced last-mile delivery and design an exact algorithm for the real-time ad hoc matching of parcel delivery tasks and drivers. Dayarian and Savelsbergh (2020) investigate a new method of last-mile delivery for retailers in which in-store customers can deliver online orders on their way home; their study proposes rolling-horizon dispatching methods for use in a dynamic and stochastic context. Wang (2019) proposes an MIP model for an emerging last-mile transportation service in which routes and schedules are optimized for a fleet of delivery vehicles to minimize passenger waiting and travel time. Sun et al. (2018) design an exact algorithm for a pickup and delivery problem considering time windows and time-dependent travel times on arcs. Sun et al. (2020) further account for the profit factor in their objective. Voccia et al. (2019) consider future order arrivals in the routing decisions of an instant delivery platform. Wang et al. (2020) propose a comprehensive decision model for services offering last-mile delivery of e-commerce and online-to-offline parcels. Using game theory to improve the efficiency of last-mile delivery, Deng et al. (2021) conduct a comparative study of urban consolidation centers and peer-to-peer capacity-sharing platforms. Their proposed methodology also provides a means to investigate the benefit of last-mile delivery services to customers and society. Among studies applying MIPs to model last-mile delivery for online retailers, some focus on instant food delivery services. Yildiz and Savelsbergh (2019) design both a model and a column-and-row generation algorithm for the timely delivery of food that becomes available within minutes of a customer's order. Considering the random time at which food will be ready, Ulmer et al. (2021) propose an order assignment model for the stochastic arrival of orders on food delivery platforms. Liu et al. (2021) study last-mile food delivery and design an order assignment decision tool that involves travel time predictors; a major contribution of their study is its consideration of the routing behavior of couriers to increase the timeliness of last-mile delivery.

In summary, few studies investigate instant delivery platforms that provide "help me buy" services. Arslan et al. (2021) conduct a pioneering work on this new type of platform and adopt a Markov decision process to investigate order splitting and consolidation strategies with the aim of increasing the number of served orders. Zhen et al. (2022) use MIP to design models and dynamic programming-based algorithms for order assignment and shopper routing. Zhen et al. (2022) investigate long-term decisions such as mode adoption and territory planning. Our study differs considerably from those in the literature in terms of our problem features and methodology. We consider order assignment, price or revenue optimization, uncertain future orders, and a rolling horizon. To solve this problem, we propose a two-stage stochastic programming model. As the two-stage model has an integer recourse function in the second stage, we design a new decomposition algorithm to solve large-scale instances. Our proposed decomposition algorithm is applicable to a wide range of two-stage stochastic integer programming models in which the second-stage subproblem has a set partitioning formulation. The aforementioned differences of our study contribute to the literature on the optimization of operational decisions for online service platforms.

3. Problem description

The decision maker in the problem studied in this paper is an instant delivery platform that operates a help me buy service. Customers place orders, indexed by r, on the platform. For simplicity, we assume that each order contains one commodity required by the customer. The platform earns a percentage n of the revenue from the commodities sold by each store. Each commodity may be sold by several stores in different locations, and the commodity's price at different stores is not identical. Stores are indexed by s. Suppose that ϱ_{rs} is the price of the commodity of known order $r$at store s. We define the parameter p_{rs} as the platform's revenue if order r's commodity is purchased from store s; here, $p_{rs} = n \cdot \varrho_{rs}$. Although several stores may sell the same commodity at different prices, these price differences are small. The platform must decide where to buy the commodity for each order, complicating order assignment.

The platform earns revenue from stores because it helps them sell their commodities, but it pays couriers, indexed by k, a fee for filling customer orders. Because instant delivery platforms are third-party platforms, they usually hire part-time couriers to avoid the costs incurred by idle couriers. For each order, the fee (i.e., the platform's cost) depends on the total distance traveled. We define the parameter d_{ijk} as the cost of courier k traveling from node i to node j; here, a node may be the location of a store, the location of a customer, or the origin of a courier. The objective of the problem is to maximize the platform's profits, which equal the aforementioned revenue minus the travel costs for the couriers who fill the orders. Thus, choosing the most expensive store is not always the most economical for the platform. The platform needs to make careful decisions about assigning orders to geographically dispersed couriers and from which store to purchase the commodity required for each order. Notably, some orders cannot be assigned to any courier because their locations are too far from the available couriers and the cost to fill them would be greater than the benefit gained. Orders not accepted by the platform will not be rolled over to the next epoch, resulting in additional waiting time and possible future demand loss. As a problem constraint, a response time limit for each order must be satisfied; here, the response time for each order is the interval between the assignment of the order to a courier and the time the commodity is delivered to the customer.

Per the usual practice of online scheduling systems, the above decisions are based on a rolling horizon, and orders are assigned in batches. In each epoch, new orders and backlogged orders are assigned to available couriers; in the remainder of this paper, these orders are referred to as “known orders.” An important feature of this problem is its consideration of future orders, which reflects decision uncertainty. We primarily consider the uncertain locations of orders arriving in the near future (i.e., orders accumulated during a batch period from the current epoch to the subsequent epoch). However, the uncertainty of the commodities requested in the orders is not considered, because it is much easier to predict the locations of future orders during a particular time slot from historical data than it is to predict the specific commodities ordered by customers.

We use an example to explain the context of this decision-making process. Suppose that the current epoch begins at 18:00 and a batch period is 5 minutes. The set of known orders includes all orders that arrive to the platform from 17:55 to 18:00 and all backlogged orders, that is, orders arriving before 17:55. For known orders, the customer locations (i.e., delivery locations) and required commodities are deterministic, and their assignment decisions must be made in the current epoch. By analyzing data from recent days on order arrivals during the 18:00–18:05 time slot, we can determine which areas may generate orders during this time slot and the probability of this occurring. Then, following the usual practice of handling uncertainty in stochastic programming, we can generate a set of future scenarios, each containing a number of future orders with deterministic locations. Notably, the “customer” for each future order cannot be an individual customer, but it can be a block within an urban district, for which it is easy to use statistics and historical data to predict future orders during a certain time slot.

The primary reason for considering future orders is that if a future order’s customer is close to a known order’s customer, the future order can be assigned to the courier working on the known order. In this case, couriers can serve more customers by traveling short distances. The cost of filling a future order is the cost of traveling from a known order’s customer location to a future order’s customer location; these two orders are assigned to the same courier. As mentioned above, the uncertain commodity requested in the future order is not considered. We assume that the future order’s commodity can be purchased at the store where the courier purchases the commodity for the known order. This assumption is rational because the commodities ordered on this help me buy service platform are usually common products sold in most stores. For simplicity, because specific information about commodities is unavailable, the unit benefit for undertaking one future order is assumed to be constant; in reality, the unit benefit is the average benefit of all of the orders in this time slot, according to historical data.

Here, we summarize the background of the problem presented above. Given a set of known orders, the platform decides whether to accept each order, assigns each accepted order to an available courier, and decides from which store to purchase the commodity required for each accepted order. The objective is to maximize the platform’s profits. The platform must make a careful decision to balance the revenue earned from the selected stores and the fees (travel costs) paid to the assigned couriers. For example, some orders' locations are too far from the available couriers and the travel cost to fill them would be greater than the revenue earned by the platform. In addition, filling the known orders of customers who are close to the customers placing future orders has high priority in the above decision-making process, as reflected in the future scenarios in the second stage of the formulated model.

Before formulating the mathematical model of the problem, we outline the following important assumptions:

(1) Each courier is assigned to at most one known order in each epoch.

(2) Couriers do not fill other orders during a decision epoch.

(3) Each order is for one commodity.

(4) Couriers stay in the same place after order completion.

For simplicity and to facilitate the presentation of our proposed methodology, the model is first formulated under the above assumptions. We subsequently extend the model by relaxing the first three assumptions so that our proposed methodology is applicable to more realistic scenarios. Details of the relaxation of these assumptions are provided in Section 7. With these assumptions relaxed, our method can be applied to contexts in which a courier can be assigned to more than one known order, couriers with ongoing orders can be assigned to new orders, and an order may require several commodities, which can be purchased from different stores.

4. Mathematical model

We propose a two-stage stochastic programming model for a third-party instant delivery platform that runs a help me buy service. The two-stage framework is used in other planning problems for instant delivery services (Khir et al., 2021). In the first stage, our proposed model decides which known orders are accepted, which courier is assigned to each accepted known order, and from which store to purchase the commodity required for each accepted order. In the second stage, the model makes decisions for the future orders in each scenario; these decisions are the same as those made in the first stage, except no store selection decision is made.

4.1 Notation

Before formulating the model, we define the parameters and decision variables used in our model. For ease of understanding, we use Roman and Greek letters to denote the parameters and the variables, respectively.

Indices and sets

R set of all known orders, indexed by r, which also denotes the location of an order's customer;

O set of all future orders, indexed by o, which also denotes the location of an order's customer;

K set of all couriers, indexed by k;

S set of all stores, indexed by s, which also denotes a store's location;

Ω set of all scenarios, indexed by ω;

$e(k)$ index of the current location for courier k, i.e., the origin of the courier's route;

i, j index of all types of nodes (locations) in the delivery network, $i, j \in R \cup O \cup S \cup \{e(k)\}$.

Parameters

p_{rs} revenue if known order r's commodity is bought from store s;

d_{ijk} courier k's travel cost from node i to node j;

t_{ijk} courier k's travel time from node i to node j;

z_{rs} waiting time at store s for handling known order r;

u_r limit of the response time for known order r;

f unit benefit for undertaking one more future order;

π^{ω} probability of scenario ω.

Decision variables

α_{rs} equals one if the commodity required by known order r is bought at store s, otherwise zero;

β_{rk} equals one if known order r is assigned to courier k, otherwise zero;

θ_{rsk} equals one if courier k buys known order r's required commodity at store s, otherwise zero;

γ_{ok}^{ω} equals one if future order o is assigned to courier k in scenario ω, otherwise zero;

$\varepsilon_{rok}^{\omega}$ equals one if future order o is assigned to courier k who undertakes order r, otherwise zero.

4.2 Two-stage stochastic programming model

Based on the above definition of parameters and decision variables, a two-stage stochastic programming model (denoted by M1) is formulated as follows.

$$\textbf{[M1]}\ \text{Max} \sum_{r\in R}\sum_{s\in S}\left[\alpha_{rs}p_{rs} - \sum_{k\in K}\theta_{rsk}\left(d_{e(k)sk} + d_{srk}\right)\right] + \mathcal{Q}(\boldsymbol{\beta}, \omega) \tag{1}$$

subject to

$$\sum_{s\in S}\alpha_{rs} \le 1 \qquad \forall r \in R \tag{2}$$

$$\sum_{k\in K}\beta_{rk} = \sum_{s\in S}\alpha_{rs} \qquad \forall r \in R \tag{3}$$

$$\sum_{r\in R}\beta_{rk} \le 1 \qquad \forall k \in K \tag{4}$$

$$\theta_{rsk} = (\alpha_{rs} + \beta_{rk} - 1)^{+} \qquad \forall r \in R, \forall s \in S, \forall k \in K \tag{5}$$

$$\sum_{s\in S}\theta_{rsk}\left(t_{e(k)sk} + z_{rs} + t_{srk}\right) \le u_r \qquad \forall r \in R, \forall k \in K \tag{6}$$

$$\alpha_{rs} \in \{0,1\} \qquad \forall r \in R, \forall s \in S \tag{7}$$

$$\beta_{rk} \in \{0,1\} \qquad \forall r \in R, \forall k \in K \tag{8}$$

$\theta_{rsk} \in \{0,1\}$ $\quad \forall r \in R, \forall s \in S, \forall k \in K. \quad (9)$

Objective (1) maximizes the total profit, which equals the revenue for filling the accepted known orders, i.e., $\sum_{r\in R}\sum_{s\in S}\alpha_{rs}p_{rs}$, minus the couriers' travel cost for filling these known orders, i.e., $\sum_{r\in R}\sum_{s\in S}\sum_{k\in K}\theta_{rsk}\left(d_{e(k)sk}+d_{srk}\right)$, and plus the expected profit of filling future orders in a set of scenarios, i.e., the objective of the second stage, denoted by $Q(\boldsymbol{\beta},\omega)$; the details of $Q(\boldsymbol{\beta},\omega)$ are elaborated later. Constraint (2) guarantees that the commodity required by a known order is bought at one store if the known order is accepted; otherwise, the commodity will not be bought at any store. Constraint (3) ensures that the two parts $\sum_{k\in K}\beta_{rk}$ and $\sum_{s\in S}\alpha_{rs}$ equal one (or zero) simultaneously, which means that the known order r is accepted (or is not accepted). Constraint (4) guarantees that each courier is assigned at most one known order. Constraint (5) links the decision variables θ_{rsk}, α_{rs}, and β_{rk} and ensures that θ_{rsk} equals one if α_{rs} and β_{rk} equal one simultaneously; otherwise, θ_{rsk} equals zero. Constraint (6) guarantees that the response time for a known order should not exceed the limit; here, the response time is a duration from the time the known order is assigned to a courier to the time the order's commodity is delivered to the customer. Constraints (7)–(9) define the domains of the decision variables in the first stage.

As the core influence on order assignment, the decision variable $\boldsymbol{\beta}$ links the first stage and the second stage. The objective of the second stage, denoted by $Q(\boldsymbol{\beta},\omega)$, is to maximize the expected profit of undertaking future orders in a set of scenarios indexed by ω. The second-stage model is as follows.

$$Q(\boldsymbol{\beta},\omega) = \text{ Max} \sum_{\omega\in\Omega}\pi^{\omega}\left[\sum_{o\in O}\sum_{k\in K}\left(\gamma_{ok}^{\omega}f-\sum_{r\in R}\varepsilon_{rok}^{\omega}d_{rok}\right)\right] \quad (10)$$

subject to

$$\sum_{k\in K}\gamma_{ok}^{\omega} \le 1 \quad \forall o \in O, \omega \in \Omega \quad (11)$$

$$\sum_{o\in O}\gamma_{ok}^{\omega} \le \sum_{r\in R}\beta_{rk} \quad \forall k \in K, \omega \in \Omega \quad (12)$$

$$\varepsilon_{rok}^{\omega} = (\beta_{rk}+\gamma_{ok}^{\omega}-1)^{+} \quad \forall r \in R, o \in O, k \in K, \omega \in \Omega \quad (13)$$

$$\gamma_{ok}^{\omega} \in \{0,1\} \quad \forall o \in O, k \in K, \omega \in \Omega \quad (14)$$

$$\varepsilon_{rok}^{\omega} \in \{0,1\} \quad \forall r \in R, o \in O, k \in K, \omega \in \Omega. \quad (15)$$

Objective (10) is to maximize the expected revenue minus the cost for undertaking future orders in a set of scenarios. Constraint (11) ensures that each future order is assigned to at most one courier. Constraint (12) guarantees that each courier is assigned at most one future order. Constraint (13) links the decision variables $\varepsilon_{rok}^{\omega}$, β_{rk}, and γ_{ok}^{ω} and ensures that $\varepsilon_{rok}^{\omega}$ equals one if β_{rk} and γ_{ok}^{ω} equal one simultaneously; otherwise, $\varepsilon_{rok}^{\omega}$ equals zero. Constraints (14)–(15) define the domains of the decision variables in the second stage.

5. A decomposition algorithm

To solve our proposed two-stage stochastic integer programming model (i.e., M1) efficiently, a decomposition algorithm is designed to solve this model with large-scale instances in a reasonable time. Section 5.1 elaborates the framework of this proposed algorithm. Details of the components embedded in the algorithm are presented in Sections 5.2–5.4.

5.1 Algorithmic framework

Our proposed algorithm is similar to the Benders decomposition approach in that it uses a decomposition framework to solve two-stage stochastic programming models. However, it is different from (i.e., the opposite of) Benders decomposition in that it passes the *primal* columns from the second-stage subproblem to the first-stage master problem; in contrast, Benders decomposition passes dual solution-based cuts (rows) to the master problem. Our proposed algorithm is efficient in solving two-stage stochastic programming models when the second-stage subproblem contains integer variables; in such cases, traditional Benders decomposition does not apply. As stated above, the columns from the second stage, called s-columns, are passed to the first stage; this implies an assumption that the second-stage subproblem (i.e., $Q(\boldsymbol{\beta}, \omega)$) has a set partitioning formulation whose relaxation can be solved effectively using a column generation (CG) procedure. The s-columns approximate the second-stage decision-making process and are passed to the first-stage master problem. To solve the first-stage problem, which also contains a decision to select s-columns for each scenario, we can use a suitable solution (or algorithm) that can efficiently solve real-world-scale instances. We adopt a CG-based method to solve the first-stage master problem containing the s-columns. In this step, another type of column, the f-column, is defined (i.e., the assumptions) for the first-stage problem. An f-column contains not only the first-stage decisions but also the selections of the s-columns for all scenarios. In addition, we use the *CG procedure*, which is only one part of the traditional CG solution method, to solve the second-stage subproblem, whereas we use the *whole traditional CG method* to solve the first-stage problem with the s-columns from the second stage.

For each specific problem, the process by which the proposed decomposition algorithm solves model M1 is briefly described below. The flowchart of the proposed algorithm contains several models that must be solved by either an algorithm or the CPLEX solver.

Step 0. Obtain an initial assignment scheme of known orders by solving model $\mathrm{M}^{\mathrm{Stage1}}$.

Step 1. Input the initial assignment scheme of known orders into model $\mathrm{M}^{\mathrm{Stage2}}$, which is constructed for each scenario.

Step 2. Run the CG procedure to generate the s-columns for each scenario. On the basis of the s-columns generated in the second stage (scenarios), the initial model, M1, is transformed into model $\mathrm{M1^{Equ}}$, which contains the decision to choose a future order assignment plan (column) for each scenario instead of scenario-related decisions.

Step 3. Run the traditional CG procedure to solve model $\mathrm{M1^{Equ}}$. If the obtained solution does not improve after a certain number of iterations, then terminate the algorithm; otherwise, return to *Step 1*.

5.2 Generating s-columns of the second-stage subproblem

Each s-column denotes an assignment plan of future order for a specific courier, determining which accepted future order is assigned to the courier in each scenario. The s-columns are generated using the column-generating procedure to solve the second-stage subproblem. As set forth above, the column-generating procedure is part of the traditional CG-based solution method; we still need to formulate the restrict master problem (RMP) and the pricing problem (PP) for the second-stage subproblem according to the usual CG practice. At the beginning of the whole algorithm, we need to obtain an initial solution for the first-stage decisions so that the second-stage subproblem can be formulated, enabling us to further formulate the RMP and the PPs. Here, the initial solution for the first-stage decisions can be obtained by solving the following model $\mathrm{M^{Stage1}}$:

$$[\mathbf{M^{Stage1}}]\ \mathrm{Max}\ \sum_{r\in R}\sum_{s\in S}\left[\alpha_{rs}p_{rs}-\sum_{k\in K}\theta_{rsk}\left(d_{e(k)sk}+d_{srk}\right)\right] \tag{16}$$

subject to Constraints (2)–(9).

We can use the CPLEX to solve the above model by setting a maximum computation time and then output a feasible solution for the remainder of the algorithmic process. Alternatively, we can use intuitive but practical decision rules to find a feasible solution for the model. For example, we can assign each known order to a store and an unassigned courier with the maximum profit, i.e., $\max_{\forall s,k}\{p_{rs}-d_{e(k)sk}-d_{srk}\}$.

Given a solution for the first-stage decision, we construct the second-stage model $\mathrm{M^{Stage2}}$, which is formulated for one of the scenarios $\omega\in\Omega$.

$$[\mathbf{M^{Stage2}}]\ \mathrm{Max}\ \sum_{\omega\in\Omega}\pi^{\omega}\left[\sum_{o\in O}\sum_{k\in K}(\gamma_{ok}^{\omega}f-\sum_{r\in R}\varepsilon_{rok}^{\omega}d_{rok})\right] \tag{17}$$

subject to Constraints (11)–(15).

As explained at the beginning of this subsection, we need to formulate the RMP and PPs for the above model according to the usual CG practice. The formulation of the RMP is based on the definition of the column, i.e., the s-column, which denotes an assignment plan of future order for a specific courier. The set of all feasible assignment plans (s-columns) for courier k in this scenario is defined by $\tilde{\mathcal{P}}_k$. Assignment plans in set $\tilde{\mathcal{P}}_k$ are indexed by $\wp$. We define a binary variable $\tilde{\lambda}_{\wp}$ for each feasible assignment plan $\wp\in\tilde{\mathcal{P}}_k$, which equals one if the future order assignment plan $\wp$ is chosen for courier

k and zero otherwise. A binary parameter $\tilde{A}_o^{\wp}$ is defined to denote whether future order o is assigned to plan $\wp$. Let $\tilde{c}_{\wp}$ be the profit of each assignment plan $\wp$. To construct an initial set of s-columns for the RMP, we assign the closest unassigned future order to each courier who filled the known order r (i.e., $\beta_{rk} = 1$) if the value of $\min_{\forall o} d_{rok}$ is no greater than f in each scenario; otherwise, the courier is not assigned to any future order in the scenario. In addition, the binary variable $\tilde{\lambda}_{\wp}$ is relaxed to a continuous variable. The linear relaxed RMP (LR-RMP) for the second-stage subproblem is as follows:

$$[\mathbf{M}_{\mathbf{LR-RMP}}^{\mathbf{Stage2}}] \text{ Max } \sum_{k \in K} \sum_{\wp \in \tilde{\mathcal{P}}_k} \tilde{c}_{\wp} \tilde{\lambda}_{\wp} \tag{18}$$

subject to

$$\sum_{\wp \in \tilde{\mathcal{P}}_k} \tilde{\lambda}_{\wp} \leq 1 \qquad \forall k \in K \tag{19}$$

$$\sum_{k \in K} \sum_{\wp \in \tilde{\mathcal{P}}_k} \tilde{A}_o^{\wp} \tilde{\lambda}_{\wp} \leq 1 \qquad \forall o \in O \tag{20}$$

$$0 \leq \tilde{\lambda}_{\wp} \leq 1 \qquad \forall k \in K, \wp \in \tilde{\mathcal{P}}_k. \tag{21}$$

Objective (18) maximizes the total profit of the selected s-columns (plans). Constraint (19) guarantees that each courier has at most one feasible plan; the dual variable for these constraints is defined as μ_k, $\forall k \in K$. Constraint (20) ensures that each future order is assigned to at most one courier; the dual variable for these constraints is defined as ν_o, $\forall o \in O$. Constraint (21) defines the domains of the decision variables. During the column-generating procedure, the dual variables of the above model are transferred to the PP to generate new future order assignment plans (i.e., the s-columns).

The PP model determines a feasible future order assignment plan with a negative reduced cost. During each iteration of the CG procedure, the PP is decomposed into $|K|$ subproblems, each of which generates a feasible assignment plan for the future orders for one courier. We define a binary variable γ_o^{ω}, which equals one if future order o is assigned to the courier in scenario ω and zero otherwise. The PP model is set forth as follows:

$$\left[\mathbf{M}_{\mathbf{PP}}^{\mathbf{Stage2}}\right] \text{ Max } \tilde{c}_{\wp} - \left(\mu_k + \sum_{o \in O} \nu_o \gamma_o^{\omega}\right) \tag{22}$$

subject to Constraints (12)–(15) in which the courier index k is omitted in the subscripts of the related parameters and the variables

$$\tilde{c}_{\wp} = \gamma_o^{\omega} f - \sum_{r \in R} \varepsilon_{ro}^{\omega} d_{ro} \qquad \forall \omega \in \Omega. \tag{23}$$

Objective (22) maximizes the reduced cost of the optimal assignment plan. Constraint (23) determines the cost of the assignment plan for the courier. An exact algorithm for solving the PP is proposed in this paper. The pseudocode of the algorithm is addressed in Appendix A. The time complexity of the algorithm is $O(|R| + |O|)$.

5.3 Solving the first-stage master problem with s-columns of scenarios

The procedure in the previous subsection derives sets of future order assignment plans (s-columns) for the couriers in each scenario. Then, given these generated s-columns in all scenarios, we can approximate the second-stage decision making in the first-stage master problem by simply choosing assignment plans (s-columns) for each scenario.

The first-stage master problem with future order assignment plans (s-columns) is defined by model $\mathrm{M1^{Equ}}$, which is equivalent to the original model M1. Before formulating model $\mathrm{M1^{Equ}}$, the newly defined parameters and decision variables are listed as follows.

Newly defined index and sets:

$\tilde{\mathcal{P}}_k(\omega)$ set of all feasible assignment plans (s-columns) for courier k in scenario ω, indexed by $\wp$.

Newly defined parameters:

$c_{k\wp}^{\omega}$ profit of assignment plan (s-column) $\wp$ for courier k in scenario ω;

$x_{ok\wp}^{\omega}$ equals one if future order o is assigned to courier k in scenario ω in assignment plan $\wp$;

$b_{rk\wp}^{\omega}$ equals one if the assignment plan $\wp$ of courier k in scenario ω covers known order r, otherwise zero.

Newly defined decision variables:

$\psi_{k\wp}^{\omega}$ equals one if assignment plan $\wp$ is selected for courier k in scenario ω, $\omega \in \Omega, \wp \in \tilde{\mathcal{P}}_k(w)$;

$\sigma_{ok\wp}^{\omega}$ equals one if assignment plan $\wp$ including future order o is selected for courier k in scenario ω, $\omega \in \Omega, \wp \in \tilde{\mathcal{P}}_k(\omega)$.

Next, the original model M1's equivalent model is formulated as follows.

$$[\mathbf{M1^{Equ}}]\ \text{Max}\quad \sum_{r\in R}\sum_{s\in S}\left[\alpha_{rs}p_{rs} - \sum_{k\in K}\theta_{rsk}\left(d_{e(k)sk} + d_{srk}\right)\right] + \sum_{\omega\in\Omega}\sum_{k\in K}\sum_{\wp\in\tilde{\mathcal{P}}_k(\omega)}\pi^{\omega}\psi_{k\wp}^{\omega}c_{k\wp}^{\omega} \tag{24}$$

subject to Constraints (2)–(9)

$$\sum_{\wp\in\tilde{\mathcal{P}}_k(w)}\psi_{k\wp}^{\omega} \le \sum_{r\in R}\beta_{rk} \qquad \forall k\in K, \omega\in\Omega \tag{25}$$

$$\psi_{k\wp}^{\omega} \le \sum_{r\in R} b_{rk\wp}^{\omega} \qquad \forall k\in K, \omega\in\Omega, \wp\in\tilde{\mathcal{P}}_k(\omega) \tag{26}$$

$$\sum_{k\in K}\sum_{\wp\in\tilde{\mathcal{P}}_k(\omega)}\sigma_{ok\wp}^{\omega} \le 1 \qquad \forall o\in O, \omega\in\Omega \tag{27}$$

$$\psi_{k\wp}^{\omega}x_{ok\wp}^{\omega} = \sigma_{ok\wp}^{\omega} \qquad \forall o\in O, k\in K, \omega\in\Omega, \wp\in\tilde{\mathcal{P}}_k(\omega) \tag{28}$$

$$\psi_{k\wp}^{\omega} \in \{0,1\} \qquad \forall k\in K, \omega\in\Omega, \wp\in\tilde{\mathcal{P}}_k(\omega) \tag{29}$$

$$\sigma_{ok\wp}^{\omega} \in \{0,1\} \qquad \forall o\in O, k\in K, \omega\in\Omega, \wp\in\tilde{\mathcal{P}}_k(\omega). \tag{30}$$

With s-columns in sets $\tilde{\mathcal{P}}_k(w)$, all of the second-stage decisions for our scenarios can be reflected by Constraints (25)–(30) in the first-stage master problem. Constraint (25) guarantees that each courier is assigned at most one assignment plan (s-column) for each scenario if the courier filled a known order; otherwise, the courier will not be assigned any assignment plan. Constraint (26) ensures that each assignment plan can be assigned to each courier if the assignment plan covers known order r. Constraint (27) guarantees that each future order is assigned to at most one assignment plan for each scenario. Constraint (28) links the newly defined variables $\psi_{k\wp}^{\omega}$ and $\sigma_{ok\wp}^{\omega}$. Constraints (29)–(30) define the domains of the newly defined variables.

For some large-scale instances, model $\mathrm{M1}^{\mathrm{Equ}}$ is an intractable integer programming model that cannot be solved effectively by CPLEX. Thus, we design a CG-based solution method to generate a near-optimal solution for this problem. Next, we reformulate model $\mathrm{M1}^{\mathrm{Equ}}$ as a set covering model for which the column is named the f-column. Each f-column captures the known order assignment and future order assignment plans of all scenarios for each courier, determining whether to accept the known order, which courier is assigned to the accepted known order, which store is selected to provide the commodity requested for the order, and which s-column is selected. The set of all feasible assignment plans (f-columns) for known order r is defined by Γ_r. An assignment plan (f-column) in set Γ_r is indexed by ℓ_r. We define a binary variable λ_{ℓ_r} for each feasible assignment plan $\ell_r \in \Gamma_r$, which is equal to one if the order assignment plan ℓ_r is chosen for known order r and zero otherwise. Let c_{ℓ_r} be the cost of each assignment plan ℓ_r. Each f-column ℓ_r contains three binary parameters: $U_k^{\ell_r}$ equals one if known order r is assigned to courier k in column ℓ_r, $L_{k\omega\wp}^{\ell_r}$ equals one if plan $\wp \in \tilde{\mathcal{P}}_k(\omega)$ is selected in column ℓ_r, and $D_{ok\omega\wp}^{\ell_r}$ equals one if future order o in plan $\wp \in \tilde{\mathcal{P}}_k(\omega)$ is selected in column ℓ_r. Based on the above definition of the f-columns, the master problem (MP) for model $\mathrm{M1}^{\mathrm{Equ}}$ is formulated as follows.

$$[\mathbf{M1}_{\mathbf{MP}}^{\mathbf{Equ}}]\ \mathrm{Max} \sum_{r\in R}\sum_{\ell_r\in\Gamma_r} c_{\ell_r}\lambda_{\ell_r} \tag{31}$$

subject to

$$\sum_{\ell_r\in\Gamma_r}\lambda_{\ell_r} = 1 \qquad \forall r \in R \tag{32}$$

$$\sum_{r\in R}\sum_{\ell_r\in\Gamma_r} U_k^{\ell_r}\lambda_{\ell_r} \le 1 \qquad \forall k \in K \tag{33}$$

$$\sum_{\wp\in\tilde{\mathcal{P}}_k(\omega)}\sum_{r\in R}\sum_{\ell_r\in\Gamma_r} L_{k\omega\wp}^{\ell_r}\lambda_{\ell_r} \le 1 \qquad \forall k \in K, \omega \in \Omega \tag{34}$$

$$\sum_{k\in K}\sum_{\wp\in\tilde{\mathcal{P}}_k(\omega)}\sum_{r\in R}\sum_{\ell_r\in\Gamma_r} D_{ok\omega\wp}^{\ell_r}\lambda_{\ell_r} \le 1 \qquad \forall o \in O, \omega \in \Omega \tag{35}$$

$$\lambda_{\ell_r} = \{0,1\} \qquad \forall r \in R, \ell_r \in \Gamma_r. \tag{36}$$

Objective (31) maximizes the total profit of plans. Constraint (32) guarantees that the known order has one feasible plan. Constraint (33) means that each courier is assigned at most one known order. Constraint (34) ensures that each courier is assigned at most one assignment plan for each scenario,

$\wp \in \tilde{\mathcal{P}}_k(\omega)$. Constraint (35) ensures that each future order is assigned to at most one courier for each scenario. Constraint (36) defines the domains of the decision variables.

To reduce the solution space and computation time, we define an RMP containing a subset of all of the feasible assignment plans in the CG-based algorithm, which is denoted by $\ell_r \subseteq \Gamma_r{}'$; in addition, the binary variable λ_{ℓ_r} is relaxed to a continuous variable. The LR-RMP for model $\mathrm{M1^{Equ}}$ is formulated below.

$$[\mathbf{M1^{Equ}_{LR-RMP}}]\ \mathrm{Max}\ \sum_{r\in R}\sum_{\ell_r\in\Gamma_r{}'} c_{\ell_r}\lambda_{\ell_r} \tag{37}$$

subject to Constraints (32)–(35) in which Γ_r is replaced by $\Gamma_r{}'$

$$0 \le \lambda_{\ell_r} \le 1 \qquad \forall r \in R, \ell_r \in \Gamma_r{}'. \tag{38}$$

At each iteration, the dual variables of the above model are transferred to the PP to generate new f-columns. μ_r, ν_k, $m_{k\omega}$, and $z_{o\omega}$ are the dual variables for Constraints (32)–(35) in which Γ_r is replaced by $\Gamma_r{}'$, respectively. Given these values of the dual variables, we construct the PP to determine a feasible order assignment plan with a positive reduced cost. During each iteration, the PP is decomposed into $|R|$ subproblems, each of which is oriented to a courier and generates a feasible plan of known order assignments and future order assignment plans of all scenarios for the courier. The PP's parameters and variables are as follows.

$$[\mathbf{M1^{Equ}_{PP}}]\ \mathrm{Max}\quad c_{\ell_r} - \Big(\mu_r + \sum_{k\in K}\nu_k\beta_k + \sum_{\omega\in W}\sum_{k\in K}\sum_{\wp\in\tilde{\mathcal{P}}_k(\omega)} m_{k\omega}\psi^{\omega}_{k\wp} + \sum_{\omega\in W}\sum_{o\in O}\sum_{k\in K}\sum_{\wp\in\tilde{\mathcal{P}}_k(\omega)} z_{o\omega}\sigma^{\omega}_{ok\wp}\Big) \tag{39}$$

subject to Constraints (2)–(3), (5)–(9), (25)–(30) in which the known order index r is omitted in the subscripts of the related parameters and the variables

$$c_{\ell_r} = \sum_{s\in S}\big[\alpha_s p_s - \theta_{sk}\big(d_{e(k)sk} + d_{sk}\big)\big] + \sum_{\omega\in\Omega}\sum_{k\in K}\sum_{\wp\in\tilde{\mathcal{P}}_k(\omega)} \pi^{\omega}\psi^{\omega}_{k\wp}c^{\omega}_{k\wp}. \tag{40}$$

Objective (39) maximizes the reduced cost of the optimal assignment plan. Constraint (40) determines the cost of the assignment plan to the known order. We propose an exact algorithm to solve the above PP efficiently. The pseudocode is shown in Appendix B. The time complexity of the algorithm is $O\big(|K||S||\Omega|\big|\ddot{\mathcal{P}}_k\big|\big)$. Here $\ddot{\mathcal{P}}_k$ is a set $\tilde{\mathcal{P}}_k(\omega)$ with maximum number of s-columns for all scenarios.

Finally, after obtaining a column pool that consists of a number of feasible assignment plans, we use the CPLEX to solve the previously proposed model $\mathrm{M1}^{Equ}_{MP}$ to obtain a first-stage integer solution.

5.4 Complexity of the decomposition algorithm

The framework of the proposed decomposition algorithm includes CG procedures that are used to solve the first-stage and second-stage problems. Therefore, the complexity of the CG procedures for solving each stage's problem is first analyzed, and then the overall complexity of the decomposition algorithm is obtained.

For each scenario ω, the time complexity of the RMP is $\mathrm{O}(2^{\tilde{n}})$, where $\tilde{n}$ is the number of variables in the RMP (Spielman and Teng, 2004); $\tilde{n} = |\cup_{k\in K} \tilde{\mathcal{P}}_k(\omega)|$ (i.e., the number of s-columns) and is equal to $|K|(|R| + |O|)$. In addition, $\mathrm{O}(|R| + |O|)$ is the time complexity of each s-column. For each scenario ω, the RMP as well as the PPs are solved at most $\tilde{n}$ times. Thus, the computational complexity for each scenario ω is $\mathrm{O}\left(\tilde{n} \times \left(|R| + |O| + 2^{\tilde{n}}\right)\right) = \mathrm{O}(\tilde{n} \times 2^{\tilde{n}})$, where $\tilde{n} = |K|(|R| + |O|)$. Similarly, for the first stage, the time complexity is $\mathrm{O}\left(n \times \left(|K||S||\Omega|\left|\ddot{\mathcal{P}}_k\right| + 2^{n}\right)\right) = \mathrm{O}(n \times 2^{n})$, where $n = |\cup_{r\in R} \Gamma_r'|$, which denotes the number of f-columns and is equal to $|R||K||S||\Omega|\left|\ddot{\mathcal{P}}_k\right|$.

Recall that the termination condition for the while loop in the algorithm (Section 5.1): "If the obtained solution does not improve after a certain number of iterations, then terminate the algorithm." Here, "a certain number" is a constant, and the number of scenarios is $|\Omega|$. Thus, the time complexity of the overall primal decomposition algorithm is $\mathrm{O}\left(|\Omega| \times \tilde{n} \times \left(|R| + |O| + 2^{\tilde{n}}\right)\right) + \mathrm{O}\left(n \times \left(|K||S||\Omega|\left|\ddot{\mathcal{P}}_k\right| + 2^{n}\right)\right)$, where $\tilde{n} = \max\{\left|\cup_{k\in K} \tilde{\mathcal{P}}_k(\omega)\right|, \forall\omega \in \Omega\}$ and $n = |\cup_{r\in R} \Gamma_r'|$. As $|\cup_{r\in R} \Gamma_r'|$ is usually larger than $\left|\cup_{k\in K} \tilde{\mathcal{P}}_k(\omega)\right|$; hence, the time complexity of the proposed decomposition algorithm can be expressed as $\mathrm{O}(n \times 2^{n})$, where $n = |\cup_{r\in R} \Gamma_r'|$.

6. Numerical experiments

Numerical experiments are conducted on a workstation with an Intel Xeon E5-2643 v3 CPU at 3.40 GHz and 128 GB of RAM to validate the proposed model and test the efficiency of the proposed decomposition algorithm. The algorithm is programmed in C# (VS2019) concert technology with the ILOG CPLEX 12.6.1 solver. The time limit for solving the instances is 3,600 seconds.

6.1 Experimental settings

We use the distribution network of an instant delivery platform company in Shanghai for our experiments. We use the platform's real data set, which contains data on approximately 2,700 orders per day. Using the platform's operational data from one day (August 31, 2022), we illustrate demand heat maps of order delivery locations and store locations providing the commodities for the orders in the left and right parts of Figure 2, respectively; the number of orders per hour for that day is shown in Figure 3. The node-to-node distance (i.e., d_{ijk}) is the real travel distance shown on Baidu Maps.

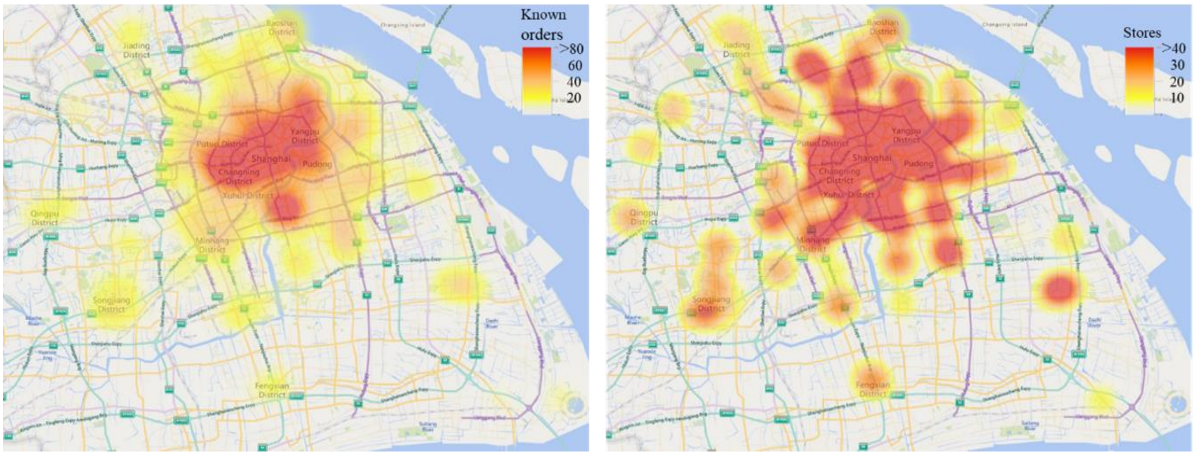


Figure 2: Heat maps of order delivery locations (left) and store locations (right) on the platform

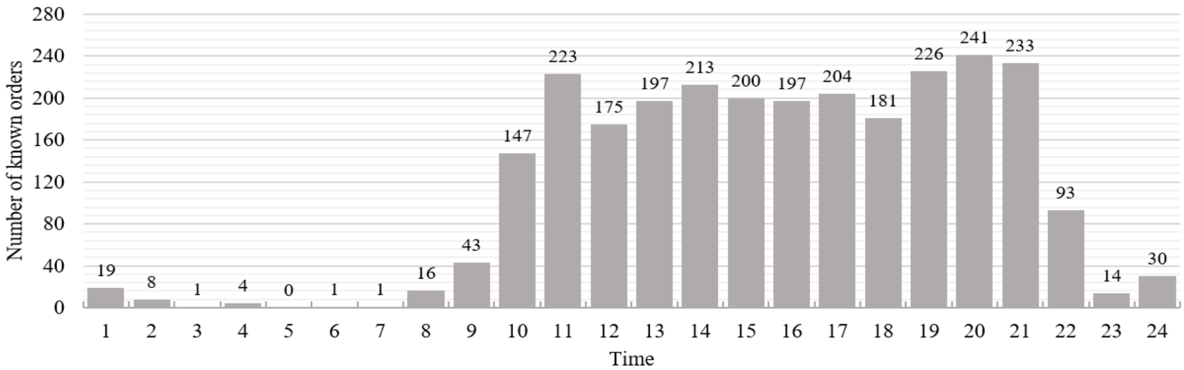


Figure 3: Number of orders per hour for one day (August 31, 2022) on the platform

Six groups of instances (ISG1–ISG6) with different scales are used in this experimental study. Table 1 shows the detailed settings for these groups. As mentioned in Section 3 (problem description), the proposed decision model is solved using a rolling horizon, and orders are assigned in batches. The time between two consecutive decision epochs is 12 minutes; the maximum number of orders accumulated per period (12 minutes) for 24 hours is approximately 48, according to the data visualized in Figure 3 (i.e., 20:00 on the horizontal axis). Thus, the instance group with the largest scale involves 50 orders. The maximum numbers of candidate stores and couriers related to orders are set accordingly, as shown in Table 1. The above batch period setting implies that the computation time for large-scale instances should not exceed the batch period length (i.e., 12 minutes); otherwise, our proposed methodology could not be applied in real-world contexts. For each instance group, the approximate numbers of model variables and constraints are listed in Table 1.

Table 1: Scale of instance groups in experiments

Group ID	No. of orders	No. of stores	No. of couriers	No. of future orders	No. of scenarios	No. of decision variables	No. of constraints
ISG1	6	3	10	6	5	2.4×10^3	4.5×10^3
ISG2	8	4	15	8	5	6.0×10^3	1.2×10^4

ISG3	10	5	20	10	5	1.2×10^4	2.4×10^4
ISG4	30	15	40	30	10	3.9×10^5	7.7×10^5
ISG5	40	20	50	40	15	1.3×10^6	2.5×10^6
ISG6	50	30	60	50	20	3.2×10^6	6.2×10^6

6.2 Solution quality

We first test small-scale instances by comparing the optimal results obtained by CPLEX, our proposed decomposition algorithm, and an LB, which removes all response time constraints (i.e., Constraint (6)) and the second stage (i.e., Constraints (11)–(15)). As shown in Table 2, the solution obtained by our decomposition algorithm has a narrow average difference from the optimal solution obtained by CPLEX (i.e., 0.35%). Our decomposition algorithm requires much less computation time than CPLEX requires for all instances. CPLEX can find the optimal solution for only eight known orders in 3,600 seconds. When the number of known orders increases from 8 to 10, CPLEX cannot obtain the optimal solutions in a reasonable time. Our decomposition algorithm runs efficiently and stably, and its average computation time is only 1.70 seconds.

As shown in Table 2, CPLEX cannot obtain the optimal solutions in a reasonable time; thus, it is unsuitable as a benchmark for evaluating the quality of the solutions obtained by our proposed decomposition algorithm. To further validate the efficiency of our proposed algorithm, we conduct experiments on large-scale instances to compare the solutions obtained by our decomposition algorithm with those obtained by the LB. The results shown in Table C-1, Appendix C reveal that our decomposition algorithm is highly efficient in solving large-scale instances, and its average computation time is 109.49 seconds. The difference between the objective values of our algorithm and the LB ranges between 11.97% and 14.86%, with an average of approximately 13.23%. Moreover, our proposed decomposition algorithm can solve instances with approximately 3 million integer variables and 6 million constraints (i.e., ISG6) in 4 minutes, which is much shorter than the previously used threshold of 12 minutes (i.e., the batch period).

Table 2: Algorithmic performance for small-scale instances

Instances		CPLEX		LB	Decomposition algorithm			
Group	ID	F_{CPLEX}	t_{CPLEX}	F_{LB}	F_{PD}	t_{PD}	GAP_1	GAP_2
ISG1	1-1	83.91	4.46	72.40	83.60	1.07	0.37%	15.47%
	1-2	77.49	4.09	66.45	77.49	1.34	0.00%	16.61%
	1-3	95.26	5.01	83.83	94.87	0.65	0.41%	13.17%
	1-4	89.60	6.74	78.21	89.38	0.84	0.25%	14.28%
	1-5	78.68	6.80	67.30	78.36	1.24	0.41%	16.43%
ISG2	2-1	109.34	260.61	94.06	108.90	1.66	0.40%	15.78%
	2-2	123.52	128.84	108.17	123.04	1.54	0.39%	13.75%
	2-3	121.00	226.69	105.59	120.42	1.97	0.48%	14.04%

	2-4	115.64	204.70	100.24	115.17	1.66	0.41%	14.89%
	2-5	134.53	121.49	119.16	134.00	2.49	0.39%	12.45%
ISG3	3-1	—	—	140.62	159.27	2.83	—	13.26%
	3-2	—	—	124.10	142.86	2.22	—	15.12%
	3-3	—	—	153.74	172.49	1.65	—	12.20%
	3-4	—	—	128.46	144.75	2.43	—	12.68%
	3-5	—	—	114.66	133.61	1.92	—	16.53%
Average						1.70	0.35%	14.44%

Notes: (1) F_{CPLEX}, F_{LB} and F_{PD} represent the objective value of solution obtained by CPLEX, LB and decomposition algorithm, respectively. (2) t_{CPLEX} and t_{PD} are the computation time of CPLEX and decomposition algorithm in seconds, respectively. (3) $Gap_1 = (F_{PD} - F_{CPLEX})/F_{CPLXE}$; $Gap_2 = (F_{PD} - F_{LB})/F_{LB}$.

6.3 Benefits of stochastic programming

One important feature of this study is its adoption of stochastic programming to account for uncertain orders in the near future. Here, we compare three methods for studying the value of the stochastic solution and the value of perfect information (Avriel and Williams, 1970). These three methods are detailed in Appendix C.2; the objectives of the three methods are denoted by F_1, F_2, and F_3. The solution obtained by Method 2 is also a feasible solution by Method 1, and F_2 is smaller than F_1. The difference between the two values represents the value of the stochastic solution (Val_{Stoc}), which evaluates the benefit of considering (or the cost of ignoring) uncertainty in decisions.

$$Val_{Stoc} = F_1 - F_2. \tag{41}$$

Information about future orders cannot be predicted. Thus, Method 3 cannot exist in reality, and F_3 serves as an upper bound (UB) for Method 1. The gap between F_1 and F_3 represents the value of perfect information (Val_{Info}), indicating how much platform operators are willing to pay to accurately predict random future orders before making their decision.

$$Val_{Info} = F_3 - F_1. \tag{42}$$

We investigate two instance groups, ISG1 and ISG2, with various travel distances for future orders. The results in Table 3 show that F_1 is greater than F_2, which illustrates the need to use a two-stage stochastic programming model to manage uncertainty. Profits from known orders account for the majority of the objective value. These profits derived by Methods 1 and 2 are shown in columns C_1 and C_2, respectively. Although C_2 is sometimes lower than C_1, the schedule obtained by Method 2 is not better than that obtained by Method 1. The optimal schedule can be obtained using a deterministic model and the estimated travel distance; however, this method may not be optimal in the face of uncertainties. As stated above, F_3 is a UB for F_1. The gap between F_1 and F_3 highlights the benefit of accurately forecasting the travel distance required to complete future orders. The platform should use historical data to improve the accuracy of its predictions of future orders.

Table 3: Value of the stochastic solution and value of perfect information

Instance		Method 1		Method 2		Method 3	Val_{Stoc}		Val_{Info}	
Group	ID	F_1	C_1	F_2	C_2	F_3	Δ_1	Gap_1	Δ_2	Gap_2
ISG1	1-6	87.45	77.65	86.07	76.84	88.67	1.38	1.58%	1.22	1.40%
	1-7	81.78	71.83	80.26	71.26	82.86	1.52	1.86%	1.08	1.32%
	1-8	80.38	70.54	79.98	70.29	81.65	0.40	0.50%	1.27	1.58%
	1-9	77.50	67.47	76.05	66.83	78.88	1.45	1.87%	1.38	1.78%
	1-10	69.82	61.52	68.85	61.08	70.97	0.97	1.39%	1.15	1.65%
ISG2	2-6	121.96	107.98	119.82	106.70	123.36	2.14	1.75%	1.40	1.15%
	2-7	102.94	89.06	101.42	88.35	103.99	1.52	1.48%	1.05	1.02%
	2-8	106.38	92.26	103.65	90.83	107.47	2.73	2.57%	1.09	1.02%
	2-9	108.42	94.72	106.33	93.30	109.91	2.09	1.93%	1.49	1.37%
	2-10	110.80	97.50	108.55	95.42	112.42	2.25	2.03%	1.62	1.46%
Average		94.74		93.10		96.02		1.70%		1.38%

Notes: (1) F_1, F_2 and F_3 represent the best solution obtained by *Method 1*, *Method 2* and *Method 3*, respectively. (2) C_1 and C_2 represent the cost of known orders assignment scheme in the first stage obtained by *Method 1*, *Method 2*, respectively. (3) $\Delta_1 = F_1 - F_2$, $\Delta_2 = F_3 - F_1$. (4) $Gap_1 = \Delta_1/F_1$, $Gap_2 = \Delta_2/F_1$.

We also compare the values of Val_{Stoc} and Val_{Info} for ISG2 with various travel distances for future orders. M_{var} denotes the average deviation of the actual travel distance from the expected travel distance in the scenarios. The results are shown in Table C-3, Appendix C. The results indicate the higher M_{var}, the more significant Val_{Stoc} and Val_{Info} values. This confirms the importance of stochastic programming, especially when the M_{var} values for scenarios are large.

We further test ISG1 and ISG2 to determine how the size of the selected scenario set affects the value of the stochastic solution. A scenario set consists of a predetermined number of scenarios, denoted by Ω. For the original scenario set, $|\Omega| = 20$. We select scenarios from the original set as test scenarios, denoted by $\Omega' \subseteq \Omega$. The size of the selected scenario set refers to the specific number of selected scenarios (i.e., $|\Omega'| \leq |\Omega|$). We change the size of the selected scenario set (i.e., $|\Omega'| = 5, 10, 15,$ or 20). The first-stage solution obtained with the selected scenario set is substituted into the stochastic model with the original scenario set. We compare the objective value obtained by Method 1 under various sizes of selected scenario sets with that obtained by Method 2. The results are shown in Table C-4, Appendix C. The results demonstrate that larger the selected scenario set, the higher the value of the stochastic solution (Val_{Stoc}). This implies that considering stochastic programming and appropriately selecting the size of the scenario set are important.

6.4 Managerial insights

This section discusses sensitivity analyses used to derive managerial insights and support platform operators' tactical decisions.

(1) Sensitivity analysis of the courier-to-order ratio

We conduct a sensitivity analysis of the courier-to-order ratio, which accounts for both known orders and future orders. Six series of experiments are conducted using ISG4, and the results are shown in Figure 4. In the first, second, and third series of experiments (Figures 4(a), 4(b), and 4(c), respectively), three numbers of orders (24, 30, and 36, respectively) are considered, while the number of stores is set to 15; in the fourth, fifth, and sixth series of experiments (Figures 4(d), 4(e), and 4(f), respectively), three numbers of stores (10, 15, and 20, respectively) are considered, while the number of orders is set to 30. The trends of these six series are similar. The objective value (profit) increases as the courier-to-order ratio increases; when the ratio exceeds a threshold, the objective value gradually flattens. However, the increasing trend is not monotonic. The reason for this nonmonotonically increasing trend is that the greater the number of couriers that can be selected by the platform is, the greater the possibility of reducing costs and thus increasing profits. However, when the number of candidate couriers reaches a threshold, this effect disappears. Thus, the platform should include an appropriate number of candidate couriers in the model when it runs each batch of orders. If too few couriers are involved, a suboptimal plan may result.

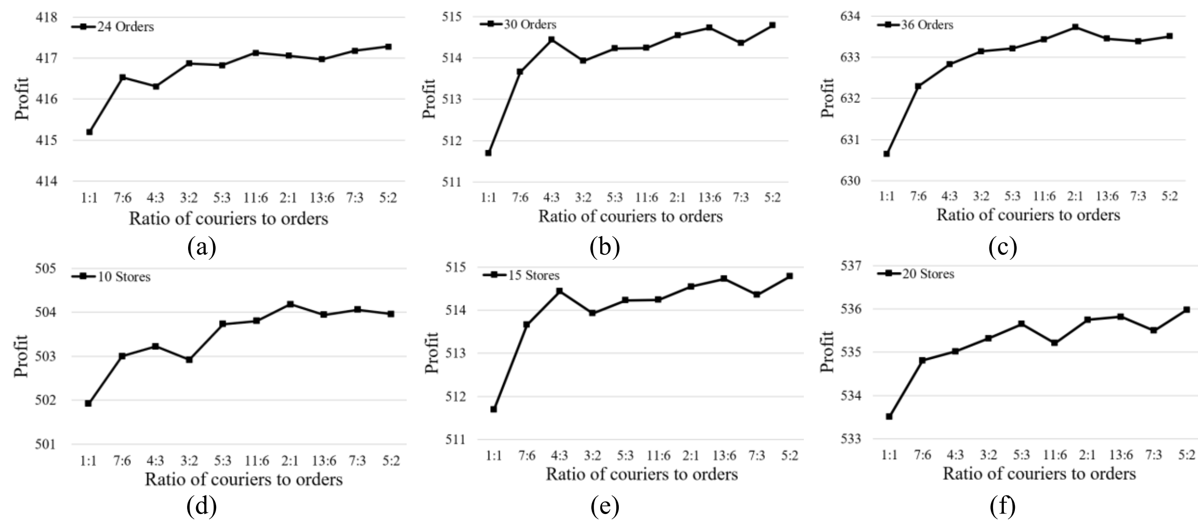


Figure 4: Sensitivity analysis of courier-to-order ratio

As mentioned, the number of stores (i.e., 15) is identical in the first, second, and third series of experiments, but as the number of orders increases from 24 to 30 and 36 (Figures 4(a), 4(b), and 4(c), respectively), the profit ranges of the three curves increase; this result is expected as more orders lead to higher profits. In the fourth, fifth, and sixth series of experiments, the number of orders is again identical (i.e., 30). When the number of stores increases from 10 to 15 and 20 (Figures 4(d), 4(e) and 4(f), respectively), the profit ranges of these three curves also increase; the reason for this finding may be that as more can be selected by the platform, the likelihood of reducing costs or increasing profits increases. However, the more stores or orders are in the model, the longer the platform's decision time (i.e., the model's computation time). As the platform's decision time may affect its level of service to customers, the platform should strike a balance between decision time and expected profits.

(2) Sensitivity analysis of response time limit

As mentioned in Section 3, the response time for each order is defined as the interval between the assignment of the order to a courier and the delivery of the commodity to the customer. The promised response time u_r reflects the service level of the platform. We conduct a sensitivity analysis of this parameter using ISG4. The response time range is 30 to 50 minutes with a step size of 1 minute; the results for the 21 instances are shown in Figure 5, in which the horizontal axis indicates the average known-order response time (i.e., $Avg_{\forall r} u_r$) for each instance. The curve in Figure 5 shows that for response time limits of less than 41 minutes, the longer the response time limit, the greater the profits obtained by the platform. When the response time limit is above this threshold, the profits of the platform cannot increase further. Therefore, the platform should not relax the service level guarantee (i.e., promised response time) too much, as doing so does not always increase profit; it should instead carefully establish an appropriate response time limit to properly balance its service level guarantee and expected profit.

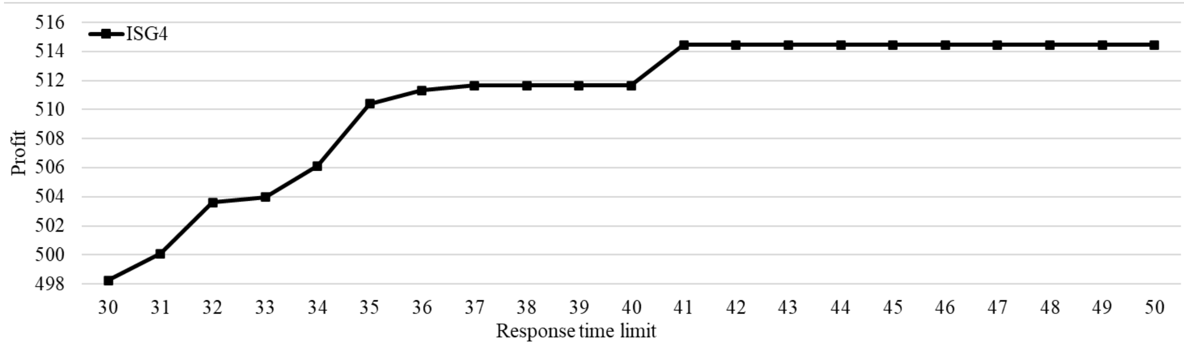


Figure 5: Sensitivity analysis of response time limit

(3) Sensitivity analysis of order, courier, and store circle distribution

The order delivery locations form a customer area; similarly, the store locations and original courier locations form store and courier areas, respectively. The relative positions of these three areas may influence the final profits of the platform. To investigate this issue, we construct instances by assuming that these three areas are circles and that the locations of 40 orders (20 stores/50 couriers) are evenly dispersed throughout the order circle (store circle/courier circle); the radius of each circle is 3 km.

First, we assume that the centers of the three circles form an equilateral triangle with a side length of 3, 4, 5, or 6 km. The results in Figure 6(a) demonstrate that the shorter the distance between each pair of circles, the greater the profits earned by the platform. The reason for this result is that when the three circles are close to each other or overlap to a large extent, this benefits the platform by reducing the travel costs of couriers, thus increasing its profits.

We gradually move the center of one of the three circles along the aforementioned triangle's axis of symmetry away from the other two centers. As indicated by the horizontal axis in Figure 6(b), the distance between this moving center and one of the other two static centers changes from the baseline case (i.e., 3 km) to 4, 5, and 6 km. The results shown in Figures 6(b) and 6(b) demonstrate similar trends; the shorter the distances between the circles are, the greater the profits earned by the platform. However, the magnitude of this effect in the three series of experiments is not identical. The effect on profits of moving the center of the store circle is the largest of the three effects, and the effect of moving the center of the courier circle is larger than that of moving the center of the order circle. In real-world platform operations, decision-makers should consider the relative positions of these three circles (areas). Incentive policies and negotiation with cooperative partners (stores) should be carefully planned and implemented so that these circles are near each other, yielding high platform profits.

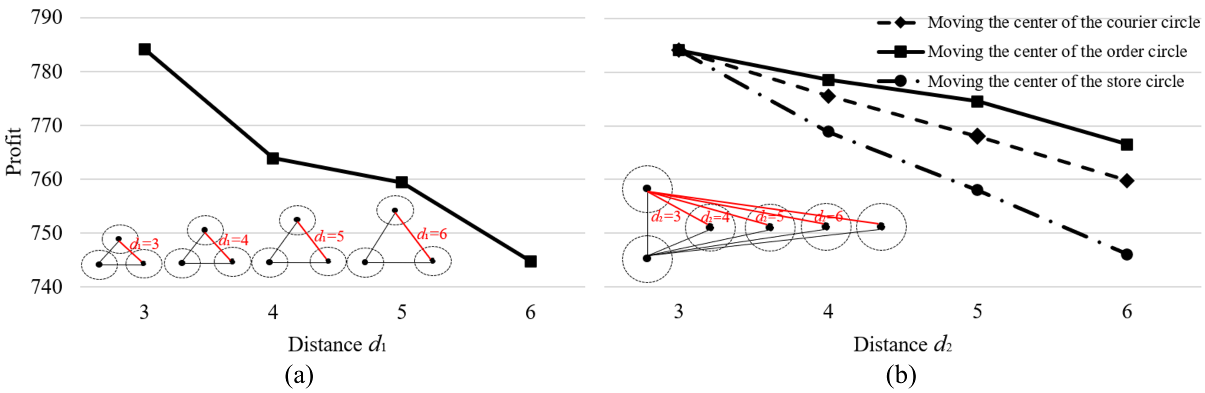


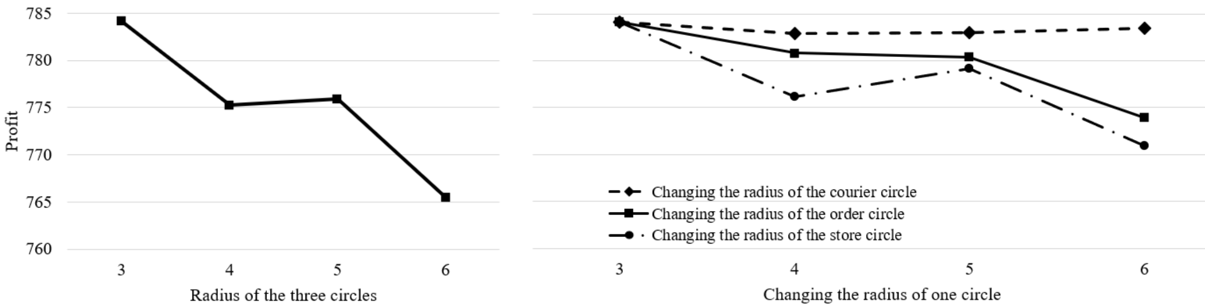
Figure 6: Sensitivity analysis of order, courier, and store circle distribution

(4) Sensitivity analysis of the order, courier, and store circle radii

In addition to the relative locations of the three areas, the sizes of these areas may influence the final outcome (i.e., platform profits). Using our baseline case in which the centers of the three circles with 3-km radii form an equilateral triangle, two series of experiments are conducted.

In the first series of experiments, the radii of the three circles increase from 3 to 4, 5, and 6 km. The results are shown in Figure 7(a). When the radii increase, the profits of the platform do not always decrease, but the overall trend is decreasing. The reason for this trend of nonmonotonically decreasing as the radii increase is that the larger the order, courier, and store circles, the greater the probability that the total distance traveled by the couriers will be long, which is likely to decrease platform profits.

In the second series of experiments, we change the radius of one of the three circles (the radius is set to 3, 4, 5, or 6 km, as indicated by the horizontal axis in Figure 7(b) and keep the other radii unchanged; thus, we present three curves in Figure 7(b). The three curves in Figure 7(b) have different trends. The radius of the courier circle does not seem to influence profits significantly. The radius of the store circle shows a similar trend to that observed our first series of experiments: profits nonmonotonically decrease as the radius increases; however, this nonmonotonic decrease is more significant than that for the curve in our first series of experiments (i.e., for the three circles). Finally, the curve for the store circle indicates that profits monotonically decrease as the radius increases. These results show that the radii of the order and store circles have an effect on profits, and that the effect of the radius of the store circle is more dynamic than that of the radius of the order circle.



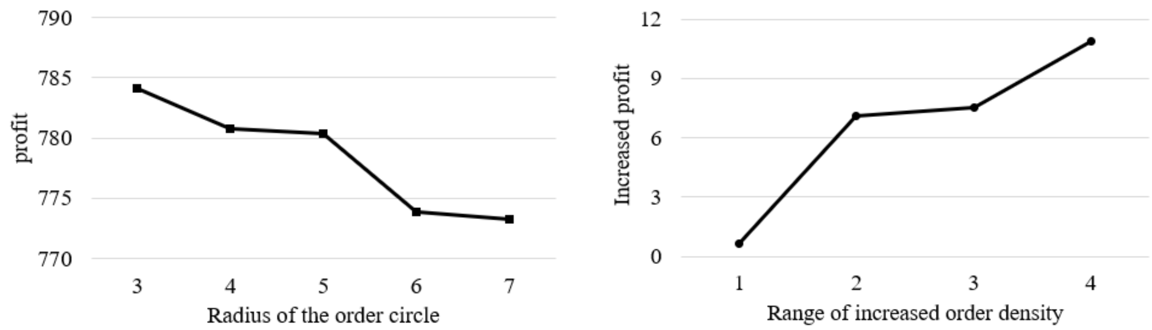
(a) (b)

Figure 7: Sensitivity analysis of the radius of the order, courier, and store circles

(5) Effect of demand density

According to the results in Figure 7(b), the radii of the order and store circles have significant effects on profits. This result reflects the possible effect of demand (order) density, which can be estimated from different areas' population density. Beginning with the three circles discussed in Section 6.4 (*4*), we further change the radius of the order circle (40 orders) from 3 to 4, 5, 6, and 7 km. The same 40 orders are distributed inside circles with decreasing radii, indicating decreasing demand density. The results in Figure 8(a) demonstrate that the greater the demand density, the higher the profits earned by the platform. The reason for this finding is that when the demand density is high, this benefits the platform by reducing the travel costs of couriers, thus increasing its profits.

We subsequently focus on the effects of the range of increased demand density. The range of increased demand density is set to 1, 2, 3, and 4. The results in Figure 8(b) indicate that the increase in profits from increasing the demand density to various ranges differs notably. More specifically, profits increase the most with the change in the range of increased demand density from 1 to 2, but profits do not increase as the range of demand density increases from 2 to 3. Although the platform is likely to obtain greater profits by offering "help me buy" services in high-density areas than in low-density areas, platform managers should set a reasonable range of increased demand density to avoid wasting resources (i.e., setting the range of increased density in the above instances to 2 rather than to 3).



(a) (b)

Figure 8: Effect of demand density

(6) Benefits of platform collaboration

Different platforms could collaborate to earn more profits by sharing their orders and couriers. We obtain real data from three platforms in Shanghai and Hangzhou (i.e., the most representative city in China in terms of e-commerce). Figure 9 shows delivery location heat maps for all orders received in one day by the three platforms in these two cities.

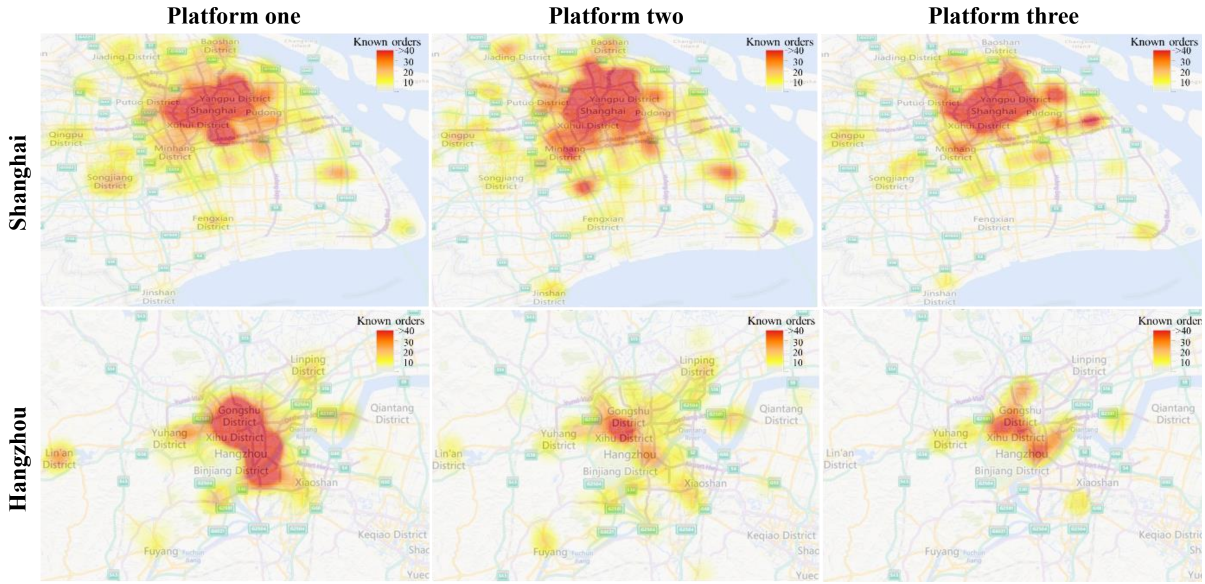


Figure 9: Heat maps of order delivery locations across three platforms in Shanghai and Hangzhou

Using our proposed model and algorithm, we conduct experiments comparing the total profits of the three platforms using collaborative and noncollaborative modes. Figure 10(a) illustrates the profits to each of the three platforms fulfilling its batch of 10 orders without collaboration; the sum of these individual profits is illustrated by the light-colored bars in Figure 10(b). The total profit for fulfilling the 30 orders when the firms collaborate is represented by the dark-colored bars in Figure 10(b). The gap between these two bars reflects the benefit of platform collaboration. The results in Figure 11 are the average values of our experiments on 20 batches of orders. According to Figure 10(b), the benefits of platform collaboration are approximately 7.2% for Shanghai and 3.6% for Hangzhou. The greater (percentage) benefits for Shanghai than for Hangzhou can be explained by the higher degree of customer overlap between platforms in Shanghai than in Hangzhou, as shown in Figure 9.

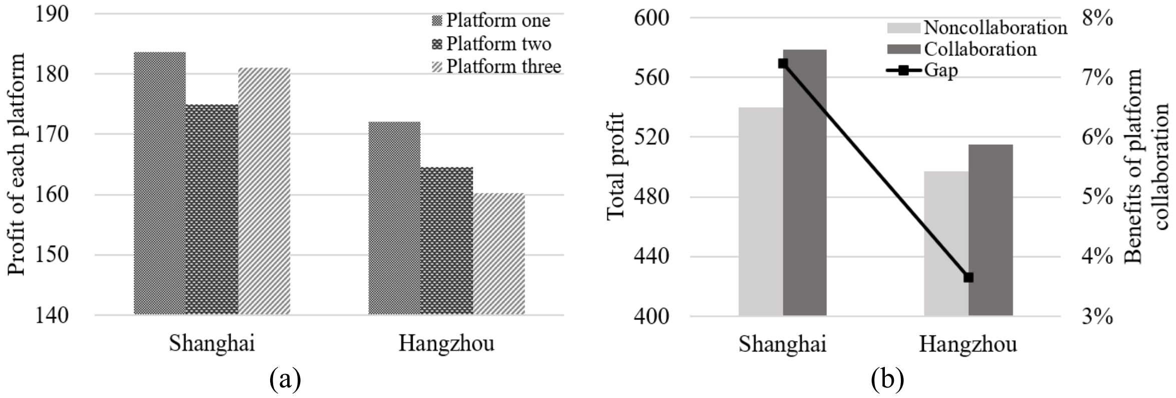


Figure 10: Comparative results for the benefits of platform collaboration

The aforementioned experiments are based on collaboration among the three platforms to fulfill 30 orders. Additional experiments are conducted to investigate whether the number of orders affects the benefits of platform collaboration. The results in Figure 11 confirm the effect. However, the effect of collaboration becomes weaker as the number of orders increases.

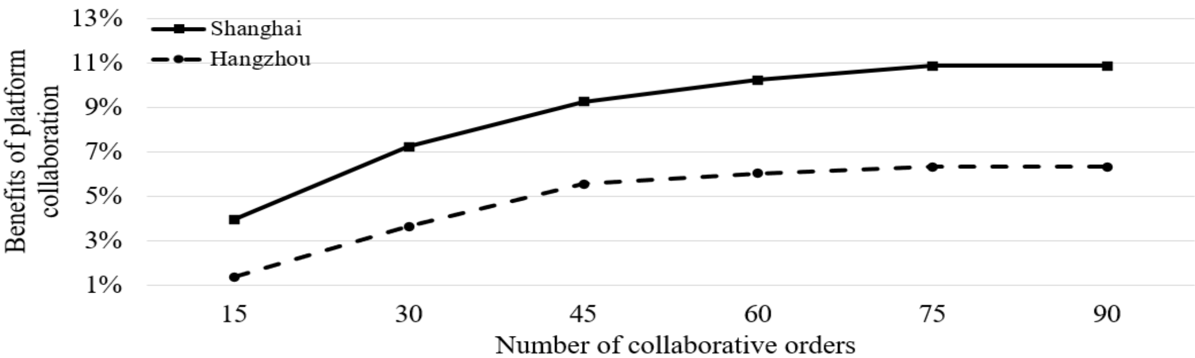


Figure 11: Effect of the number of collaborative orders on the benefits of platform collaboration

(7) Effect of decision frequency

In our previous experiments, the instant delivery platform adopts a decision interval of 12 minutes. It is of great interest to the manager to determine whether a higher decision frequency would increase profits. Although increasing the decision frequency reduces the potential idle time of couriers and results in greater utilization of delivery capacity, making more frequent decisions limits the possibility of bundling orders (i.e., bundling a known order with a future order from a nearby location) and reduces the efficiency of delivery routes. Thus, the optimal decision frequency should appropriately balance these aspects.

We conduct numerical experiments to evaluate performance with different decision frequencies. The planning horizon is set to 1 hour. The decision frequency varies from every 12 minutes to every 10, 6, 4, and 2 minutes, which result in 5, 6, 10, 15, and 30 decision waves, respectively. For fair comparison, the sizes of the sets of known orders in the planning horizon are the same (150 orders), and the setting of all couriers' initial locations is identical. The results in Figure 12 show that a decision frequency of every 10 minutes yields the best performance. Selecting a higher decision frequency does not necessarily increase platform profits, as is observed for decision frequencies of every 6, 4 and 2 minutes. The reason for this finding is that although additional supply (available couriers) may be more effectively utilized when the decision waves are more frequent, a high decision frequency may limit the opportunities to bundle orders for cost savings.

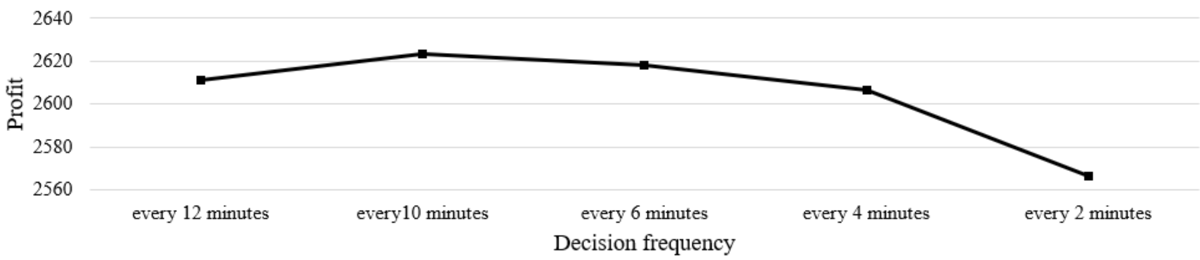


Figure 12: Effect of decision frequency

(8) Robustness test for percentage of the revenue

The sales amount percentage as revenue to the platform, i.e. parameter n, is an important feature of help me buy services. Thus the robustness tests are conducted to investigate whether the performance is robust to the information on the parameter n. The results shown in Figure 13 validate the model's robustness with respect to the sales amount percentage as revenue to the platform. When the sales amount percentage is higher than their estimated values by 20%, 40%, …, 120%, the profit deviation (percentage gap) is less than 2%.

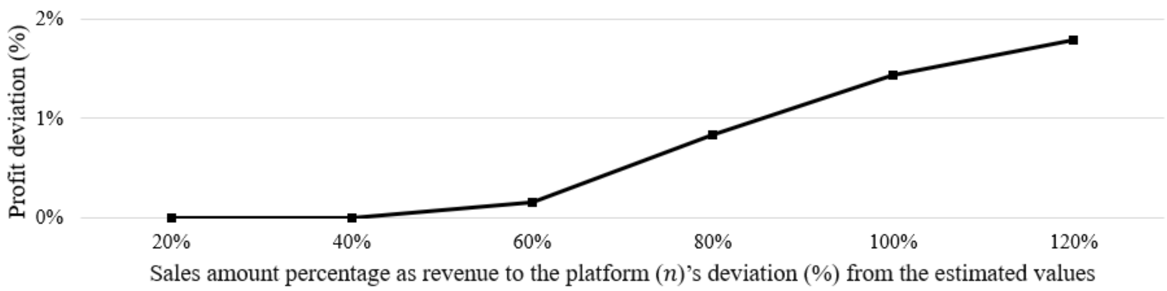


Figure 13: Robustness test on the sales amount percentage as revenue to the platform

7. Extensions for more generic contexts

As mentioned at the end of Section 3, we base our study on three assumptions to easily explain our proposed methodology. In this section, we extend the proposed model to more realistic environments by relaxing these three assumptions. In the following three subsections, we consider contexts in which (1) each courier can be assigned to more than one known order, (2) couriers with ongoing orders can be assigned to new orders, and (3) each order may contain several commodities that can be purchased from different stores. These extensions render our proposed methodology applicable to more general contexts.

7.1 Assignment with multiple known orders per batch

Our main study assumes that each courier is assigned to at most one known order at each epoch. This subsection relaxes this assumption and allows multiple known orders to be assigned to a courier at each epoch. Different stores and different destinations for multiple orders form many possible sequences of locations, which makes the routing decision for each courier extremely complex, especially when the number of orders is large. Therefore, this extension assumes that at most two orders can be assigned to one courier at each epoch. This assumption is reasonable because the most important feature for this type of delivery service is timeliness; if a platform allows each courier to take too many orders in one batch, the delivery time for some customers will be too long, which is not in line with the aim of this type of service. Before formulating this extended model, the new parameters and variables are listed below.

Newly defined indices and decision variables:

$e'(k)$ index of the end location for courier k, i.e., the end of the courier's route;

ξ_{sk} equals one if courier k visits store s, otherwise zero;

ϵ_{rk} equals one if known order r is the last order assigned to courier k, otherwise zero;

τ_{ijk} equals one if courier k visits node j immediately after node i, otherwise zero;

η_{ik} time when the courier k arrives at node i.

[M2] Max $\sum_{r\in R}\sum_{s\in S}\alpha_{rs}p_{rs} - \sum_{k\in K}\sum_{i\in e(k)\cup R\cup S}\sum_{j\in R\cup S\cup e'(k)} d_{ijk}\tau_{ijk} +$
$\sum_{\omega\in\Omega}\pi^{\omega}[\sum_{o\in O}\sum_{k\in K}(\gamma_{ok}^{\omega}f - \sum_{r\in R}\varepsilon_{rok}^{\omega}d_{rok})]$ (43)

subject to Constraints (2)–(3), (5), (7)–(9), (11), (14)–(15)

$$\sum_{r\in R}\beta_{rk} \le 2 \qquad \forall k\in K \tag{44}$$

$$\theta_{rsk} \le \xi_{sk} \qquad \forall r\in R, \forall s\in S, \forall k\in K \tag{45}$$

$$\xi_{sk} \le M\sum_{r\in R}\theta_{rsk} \qquad \forall s\in S, \forall k\in K \tag{46}$$

$$\sum_{i\in R\cup S}\tau_{irk} = \sum_{i\in R\cup S\cup e'(k)}\tau_{rik} = \beta_{rk} \qquad \forall r\in R, \forall k\in K \tag{47}$$

$$\sum_{i\in e(k)\cup R\cup S}\tau_{isk} = \sum_{i\in R\cup S}\tau_{sik} = \xi_{sk} \qquad \forall s\in S, \forall k\in K \tag{48}$$

$$\sum_{i\in S\cup e'(k)}\tau_{e(k)ik} = \sum_{i\in e(k)\cup R}\tau_{ie'(k)k} = 1 \qquad \forall k\in K \tag{49}$$

$$\tau_{re'(k)k} = \epsilon_{rk} \qquad \forall r\in R, \forall k\in K \tag{50}$$

$$\eta_{jk} \ge \eta_{ik} + t_{ijk} - M(1-\tau_{ijk}) \qquad \forall i\in e(k)\cup R\cup S, \forall j\in R\cup S\cup e'(k), \forall k\in K \tag{51}$$

$$\eta_{jk} \ge \eta_{sk} + \sum_{r\in R}z_{rs}\theta_{rsk} + t_{sjk} - M(2-\tau_{sjk}-\theta_{r'sk}) \qquad r'\in R, \forall s\in S, \forall j\in R\cup S, \forall k\in K \tag{52}$$

$$\eta_{rk} \leq u_r \qquad \forall r \in R, \forall k \in K \qquad (53)$$

$$\eta_{rk} \geq \eta_{sk} - M(1 - \theta_{rsk}) \qquad \forall r \in R, \forall s \in S, \forall k \in K \qquad (54)$$

$$\sum_{o \in O} \gamma_{ok}^{\omega} \leq \sum_{r \in R} \epsilon_{rk} \qquad \forall k \in K, \omega \in \Omega \qquad (55)$$

$$\varepsilon_{rok}^{\omega} = (\epsilon_{rk} + \gamma_{ok}^{\omega} - 1)^{+} \qquad \forall r \in R, o \in O, k \in K, \omega \in \Omega \qquad (56)$$

$$\xi_{sk} \in \{0,1\} \qquad \forall s \in S, \forall k \in K \qquad (57)$$

$$\epsilon_{rk} \in \{0,1\} \qquad \forall r \in R, \forall k \in K \qquad (58)$$

$$\tau_{ijk} \in \{0,1\} \qquad \forall i \in e(k) \cup R \cup S, \forall j \in R \cup S \cup e'(k), \forall k \in K \qquad (59)$$

$$\eta_{ik} \geq 0 \qquad \forall i \in e(k) \cup R \cup S \cup e'(k), \forall k \in K. \qquad (60)$$

Objective (43) maximizes the total profit. Constraints (44) guarantee that each courier is assigned with at most two known orders. Constraints (45)–(46) link the decision variables θ_{rsk} and ξ_{sk}. Constraints (47)–(49) ensure that the route of each courier is continuous. Constraints (50) link the decision variables τ_{ijk} and ϵ_{rk}, and guarantee that the last order assigned to each courier is the penultimate node of the courier's route. Constraints (51)–(52) connect the arrival times of two consecutive nodes. Constraints (53) guarantee that the response time for a known order should not exceed the limit. Constraints (54) ensure that each courier visits the store with the commodity required by a known order before visiting the location of the order's customer. Constraints (55) guarantee that each courier is assigned with at most one future order; if a courier has not been assigned with any known order, none future order will be assigned to the courier. Constraints (56) link the decision variables $\varepsilon_{rok}^{\omega}$, ϵ_{rk}, γ_{ok}^{ω}, and ensure that $\varepsilon_{rok}^{\omega}$ equals one if ϵ_{rk} and γ_{ok}^{ω} equal one simultaneously, otherwise $\varepsilon_{rok}^{\omega}$ equals zero. Constraints (57)–(60) define the variables.

Experiments are conducted to investigate the benefit of this extension that allows assigning multiple known orders to a courier. The results are shown in Table D-1, Appendix D. The results demonstrate that the profit could be increased by 2.81% on average; however, the computation time of the extended model M2 is longer than (about 1.75 times of) the time of the original model M1. If the decision time per batch is acceptable in reality, it is worth considering this extension in order assignment.

7.2 Involvement of couriers with ongoing orders

Our main study assumes that candidate couriers do not fulfill other orders at a decision epoch. This subsection relaxes this assumption so that an order can be assigned to a courier who is currently fulfilling another known order at the decision epoch. If it is convenient for a courier to fulfill a new known order, that courier should be considered even if they have an ongoing order, which may be completed in a short time and whose destination is close to the store containing the commodity required by the new known order. Due to space limitations, the model formulation and the new parameters and variables are elaborated in Appendix E.

We conduct experiments to investigate the benefits of this extension. The results are shown in Table D-2, Appendix D. These results demonstrate that the profits of the platform can increase by 1.66% on average; however, the computation time of the extended model M3 is about 1.58 times the computation time of the original model M1. If the decision time per batch is acceptable in reality, this extension should be considered for order assignment.

7.3 Multiple commodities in one order

Our main study assumes that each order contains only one commodity. This subsection relaxes this assumption and allows an order to contain multiple commodities. If the commodities required by one order can be purchased from a single store, the combination of these commodities can be regarded as one "commodity"; therefore, our proposed methodology still works. Here, we consider that the multiple commodities required by one order can be purchased from more than one store, which may increase the platform's profits but complicate the decision model. Due to space limitations, the model formulation and the new parameters and variables are elaborated in Appendix F.

We conduct experiments to investigate the benefits of this extension. The results are shown in Table D-3, Appendix D. These results demonstrate that the increase in profits is not as significant as that of the two previous extensions. In addition, the computation time of this extended model is much longer (about 6.84 times) than the computation time of the original model. These results imply that it may not be necessary for couriers to purchase commodities for an order from different stores, even if different stores offer different prices for these commodities.

8. Conclusions

As an on-demand e-commerce mode and alternative to traditional last-mile delivery services, help me buy services have become increasingly popular. To determine how to efficiently operate this new mode, this study proposes a comprehensive methodology for accepting orders, assigning orders to couriers, and selecting stores to supply the required commodities. This study contributes to the literature from the following three perspectives.

From the perspective of problem modeling, this study considers realistic factors and comprehensive decisions by formulating a two-stage stochastic programming model to maximize platform profits in the context of the uncertain arrival of future orders by deciding which orders to accept, which courier to assign to an accepted order, and from which store to visit to purchase the commodity required for an order. This study also extends the model in three aspects to consider more general contexts, specifically those in which multiple orders can be assigned to one courier, couriers with ongoing orders can be assigned to new orders, and multiple commodities in one order can be bought from different stores. The benefits of these three extensions are estimated quantitatively, showing that they increase the profits of the platform by 2.81%, 1.66%, and 0.87%, respectively. By considering a variety of realistic factors, this study offers one of the most comprehensive MIP models for research on help me buy services.

Contributing to algorithm design, we propose a new decomposition algorithm to solve our proposed two-stage stochastic programming model, which features an integer recourse function in the second stage. Different from Benders decomposition, which passes rows from the second stage to the first stage, our proposed decomposition algorithm passes primal columns from the second stage to the first stage. This enables the solution of the integer programming submodel contained in the second stage; in contrast, the Benders decomposition approach can mainly be applied to two-stage stochastic programming models with a linear recourse function in the second stage. Our proposed algorithm can also be used to solve other two-stage stochastic integer programming models. Our algorithm's efficiency is validated experimentally. Our proposed decomposition algorithm requires approximately 2 seconds to solve large-scale instances that can be solved by CPLEX, whereas CPLEX needs approximately 200 seconds; in addition, the optimality gap between the CPLEX solution and the solution obtained by our algorithm is about 0.35%.

We conduct numerical experiments based on real data from an instant delivery platform company in Shanghai to derive managerial insights for practitioners. The results indicate that our proposed methodology can be applied to make decisions for the assignment of new orders in batches. According to real data, when 240 orders arrive to the platform per hour, the batch period can be set to 12 minutes, but our proposed methodology can make decisions in 4 minutes. In addition, the value of the stochastic solution is estimated at about 1.70% of the platform's profits, and the value of perfect information is estimated to be 1.38% of the profits. Sensitivity analyses are also performed to derive insights. For example, setting the appropriate number of couriers and appropriate response time is important as more couriers or a longer response time does not always yield greater profits for the platform. Moreover, the geographical distributions of couriers, stores, and orders (customers) affect the platform's profits. Finally, the benefits of platform collaboration are also investigated in this study.

This study has some limitations. For example, although the proposed methodology is validated using real data from approximately 2,000 orders per day on the platform, the algorithm could be further improved to solve larger-scale instances and be applied to larger platforms. Machine learning techniques could be combined with the proposed MIP models to derive decision rules, which are more suitable for platform operators than the purely MIP-based decisions used in very-large-scale operational environments. Finally, the proposed model could be extended to support strategic decisions such as whether to adopt help me buy services, how to determine reference prices (pricing problems) considering customer preferences, and whether to consider the relationship product value and delivery fee, which may be of great concern to executives of e-commerce or instant delivery companies. The above extensions offer future directions for research in this field.

References

Arslan, A.M., Agatz, N., Kroon, L., Zuidwijk, R. (2019) Crowdsourced delivery: A dynamic pickup and delivery problem with ad hoc drivers. *Transportation Science* 53(1): 222–235.

Arslan, A.M., Agatz, N., Klapp, M.A. (2021) Operational strategies for on-demand personal shopper services. *Transportation Research Part C* 130: 103320.

Bahrami, S., Nourinejad, M., Yin, Y., Wang, H. (2021) The three-sided market of on-demand delivery. *Transportation Research Part E* 179: 103313.

Bai, J., So, K.C., Tang, C.S., Chen, X., Wang, H. (2019) Coordinating supply and demand on an on-demand service platform with impatient customers. *Manufacturing & Service Operations Management* 21(3): 556–570.

Benjaafar, S., Ding, J.-Y., Kong, G., Taylor, T. (2022) Labor welfare in on-demand service platforms. *Manufacturing & Service Operations Management* 24(1): 110–124.

Boysen, N., Briskorn, D., Schwerdfeger, S. (2019) Matching supply and demand in a sharing economy: Classification, computational complexity, and application. *European Journal of Operational Research* 278(2): 578–595.

Dayarian, I., Savelsbergh, M.W.P. (2020) Crowdshipping and same-day delivery: Employing in-store customers to deliver online orders. *Production and Operations Management* 29(9): 2153–2174.

Deng, Q., Fang, X., Lim, Y.F. (2021) Urban consolidation center or peer-to-peer platform? The solution to urban last-mile delivery. *Production and Operations Management* 30(4): 997–1013.

Fatehi, S., Wagner, M.R. (2022) Crowdsourcing last-mile deliveries. *Manufacturing & Service Operations Management* 24(2): 791–809.

He, B., Mirchandani, P., Wang, Y. (2020) Removing barriers for grocery stores: O2O platform and self-scheduling delivery capacity. *Transportation Research Part E* 141: 102036.

Klapp, M., Erera, A., Toriello, A. (2020) Request acceptance in same-day delivery. *Transportation Research Part E* 143: 102083.

Khir, R., Erera, A., Toriello, A. (2021) Two-stage sort planning for express parcel delivery. *IISE Transactions* 53(12): 1353–1368.

Kung, L.C., Zhong, G.Y. (2017) The optimal pricing strategy for two-sided platform delivery in the sharing economy. *Transportation Research Part E* 101: 1–12.

Li, X., Li, J., Aneja, Y.P., Guo Z., Tian, P. (2019) Integrated order allocation and order routing problem for e-order fulfillment. *IISE Transactions* 51(10): 1128–1150.

Liu, S., He, L., Shen, Z.-J. M. (2021) On-time last-mile delivery: Order assignment with travel-time predictors. *Management Science* 67(7): 4095–4119.

Özkan, E. (2020) Joint pricing and matching in ride-sharing systems. *European Journal of Operational Research* 287(3): 1149–1160.

Qi, W., Li, L., Liu, S., Shen, Z.-J. M. (2018) Shared mobility for last-mile delivery: Design, operational prescriptions, and environmental impact. *Manufacturing & Service Operations Management* 20(4): 737–751.

Spielman D.A., Teng S.-H. (2004) Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. *Journal of the ACM* 51(3): 385–463.

Sun, P., Veelenturf, L.P., Hewitt, M., Van Woensel, T. (2018) The time-dependent pickup and delivery problem with time windows. *Transportation Research Part B* 116: 1–24.

Sun, P., Veelenturf, L.P., Hewitt, M., Van Woensel, T. (2020) Adaptive large neighborhood search for the time-dependent profitable pickup and delivery problem with time windows. *Transportation Research Part E* 138: 101942.

Taylor, T. (2018) On-demand service platforms. *Manufacturing & Service Operations Management* 20(4): 704–720.

Ulmer, M.W., Savelsbergh, M.W.P. (2020) Workforce scheduling in the era of crowdsourced delivery. *Transportation Science* 54(4): 1113–1133.

Ulmer, M.W., Thomas, B.W., Campbell, A.M., Woyak, N. (2021) The restaurant meal delivery problem: dynamic pickup and delivery with deadlines and random ready times. *Transportation Science* 55(1): 75–100.

Voccia, S.A., Campbell, A.M., Thomas, B.W. (2019) The same-day delivery problem for online purchases. *Transportation Science* 53(1): 167–184.

Wang, H. (2019) Routing and scheduling for a last-mile transportation system. *Transportation Science* 53(1):131–147.

Wang, Y., Lei, L., Zhang, D., Lee, L.H. (2020) Towards delivery-as-a-service: Effective neighborhood search strategies for integrated delivery optimization of E-commerce and static O2O parcels. *Transportation Research Part B* 139: 38–63.

Yildiz, B., Savelsbergh, M.W.P. (2019) Provably high-quality solutions for the meal delivery routing problem. *Transportation Science* 53(5): 1372–1388.

Yildiz, B., Savelsbergh, M.W.P. (2020) Pricing for delivery time flexibility. *Transportation Research Part B* 133: 230–256.

Zhen, L., He, X., Wang, H., Laporte, G., Tan, Z. (2022) Decision models on personal shopper platform operations optimization. *Transportation Research Part C* 142: 103782.