Distributed Order Assignment and Scheduling Problem under Processing and Delivery Time Uncertainty

Yantong Li
School of Maritime Economics and Management, Dalian Maritime University, 116026 Dalian, China, yantongli@163.com

Jean-François Côté
CIRRELT, Université Laval, Québec, Canada, jean-francois.cote@fsa.ulaval.ca

Leandro C. Coelho
CIRRELT, Université Laval, Québec, Canada research chair in integrated logistics, leandro.coelho@fsa.ulaval.ca

Chuang Zhang
Department of equipment support and remanufacturing, Army Academy of Armored Forces, 100072 Beijing, China, chuangzhang1225@163.com

Shuai Zhang
Business School, University of Greenwich, London SE10 9LS UK. shuai.zhang@greenwich.ac.uk

In response to increasingly fierce competition and highly customized demands, many companies adopt a distributed production model but manage their orders in a centralized manner. Coordination between multiple factories requires unified information and resources to provide a close match between supply and demand. One of the crucial tasks is to solve the order assignment and scheduling (OAS) problem with uncertainties introduced by unexpected changes in upstream supply, labor supply, and transportation capacity. Managing uncertainties in production and distribution is important, as they can significantly interrupt and delay the timely and constant supply of orders if not appropriately managed properly. We address an order assignment and scheduling problem with direct distribution under uncertainties in processing and delivery time. The aim is to achieve a minimum of weighted sum cost and timeliness, which involves the optimization of the order assignments to multi-factory and production scheduling for orders at each site. We first formulate the problem as a two-stage stochastic programming model. To manage a large scale of possible scenarios, we apply a sample average approximation (SAA) method to approximate the model. We propose a novel model with fewer binary variables and big-M constraints. An exact logic-based Benders decomposition (LBBD) method is developed to deal with practical-sized instances. Numerical results indicate the superiority of our new model and the LBBD method. Our solution provides lower cost and higher timeliness than a few sequential models. Managerial implications are discussed to demonstrate its advantages and potential applicability in practice.

Key words: Order assignment and scheduling; Stochastic optimization; makespan; tardiness; Logic-based Benders decomposition.

1. Introduction

The world has witnessed several severe emergency accidents in the past two decades, e.g. the 2004 Indian Ocean tsunami, the 2011 East Japan earthquake, and the 2019 COVID-19 pandemic. These events have significantly impacted supply chains (SC), leading to SC uncertainties. In response to such uncertainties, many companies split their factories around the world to make distributed production, such that their SCs are more resilient to SC disruptions. In this case, a company has to deal with orders from all over the world and make production and delivery in a distributed manner, leading to a distributed order assignment and scheduling (OAS) problem. An OAS problem determines the order assignment to distributed manufacturers and the production scheduling at each manufacturer to optimize a given performance measure. An optimal solution to the OAS problem can significantly improve the timeliness of order delivery, reduce operational costs, and balance workloads. Ultimately, optimized solutions for this problem have the potential to mitigate the vulnerability of the SC under uncertainty.

Given its significance with a wide range of applications, the OAS problem has been widely studied. However, most studies on the OAS problem assume that the production and distribution times of orders are deterministic (Chen and Pundoor 2006, Sun et al. 2015). When confronting a severe emergency event, a global SC might be disrupted for various reasons, such as limited delivery capacity due to road damage or traffic restrictions, or a labor shortage. Taking the COVID-19 outbreak as an example, the rapid and widespread of the COVID-19 has created severe uncertainties in global SC: first, the epidemic control efforts have interrupted flows of raw materials and finished goods between different countries and regions; second, the blockage of people and materials has impacted the production of manufacturers; third, the demand surges drastically, especially for medical materials or personal protective equipment. These uncertainties cause high operational costs and frequent delivery delays. To address these challenges, it is essential to integrate distributed manufacturers and make centralized production and distribution plans to increase supplies and mitigate the severity of supply shortages. In particular, the uncertainty in production could result from delays in upstream suppliers or uncertainties in humanitarian operations, affecting timely production (Kovacs and Moshtari 2019, Fattahi and Govindan 2020). Considering the vulnerability of the transportation, distributors also struggle with order delivery, which may lead to uncertainties in distribution timeliness. It is significant to investigate the stochastic OAS problem in real-life settings, which helps prompt response to unexpected changes and random order releases.

This paper investigates the stochastic order assignment and scheduling problem with direct distribution (S-OASD). We aim to develop a joint order production and distribution plan such that order deliveries can be provided timely and economically. The S-OASD problem differs from the traditional production and distribution problem as it integrates order scheduling at each manufacturer. It is also distinct from the parallel machine scheduling problem since manufacturers (machines) are decentralized and dispersed at different locations. Moreover, the order production and distribution times from various manufacturers for the same order are different and uncertain. We study the S-OASD problem that minimizes the weighted sum of total cost and the expected makespan. We formulate it as a two-stage stochastic programming model. We then make the model tractable by using the sample average approximation (SAA) method. Specifically, we propose a new model based on a predetermined processing sequence at each manufacturer. The new model has significantly fewer binary variables and big-M constraints and is thus more suitable to be solved using off-the-shelf solvers. We develop a logic-based Benders decomposition (LBBD) method for the problem. The LBBD decomposes the S-OASD problem into a master problem (MP) determining order assignments and a subproblem taking charge of order scheduling at each manufacturer. Benders cuts are iteratively generated and added to the MP. We first introduce a basic optimality cut and then strengthen it by finding the minimal infeasible subset and deriving analytical cuts. The performance of the proposed models and algorithms is demonstrated by numerical experiments on random instances. We then extend the S-OASD problem to consider due dates of order delivery, denoted as S-OASD-T. The objective is to minimize the weighted sum of total cost and total tardiness. For the problem, we first present a two-stage stochastic programming formulation. We then develop an LBBD method to solve it. We perform numerical experiments to show the performance of the proposed method.

The remainder of the paper is structured as follows. In Section 2, we briefly review related literature. Section 3 formally describes the studied S-OASD problem and presents their mathematical formulations. Section 4 presents the developed LBBD methods. In Section 5, we conduct numerical experiments to demonstrate the performance of the proposed models and algorithms, introduce the extended S-OASD-T problem, and discuss some practical implications based on the obtained results. Finally, we conclude the paper and indicate some future research directions in Section 6.

2. Literature review

This section reviews the related literature to the studied S-OASD problem. We first summarize papers on the OAS problem and multi-factory scheduling problem which are closely relevant to the studied problem. Since our main contribution comes from the design of the LBBD method for solving the studied problem, we then review papers concerning the LBBD as well as its applications.

2.1. Order assignment and scheduling in multi-factory environment

OAS problems have attracted increasing attention since the seminal work of Chen and Pundoor (2006). Existing studies can be categorized by the factory settings, e.g., single machine, multi-machine, and the performance measures to be optimized. In particular, as the factory locations vary, it is imperative to integrate some realistic elements such as order production (processing) cost/time, and the delivery cost/time within the scheduling planning. This is also the main focus of this research. Thus, we systematically review the integrated OAS problem considering multiple factories and delivery times.

Behnamian and Ghomi (2013) address a multi-factory production scheduling problem by parallel machines in each factory to minimize the makespan. They present a MILP model, a modified largest processing time heuristic, and a genetic algorithm for the problem. Behnamian (2015) then impels the study by considering transshipments among factories to minimize the total processing cost of all orders. In particular, each order must be completed before a given due date. The author develops a MILP model and Benders decomposition method for the problem. Behnamian (2017) studies a similar problem with makespan minimization and develops a particle swarm optimization algorithm for it. Yazdani et al. (2015) refine the model of Behnamian and Ghomi (2013) and propose an artificial bee colony metaheuristic for the multi-factory parallel scheduling problem. Chung et al. (2012) study the decentralized distributed planning problem where new orders must be properly assigned to multiple collaborating factories to minimize the total tardiness. We refer interested readers to the review papers of Behnamian and Ghomi (2016) and Lohmer and Lasch (2020) summarizing the literature on multi-factory scheduling. However, we observe from their reviews that the delivery time to final customers has rarely been integrated despite its importance in decentralized production practice where multiple factories are located at different sites. This is particularly important in some OAS applications for time-sensitive products.

Chen and Pundoor (2006) consider a supply chain in which products are produced in multiple overseas factories with a single machine and distributed to a domestic distribution center. Four variants with different performance measures are investigated separately, each representing a decision maker's preference for total lead time and total cost. They analyze the complexities of the four variants and provide fast heuristics for the intractable ones and exact algorithms for the other ones that are polynomially solvable. Li and Ou (2007) investigate the production scheduling and distribution concerning two machines at different locations to minimize delivery and customer waiting costs. They prove that the problem is strongly NP-hard and develop heuristics for solving it. Kerkhove and Vanhoucke (2014) extend the work of Behnamian and Ghomi (2013) to solve a scheduling problem faced by a knitted fabrics producer. Their objective is to minimize the weighted sum of the order lateness and tardiness. They consider different due dates for different factories due

to the difference in production locations. A hybrid metaheuristic combining simulated annealing and genetic algorithms is developed to solve a real case. Guo et al. (2013) address an OAS problem considering transportation time to minimize three time-related objectives, i.e., the total tardiness, total throughput time, and total idle times. They formulate the problem into a multi-objective MILP and design a non-dominated sorting genetic algorithm II (NSGA-II) method to solve it. Sun et al. (2015) propose a genetic algorithm for solving a multi-factory scheduling problem with maritime transport consideration to minimize total production, transportation, and penalty costs. Karimi and Davoudpour (2017) investigate a multi-factory scheduling problem considering batch deliveries and transshipments among factories to minimize the sum of total tardiness and transportation costs. They propose a MILP and a knowledge-based heuristic for the problem. These studies extend the multi-factory scheduling problem by integrating distribution activities in which all finished orders have to be transported to a central warehouse.

Direct distribution to customers has rarely been studied in multi-factory OAS planning. However, a few researchers have recently realized the importance of such direct distribution of orders to multiple final customers. Li et al. (2017) consider a multi-factory and multi-customer production scheduling and delivery problem, in which orders have to be assigned to one factory that operates a single machine and then delivered to the final customer upon completion of its processing. They optimize the makespan and the total completion time subject to a given threshold separately by several heuristics. He et al. (2019) address the multi-factory (each with a single machine) OAS problem to minimize the total costs of labor, inventory holding, transportation, and penalty costs. A novel feature of the problem is to determine the transportation modes to deliver final products to customers. A memetic heuristic is designed for the problem. Marandi and Fatemi Ghomi (2019) investigate the multi-factory scheduling problem in which vehicle routing decisions are integrated. They develop several metaheuristics to tackle the proposed problem.

In summary, the multi-factory OAS problem has been extensively studied lately. However, there are still some gaps to be filled. First, most existing works streamline the distribution part by assuming that orders are delivered to a distribution center instead of to the customers. Second, many effective heuristic algorithms are developed for the OAS problem, while exact approaches are rarely proposed. Third, most of the studies are under deterministic environments. However, in many situations, particularly in emergencies, the processing time of orders and the delivery time is hard to be precisely estimated, indicating the necessity to consider S-OASD problems.

2.2. Logic-based Benders decomposition

The Benders Decomposition (BD) method is widely applied to solve large-scale MILPs that can be decomposed into easier problems through the BD procedure. It first decomposes the original problem into a master problem (MP) and a subproblem (SP) that are solved iteratively by dynamically

adding Benders cuts to the MP. In a classical BD scheme, Benders cuts are generated according to the dual solution of the linear program SP. Recently, Hooker (2000) and Hooker and Ottosson (2003) introduce LBBD to extend the classic BD to enable the SP to take any form (e.g., MILP or nonlinear).

The LBBD framework has been proved effective in solving a wide range of combinatorial optimization problems. We briefly review some papers on distributed operating room scheduling (DORS). In this problem, decision-makers need to assign patients in a specific area to one of the available hospitals and determine when and which operating room (OR) to assign to each patient in a hospital. Guo et al. (2021) investigated the stochastic DORS problem to determine assignments of surgeries to ORs in collaborative hospitals. Considering the stochastic surgery duration, the authors formulated the problem as a two-stage stochastic integer program and then solved it via SAA. An LBBD algorithm with various Benders cuts was proposed to solve the problem. Several enhancement techniques were also developed to find an initial solution, tighten the MP, and quickly eliminate suboptimal master solutions. Roshanaei et al. (2020) studied the balanced DORS problem to determine the macro balancing decision on the number of ORs among hospitals and micro-decisions on patients-OR assignments in each hospital. A mixed-integer nonlinear programming model was formulated to minimize the total costs, including the opening cost of surgical suites and ORs, the scheduling and rejection cost of patients, and the imbalance penalty costs at both macro and micro levels. The model was then reformulated into three variants with various reformulation-decomposition techniques. The authors proposed uni- and bi-level LBBD algorithms for the problem, which were verified effective through real-world datasets. Roshanaei et al. (2017) proposed three novel LBBD algorithms and a cut propagation mechanism for the DORS problem in which ORs and patients are assigned and scheduled collaboratively in a planning horizon. The objective is to minimize the total costs, including the opening cost of surgical suites and ORs and the waiting cost of patients. Note that the waiting cost has two meanings, i.e., the cost of patients scheduled later than the referral date and the costs of patients who remain unscheduled at the end of the planning horizon.

Other applications of LBBD algorithms can be found in parallel machine scheduling (Tran et al. 2016), two-dimensional bin-packing (Côté et al. 2021), planning and scheduling (Hooker 2007b), strip packing (Côté et al. 2014), capacity- and distance-constrained plant store (Fazel-Zarandi and Beck 2012), parallel machine scheduling problems with setups (Tran et al. 2016), scheduling and location (Li et al. 2021), home healthcare delivery (Heching et al. 2019), and parcel delivery with drones (Baloch and Gzara 2020).

The literature review allows us to better position our contributions as follows: 1) study a stochastic distributed order assignment and scheduling problem with direct distribution; 2) consider two

distinct objectives, i.e., the makespan and total weighted tardiness costs; 3) we consider a more practical situation where the processing time and distribution time of orders is uncertain, which is relevant and practical; 4) we develop new models and solution methods for the studied problem; 5) we propose cut strengthening techniques to accelerate the convergence of the developed LBBD method. Next, we describe the problem in detail and introduce the formulations.

3. Problem description and formulation

In this section, we elaborate on the studied S-OASD problem and present two-stage stochastic programming formulations.

3.1. Two-stage stochastic programming

We first present some preliminaries of two-stage stochastic programming based on those of Elci and Hooker (2020). We are interested in solving two-stage stochastic programs with the following form:

$$\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}) + \mathbb{E}_{\omega}[Q(\boldsymbol{x},\omega)], \tag{1}$$

where $\boldsymbol{x}$ is the vector of first-stage variables, $\mathcal{X}$ is the solution space of $\boldsymbol{x}$, $f(\boldsymbol{x})$ is the first-stage objective function, ω ranges over a finite set $\Omega = \{1, 2, \ldots, N\}$ of scenarios, and $\mathbb{E}_{\omega}[Q(\boldsymbol{x},\omega)]$ represents the expectation of the second-stage objective with $Q(\boldsymbol{x},\omega)$ being the optimal value of the second-stage problem under scenario ω. The value of the second-stage problem under scenario ω is formulated as follows:

$$Q(\boldsymbol{x},\omega) = \min_{\boldsymbol{y}\in Y(\omega)} \{g(\boldsymbol{y})\}, \tag{2}$$

where $\boldsymbol{y}$ and $Y(\omega)$ are the second-stage variable vector and its solution space, respectively, and $g(\boldsymbol{y})$ is the optimal objective value of the second-stage problem under scenario ω. If we know the probability p_{ω} of each scenario ω, the two-stage stochastic problem can be formulated as:

$$\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}) + \sum_{\omega\in\Omega} p_{\omega} Q(\boldsymbol{x},\omega). \tag{3}$$

3.2. Problem description and notations

In this section, we formally describe the studied problem, define related notations, and formulate the problem as a two-stage stochastic program. Decision-makers need to determine a production and distribution plan with multiple manufacturers, third-party logistic providers, and customers with orders. Both the manufacturers and customers are dispersed on a network at different locations. Customers place orders and expect to receive them as soon as possible. The requested orders are assigned to manufacturers for processing. The manufacturers have contracts with emergency

management authorities and agree to comply with the centralized production plan. Finished orders are then directly distributed to customers by third-party logistics providers.

Given a set of orders and manufacturers that are dispersed at different locations, decision-makers need to determine how to assign these orders to the manufacturers, when to process each order at each manufacturer, and when to deliver the finished orders to the corresponding customer location. The objective is to minimize the weighted sum of the total costs including order processing and delivery costs, and the timeliness related costs. In particular, we consider two distinct timeliness objectives: the makespan indicating when all orders are delivered to their corresponding customers, and the total tardiness indicating the total time delayed of all delivered orders.

Consider a set of $F = \{1, 2, \ldots, m\}$ of manufacturers whose production plants are located at different locations with coordinates (X_k, Y_k). Each manufacturer $k \in F$ has a specific release time r_k indicating the earliest time that manufacturer k can process the first order. A set $J = \{1, 2, \ldots, n\}$ contains the orders are placed by customers from different locations. Each order must be delivered to a specific location with coordinates (X_j, Y_j). The producing cost and delivery cost for order j are dependent on the assigned manufacturer k, denoted as g_{jk} and c_{jk}. Due to various factors, the processing time and delivery time for order j are uncertain. These uncertainties are represented by a finite set $\Omega = \{1, 2, \ldots, N\}$ of scenarios. The processing time p^{ω}_{jk} and delivery time t^{ω}_{jk} are realized under each scenario $\omega \in \Omega$. The decisions to be made are to assign orders to manufacturers and to arrange the production sequence of the assigned orders at each manufacturer.

The S-OASD aims to minimize the weighted sum of the total cost and makespan, which is a primary objective in some scheduling problems (Han et al. 2015). We define non-negative continuous variables C^{ω}_{j} and C^{ω}_{max} as the completion time of order j and the makespan under scenario ω, respectively. The completion time of a single order refers to the time it finishes processing in its assigned manufacturer, while the makespan denotes the time when all orders have been processed and delivered to their customer locations, representing the timeliness of order fulfillment.

The following assumptions are made: 1) each order must be processed by exactly one manufacturer; 2) each manufacturer can process at most one order at a time; 3) the distribution of each order starts immediately after the production is completed; 4) each order must be processed non-preemptively.

To formulate the problem, the following notation is defined:

Parameters:

p^{ω}_{jk} processing time of order j at manufacturer k under scenario ω;

g_{jk} processing cost of order j at manufacturer k;

t^{ω}_{jk} delivery time of order j from manufacturer k under scenario ω;

c_{jk} delivery cost of order j from manufacturer k to its distribution location;

r_k earliest available time for manufacturer k;

M a large number, which is set to $\max_{k\in F} r_k + \max_{j\in J,k\in F,\omega\in\Omega} t_{jk}^{\omega} + \sum_{j\in J}(\max_{k\in F} p_{jk})$.

Decision variables:

v_{jk} equal to 1 if order j is assigned to manufacturer k;

x_{ij}^{ω} equal to 1 if the production of order i (immediately or not) precedes order j at the same manufacturer under scenario ω;

C_j^{ω} completion time of order j under scenario ω;

C_{max}^{ω} makespan, i.e., the completion time of all orders under scenario ω;

T_j^{ω} tardiness of order j under scenario ω.

3.3. Formulations

We first present a two-stage stochastic programming model for the studied problem based on the linear ordering (LO) model.

3.3.1. Linear ordering formulation The objective function of the S-OASD is to minimize the weighted sum of the total cost and the expected makespan. To formulate the problem, we first adopt the LO model, which is widely used to model single and parallel machine scheduling problems (Naderi and Roshanaei 2020, Li et al. 2021), to formulate the studied S-OASD problem.

$$(P_1)\ \min\ w_1 \sum_{j\in J}\sum_{k\in F}(g_{jk}+c_{jk})v_{jk} + w_2\mathbb{E}_{\omega}[C_{max}^{\omega}] \tag{4}$$

s.t.

$$\sum_{k\in F} v_{jk} = 1 \qquad \forall j\in J \tag{5}$$

$$C_{max}^{\omega} \geq C_j^{\omega} + \sum_{k\in F} t_{jk}^{\omega} v_{jk} \qquad \forall j\in J,\omega\in\Omega \tag{6}$$

$$C_j^{\omega} \geq \sum_{k\in F}(p_{jk}^{\omega}+r_k)v_{jk} \qquad \forall j\in J,\omega\in\Omega \tag{7}$$

$$C_j^{\omega} \geq C_i^{\omega} + p_{jk}^{\omega} - M(3-x_{ij}^{\omega}-v_{jk}-v_{ik}) \qquad \forall i,j\in J, i<j, k\in F,\omega\in\Omega \tag{8}$$

$$C_i^{\omega} \geq C_j^{\omega} + p_{ik}^{\omega} - M(2+x_{ij}^{\omega}-v_{jk}-v_{ik}) \qquad \forall i,j\in J, i<j, k\in F,\omega\in\Omega \tag{9}$$

$$C_j^{\omega} \geq 0 \qquad \forall j\in J,\omega\in\Omega \tag{10}$$

$$v_{jk} \in \{0,1\} \qquad \forall j\in J,k\in F \tag{11}$$

$$x_{ij}^{\omega} \in \{0,1\} \qquad \forall i,j\in J, i<j,\omega\in\Omega. \tag{12}$$

The objective function (4) minimizes the total weighted cost and the expected makespan. The weights w_1 and w_2 represent the preference of decision-makers on the cost and makespan components. Constraints (5) ensure that each order is assigned to exactly one manufacturer. Constraints (6) compute the makespan under scenario ω. Constraints (7) indicate that the completion time

of order j under scenario ω must be greater than or equal to the total of its processing time and the manufacturer's release time. Constraints (8) and (9) represent the completion time sequence constraints. Constraints (10)–(12) bound the variables. The studied S-OASD problem is NP-hard since its deterministic counterpart can be reduced to a parallel machine scheduling problem to minimize the makespan, which is well known to be strongly NP-hard (Lenstra et al. 1977).

3.3.2. A new formulation with predetermined sequences We propose next a new model for the studied S-OASD problem. In the model, the order processing sequence is predetermined under each scenario. In particular, each manufacturer process the assigned orders according to the longest delivery time (LDT) rule, i.e., orders with longer delivery times are processed first. Because of the predetermined sequence, the new model does not require any binary sequence variables and has significantly fewer big-M constraints. Moreover, the new model is still valid for the S-OASD problem despite the fixed sequence. This is reflected by the following proposition.

Proposition 1. *For the S-OASD problem with total cost and expected makespan minimization, there exists an optimal schedule in which orders at each manufacturer are scheduled using the LDT rule under each scenario.*

Proof. In the objective of the S-OASD problem, the production and distribution costs are fixed when orders are assigned to manufacturers, which indicates that the cost is independent of the order processing sequence. Thus, the predetermined processing sequence may only affect the makespan for a given order assignment. Next, we prove that under each scenario, the expected makespan of S-OASD with predetermined order processing sequence is still optimal. We note that for a given set of orders assigned to a manufacturer under scenario ω, the resulting problem is a single machine scheduling problem with delivery time and makespan minimization $1|t^{\omega}_{jk}|C^{\omega}_{max}$, where t^{ω}_{jk} is the delivery time of order j from manufacturer k. This problem can be optimally solved in polynomial time using the LDT rule (Vakhania 2004). In an optimal solution of the S-OASD problem, let set R^{ω}_k represent the orders assigned to manufacturer k under scenario ω and $C^{\omega}_{max}(R^{\omega}_k)$ be the corresponding makespan of manufacturer k. In the solution, if the corresponding schedule to obtain $C^{\omega}_{max}(R^{\omega}_k)$ does not comply with the LDT rule, one can always sort the orders in the set R^{ω}_k following the LDT rule to obtain another optimal schedule with a makespan $\hat{C}^{\omega}_{max}(R^{\omega}_k)$, where $\hat{C}^{\omega}_{max}(R^{\omega}_k) \leq C^{\omega}_{max}(R^{\omega}_k)$. □

Given the validity of Proposition 1, we can devise a new model P_2 which does not rely on binary sequence variables x^{ω}_{ij} and has significantly fewer big-M constraints. We define a set J^{ω}_k of orders for each manufacturer $k \in F$ under scenario $\omega \in \Omega$, where J^{ω}_k contains all orders in set J sorted in non-decreasing order of delivery time from manufacturer k, i.e., $J^{\omega}_k = \{j^{\omega}_{1k}, j^{\omega}_{2k}, \ldots, j^{\omega}_{nk}\}$ and $t^{\omega}_{j_{1k}k} \geq t^{\omega}_{j_{2k}k}, \ldots, \geq t^{\omega}_{j_{2k}k}$. We then define a continuous variable C^{ω}_{jk} indicating the completion

time of order j at manufacturer k under scenario ω. The proposed new model P_2 can be formulated as follows:

$$(P_2) \quad \min\ w_1 \sum_{j\in J}\sum_{k\in F}(g_{jk}+c_{jk})v_{jk} + w_2\mathbb{E}_\omega[C^\omega_{max}] \tag{13}$$

s.t. (5)–(7), (10), (11), and to

$$C^\omega_{jk} \geq (r_k + p^\omega_{jk})v_{jk} \qquad \forall\, j\in J, k\in F, \omega\in\Omega \tag{14}$$

$$C^\omega_{jk} \geq C^\omega_{j-1,k} + p^\omega_{jk}v_{jk} \qquad \forall\, k\in F, j\in J_k, j\neq 1, \omega\in\Omega \tag{15}$$

$$C^\omega_{j} \geq C^\omega_{jk} - M(1-v_{jk}) \qquad \forall\, j\in J, k\in F, \omega\in\Omega \tag{16}$$

$$C^\omega_{jk} \geq 0 \qquad \forall\, j\in J, k\in F, \omega\in\Omega. \tag{17}$$

Constraints (14) indicate that if order j is processed at manufacturer k, its completion time at that manufacturer must be greater than or equal to its processing time plus the release time of the manufacturer. Note that at each manufacturer k, orders in the set J_k are ordered in non-increasing order of their delivery time. Constraints (15) give the completion time consistency for two consecutive orders at the same manufacturer, indicating that the completion time of order j at manufacturer k cannot be earlier than the total of its processing time and the completion time of order $j-1$. If order j is not processed at manufacturer k, its completion time on the manufacturer can be at least the completion time of order $j-1$. Constraints (16) identify the completion time of order j, indicating that if it is assigned to manufacturer k, then its completion time should be greater than or equal to C^ω_{jk}. Constraints (17) bound the decision variables.

The proposed new model P_2 is superior to the adapted model P_1 mainly in two aspects. First, P_2 has significantly fewer binary variables than P_1. Second, P_2 requires significantly fewer big-M constraints, regarded as difficult constraints in a MILP model. The numbers of binary variables, non-binary variables, and big-M constraints of the two models are listed in Table 1, in which the first three columns report the numbers of binary variables (#BV), non-binary variables (#NBV), and Big-M constraints (#BMC) in the two models, while the last three columns correspond to those numbers of an instance with 100 scenarios, 100 orders, and ten manufacturers.

Table 1 **Comparison between the proposed models for $\omega=100$, $n=100$ and $m=10$**

Model	#BV	#NBV	#BMC	#BV	#NBV	#BMC
P_1	$\frac{n(n-1)\omega}{2}+mn$	$(n+1)\omega$	$n(n-1)m\omega$	496,000	10,100	99,000,000
P_2 (new)	mn	$(mn+n+1)\omega$	$mn\omega$	1,000	110,100	100,000

4. Solution methods

Given the continuous nature of the processing time and distribution time of orders, there may be a large number of scenarios to handle in practice. We see that the above proposed models are dependent on the number of scenarios. Thus, it becomes intractable when the number of scenarios gets very large. In this section, we first introduce the SAA method to approximate the proposed model using a set of sample scenarios. Then, we propose an LBBD method to solve the SAA model.

4.1. Sample Average Approximation

SAA method is a Monte Carlo simulation-based approach for solving stochastic discrete optimization problems (Kleywegt et al. 2002). The basic idea is to approximate the expected value function with a set of random samples. The expected value function is then replaced by the sample average function, while the resulted sample average optimization problem is solved to obtain approximate solutions to the original stochastic problem. The SAA method has been successfully applied to solve a wide range of scheduling problems (Mancilla and Storer 2012, Bentaha et al. 2014, Liu et al. 2020).

We generate a set $S=\{1,2,\ldots,|S|\}$ of samples of the random parameter vector to approximate the set Ω. These samples can be generated by Monte Carlo sampling techniques or can be viewed as historical data of $|S|$ observations of the random vector ω (Shapiro et al. 2014). For any $x\in\mathcal{X}$, we can estimate the expected value $\mathbb{E}_\omega[Q(\boldsymbol{x},\omega)]$ by averaging values $\mathbb{E}_\omega[Q(\boldsymbol{x},\omega^s)]$, $s=1,2,\ldots,|S|$, i.e., $\frac{1}{|S|}\sum_{s\in S}C^s_{max}$. The original S-OASD problem can be reformulated using the SAA method as a deterministic MILP, denoted as model P_3:

$$\min\ w_1\sum_{j\in J}\sum_{k\in F}(g_{jk}+c_{jk})v_{jk}+w_2\frac{1}{|S|}\sum_{s\in S}C^s_{max} \tag{18}$$

s.t. (5), (11) and to

$$C^s_{max}\geq C^s_j+\sum_{k\in F}t^s_{jk}v_{jk} \qquad \forall\, j\in J, s\in S \tag{19}$$

$$C^s_j\geq\sum_{k\in F}(p^s_{jk}+r_k)v_{jk} \qquad \forall\, j\in J, s\in S \tag{20}$$

$$C^s_j\geq C^s_i+p^s_{jk}-h(3-x^s_{ij}-v_{jk}-v_{ik}) \qquad \forall\, i,j\in J, i<j, k\in F, s\in S \tag{21}$$

$$C^s_i\geq C^s_j+p^s_{ik}-h(2+x^s_{ij}-v_{jk}-v_{ik}) \qquad \forall\, i,j\in J, i<j, k\in F, s\in S \tag{22}$$

$$C^s_j\geq 0 \qquad \forall\, j\in J, s\in S \tag{23}$$

$$x^s_{ij}\in\{0,1\} \qquad \forall\, i,j\in J, i<j, s\in S. \tag{24}$$

The above model can be directly solved by an MIP solver. However, given the NP-hard nature of the studied problem, current MIP solvers can only handle instances with a small number of scenarios. Next, we propose an LBBD method to solve the problem.

4.2. Logic-based Benders decomposition

As introduced in the literature review section, the LBBD method is widely used in solving complex combinatorial optimization problems. The LBBD method is particularly effective for those problems with a problem-specific decomposition structure. The studied S-OASD problem consists of two classes of decisions, i.e., the order assignment variables and the order scheduling variables, which shows a good decomposition structure. This motivates us to tailor the LBBD method to solve the proposed S-OASD problem. We next present the proposed LBBD method, sequentially introducing the MP, SP, and Benders cuts.

4.3. Solving the MP

For the S-OASD problem, the MP disregards the sequencing variables and constraints, only considering the order assignment decisions. Optimally solving the MP gives a lower bound to the S-OASD problem. The MP can be formulated as follows:

$$\min w_1 \sum_{j\in J}\sum_{k\in F}(g_{jk}+c_{jk})v_{jk} + w_2 \frac{1}{|S|}\sum_{s\in S} C^s_{max} \tag{25}$$

s.t. (5), (11), and to

$$C^s_{max} \geq r_k + \sum_{j\in O} p^s_{jk} v_{jk} + \min_{j\in O} t^s_{jk} \qquad \forall k \in F, s \in S \tag{26}$$

$$cuts. \tag{27}$$

The objective function (25) minimizes the weighted sum of total cost and expected makespan. Note that the makespan C^s_{max} represents a lower bound on the real makespan, since order sequencing constraints are relaxed and not included in the MP. Constraints (26) ensure that the makespan must be greater than or equal to the total processing time of all orders assigned to a manufacturer, the minimum delivery time of all orders from the manufacturer, and the release time of the manufacturer. Constraints (27) ($cuts$) are the Benders cuts generated by solving the SP with a given incumbent solution to the MP. Initially, (27) is set to $\emptyset$. The MP is solved by a branch-and-cut (B&C) method and augmented by dynamically adding Benders cuts upon obtaining a feasible solution. Once a feasible solution to the MP is identified, the SP is solved to get an upper bound to the S-OASD and generate Benders cuts.

4.4. Solving the SP

In a feasible solution to the MP, we obtain the values of the order assignment variable v_{jk} and the corresponding set J_k of orders assigned to manufacturer k. The SP consists of scheduling the orders assigned to each manufacturer k under each scenario s, which can be decomposed into a set of independent single machine scheduling problems.

We define another binary variable $y_{ij}^s = 1$ if order i is processed immediately before order j at manufacturer k ($i, j \in J_k$). Then, the SP for manufacturer k (SP_k) can be written as:

$$(SP_k) \ \min \frac{1}{|S|} \sum_{s \in S} C_{max}^s \tag{28}$$

s.t.

$$y_{ij}^s + y_{ji}^s \leq 1 \qquad \forall i, j \in J_k, s \in S \tag{29}$$

$$C_j^S \geq C_i^s + p_{jk}^s - M(1 - y_{ij}^s) \qquad \forall i, j \in J_k, i < j, k \in F, s \in S \tag{30}$$

$$C_j^s \geq p_{jk}^s + r_k \qquad \forall j \in J_k, s \in S \tag{31}$$

$$C_{max}^s \geq C_j^s \qquad \forall j \in J_k, s \in S \tag{32}$$

$$C_j^s \geq 0 \qquad \forall j \in J_k, s \in S \tag{33}$$

$$y_{ij}^s \in \{0, 1\} \qquad \forall i, j \in J_k, s \in S. \tag{34}$$

The objective function (28) minimizes the expected makespan. Constraints (29) indicate that order i cannot be the predecessor and successor of order j simultaneously. Constraints (30) and (31) compute the completion time of orders. Constraints (32) determine the makespan.

The above model can be further decomposed into $|S|$ subproblems, and each corresponds to a $1|t_{jk}^s|C_{max}^s$ problem for a scenario s. As mentioned, the $1|t_{jk}^s|C_{max}^s$ problem can be optimally solved in polynomial time using the LDT rule. Optimally solving the SP provides an upper bound to the S-OASD problem. The obtained solution of the SP is then used to form Benders cuts to be added to the MP.

4.5. Generating Benders cuts

The generation of Benders cuts plays an important role in the LBBD method. The MP is augmented by Benders cuts such that the upper bound of the SP and the lower bound of the MP converge to optimality. For the studied S-OASD problem, given a feasible solution to the MP, which determines the order assignment variables, the resulted SP is always feasible. Thus we propose an optimality cut which is dynamically added to the MP.

4.5.1. Optimality cut The basic optimality cut is presented as follows:

$$C_{max}^s \geq \hat{C}_{max}^s(J_k) \left(\sum_{j \in J_k} v_{jk} - |J_k| + 1 \right) \qquad \forall k \in F, s \in S \tag{35}$$

where $\hat{C}_{max}^s(J_k)$ is the corresponding makespan in the current iteration. Cuts (35) ensure that, in subsequent iterations, if all orders in the set J_k are assigned to manufacturer k, the makespan under

scenario s should be greater than or equal to the makespan obtained in the current iteration. The cut is weak since once an order is removed from J_k, the cut becomes non-binding as its right-hand side takes a non-positive value. We next introduce two ways to strengthen the cut.

The first way is to strengthen the cut by finding the minimal infeasible subset of J_k. The key idea is to reduce the size of J_k to obtain a minimal subset J_k^* such that the resulting makespan under scenario s is still equal to $\hat{C}^s_{max}(J_k)$. To obtain J_k^*, we gradually remove orders, one at a time, from J_k and recalculate the makespan; if the expected makespan is still equal to $\hat{C}^s_{max}(J_k)$, we exclude j from J_k and continue to check the following order until all orders are enumerated. Finally, we can replace J_k in (35) with the reduced set and get the cut below:

$$C^s_{max} \geq \hat{C}^s_{max}(J_k^*)\left(\sum_{j\in J_k^*} v_{jk} - |J_k^*| + 1\right) \qquad \forall\, k \in F, s \in S. \tag{36}$$

In subsequent iterations, cut (36) is only useful for solutions where all assignment variables in J_k^* take exactly the same value as in the current iteration, i.e., $v_{jk} = 1, \forall j \in J_k^*$. We expect the cut to be useful when some orders in the set J_k^* are assigned to other manufacturers. To this end, we can derive a stronger cut by exploiting the analytical information of the cut, denoted as analytical Benders cut (Hooker 2007a). The cut is presented as follows:

$$C^s_{max} \geq \hat{C}^s_{max}(J_k^*)v_{qk} - \sum_{j\in J_k^*\setminus\{q\}} p^s_{jk}(1 - v_{jk}) \qquad \forall\, k \in F, s \in S, \tag{37}$$

where q represents the first processed order in set J_k^* with the largest delivery time, i.e., $q = \arg\max_{j\in J_k^*} t^s_{jk}$. Cuts (37) contain analytical information that explores the neighbors of the current order assignment scheme. In subsequent iterations, if order j is excluded from the current assignment J_k^*, the makespan value can be reduced by p^s_{jk}, and the right-side value of the cut remains positive. Thus the cuts can still impose lower bounds on the makespan even if order assignments change in subsequent iterations. Next, we prove the validity of this cut.

4.5.2. Validity of the proposed Benders cut To prove the validity of the proposed analytical Benders cut (37), we must prove that the cut does not remove any feasible solutions. To this end, we start with the following proposition.

Proposition 2. *Let J_k^* be the set of orders assigned to manufacturer k, $C^s_{max}(J_k^*)$ be the corresponding makespan of the set of orders J_k^* processed at manufacturer k under scenario s, and q be the first processed order in J_k^*. The optimal makespan under scenario s can decrease by at most p^s_{ik} when order $i \in J_k^*$ is removed from J_k^*, $i \neq \arg\max_{j\in J_k^*} t^s_{jk}$.*

Proof. The optimal makespan $C^s_{max}(J^*_k)$ under scenario s can be obtained using the LDT rule, which indicates that the first processed order is $q = \arg\max_{j \in J^*_k} t^s_{jk}$. Given an order i to be removed from J^*_k, there is always an order j that precedes order i since the first order q is always in the set. We can obtain the optimal makespan of the reduced set $J'_k = J^*_k \setminus \{i\}$ by excluding i, and the sequence of other orders in the set remains the same. The makespan reduction under scenario s is of at most $p^s_{ik} + t^s_{ik} - t^s_{jk}$, where order j is an order that precedes order i. Since the orders in J'_k are sequenced using the LDT rule with $t^s_{ik} \leq t^s_{jk}$, the resulting makespan reduction is thus of at most p^s_{ik}. Hence, the optimal makespan decrease of removing order i from J^*_k is less than or equal to p^s_{ik}. □

Corollary 1. *Given a set J^*_k of orders assigned to manufacturer k, the optimal makespan reduction under scenario s obtained by excluding the set $H \subseteq J^*_k$ of orders from J^*_k is of at most $\sum_{j \in H} p^s_{jk}$, where H does not contain the first processed order q in set J^*_k, i.e., $q \notin H, q = \arg\max_{j \in J^*_k} t^s_{jk}$.*

Proof. Given the optimal makespan $C^s_{max}(J^*_k)$ of the set J^*_k of orders processed at manufacturer k under scenario s, orders in the set H can be removed, one by one, from J^*_k, and each removal of a order $j \in H$ will decrease the makespan of the resulting set $J^*_k \setminus \{j\}$ by at most p^s_{jk}. The sum of the decreases is at most $\sum_{j \in H} p^s_{jk}$. □

Proposition 3. *The proposed optimality cut* (37) *is valid.*

Proof. Let the cut generated in the current iteration be $C^s_{max} \geq \hat{C}^s_{max}(J^*_k) v_{qk} - \sum_{j \in J^*_k \setminus \{q\}} p^s_{jk}(1 - v_{jk})$, where $\hat{C}^s_{max}(J^*_k)$ is the optimal makespan at manufacturer k under scenario s. If the same set of orders is assigned to manufacturer k in subsequent iterations, then the value of $\sum_{j \in J^*_k \setminus \{q\}} p^s_{jk}(1 - v_{jk})$ is zero and $v_{qk} = 1$. The right-hand side value becomes $\hat{C}^s_{max}(J^*_k)$, which indicates that the cut is valid. Otherwise, a set $H \subseteq J^*_k$ of orders are excluded from J^*_k, and there are two possible cases: 1) if the first processed order in J^*_k is removed from the set, i.e., $q \in H$, the right-hand side of (37) becomes non-positive and the cut is valid; 2) if the first processed order remains in the set, i.e., $q \notin H$, then the right-hand side is reduced by $\sum_{j \in H} p^s_{jk}$. We can derive from Corollary 1 that the optimal makespan of the reduced set of orders at manufacturer k is $C^s_{max}(J^*_k \setminus H) \geq \hat{C}^s_{max}(J^*_k) - \sum_{j \in H} p^s_{jk}$. Hence, the reduction makes the right-hand side less than or equal to the real value of $C^s_{max}(J^*_k \setminus H)$ and the cut remains valid. □

4.6. Outline of LBBD method for solving the S-OASD problem

The above procedure is implemented by a *branch-and-check* routine, which is an advanced implementation method of the LBBD algorithm (Tran et al. 2016). The *branch-and-check* implementation has proved to be efficient in solving problems with hard MPs and easy SPs. It operates on a single search tree, generates cuts upon finding each feasible solution, and adds cuts to the nodes that remain to be fathomed. In this case, the MP is solved only once, and no node is evaluated

repeatedly. In a classic implementation of the LBBD method, the MP is repeatedly solved to optimality, and cuts are only generated and added at the end of each optimal solution. The general framework of the proposed LBBD with *branch-and-check* implementation is depicted in Algorithm 1.

Algorithm 1 LBBD for the S-OASD problem

1. Formulate the MP and SP, and initialize $cuts \leftarrow \emptyset$
2. **While** (an optimal solution is not found and the time limit is not reached)
3. Solve the MP with the B&C procedure in a single search tree:
4. **If** a feasible solution is found, **do**
5. Solve the SP
6. Add cut (37) to *cuts*. Then, go to step 2
7. **End if**
8. **End while**
9. Output the best obtained solution

5. Computational study

We conduct numerical experiments on randomly-generated instances for both problems to demonstrate the performance of the proposed model and algorithms. All the computational experiments are conducted on a PC equipped with a Core i7 CPU at 3.2 GHz and 16 GB RAM. The developed models and the LBBD method are coded in C++ linked with IBM ILOG CPLEX 12.10. All MILP models are solved by CPLEX using the default branch-and-cut (B&C) algorithm. The MP of the LBBD is also solved with B&C, while the iterations between MP and SP are achieved by the LazyCallback function of CPLEX. A maximum time limit for each run is set to 1800 seconds.

5.1. Instance generation and parameter setting

For the S-OASD with total cost and makespan minimization, we generated 100 instances with up to 100 orders, 20 manufacturers, and 300 scenarios. The number of orders (customers) is set to $n = \{20, 40, \ldots, 100\}$, the number of manufacturers is set to $m = \{5, 10, \ldots, 20\}$, and the number of scenarios is set to $s = \{20, 60, 100, 200, 300\}$.

Considering the high complexity, the instances are comparatively smaller in size for S-OASD-T. We generated 72 instances with up to 100 orders, ten manufacturers, and 100 scenarios as follows. The number of orders (customers) is set to $n = \{10, 20, 40, 60, 80, 100\}$, the number of manufacturers is set to $m = \{2, 5, 8, 10\}$, and the number of scenarios is set to $s = \{20, 60, 100\}$.

The parameter setting for both problems are generally the same. The coordinates of manufacturers and order distribution location are generated from the interval $U[1, 200]$. The transportation

cost is set to be equal to the euclidean distance between the order i and its assigned manufacturer k which is calculated as $\left\lfloor \sqrt{(X_i - X_k)^2 + (Y_i - Y_k)^2} \right\rfloor$. The production cost g_{jk} for order j at manufacturer k is generated by first generating the value β_j from the interval $U[30, 80]$, and then generate g_{jk} from interval $U[\beta_j - 20,\ \beta_j + 20]$.

The mean processing time $\mathbb{E}[p_{jk}]$ of order j at manufacturer k was randomly generated from $U[\beta_j - 20,\ \beta_j + 20]$, where β_j was randomly generated from the interval $U[30, 80]$. The maximum deviations of the processing time was set to $0.2 \cdot \mathbb{E}[p_{jk}]$. The release time r_k ranges from the interval $U[0,\ 1/2 \sum_{j \in J, k \in F, s \in S} p^s_{jk}/(m^2 s)]$. The mean distribution time $\mathbb{E}[t_{jk}]$ was set to be equal to c_{jk}. The maximum deviations of the distribution time, similar to that of p^s_{jk}, was set to $0.2 \cdot \mathbb{E}[t_{jk}]$. Specifically, each order j was given a due date d_j in S-OASD-T to minimize the total tardiness. Here, we followed the method in Shim and Kim (2007) and Ching-Fang Liaw and Chen (2003) that d_j was generated from a uniform distribution from $[\bar{P}(1 - TF - RF/2), \bar{P}(1 - TF + RF/2)]$, where $\bar{P} = \sum_{j \in J, k \in F, s \in S} p^s_{jk}/(m^2 s)$, and the tightness factor TF and range factor RF were set to $TF = RF = 0.2$.

5.2. Weight determination

The considered S-OASD problem minimizes the weighted sum of the total cost and makespan. The weights of different objective components are the keys for the problem as they indicate the preference of the decision-maker and have great impact on the results of the problem. We denote F_1 and F_2 as the two objective components and denote the objective function as $min\ w_1 F_1 + w_2 F_2$. Then we apply the following method to determine the value of w_1 and w_2.

$$w_1 + w_2 = 1 \tag{38}$$

$$w_1 (F_1^{max} - F_1^{min}) = w_2 (F_2^{max} - F_2^{min}), \tag{39}$$

where F_1^{min}, F_1^{max}, F_2^{min} and F_2^{max} are calculated as follows.

$$F_1^{min} = \{min f_1\}, \qquad i.e., w_1 = 1, w_2 = 0 \tag{40}$$

$$F_2^{min} = \{min f_2\}, \qquad i.e., w_1 = 0, w_2 = 1 \tag{41}$$

$$F_1^{max} = \{f_1 | f_2 = F_2^{min}\} \tag{42}$$

$$F_2^{max} = \{f_2 | f_1 = F_1^{min}\} \tag{43}$$

It is difficult to compute the $\mathbb{E}_\omega[Q(\boldsymbol{x}, \omega)]$ for a given order assignment to manufacturers, since the size of the set Ω may be very large (Bentaha et al. 2014). To tackle this difficulty, we apply the popular SAA method to solve the two-stage S-OASD problem. Based on the SAA method,

we then propose a new MILP model, which can be efficiently solved using off-the-shelf solvers, to approximately solve the S-OASD problem. We develop a stochastic LBBD method for solving the SAA of the S-OASD problem.

To determine the values of w_1 and w_2, we solve the new model P_2 using 18 instances with up to 60 orders, ten manufacturers, and 100 scenarios based on the above method. We first set $w_1 = 1$, $w_2 = 0$, and then set $w_1 = 0$, $w_2 = 1$ and obtain the objective function components of the new model, respectively. Then, the weights are determined using equations (38) and (39). The computational results are reported on Table 2. Therefore, in the following computational experiments, the weights are set $w_1 = 0.18$ and $w_2 = 0.82$.

Table 2 **Results of w_1 and w_2**

Weight	Result
F_1^{min}	3662.50
F_2^{min}	407.49
F_1^{max}	4988.56
F_2^{max}	738.94
w_1	0.18
w_2	0.82

5.3. Performance of the new model

As mentioned above, we proposed a new formulation with predetermined processing sequences at each manufacturer for the S-OASD problem. In this section, we test the 100 instances with up to 100 orders and 20 manufacturers to evaluate the performance of the new formulation. We compare the new model with the adapted LO model. The results obtained by the two models are presented in Table 3, where each entity is the average value of four instances with various manufacturers and the symbol "$-$" means that at least one of the four instances is not feasibly solved. In the table, the first two columns show the number of scenarios and orders, respectively. We then report the obtained upper bound (UB), lower bound (LB), average gap (Gap(%)), the number of instances that are solved to optimality (#opt), and the corresponding computation times for the two models, respectively. Note that the relative gap is calculated as follows.

$$\text{Gap}(\%) = \frac{\text{UB} - \text{LB}}{\text{UB}} \times 100. \tag{44}$$

We can see from Table 3 that the new model significantly outperforms the LO model. The new model can provide feasible solutions for all the 100 instances, while LO fails on several instances with 100 or more scenarios. The new model obtains better bounds, smaller gaps, and more optimal solutions in shorter computation time. It solves 15 out of the 100 tested instances, while the LO model only solves one instance to optimality. The average gap is also much smaller than LO on

Table 3 **Comparison results of the LO model and the new model**

s	n	LO model					New model				
		UB	LB	Gap (%)	#Opt	Time (s)	UB	LB	Gap (%)	#Opt	Time (s)
20	20	526.13	505.26	3.55	1	1441.39	523.77	523.77	0.00	4	64.43
	40	938.14	879.12	5.98	0	1800.61	927.04	862.14	6.09	0	1800.09
	60	2308.83	1267.84	26.75	0	1801.12	1342.12	1228.68	8.06	0	1800.19
	80	5370.54	1779.25	62.63	0	1801.74	1906.36	1639.69	12.56	0	1800.27
	100	7962.02	2016.44	74.66	0	1802.56	2173.74	1774.17	17.16	0	1800.53
60	20	515.78	495.83	3.24	1	1760.11	512.60	512.60	0.00	4	248.77
	40	1931.12	769.43	44.54	0	1801.62	824.31	761.26	7.21	0	1800.27
	60	4507.94	1234.83	72.68	0	1803.03	1331.87	1154.03	12.44	0	1800.36
	80	5829.59	1572.77	73.00	0	1805.24	1743.57	1435.68	16.60	0	1800.61
	100	8049.42	2036.51	74.87	0	1808.91	2244.16	1808.95	17.94	0	1800.74
100	20	498.48	475.22	4.14	0	1800.91	492.48	492.48	0.00	4	117.73
	40	2741.99	835.61	68.67	0	1802.76	904.67	802.33	10.46	0	1800.47
	60	4497.96	1161.53	74.29	0	1805.19	1248.66	1049.73	15.29	0	1800.56
	80	5977.82	1669.89	72.33	0	1809.32	1852.34	1550.45	14.97	0	1802.04
	100	-	-	-	-	-	2428.44	1791.77	23.16	0	1801.73
200	20	587.32	546.73	6.68	0	1801.85	573.41	556.01	2.88	1	1638.29
	40	3082.74	860.71	71.99	0	1805.41	916.46	810.77	10.29	0	1800.99
	60	4636.48	1238.16	73.45	0	1810.02	1383.32	1108.77	19.31	0	1801.68
	80	-	-	-	-	-	1886.06	1440.22	22.00	0	1802.38
	100	-	-	-	-	-	5097.52	1761.64	54.05	0	1805.98
300	20	-	-	-	-	-	510.19	502.44	1.27	2	1191.84
	40	-	-	-	-	-	913.12	788.80	12.99	0	1801.70
	60	-	-	-	-	-	1460.68	1133.29	21.31	0	1802.07
	80	-	-	-	-	-	3106.63	1498.82	39.88	0	1802.78
	100	-	-	-	-	-	5009.00	1350.01	58.58	0	1812.47

instances that LO can solve. The performance of LO is heavily affected by the size of the instances. For example, LO obtains an average gap of 3.55% when $n = 20$, but it increases sharply when the number of orders becomes large and rises to 75.66% when $n = 100$. At the same time, the average gap obtained by the new model grows from 0.00% to 17.16%, which is much lower than that of LO.

In summary, the proposed new model significantly outperforms the LO model. It can generally handle instances with up to 300 scenarios, 100 orders, and 20 manufacturers. However, the new model's performance becomes poor when the instances become extremely large. For example, the average gap between instances with 200 scenarios and 100 orders is 54.05%, and it further

increases to 58.58% on instances with 300 scenarios. This means that the model can hardly provide satisfactory solutions to large-sized problems, and we need more efficient algorithms. Next, we report results obtained by the proposed LBBD method and make comparisons with the new model.

5.4. Performance of the new model and the LBBD method

We further evaluate the performance of the proposed LBBD method and the new model on the same 100 instances. The results are presented in Table 4, in which the average gaps (Gap (%)), the number of optimal solutions (#Opt), and the average computation time (Time (s)) of the two methods are reported.

Table 4 Comparison between the new model and the LBBD method

s	n	New model			LBBD		
		Gap (%)	#Opt	Time (s)	Gap (%)	#Opt	Time (s)
20	20	0.00	4	64.43	0.00	4	5.17
	40	6.09	0	1800.09	0.00	4	4.69
	60	8.06	0	1800.19	0.05	3	450.75
	80	12.56	0	1800.27	0.06	2	902.57
	100	17.16	0	1800.53	0.00	4	129.58
60	20	0.00	4	248.77	0.00	4	3.75
	40	7.21	0	1800.27	0.00	4	14.24
	60	12.44	0	1800.36	0.00	4	90.27
	80	16.60	0	1800.61	0.00	4	103.26
	100	17.94	0	1800.74	0.03	3	541.12
100	20	0.00	4	117.73	0.00	4	4.12
	40	10.46	0	1800.47	0.00	4	72.86
	60	15.29	0	1800.56	0.00	4	102.36
	80	14.97	0	1802.04	0.00	4	315.04
	100	23.16	0	1801.73	0.00	4	301.92
200	20	2.88	1	1638.29	0.00	4	52.53
	40	10.29	0	1800.99	0.00	4	83.73
	60	19.31	0	1801.68	0.02	3	488.56
	80	22.00	0	1802.38	0.00	4	504.49
	100	54.05	0	1805.98	0.00	4	330.92
300	20	1.27	2	1191.84	0.00	4	450.58
	40	12.99	0	1801.70	0.00	4	175.98
	60	21.31	0	1802.07	0.04	3	762.96
	80	39.88	0	1802.78	0.09	2	1378.37
	100	58.58	0	1812.47	0.12	2	1085.58
Avg./Tot		16.18	15	1571.96	0.02	90	334.22

Table 4 reports the computational results of the new model and the LBBD method. The LBBD method outperforms the new model since it obtains smaller gaps in shorter computation time. The LBBD method obtains an average gap of 0.02%, indicating that all the instances are solved optimally or to near optimality, while that obtained by the new model is 16.18%, more than 800 times higher than that of LBBD. The LBBD method solves 90 out of the 100 instances to optimality, compared to 15 by the new model. The LBBD method is generally faster than the new model.

The average computation time of the new model is 1571.96 s, almost five times that of LBBD (334.22 s). We can also observe that the number of orders has a great impact on the performance of the new model since all the instances solved to optimality have only 20 orders. For each set of instances with the same number of scenarios, the increase in the number of orders makes the average gap grow. For example, when $s = 200$, the average gap obtained by the new model is 2.88% when $n = 20$, and it becomes 54.05% when $n = 100$. Since all the average gaps obtained by the LBBD are small, it makes little sense to compare the differences when instance parameters change. However, we can find another trend that the larger the instances get, the longer computation time LBBD needs to solve the instances. In summary, LBBD dominates the new model. The new model consumes nearly five times the computation time, more than 800 times in average gaps, but only gets one-sixth the number of optimal solutions compared with the LBBD method.

5.5. Performance of the Benders cuts

In this paper, we first proposed Benders cut (35) and then strengthened it into cut (37). To evaluate the performance of the new cut, we choose all the 60 instances with 100, 200, and 300 scenarios and solve them with LBBD using cut (37) and cut (35), respectively. The computational results are reported in Table 5, in which the LBBD method with cut (37) and cut (35) are denoted as LBBD+(37) and LBBD+(35), respectively. To better illustrate the differences between the two cuts, we output the number of iterations and the number of Benders cuts added to the MP and denote them as #Iter and #Cuts, respectively.

Table 5 Comparison of cut (37) and cut (35)

s	n	LBBD+(37)					LBBD+(35)				
		Gap (%)	#Opt	Time (s)	#Iter	#Cuts	Gap	Optimal	Time	#Iter	#Cuts
100	20	0.00	4	4.12	11.75	3349.50	0.00	4	7.59	15.25	4050.00
	40	0.00	4	72.86	16.25	4775.00	0.00	4	82.19	16.50	5175.00
	60	0.00	4	102.36	20.00	5175.00	0.00	4	117.93	18.75	4775.00
	80	0.00	4	315.04	31.00	7950.00	0.00	4	500.37	29.50	7700.00
	100	0.00	4	301.92	16.75	4500.00	0.00	4	336.07	17.25	4800.00
200	20	0.00	4	52.53	25.00	11943.00	0.00	4	128.87	35.00	16100.00
	40	0.00	4	83.73	16.00	7400.00	0.00	4	96.27	18.00	8450.00
	60	0.02	3	488.56	21.25	11050.00	0.07	3	506.34	27.50	12650.00
	80	0.00	4	504.49	18.25	10000.00	0.00	4	410.35	18.75	10350.00
	100	0.00	4	330.92	31.00	16000.00	0.00	4	508.46	38.25	18800.00
300	20	0.00	4	450.58	40.00	30150.00	0.64	3	720.91	49.75	40725.00
	40	0.00	4	175.98	37.75	23925.00	0.00	4	179.99	37.50	23100.00
	60	0.04	3	762.96	25.00	18300.00	0.06	3	784.57	24.75	17400.00
	80	0.09	2	1378.37	59.25	41025.00	0.09	2	1304.33	53.75	37500.00
	100	0.12	2	1085.58	30.75	28500.00	0.13	2	1103.59	31.50	29700.00
Avg./Tot		0.02	54	407.33	26.67	14936.17	0.07	53	452.52	28.80	16085.00

From Table 5, we can find that both LBBD+(37) and LBBD+(35) perform well in the proposed LBBD method. However, the performance of LBBD+(37) is still better than that of LBBD+(35)

with lower average gaps, more optimal solutions, and shorter computation time. Specifically, the average gap of LBBD+(37) is 0.02% and that of LBBD+(35) is 0.07%. LBBD+(37) obtains 54 out of 60 optimal solutions, one more than LBBD+(35). Lower average gaps and more optimal solutions reflect the slight advantage of LBBD+(37), but the comparison on computational efficiency of LBBD+(37) with LBBD+(35) seems more obvious. The average computation time of LBBD+(37) is 407.33 s, about 10% faster than LBBD+(35) (452.52 s). The number of iterations and Benders cuts of LBBD+(37) are 26.67 and 14936.17, repectively, both of which are fewer than that of LBBD+(35). This shows that the cut (37) is more effective and efficient than (35).

5.6. Sensitivity analysis

In the S-OASD problem investigated in this paper, we set the objective function as the weighted sum of total cost and makespan and determined the value of the two weights with the method of Section 5.2. However, the preference of decision-makers might vary with circumstances since the importance of cost minimization and makespan minimization differs when dealing with different problems or situations. The weight change will affect the solution of the problem and, at the same time, may have an impact on the computational complexity. In this section, we conduct a sensitivity analysis to better understand the influence of weight changes in solving the problem. We choose 36 instances with up to 100 scenarios and 60 orders and change the value of w_1 and w_2. Specifically, we set $w_1 = \{0.1, 0.3, 0.5, 0.7\}$ for each run, and, w_2, due to $w_1 + w_2 = 1$, is also fixed correspondingly. The instances are solved using the proposed LBBD method with a time limit of 1800 seconds for each run. We output the computational results in Table 6 where the first three columns are the combination of w_1 and w_2, the upper bound (UB), and the lower bound (LB). The next two columns show the total cost and the expected makespan, which are denoted as TC and C_{max}, respectively. The last three columns display the average gap (Gap (%)), the number of optimal solutions (#Opt), and the computation time (Time (s)) to show the solution quality and computational efficiency of each combination.

Table 6 Sensitivity analysis

(w_1, w_2)	UB	LB	TC	C_{max}	Gap (%)	#Opt	Time (s)
(0.1,0.9)	628.75	627.99	3662.72	291.64	0.11	29	570.05
(0.3,0.7)	1265.78	1265.78	3395.83	352.91	0.00	36	2.26
(0.5,0.5)	1861.59	1861.59	3321.47	401.71	0.00	36	0.56
(0.7,0.3)	2439.17	2439.17	3290.22	453.37	0.00	36	0.30
(0.9,0.1)	2997.31	2997.31	3272.08	524.39	0.00	36	0.14

We can observe from Table 6 that when $(w_1, w_2) = (0.1, 0.9)$, LBBD solves 29 out of 36 instances to optimality and the average gap is 0.11%. When w_1 gets larger, the value of TC decreases while the C_{max} increases. At the same time, the objective value, i.e., the UB increases since TC takes a larger

Table 7 Computational results of out-of-sample analysis

n	m	New model				LBBD method			
		Obj	*PT*85	*PT*99	Time (s)	*Obj*	*PT*85	*PT*99	Time (s)
20	5	557.06	563.94	577.06	1464.65	557.056	563.94	577.06	8.098
	10	500.22	507.18	519.48	1810.83	493.588	500.16	510.00	11.303
40	5	1383.22	1395.02	1417.16	1830.08	1320.35	1330.82	1348.04	359.139
	10	856.93	863.52	875.82	1812.63	849.304	856.34	868.64	55.356
60	5	1847.32	1868.00	1904.90	1807.63	1750.84	1765.80	1791.22	1.176
	10	1378.01	1388.82	1410.96	2091.99	1338.45	1347.14	1364.36	26.733
80	5	3281.40	3315.40	3364.60	1826.81	2508.68	2522.96	2560.68	4.732
	10	1967.31	1984.34	2021.24	1832.28	1823.56	1834.58	1856.72	20.431
100	5	3308.06	3337.42	3392.36	1819.81	2916.84	2933.56	2964.72	3.758
	10	2660.71	2688.20	2745.60	1831.00	2130.22	2139.92	2158.78	270.086
Average		1774.02	1791.18	1822.92	1812.77	1568.89	1579.52	1600.02	76.08

portion in UB than C_{max}. Also, when w_1 equals 0.3 or larger, all 36 instances are optimally solved, and the computation time gets shorter. The average computation time is 570.05 s when $w_1 = 0.1$, it decreases sharply to 2.26 s when $w_1 = 0.3$ and further becomes as short as 0.14 s when $w_1 = 0.9$. This means that when w_1, the weight of total cost, increases, the problem becomes easier to solve. The reason might be that the decisions on total cost only concern the order assignments, which is much easier to be solved as decision-makers can assign orders to manufacturers that cost less. On the contrary, decisions concerning makespan minimization need to determine sequencing assigned orders at each manufacturer, which is much more complicated than the assignment decisions.

5.7. Out-of-sample analysis

In this section, we analyze the performances of the new model P_2 and the proposed LBBD method when tackling some difficult problems with numerous scenarios. We generate ten instances with $n = \{20, 40, \ldots, 100\}$, $k = \{5, 10\}$ and $\Omega = 1000$. The parameter settings of the ten instances are consistent with those of Section 5.1. Preliminary experiments show that the problem is extremely hard to solve due to the number of scenarios. Therefore, we take an out-of-sample method with two steps. Firstly, we solve the instances by selecting 200 samples ($S = 200$) out of the 1000 scenarios and determine the assignments of the orders to the manufacturers. Secondly, we solve the 1000-scenario instances based on the assignment results and determine the sequencing of orders at each manufacturer under each scenario. The computation time limit is set to 1800 seconds, and the computational results are reported in Table 7. In the table, the first two columns show the number of orders (n) and manufacturers (m), respectively. To evaluate the performances of the two approaches, we output four key indicators, i.e., the objective values (Obj), the 85 percentile and the 99 percentile of the objective value (denoted as $PT85$ and $PT99$, respectively), and the computation time.

From the table, we can find that both the new model and the LBBD method solve all the ten instances and LBBD significantly outperforms the new model. The average objective value obtained

Table 8 Out-of-sample analysis with normal distribution

n	m	New model				LBBD method			
		Obj	*PT*85	*PT*99	Time (s)	*Obj*	*PT*85	*PT*99	Time (s)
20	5	736.05	738.46	741.74	1812.68	734.69	736.82	739.28	1818.17
	10	449.73	450.56	453.84	246.41	449.73	450.56	453.84	2.59
40	5	1167.01	1169.76	1174.68	1816.85	1146.44	1148.46	1152.56	2.51
	10	976.77	979.20	982.48	1848.85	958.03	960.40	964.50	1826.28
60	5	1796.15	1799.20	1804.94	1850.00	1758.16	1760.16	1765.90	17.50
	10	1316.44	1318.64	1321.92	1860.33	1291.59	1293.24	1297.34	546.52
80	5	2587.03	2589.64	2597.02	1868.99	2390.08	2392.76	2400.14	18.31
	10	1839.76	1842.60	1847.52	1822.79	1682.91	1685.26	1688.54	68.54
100	5	3719.87	3725.00	3733.20	2252.19	2921.32	2924.58	2930.32	63.60
	10	2584.10	2588.52	2595.08	2330.49	2175.56	2177.62	2185.00	321.52
Average		1717.29	1720.16	1725.24	1770.96	1550.85	1552.99	1557.74	468.56

by the new model is 1774.02, about 13.08% higher than that of LBBD (1568.89). The *PT*85 and *PT*99 of the new model are 1791.18 and 1822.92, while they are 1579.52 and 1600.02 for LBBD, with a difference of 13.40% and 13.94%, respectively. The differences show that LBBD can obtain solutions with higher solution qualities. For the computation time, LBBD is significantly more efficient than the new model. LBBD solves all the ten instances within 360 seconds each, while the new model needs at least 1464.65 s to solve any of the instances. The average computation time of the new model is 1812.77 s, about 23.8 times that of the LBBD method (76.08 s). In summary, LBBD is superior to the new model in both solution quality and computational efficiency.

In this paper, the processing time p^s_{jk} and distribution time t^s_{jk} are generated from a uniform distribution in the computational study. Now, we proceed to explore the influence on the problem solving when p^s_{jk} and t^s_{jk} are generated from normal distributions. For this end, we generate another ten instances with the same parameter setting as the above out-of-sample analysis, except the generation of p^s_{jk} and t^s_{jk}. Specifically, p^s_{jk} is generated from a normal distribution whose average and variance are $\mathbb{E}[p_{jk}]$ and 1, respectively. Similarly, t^s_{jk} is generated from a normal distribution whose average and variance are $\mathbb{E}[t_{jk}]$ and 1, respectively. Note that $\mathbb{E}[p_{jk}]$ and $\mathbb{E}[t_{jk}]$ are defined in Section 5.1. The computational results are reported in Table 8.

From Table 8, we can observe that both the two approaches can provide feasible solutions for all the instances. The average objective value of the new model and LBBD are 1717.29 and 1550.85, indicating that LBBD can find solutions of better quality. The 85 percentile and 99 percentile objective values obtained by the new model are 1720.16 and 1725.24, while they are 1552.99 and 1557.74 for the LBBD, respectively. The new model obtains an average computation time of 1770.96 seconds, about 3.78 times that of LBBD (468.58 s).

5.8. Extended problem minimizing total tardiness

In this subsection, we extend the S-OASD problem to a S-OASD-T, which minimizes the weighted sum of the total cost and the total tardiness. In many practical applications, orders are expected

to be completed before a due date to meet the need of customers, otherwise they might suffer great loss. To describe the S-OASD-T, we define the due date of order j as d_j indicating the time when the order arrives at its customer. If the actual arrival time exceeds the due date under scenario ω, we say that the order is delayed. The tardiness of order j can be formulated as $T_j^\omega = \max(C_j^\omega + t_{jk}^\omega - d_j, 0)$. Note that t_{jk}^ω indicates the distribution time of order j from manufacturer k to its designated location under scenario ω. The total tardiness under scenario ω is defined as $T_{tot}^\omega = \sum_{j\in J} T_j^\omega$. Based on the parameters and decision variables defined in Section 3.2, we can formulate S-OASD-T as follows.

$$(P_4)\ \min\ w_1 \sum_{j\in J}\sum_{k\in F}(g_{jk}+c_{jk})v_{jk} + w_2\mathbb{E}_\omega[T_{tot}^\omega] \tag{45}$$

s.t. (5), (7)–(12), and to

$$T_j^\omega \geq C_j^\omega + \sum_{k\in F} t_{jk}^\omega v_{jk} - d_j \qquad \forall\, j\in J, \omega\in\Omega \tag{46}$$

$$T_{tot}^\omega \geq \sum_{j\in J} T_j^\omega \qquad \forall\, \omega\in\Omega \tag{47}$$

$$T_j^\omega, T_{tot}^\omega \geq 0 \qquad \forall\, j\in J, \omega\in\Omega. \tag{48}$$

5.8.1. LBBD for the S-OASD-T The total tardiness minimization is much more difficult to solve compared with the makespan minimization. It has been proved that a single machine scheduling problem with release date and total tardiness minimization is NP-hard (Lawler 1977), while the problem is polynomial-time solvable when the objective is makespan minimization (Baker and Trietsch 2013). We next solve the S-OASD-T using the proposed LBBD framework and introduce the MP, the SP, and the Benders cuts, respectively.

The MP for the S-OASD-T that minimizes the weighted sum of total cost and tardiness. We define an auxiliary variable τ_k^s indicating the total tardiness of manufacturer k under scenario s. Then, the MP can be written as:

$$\min\ w_1 \sum_{j\in J}\sum_{k\in F}(g_{jk}+c_{jk})v_{jk} + w_2\frac{1}{|S|}\sum_{s\in S} T_{tot}^s \tag{49}$$

s.t. (5), (11), (48), and to

$$T_{tot}^s \geq \sum_{j\in J} T_j^s \qquad \forall\, s\in S \tag{50}$$

$$T_{tot}^s \geq \sum_{k\in F} \tau_k^s \qquad \forall\, s\in S \tag{51}$$

$$T_j^s \geq \sum_{k\in F}(r_k + p_{jk}^s + t_{jk}^s)v_{jk} - d_j \qquad \forall\, j\in J, s\in S \tag{52}$$

$$\tau_k^s \geq r_k + \sum_{j \in J} p_{jk}^s v_{jk} - \max_{j \in J}\{d_j - t_{jk}^s\} \qquad \forall\, k \in F, s \in S \tag{53}$$

$$\tau_k^s \geq 0 \qquad \forall\, k \in F, s \in S \tag{54}$$

$$cuts. \tag{55}$$

The objective (49) minimizes the weighted sum of total cost and expected total tardiness. Constraints (50) indicate that the total tardiness under each scenario s is at least the sum of the tardiness of all the orders. Constraints (51) indicate that the total tardiness is at least the sum of the total tardiness of each manufacturer. Constraints (52) and (53) are two inequalities providing lower bounds for T_j^s and τ_k^s, respectively. Constraints (52) ensure that the tardiness of each order j is at least the sum of the earliest available time, the processing time, and the delivery time of the order to the manufacturer that it is assigned to. Constraints (53) limit the tardiness of each manufacturer to be at least its earliest available time plus the total processing time of the orders assigned to it, minus the maximum modified due date of an order to the manufacturer, i.e., $\max_{j \in J}\{d_j - t_{jk}^s\}$.

For S-OASD-T, the SP at manufacturer k (SPT_k) is a single machine scheduling problem under uncertainty to minimize the expected total tardiness, which can be denoted as $1|t_{jk}^s|\sum T_j^s$. We formulate the SPT_k as follows.

$$(SPT_k) \ \min \frac{1}{|S|} \sum_{j \in J_k, s \in S} T_j^s \tag{56}$$

s.t. (29)–(31), (33), (34), and to

$$T_j^s \geq C_j^s + t_{jk}^s - d_j \qquad \forall\, j \in J_k, s \in S \tag{57}$$

$$T_j^s \geq 0 \qquad \forall\, j \in J_k, s \in S, \tag{58}$$

where the objective function (56) minimizes the expected total tardiness and constraints (57) compute the tardiness of order j under scenario s.

The $1|t_{jk}^s|\sum T_j^s$, i.e., the single machine scheduling problem with release date and total tardiness minimization problem is NP-hard (Pinedo and Rammouz 1988) and is much more complicated than the $1|t_{jk}^s|C_{max}^s$. Some researchers have proposed various approaches to solve the problem (França et al. 2001, Cheng et al. 2005, Cordone and Hosteins 2019). Among them, Tanaka and Fujikuma (2012) developed an exact algorithm based on dynamic programming (DP) which has been proven to be effective. The DP-based algorithm has also successfully solved some similar problems (Tanaka and Sato 2013, Tanaka and Araki 2013, Şen and Bülbül 2015). In this paper, we also use this DP-based algorithm.

The optimality cut for the S-OASD-T is similar to that of the S-OASD and can be formulated as:

$$\tau_k^s \geq \hat{\tau}_k^s(J_k)\left(\sum_{j\in J_k} v_{jk} - |J_k| + 1\right) \qquad \forall\, k \in F, s \in S. \tag{59}$$

Note that the $\hat{\tau}_k^s$ is the total tardiness in the current iteration. Cuts (59) indicate that for each manufacturer k, the total tardiness under scenario s is at least $\hat{\tau}_k^s$ in subsequent iterations if all orders in the set J_k are assigned to it.

5.8.2. Computational results We compare the proposed LBBD method with the LO formulation for S-OASD-T that minimizes the weighted sum of total cost and tardiness. We run the model and algorithm on 72 randomly generated instances with up to 100 scenarios, 100 orders, and ten manufacturers. The weights are set $w_1 = 0.18$, $w_2 = 0.82$. The computational results are reported in Table 9, in which #Feas indicates the number of feasible solutions the model and algorithm provide.

Table 9 Computational results of S-OASD-T

s	n	LO				LBBD			
		#Feas	#Opt	Gap (%)	Time (s)	#Feas	#Opt	Gap (%)	Time (s)
20	10	4	2	7.98	900.71	4	4	0.00	0.14
	20	4	0	25.73	1800.08	4	4	0.00	3.15
	40	4	0	57.68	1800.39	4	3	16.25	644.55
	60	4	0	81.29	1800.56	4	3	14.87	890.69
	80	4	0	87.00	1800.74	4	0	20.40	1800.47
	100	4	0	88.59	1801.06	4	0	11.85	1801.09
60	10	4	2	3.74	902.74	4	4	0.00	0.04
	20	4	0	26.02	1800.21	4	4	0.00	2.08
	40	4	0	65.39	1800.63	4	3	0.05	456.75
	60	4	0	88.06	1801.31	4	0	2.59	1800.39
	80	4	0	98.73	1802.23	4	0	19.06	1801.12
	100	4	0	98.78	1803.00	4	0	8.52	1802.07
100	10	4	2	5.46	974.15	4	4	0.00	0.08
	20	4	0	28.13	1800.29	4	3	7.11	450.40
	40	4	0	73.51	1800.92	4	4	0.00	289.37
	60	4	0	98.20	1802.36	4	1	1.92	1680.05
	80	1	-	-	-	4	0	8.29	1801.54
	100	0	-	-	-	4	0	8.69	1805.09
Avg./Tot		65	6	-	-	72	37	6.64	946.06

From Table 9 we can find that LBBD outperforms LO significantly in both capability, solution quality, and computational efficiency. First, LBBD can solve all 72 instances while LO can only provide 65 feasible solutions. LO fails to solve most instances with 100 scenarios and 80 or more orders. LBBD obtains 37 optimal solutions, slightly more than half of the total instances, while

Table 10 Out-of-sample analysis of S-OASD-T

n	m	New model				LBBD method			
		Obj	*PT*85	*PT*99	Time (s)	*Obj*	*PT*85	*PT*99	Time (s)
10	5	172.21	181.92	199.14	1842.07	172.21	181.92	199.14	0.85
	10	171.61	182.50	198.08	206.86	171.61	182.50	198.08	0.43
20	5	587.09	620.12	670.96	3021.11	450.87	459.20	487.90	10.96
	10	506.97	534.74	576.56	2652.06	356.12	368.78	393.38	2.20
30	5	4060.41	4263.34	4671.70	3770.38	653.27	663.16	701.70	28.74
	10	806.95	852.54	944.38	4511.58	537.71	546.68	566.36	21.36
40	5	-	-	-	-	1015.89	1051.14	1122.48	1814.64
	10	-	-	-	-	636.22	643.94	671.00	10.74
50	5	-	-	-	-	1148.02	1173.78	1226.26	1926.13
	10	-	-	-	-	816.09	827.58	864.48	1276.31
Average		-	-	-	-	595.80	609.87	643.08	509.23

LO only solves six instances with ten orders to optimality. Considering the average gaps, LO only performs well on the smallest instances ($n = 10$) with average gaps below 10%. When the number of orders becomes larger, the average gap increases sharply. For example, when $s = 60$, the average gap of LO is 3.74% when $n = 10$, and it becomes more than 88% when $n = 60$. When $s = 60, n = 100$, the average gap of LO is 98.78%, indicating that LO has lost the capability of finding good solutions. As for LBBD, the average gap is significantly lower than that of LO. The average gap of all the instances obtained by LBBD is 6.64% and the largest one is 20.40% when $s = 20, n = 80$. The size of the instances also has a significant impact on the performance of LBBD. For each group of scenarios, most of the instances with $n \leq 60$ are optimally solved, but none of those with $n \geq 80$ is solved to optimality. The computation time of LBBD has a similar trend as the larger the instances are, the longer computation time LBBD consumes.

Next, we compare the performances of the LO model with LBBD in the out-of-sample analysis as in Section 5.7. Considering to the complexity of S-OASD-T, we generate ten instances with $n = \{10, 20, \ldots, 50\}$, $k = \{5, 10\}$ and $\Omega = 1000$. Here, we determine the order assignments using 100 samples. The computational results are reported in Table 10.

From Table 10, we can observe that the LO model only solves six of the ten instances with $n \leq 30$, while LBBD can provide feasible solutions for all the ten instances. For the first six instances, LBBD can obtain solutions with equal or lower objective values, indicating that the average solution quality obtained by LBBD is higher than that by LO. The computational efficiency of LBBD is also better than LO. For example, LBBD and LO obtain the same solution quality on the first two instances, but the computation time of LO is 1842.07 s and 206.86, while they are both less than one second by LBBD. The average computation times of the first six instances obtained by LO and LBBD are 2667.34 s, which is about 248 times that of LBBD (10.75 s).

Table 11 Out-of-sample analysis with normal distribution of S-OASD-T

n	m	New model				LBBD method			
		Obj	$PT85$	$PT99$	Time (s)	Obj	$PT85$	$PT99$	Time (s)
10	5	215.53	216.98	220.26	1885.94	213.95	215.90	220.00	1.30
	10	173.62	176.00	181.74	2335.17	173.62	176.00	181.74	1.92
20	5	5129.22	5161.96	5248.88	4445.66	472.91	477.36	486.38	11.07
	10	488.94	493.90	505.38	2987.39	331.59	333.78	337.88	2.14
30	5	2716.51	2742.92	2783.10	4362.81	656.35	662.14	673.62	419.92
	10	1299.15	1312.24	1334.38	4396.43	491.58	491.58	491.58	6.29
40	5	-	-	-	-	732.55	736.26	743.64	137.35
	10	-	-	-	-	659.13	660.26	664.36	27.22
50	5	-	-	-	-	1045.72	1048.30	1057.32	313.55
	10	-	-	-	-	776.26	778.84	783.76	9.94
Average		-	-	-	-	555.37	558.04	564.03	93.07

The out-of-sample analysis with normal distribution is conducted in which the processing time and the distribution time are generated following the method of Section 5.7. The computational results are shown in Table 11.

In the table, LO solves six of the ten instances, while LBBD obtains feasible solutions for all the instances. All the instances are solved by LBBD within 420 seconds and the average computation time is 93.07 s. This is a significant advantage over LO as the computational times of LO on the first six instances range from 1885.94 s to 4396.43 s, indicating that LO loses its capability with the increase of instance size. Concerning the solution quality, the performance of LO is competitive with LBBD on instances with ten orders but decreases quickly when the instances get larger. For example, the objective value of LO on instances with $n = 20, m = 5$ is 5129.22, while it is only 472.91 by LBBD. LO fails to solve instances with 40 and 50 orders but LBBD provides solutions for all instances within reasonable computation time.

5.9. Practical implications

In this section, we draw some managerial implications. We address the critical problem encountered in coordinating different sectors to provide unified resource utilization of emergency resources under uncertainties, achieving the assignment of orders to manufacturers, and the timely delivery of orders. We abstract the problem to be the order assignment and scheduling problem with direct distribution under uncertainty. We consider two variants of the problem with different objective functions. We formulate the problems as scenario-based MILP models adapted from existing literature. Particularly, the first one minimizes the total cost and makespan, for which we analyze the properties of the problem and then propose an efficient MILP model that significantly outperforms the adapted model. We further develop an accurate method to find optimal or near-optimal solutions in reasonable computation time for practical-sized instances. An optimal solution to the studied S-OASD problem would help decision-makers integrate and utilize resources, improve operations efficiency, and respond quickly to the requested orders. Our successful implementation of

an S-OASD solution benefits all participants in the system. It proves that the proposed decision framework, models, and solution method can efficiently handle the order assignment and scheduling problems under uncertain circumstances such as natural disasters or emergency accidents.

First, our efficient models and algorithms ensure leveraging the performance of the platform. According to our solution, such information as 1) the allocation of orders to manufacturers; 2) the expected completion time of each order; and 3) the expected delivery time of each order would be generated quickly and then transferred to the corresponding manufacturers, third-party logistics providers, and public or private customers so that participants in the system can work coordinately and efficiently. For manufacturers, after receiving the allocated orders, they could start production in the light of the given schedules; for the third-party logistics providers, knowing the expected completion time of each order from manufacturers, they can arrange distribution capacities to perform timely order distribution; for the customers, they would get the expected delivery date of the requested orders, which helps them make further work plan. All stakeholders have lots of visibility and benefit from the transparency of our method.

Second, our solution to the S-OASD problem provides the flexibility to decision-makers to balance the trade-off between cost and timeliness. By varying the weights allocation between cost and makespan or total tardiness components in the objective function, decision-makers can easily control the desired solutions based on their preferences. For example, decision-makers could allocate a greater weight to the makespan component to ensure timely order delivery for urgent orders. In some other situations, the customers face some emergencies and want to receive the products before the due date, making the total tardiness minimization critical in the objective function. Once the emergencies are solved, decision-makers could increase the weight of the cost component to alleviate budget pressures.

Third, the proposed models and algorithm can be easily integrated into the intelligent support framework of some decision-making centers, so that powerful decision support tools can be established to help decision-makers make production and scheduling plans. In particular, our proposed model and method can provide reliable solutions through frequent rescheduling. It is of practical importance when dealing with uncertain tasks since manufacturers may face disruption risks. In the case that some manufacturers are disrupted by emergency events, decision-makers can reschedule the order production by resolving the S-OASD problem such that a feasible solution to the S-OASD problem can be quickly obtained.

Last but not least, our study theoretically contributes to the literature in the following aspects. We first introduce a new order assignment and scheduling problem with the direct distribution of orders. This is the first work integrating order assignment, scheduling, and direct distribution. We propose scenario-based MILP models for the S-OASD problem considering two different objective

functions. To minimize the weighted sum of total cost and makespan, we propose a new model that explores the property of the S-OASD problem, i.e., the optimal schedule of orders at each manufacturer complies with the LDT rule. This property helps derive a much lighter model with significantly fewer binary variables and big-M constraints. The model is thus easier to be solved by commercial solvers. This model may be applied to other similar parallel machine scheduling problems with makespan minimization, in which the optimal schedule at each machine comply with some rules. We further propose an exact LBBD method for each of them, which fills the research gap given that most studies on the OAS problem focus on deterministic environments and developing heuristics.

6. Conclusions

In this study, we investigated the order assignment and scheduling problem with direct distribution under uncertainty. The problem is of practical significance, both theoretically and practically. An optimal solution to the S-OASD problem efficiently assigns orders to manufacturers and sequences orders at each manufacturer. Implementing such a solution aims to achieve clear assignments of the orders and manufacturers, a cost-effective order production and distribution plan, and timely delivery of orders. The objective function was to minimize the weighted sum of total cost and makespan. We presented a two-stage scenario-based MILP model adapted from the literature and then proposed a novel MILP model with a predetermined order processing sequence. The new model does not require binary sequencing variables and has significantly fewer big-M constraints since the processing sequence of orders at each manufacturer was fixed according to the LDT rule. Due to the NP-hardness of the studied problem, we further proposed a logic-based Benders decomposition method. The LBBD method decomposed the original problem into an assignment master problem and a sequencing subproblem. Then the MP and SP were solved in an iterative manner where Benders cuts were generated and added to the MP. We introduced a basic optimality cut, which was then improved by finding the minimal infeasible subsets and deriving analytical information.

Extensive computational experiments are conducted on the S-OASD problem. Numerical results on 100 random instances with up to 300 scenarios, 100 orders, and 20 manufacturers indicate that the proposed new model and LBBD method are superior to the basic LO model. The new model can provide feasible solutions to all the instances and solve some small instances to optimality. However, it loses efficiency in solving large-sized instances. The LBBD manages to provide optimal or near-optimal solutions for all instances in reasonable computation time. In particular, it significantly outperforms the new model. The sensitivity analysis shows that the larger the weight of the total cost is, the easier the problem is to solve. We also conducted the out-of-analysis on instances with

1000 scenarios and normal-distributed parameters, respectively. We then extend the S-OASD into an S-OASD-T that minimizes the weighted sum of total cost and total tardiness. Considering the complexity of the S-OASD-T, the numerical experiments are conducted on instances with up to 100 scenarios, 100 orders, and ten manufacturers. The LO model performs poorly in most instances and fails to solve some large-sized ones. The LBBD outperforms the LO significantly in both solution quality and computational efficiency. We presented managerial implications based on the obtained numerical results and indicated the potential benefits of implementing our solution.

Operations research-based decision support tools play crucial roles in production scheduling and logistics, especially under uncertain circumstances. This paper aims to develop an OR-based decision support tool for the order assignment and scheduling practice considering uncertainty. Further studies may include multi-level manufacturers. Manufacturers with different levels may have various production capabilities and resilience in uncertain environments. It is of practical significance to consider uncertainty in order assignment and scheduling planning for a reliable supply of products.

Acknowledgments

This work was partly supported by grants 2021-04037 and 2019-00094 from the Canadian Natural Sciences and Engineering Research Council (NSERC). These supports are gratefully acknowledged. We thank an associate editor and three anonymous referees for their valuable suggestions on an earlier version of this paper.

References

K. R. Baker and D. Trietsch. *Principles of sequencing and scheduling*. John Wiley & Sons, 2013.

G. Baloch and F. Gzara. Strategic network design for parcel delivery with drones under competition. *Transportation Science*, forthcoming, 2020.

J. Behnamian. Multi-cut Benders decomposition approach to collaborative scheduling. *International Journal of Computer Integrated Manufacturing*, 28(11):1167–1177, 2015.

J. Behnamian. Heterogeneous networked cooperative scheduling with anarchic particle swarm optimization. *IEEE Transactions on Engineering Management*, 64(2):166–178, 2017.

J. Behnamian and S. F. Ghomi. The heterogeneous multi-factory production network scheduling with adaptive communication policy and parallel machine. *Information Sciences*, 219:181–196, 2013.

J. Behnamian and S. F. Ghomi. A survey of multi-factory scheduling. *Journal of Intelligent Manufacturing*, 27(1):231–249, 2016.

M. L. Bentaha, O. Battaïa, and A. Dolgui. A sample average approximation method for disassembly line balancing problem under uncertainty. *Computers & Operations Research*, 51:111–122, 2014.

Z.-L. Chen and G. Pundoor. Order assignment and scheduling in a supply chain. *Operations Research*, 54 (3):555–572, 2006.

T. E. Cheng, C. Ng, J. Yuan, and Z. Liu. Single machine scheduling to minimize total weighted tardiness. *European Journal of Operational Research*, 165(2):423–443, 2005.

C.-Y. C. Ching-Fang Liaw, Yang-Kuei Lin and M. Chen. Scheduling unrelated parallel machines to minimize total weighted tardiness. *Computers & Operations Research*, 30(12):1777–1789, 2003.

T.-P. Chung, C.-J. Liao, and C.-H. Lin. Minimizing makespan on parallel machines with batch arrivals. *Engineering Optimization*, 44(4):467–476, 2012.

R. Cordone and P. Hosteins. A bi-objective model for the single-machine scheduling problem with rejection cost and total tardiness minimization. *Computers & Operations Research*, 102:130–140, 2019.

J.-F. Côté, M. Dell'Amico, and M. Iori. Combinatorial Benders' cuts for the strip packing problem. *Operations Research*, 62(3):643–661, 2014.

J.-F. Côté, M. Haouari, and M. Iori. Combinatorial Benders decomposition for the two-dimensional bin packing problem. *INFORMS Journal on Computing*, forthcoming, 2021.

O. Elci and J. Hooker. Stochastic planning and scheduling with logic-based Benders decomposition. *arXiv preprint arXiv:2012.14074*, 2020.

M. Fattahi and K. Govindan. Data-driven rolling horizon approach for dynamic design of supply chain distribution networks under disruption and demand uncertainty. *Decision Sciences*, forthcoming, 2020.

M. M. Fazel-Zarandi and J. C. Beck. Using logic-based Benders decomposition to solve the capacity-and distance-constrained plant location problem. *INFORMS Journal on Computing*, 24(3):387–398, 2012.

P. M. França, A. Mendes, and P. Moscato. A memetic algorithm for the total tardiness single machine scheduling problem. *European Journal of Operational Research*, 132(1):224–242, 2001.

C. Guo, M. Bodur, D. M. Aleman, and D. R. Urbach. Logic-based benders decomposition and binary decision diagram based approaches for stochastic distributed operating room scheduling. *INFORMS Journal on Computing*, 33(4):1551–1569, 2021.

Z. Guo, W. K. Wong, Z. Li, and P. Ren. Modeling and pareto optimization of multi-objective order scheduling problems in production planning. *Computers & Industrial Engineering*, 64(4):972–986, 2013.

B. Han, W. Zhang, X. Lu, and Y. Lin. On-line supply chain scheduling for single-machine and parallel-machine configurations with a single customer: Minimizing the makespan and delivery cost. *European Journal of Operational Research*, 244(3):704–714, 2015.

Z. He, Z. Guo, and J. Wang. Integrated scheduling of production and distribution operations in a global mto supply chain. *Enterprise Information Systems*, 13(4):490–514, 2019.

A. Heching, J. N. Hooker, and R. Kimura. A logic-based Benders approach to home healthcare delivery. *Transportation Science*, 53(2):510–522, 2019.

J. Hooker. Logic based methods for optimization: Combining optimization and constraint satisfaction. John Willey and Sons. *Inc. New York*, 2000.

J. N. Hooker. *Integrated Methods for Optimization*, volume 100. Springer Science & Business Media, Heidelberg, 2007a.

J. N. Hooker. Planning and scheduling by logic-based Benders decomposition. *Operations Research*, 55(3): 588–602, 2007b.

J. N. Hooker and G. Ottosson. Logic-based Benders decomposition. *Mathematical Programming*, 96(1): 33–60, 2003.

N. Karimi and H. Davoudpour. A knowledge-based approach for multi-factory production systems. *Computers & Operations Research*, 77:72–85, 2017.

L.-P. Kerkhove and M. Vanhoucke. Scheduling of unrelated parallel machines with limited server availability on multiple production locations: a case study in knitted fabrics. *International Journal of Production Research*, 52(9):2630–2653, 2014.

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

G. Kovacs and M. Moshtari. A roadmap for higher research quality in humanitarian operations: A methodological perspective. *European Journal of Operational Research*, 276(2):395–408, 2019.

E. L. Lawler. A "pseudopolynomial" algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics*, 1(08):331–342, 1977.

J. K. Lenstra, A. R. Kan, and P. Brucker. Complexity of machine scheduling problems. In *Annals of Discrete Mathematics*, volume 1, pages 343–362. Elsevier, 1977.

C.-L. Li and J. Ou. Coordinated scheduling of customer orders with decentralized machine locations. *IIE Transactions*, 39(9):899–909, 2007.

K. Li, X. Zhang, J. Y. Leung, and B.-y. Cheng. Integrated production and delivery with multiple factories and multiple customers. *International Journal of Systems Science: Operations & Logistics*, 4(3):219–228, 2017.

Y. Li, J.-F. Côté, L. Callegari-Coelho, and P. Wu. Novel formulations and logic-based benders decomposition for the integrated parallel machine scheduling and location problem. *INFORMS Journal on Computing*, 2021.

X. Liu, F. Chu, F. Zheng, C. Chu, and M. Liu. Parallel machine scheduling with stochastic release times and processing times. *International Journal of Production Research*, pages 1–20, 2020.

J. Lohmer and R. Lasch. Production planning and scheduling in multi-factory production networks: a systematic literature review. *International Journal of Production Research*, pages 1–27, 2020.

C. Mancilla and R. Storer. A sample average approximation approach to stochastic appointment sequencing and scheduling. *IIE Transactions*, 44(8):655–670, 2012.

F. Marandi and S. Fatemi Ghomi. Integrated multi-factory production and distribution scheduling applying vehicle routing approach. *International Journal of Production Research*, 57(3):722–748, 2019.

B. Naderi and V. Roshanaei. Branch-relax-and-check: A tractable decomposition method for order acceptance and identical parallel machine scheduling. *European Journal of Operational Research*, 286(3):811–827, 2020.

M. Pinedo and E. Rammouz. A note on stochastic scheduling on a single machine subject to breakdown and repair. *Probability in the Engineering & Informational Sciences*, 2(01):41–49, 1988.

V. Roshanaei, C. Luong, D. M. Aleman, and D. Urbach. Propagating logic-based Benders' decomposition approaches for distributed operating room scheduling. *European Journal of Operational Research*, 257 (2):439–455, 2017.

V. Roshanaei, C. Luong, D. M. Aleman, and D. R. Urbach. Reformulation, linearization, and decomposition techniques for balanced distributed operating room scheduling. *Omega*, 93:102043, 2020.

H. Şen and K. Bülbül. A strong preemptive relaxation for weighted tardiness and earliness/tardiness problems on unrelated parallel machines. *INFORMS Journal on Computing*, 27(1):135–150, 2015.

A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on stochastic programming: modeling and theory*. SIAM, Philadelphia, 2014.

S. O. Shim and Y. D. Kim. Scheduling on parallel identical machines to minimize total tardiness. *European Journal of Operational Research*, 177(1):135–146, 2007.

X. Sun, S. Chung, and F. T. Chan. Integrated scheduling of a multi-product multi-factory manufacturing system with maritime transport limits. *Transportation Research Part E: Logistics and Transportation Review*, 79:110–127, 2015.

S. Tanaka and M. Araki. An exact algorithm for the single-machine total weighted tardiness problem with sequence-dependent setup times. *Computers & Operations Research*, 40(1):344–352, 2013.

S. Tanaka and S. Fujikuma. A dynamic-programming-based exact algorithm for general single-machine scheduling with machine idle time. *Journal of Scheduling*, 15(3):347–361, 2012.

S. Tanaka and S. Sato. An exact algorithm for the precedence-constrained single-machine scheduling problem. *European Journal of Operational Research*, 229(2):345–352, 2013.

T. T. Tran, A. Araujo, and J. C. Beck. Decomposition methods for the parallel machine scheduling problem with setups. *INFORMS Journal on Computing*, 28(1):83–95, 2016.

N. Vakhania. Single-machine scheduling with release times and tails. *Annals of Operations Research*, 129 (1-4):253–271, 2004.

M. Yazdani, S. Gohari, and B. Naderi. Multi-factory parallel machine problems: improved mathematical models and artificial bee colony algorithm. *Computers & Industrial Engineering*, 81:36–45, 2015.