Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 7209303, 18 pages
https://doi.org/10.1155/2017/7209303

Research Article

Fix-and-Optimize and Variable Neighborhood Search Approaches for Stochastic Multi-Item Capacitated Lot-Sizing Problems

Liuxi Li,[1] Shiji Song,[1] Cheng Wu,[1] and Rui Wang[1,2]

[1]*Department of Automation, Tsinghua University, Beijing 100084, China*
[2]*Department of Basic Science, Military Transportation University, Tianjin 300161, China*

Correspondence should be addressed to Shiji Song; shijis@mail.tsinghua.edu.cn

Received 19 October 2016; Revised 18 January 2017; Accepted 1 March 2017; Published 12 April 2017

Academic Editor: Honglei Xu

We discuss stochastic multi-item capacitated lot-sizing problems with and without setup carryovers (also known as link lot size), S-MICLSP and S-MICLSP-L. The two models are motivated from a real-world steel enterprise. To overcome the nonlinearity of the models, a piecewise linear approximation method is proposed. We develop a new fix-and-optimize (FO) approach to solve the approximated models. Compared with the existing FO approach(es), our FO is based on the concept of "k-degree-connection" for decomposing the problems. Furthermore, we also propose an integrative approach combining our FO and variable neighborhood search (FO-VNS), which can improve the solution quality of our FO approach by diversifying the search space. Numerical experiments are performed on the instances following the nature of realistic steel products. Our approximation method is shown to be efficient. The results also show that the proposed FO and FO-VNS approaches significantly outperform the recent FO approaches, and the FO-VNS approaches can be more outstanding on the solution quality with moderate computational effort.

1. Introduction

The stochastic multi-item capacitated lot-sizing problem (S-MICLSP) and its setup carryover extension (also known as linked lot size extension, in [1], abbreviated to "-L"), S-MICLSP-L, are designed to map an industrial optimization problem in a realistic steel enterprise. The problem setting is as follows: there are several types of steel products. These products differ in various attributes: chemical composition (mixture), width, thickness, shape (bar, rod, tube, pipe, plate, sheet, etc.), microstructure (ferritic, pearlitic, martensitic, etc.), physical strength, and other attributes. Usually, each type of steel products should be used for only one particular purpose, and each purpose can be satisfied by one or several steel products. Hence, in this steel enterprise, a typical production schedule is made based on the need of one particular purpose, rather than the need of one particular customer. There are two categories of production scheduling, solid scheduling and flexible scheduling, applied in different factories in this steel enterprise. The solid one schedules all types of products simultaneously with a high frequency of restarting production states, while the flexible one schedules parts of the types simultaneously with a low frequency of restarting production states. All of the production schedules are made before the whole planning horizon. The problems with these settings can be suitably mapped to S-MICLSP and S-MICLSP-L.

Both S-MICLSP and S-MICLSP-L are stochastic generalizations of the capacitated lot-sizing problems (CLSP, see [2]) and they consider backlogging and setup carryovers jointly. The deterministic CLSPs with backlogging or setup carryovers individually have been tackled by various models in the literature. We refer the interested readers to [3, 4] for the most recent review on CLSPs. For the lot-sizing problems considering backlogging and setup carryovers jointly, models were treated in [5–7]. Their problem formulations were similar to [8], who first solved problems with setup carryovers. All of the above studies focused on how to

solve the lot-sizing problems by designing heuristics. As the authors highlighted, although there is a significant amount of research literature on CLSPs, the literature on problems that consider backlogging and setup carryover jointly is rather scarce.

In this paper, we assume that demand is continuously stochastic which can cover majority of demand environment. Due to the model uncertainty, approximation methods are applied to reformulate the lot-sizing models for performing deterministic mixed integer programming (MIP) in the literature. Haugen et al. [9] generated subproblems for each scenario solved heuristically to capture the nature of demand uncertainty and specify a reasonable number of representative scenarios. Brandimarte [10] modeled the demand uncertainty through generating scenario trees. They made the generated scenario trees match the first, second, third, and fourth moments of the given distribution. The scenario method or scenario-generated method can also be found in [11–14]. Almost all approximation methods for lot-sizing problems are scenario methods. Nevertheless, Mietzner and Reger [15] stated the advantages and disadvantages of scenario methods. One of the crucial disadvantages is as follows: to capture more properties of the uncertainty, the approximated models should ensure an adequate number of scenarios, but the practice of scenario methods can be very time-consuming. This leads to the contradiction between computational time and approximation accuracy. In the following part, we will propose our approximation method to overcome this drawback.

Since the approximated models can perform deterministic MIP, the methods used for deterministic CLSP and its extensions can be also applied to the approximated models. Historically, exact methods (branch & bound technique, Lagrangian relaxation, cut-generation technique, etc.) and metaheuristics (genetic algorithm, particle swarm optimization, tabu search, etc.) are adopted in the deterministic lot-sizing models. We refer interested readers to [16] for further review. Recently, MIP-based heuristics are developed to solve lot-sizing models since they combine the advantages of exact methods and (meta-)heuristics. An MIP-based heuristic shown to be outstanding is called fix-and-optimize (FO) approach, which is proposed by Sahling et al. [17]. The authors presented three types of decomposition method: product decomposition, resource decomposition, and time periods decomposition. Based on the work of [17], variants of FO are developed by [18–20]. However, all of the variants follow the decomposition framework of [17].

Although FO exhibits its efficiency and effectiveness in the literature, it follows a prespecified trajectory and hence it is a local search method. This may result in low solution quality. To enhance the search space of FO approach, one can apply variable neighborhood search (VNS) proposed by [21]. VNS is a metaheuristic which involves two key steps. The first key step is using a local search method to obtain local optimum and the second is systematically changing the neighborhood structure of each local search. Unlike other metaheuristics, VNS does not follow a prespecified trajectory but explores increasingly distant neighborhoods of the current incumbent solution. Since VNS can enhance the search space, many integrative frameworks with VNS are proposed to solve lot-sizing problems. Hindi et al. [22] proposed an integrative Lagrangian relaxation- (LR-) VNS framework for the CLSP with setup times and got good feasible solutions. Zhao et al. [23] and Seeanner et al. [24] developed another type of VNS, the so-called variable neighborhood decomposition search (VNDS) to solve multilevel lot-sizing problems, and provided promising computational results. All of the above studies throw light upon solving lot-sizing problems by combining VNS.

Newly, Chen [25] proposed an excellent integrative framework combining FO and VNS for deterministic lot-sizing problems. Since our models have *"many-to-one" demand structure*, his framework cannot be applied to our models. However, motivated from his work, we propose our FO and integrative FO-VNS for our stochastic lot-sizing problems. Compared with the work of [25], our proposed FO allows capacity-infeasible (overtime cost is not zero) solutions and can be applied to *"many-to-one" demand structure*, while he prohibited capacity-infeasible solutions and his framework was only valid for one-to-one demand structure. Thus, we apply the integrative framework to models without setup carryovers, S-MICLSP, and successfully extend it to our setup carryovers version, S-MICLSP-L, while Chen [25] only applied his framework to models without setup carryovers.

In this paper, we follow a similar analytical procedure of solving stochastic lot-sizing problems to the reviewed literature. However, despite the above, our paper demonstrates other unique characteristics which distinguish from the existing related literature as follows:

(1) Derived from realistic industrial problems, we formulate S-MICLSP and S-MICLSP-L models considering backlogging, production overtime, and initial inventory at the same time, which is much more complicated than the existing models in the literature.

(2) We propose a piecewise linear method to approximate S-MICLSP and S-MICLSP-L models. This method is simple and easy-to-implement, providing a good trade-off between computational time and approximation accuracy. This method overcomes the drawback of scenarios generating on the computational end.

(3) A new FO approach is proposed for our approximated models. Differing from the decomposition framework of [17], this approach decomposes the main problem based on the combined information of products, resources, demands, and time periods.

(4) An integrative VNS heuristic which uses FO as the local search engine is proposed to solve our approximated models. This combined approach is running on a specially designed neighborhood structure.

The outline of this paper is structured as follows: we formulate our S-MICLSP and S-MICLSP-L models and propose our piecewise linear approximation method in Section 2. Our proposed FO approach and combined method (FO-VNS) are described in Sections 3 and 4. Numerical experiments of

the two approaches on instances generated from a realistic case are presented in Section 5. In Section 6, the concluding remarks as well as discussions on future research are provided. The generating method is lengthy and is relegated to Appendix.

2. Models Formulation and Approximation

In this section, we first formulate S-MICLSP and S-MICLSP-L models. To overcome the nonlinearity and intractability of the models, we then propose a piecewise linear approximation method to reformulate the models. These approximated models are deterministic and hence can be tractably solved by our following proposed algorithms.

2.1. Model Description. In our models, demands have no *one-to-one correspondence* to products. Demands can be satisfied by multiple products and categorized into different classes by the purposes. We can use the term "demand class" to describe one demand for purpose. The term "demand class" can help readers recognize the unique structure of demands in our models. But to avoid ambiguity, we equate the term "demand" to the term "demand class" and use "demand" mostly in the context. For detailed description, we make additional assumptions as follows:

(i) General capacitated lot-sizing problems assumptions:

  (a) lot-sizing for multiple products
  (b) finite time of planning horizon
  (c) initial inventories
  (d) capacitated production resource
  (e) decision before planning horizon

(ii) Demands assumptions:

  (a) continuously randomized on a known distribution with a finite support, independent, and identically distributed from period to period for each demand (class)
  (b) *many-to-one structure*: each product can only satisfy one demand (class), while each demand (class) can be satisfied by multiple products

(iii) Big-bucket assumption (see [26]):

  (a) permit the production of multiple products during a single period

(iv) Linked lot sizes assumption (see [1]):

  (a) the setup state of a resource to be carried over from the current period to the next period

(v) Other assumptions:

  (a) overtime production and backlogging setting are allowed, with high penalty costs.
  (b) no lead times
  (c) expected cost minimization objective
  (d) continuous variables for lot sizes

Note that the overtime production is allowed since the requirement of flexibility. This assumption is often used in practice if no feasible production plan could be found otherwise the following two facts: one is the production capacity limits are frequently "soft" as machines could run longer than the planned daily operating time, the other one is the total volume of production could be increased slightly if machines could run below their technical limits by default.

Using the symbols given in Notations, the S-MICLSP can be formulated as given below.

$$\min \quad \sum_{t\in\mathcal{T}}\left(\sum_{i\in\mathcal{P}}\left(f_i X_{it} + p_i Q_{it} + h_i E\left[I_{it}\right]\right) + \sum_{j\in\mathcal{D}} b_j E\left[B_{jt}\right] + \sum_{r\in\mathcal{R}} \mathrm{oc}_r O_{rt}\right) \tag{1}$$

subject to constraints

$$I_{it} = I_{i,t-1} + Q_{it} - \sum_{j\in\mathcal{D}_i} S_{ijt}, \quad i\in\mathcal{P},\ t = 1,\ldots,T, \tag{2}$$

$$B_{jt} = \left[\sum_{i\in\mathcal{P}_j}\frac{S_{ijt}}{e_{ij}} - D_{jt} - B_{j,t-1}\right]^{-}, \tag{3}$$

$$B_{j0} = 0,\ j\in\mathcal{D},\ t = 1,\ldots,T,$$

$$\sum_{j\in\mathcal{D}_i} S_{ijt} \leqslant Q_{it} + I_{i,t-1}, \quad i\in\mathcal{P},\ t = 1,\ldots,T, \tag{4}$$

$$\sum_{i\in\mathcal{P}_r}\left(c_i Q_{it} + \mathrm{st}_i X_{it}\right) \leqslant k_r + O_{rt}, \tag{5}$$

$$r\in\mathcal{R},\ t = 1,\ldots,T,$$

$$c_i Q_{it} \leqslant m_i X_{it}, \quad i\in\mathcal{P},\ t = 1,\ldots,T, \tag{6}$$

$$S_{ijt}, Q_{it}, O_{rt} \geqslant 0, \tag{7}$$

$$i\in\mathcal{P},\ j\in\mathcal{D}_i,\ r\in\mathcal{R},\ t = 1,\ldots,T,$$

$$X_{it}\in\{0,1\}, \quad i\in\mathcal{P},\ t = 1,\ldots,T. \tag{8}$$

The objective function (1) to be minimized is the sum of setup costs, production costs, inventory holding costs, backlogging

penalty costs, and overtime costs. Constraints (2) and (3) are the inventory-balanced equations that each demand (class) can be satisfied by multiple products. Additionally, constraints (4) imply that the quantity of one product used to fulfill one demand (class) should not exceed the sum of inventory and production quantity. Constraints (5) give the capacity constraint of each resource in each period with overtime. Constraints (6) are the coupling constraints linking each production variable Q_{it} with its corresponding setup variables X_{it}, where the choice of each large positive number m_i must not limit any feasible production quantity of product i in period t. The coupling constraints imply that $Q_{it} = 0$ if $X_{it} = 0$ for all i and t. The nonnegative real or binary nature of each variable in the model is indicated by constraints (7) and (8).

The S-MICLSP-L allows the setup state of each resource to be carried over from the current period to the next period. To formulate the S-MICLSP-L, additional binary variables indicating setup carryovers and additional constraints linking the setup state variables with the setup carryover variables are required. We adopt the formulation of [27] with overtime. Additional variables can also be found in Notations. The S-MICLSP-L can be formulated as given below.

$$\min \quad \sum_{t\in\mathcal{T}} \left(\sum_{i\in\mathcal{P}} \left(f_i \left[X_{it} - Z_{it} \right] + p_i Q_{it} + h_i E\left[I_{it} \right] \right) + \sum_{j\in\mathcal{D}} b_j E\left[B_{jt} \right] + \sum_{r\in\mathcal{R}} \mathrm{oc}_r O_{rt} \right) \tag{9}$$

subject to constraints (2)~(4), (6)~(8), and constraints

$$\sum_{i\in\mathcal{P}_r} \left(c_i Q_{it} + st_i \left[X_{it} - Z_{it} \right] \right) \leqslant k_r + O_{rt}, \quad r \in \mathcal{R},\ t = 1, \ldots, T, \tag{10}$$

$$\sum_{i\in\mathcal{P}_r} Z_{it} \leqslant 1, \quad r \in \mathcal{R},\ t = 1, \ldots, T, \tag{11}$$

$$Z_{it} \leqslant X_{it}, \qquad Z_{it} \leqslant X_{i,t-1}, \quad i \in \mathcal{P},\ t = 1, \ldots, T, \tag{12}$$

$$\left| \mathcal{P}_r \right| \left(2 - Z_{it} - Z_{i,t+1} \right) + 1 \geqslant \sum_{i'\in\mathcal{P}_r} X_{i't}, \quad i \in \mathcal{P}_r,\ t = 1, \ldots, T, \tag{13}$$

$$Z_{it} \in \{0, 1\}, \qquad Z_{i1} = 0, \quad i \in \mathcal{P},\ t = 1, \ldots, T. \tag{14}$$

Constraints (10) are similar to constraints (5). Constraints (11) imply that, in each period, the setup carryover of a resource is possible only for at most one product. Constraints (12) indicate that the setup carryover of a resource for product i occurs in period t only if the resource is set up for the item in both periods $t-1$ and t. Constraints (13) indicate multiperiod setup carryovers. Constraints (14) specify the binary nature of setup carryover variables.

2.2. Piecewise Linear Approximation. Both $E[I_{it}]$ and $E[B_{jt}]$ in S-MICLSP and S-MICLSP-L models are nonlinear stochastic functions. We have specified the continuous nature of demand uncertainty. Thus it is intractable to solve these models. We could apply scenario method to approximate the models. However, we have discussed in Section 1 that the computational efforts can be unacceptable and the precision of approximation can be low. Fortunately, it is possible to replace the functions of $E[I_{it}]$ and $E[B_{jt}]$ by suitably chosen piecewise linear functions. The functions of $E[I_{it}]$ and $E[B_{jt}]$ can be approximated as follows. Let q_{jt} denote the total amount available to fill the cumulated demand j from period 1 to period t (cumulated quantity produced up to period t plus initial inventory in period 1). Let y_{jt} denote the cumulated demand from period 1 up to period t and let $f_{y_{jt}}$ denote the associated density function. Denote $E[I_{jt}]_{(q_{jt})}$ the expected physical inventory on hand at the end of period t for demand j corresponding to q_{jt}. Then consequently $E[I_{jt}]_{(q_{jt})}$ is equal to

$$\begin{aligned} E\left[I_{jt} \right]_{(q_{jt})} &= \int_0^{q_{jt}} \left(q_{jt} - y_{jt} \right) \cdot f_{y_{jt}}\left(y_{jt} \right) \mathrm{d}y_{jt} \\ &= q_{jt} - E\left[y_{jt} \right] + \int_{q_{jt}}^{\infty} \left(y_{jt} - q_{jt} \right) \cdot f_{y_{jt}}\left(y_{jt} \right) \mathrm{d}y_{jt} \\ &= q_{jt} - E\left[y_{jt} \right] + P_{y_{jt}}\left(q_{jt} \right), \end{aligned} \tag{15}$$

where $P_{y_{jt}}(q_{jt})$ is the well-known expected loss function or the failure function of the random variable y_{jt} with respect to the quantity q_{jt}. Figure 1 illustrates the function $P_{y_{jt}}(q_{jt})$.

A backlog of demand j occurs at the end of period t, if the cumulated demand up to period t, y_{jt}, is greater than the cumulated production quantity up to period t, q_{jt}. Hence the expected backlog of demand j at the end of period t, denoted by $E[I_{jt}^{\mathrm{end}}]_{(q_{jt})}$, is

$$\begin{aligned} E\left[I_{jt}^{\mathrm{end}} \right]_{(q_{jt})} &= \int_{q_{jt}}^{\infty} \left(y_{jt} - q_{jt} \right) \cdot f_{y_{jt}}\left(y_{jt} \right) \mathrm{d}y_{jt} \\ &= P_{y_{jt}}\left(q_{jt} \right). \end{aligned} \tag{16}$$

Consider the backlog just after production but before demand occurrence in period t. This backlog cannot be affected by

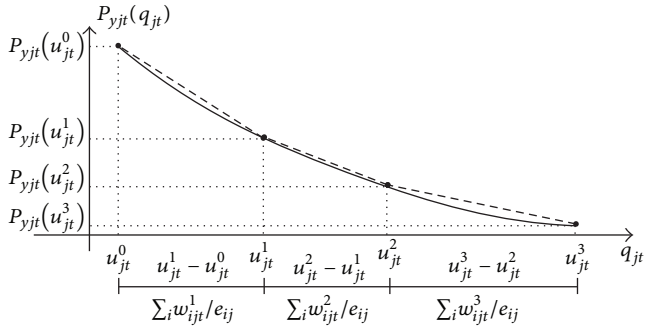

Figure 1: Illustration of piecewise linear approximation for $P_{y_{jt}}(q_{jt})$ with segments number $L = 3$.

demand since demand does not occur. Hence the expected backlog just after production but before demand occurrence of demand j corresponding to q_{jt} in period t, denoted by $E[I^{\text{prod}}_{jt}]_{(q_{jt})}$, is

$$E\left[I^{\text{prod}}_{jt}\right]_{(q_{jt})} = \int_{q_{jt}}^{\infty} \left(y_{j,t-1} - q_{jt}\right) \cdot f_{y_{j,t-1}}\left(y_{j,t-1}\right) \mathrm{d}y_{j,t-1} = P_{y_{j,t-1}}\left(q_{jt}\right). \tag{17}$$

The expected backlog number of demand j in period t can be expressed as the difference between the backlog at the end of period t and the expected backlog just after production but before demand occurrence of period t.

$$E\left[B_{jt}\right]_{(q_{jt})} = E\left[I^{\text{end}}_{jt}\right] - E\left[I^{\text{prod}}_{jt}\right] = P_{y_{jt}}\left(q_{jt}\right) - P_{y_{j,t-1}}\left(q_{jt}\right). \tag{18}$$

Define L line segments with interval limits u^l_{jt} for demand j that mark the cumulated production up to period t. Let u^0_{jt} be the lower limit of the relevant region for demand j. Accordingly, the slope of the inventory on hand function for the line segment l is

$$a^l_{I_{jt}} = \frac{\left[E\left[I_{jt}\right]_{(u^l_{jt})} - E\left[I_{jt}\right]_{(u^{l-1}_{jt})}\right]}{\left[u^l_{jt} - u^{l-1}_{jt}\right]} = \frac{\left[u^l_{jt} - E\left[y_{jt}\right] + P_{y_{jt}}\left(u^l_{jt}\right)\right] - \left[u^{l-1}_{jt} - E\left[y_{jt}\right] + P_{y_{jt}}\left(u^{l-1}_{jt}\right)\right]}{u^l_{jt} - u^{l-1}_{jt}} = \frac{\left[u^l_{jt} + P_{y_{jt}}\left(u^l_{jt}\right)\right] - \left[u^{l-1}_{jt} + P_{y_{jt}}\left(u^{l-1}_{jt}\right)\right]}{u^l_{jt} - u^{l-1}_{jt}}. \tag{19}$$

Similar to the above calculation, the nonlinear function of backlogging in period t can be approximated, whereby the slope can be calculated as

$$a^l_{B_{jt}} = \frac{\left[P_{y_{jt}}\left(u^l_{jt}\right) - P_{y_{j,t-1}}\left(u^l_{jt}\right)\right] - \left[P_{y_{jt}}\left(u^{l-1}_{jt}\right) - P_{y_{j,t-1}}\left(u^{l-1}_{jt}\right)\right]}{u^l_{jt} - u^{l-1}_{jt}}. \tag{20}$$

Both $a^l_{I_{jt}}$ and $a^l_{B_{jt}}$ are calculated from the function $P_{y_{jt}}(q_{jt})$ when L segments are defined (see Figure 1). Once slope values of expected inventories and backlogging can be found from the distribution of the random variable $y_{jt}, j \in \mathscr{D}, t \in \mathscr{T}$, we can approximate the original model in the following. Let w^l_{ijt} be the production quantity of product i for demand j in period t associated with interval l. As $P_{y_{jt}}(x)$ is convex, $\sum_{i\in\mathscr{P}_j}(w^l_{ijt}/e_{ij}) \leqslant u^l_{jt} - u^{l-1}_{jt}$ should be satisfied. Also, $\widehat{q}_{ijt} = \sum_{l=1}^{L} w^l_{ijt}$ is the cumulated production quantity of product i for demand j up to period t and $q_{ijt} = \widehat{q}_{ijt} - \widehat{q}_{ij,t-1}$ (see Figure 1). Thus constraints (2)~(4) can be rewritten. We introduce the slope values into the model. Let $a^0_{I_{jt}}$ be the expected inventory and $a^0_{B_{jt}}$ be the expected backlogging at point u^0_{jt}. In that sense, the physical inventory of product i in period t can be approximated as $E[I_{it}] \approx \sum_{j\in\mathscr{D}_i}(a^0_{I_{jt}} + \sum_{l=1}^{L} a^l_{I_{jt}} w^l_{ijt})$, and the backlog number can be approximated as $E[B_{jt}] \approx \sum_{i\in\mathscr{P}_j}(a^0_{B_{jt}} + \sum_{l=1}^{L} a^l_{B_{jt}}(w^l_{ijt}/e_{ij}))$. All the additional symbols in this section are listed in Notations. The following linear approximated S-MICLSP is obtained:

$$\begin{aligned}\min \quad & \sum_{t\in\mathscr{T}}\left(\sum_{i\in\mathscr{P}}\left(f_i X_{it} + p_i \sum_{j\in\mathscr{D}_i} q_{ijt}\right) + \sum_{i\in\mathscr{P}} h_i \cdot \sum_{j\in\mathscr{D}_i}\left[a^0_{I_{jt}} + \sum_{l=1}^{L} a^l_{I_{jt}} w^l_{ijt}\right] \right. \\ & \left. + \sum_{j\in\mathscr{D}} b_j \cdot \sum_{i\in\mathscr{P}_j}\left[a^0_{B_{jt}} + \sum_{l=1}^{L} a^l_{B_{jt}} \frac{w^l_{ijt}}{e_{ij}}\right] + \sum_{r\in\mathscr{R}} \mathrm{oc}_r O_{rt}\right)\end{aligned} \tag{21}$$

subject to constraints (8) and constraints

$$\sum_{l=1}^{L} w^l_{ij,t-1} \leqslant \sum_{l=1}^{L} w^l_{ijt}, \quad j \in \mathscr{D},\ i \in \mathscr{P}_j,\ t = 2, \ldots, T, \tag{22}$$

$$\sum_{i\in\mathscr{P}_j} \frac{w^l_{ijt}}{e_{ij}} \leqslant u^l_{jt} - u^{l-1}_{jt}, \quad j \in \mathscr{D},\ l = 1, \ldots, L,\ t = 1, \ldots, T, \tag{23}$$

$$\sum_{l=1}^{L} w_{ijt}^{l} - \sum_{l=1}^{L} w_{ij,t-1}^{l} = q_{ijt}, \quad j \in \mathscr{D},\ i \in \mathscr{P}_j,\ t = 1, \ldots, T, \tag{24}$$

$$\sum_{i \in \mathscr{P}_r} \left(c_i \sum_{j \in \mathscr{D}_i} q_{ijt} + \mathrm{st}_i X_{it} \right) \leqslant k_r + O_{rt}, \quad r \in \mathscr{R},\ t = 1, \ldots, T, \tag{25}$$

$$c_i \sum_{j \in \mathscr{D}_i} q_{ijt} \leqslant m_i X_{it}, \quad i \in \mathscr{P},\ t = 1, \ldots, T, \tag{26}$$

$$O_{rt}, w_{ijt}^{l} \geqslant 0, \quad j \in \mathscr{D},\ i \in \mathscr{P}_j,\ l = 1, \ldots, L,\ r \in \mathscr{R},\ t = 1, \ldots, T. \tag{27}$$

Constraints (22)~(24) are approximated constraints of (2)~(4) by piecewise linear approximation. Constraints (25)~(27) are similar to constraints (5)~(7).

The linear approximated S-MICLSP-L is as follows:

$$\min \sum_{t \in \mathscr{T}} \left(\sum_{i \in \mathscr{P}} \left(f_i \left[X_{it} - Z_{it} \right] + p_i \sum_{j \in \mathscr{D}_i} q_{ijt} \right) + \sum_{i \in \mathscr{P}} h_i \cdot \sum_{j \in \mathscr{D}_i} \left[a_{I_{jt}}^{0} + \sum_{l=1}^{L} a_{I_{jt}}^{l} w_{ijt}^{l} \right] + \sum_{j \in \mathscr{D}} b_j \cdot \sum_{i \in \mathscr{P}_j} \left[a_{B_{jt}}^{0} + \sum_{l=1}^{L} a_{B_{jt}}^{l} \frac{w_{ijt}^{l}}{e_{ij}} \right] + \sum_{r \in \mathscr{R}} \mathrm{oc}_r O_{rt} \right) \tag{28}$$

subject to constraints (22)~(24), (26)~(27), (8), and (11)~(14) and constraints

$$\sum_{i \in \mathscr{P}_r} \left(c_i \sum_{j \in \mathscr{D}_i} q_{ijt} + \mathrm{st}_i \left(X_{it} - Z_{it} \right) \right) \leqslant k_r + O_{rt}, \quad r \in \mathscr{R},\ t = 1, \ldots, T. \tag{29}$$

Constraints (29) are similar to constraints (25).

Models (21) and (28) are both deterministic models; hence these models can be tractably solved by our next proposed algorithms.

3. New Fix-and-Optimize (FO) Approach

In the FO approach of [17], a series of MIP subproblems is solved in each of which most of the binary setup variables are tentatively fixed to 0 or 1. Only a subset of binary variables of the original model is treated as decision variables and "optimized" by a run of an MIP solver. MIP subproblems are generated using three types of basic decompositions, product decomposition, resource decomposition, and time periods decomposition. The authors also proposed three more combined decomposition methods: (1) product decomposition first and then resource decomposition, (2) product decomposition first and then time periods decomposition, and (3) product decomposition first, then resource decomposition, and finally time periods decomposition. Figure 2 exemplifies the FO approach of product decomposition with 4 products. In this section, we will propose our FO approach, which differs from [17]. In the following, we first define the so-called "k-degree-connection" to combine the decision of resources, products, demands, and time periods. Then the subproblems of the fix-and-optimize approach can be redefined based on the concept "k-degree-connection" (as we have discussed before, our models have "many-to-one" demand structure; we need to state that, in the work of [25], a similar concept "Interrelatedness" is defined; however, his concept is for one-to-one demand structure and cannot be applied to our models). Finally, we present our FO approach for both the S-MICLSP and S-MICLSP-L models.

3.1. Definition of "k-Degree-Connection".

In S-MICLSP, the binary setup variables are closely connected to other decision variables. We can infer from constraints (26) that if the setup variable X_{it} is set to be zero, no production can be planned in this period. If the setup variable X_{it} is set to be one, the corresponding production q_{ijt} can be made. If the value of q_{ijt} changes, the value of $q_{i,j,t-1}$ and $q_{i,j,t+1}$ may also change due to constraints (24). The change of X_{it} may also cause the change of $q_{i',j,t}$, whereby $\mathscr{D}_{i'} = \mathscr{D}_i = j$, due to constraints (23). Similarly, the change of X_{it} may also cause the change of $q_{i'',j,t}$, whereby $i, i'' \in \mathscr{P}_r$, $r \in \mathscr{R}$, due to constraints (25).

First we define "1-degree-connection." Let $\widetilde{\Omega} = \{X_{it} \mid i \in \mathscr{P}, t \in \mathscr{T}\}$ denote the set of all binary setup variables. We say two setup variables $X_{it} \in \widetilde{\Omega}$ and $X_{i't'} \in \widetilde{\Omega}$ have "1-degree-connection" or X_{it} is "1-degree-connected" to $X_{i't'}$ if one of the following conditions holds:

(1) Period time t and period time t' are consecutive; that is, $i' = i$ and $t' \in \{t-1, t+1\}$;

(2) Product i and i' both satisfy demand j; that is, $\mathscr{D}_{i'} = \mathscr{D}_i = j$ and $t' = t$;

(3) Product i and i' are produced by the same resource r; that is, $i, i' \in \mathscr{P}_r$ and $t' = t$;

then we can define the set of binary setup variables that are "1-degree-connected" to X_{it}, denoted by $\mathrm{DC}(X_{it})$, as follows:

$$\mathrm{DC}\left(X_{it}\right) = \left\{X_{i,t-1}, X_{it}, X_{i,t+1}\right\} \cup \left\{X_{i't} \mid \mathscr{D}_{i'} = \mathscr{D}_i = j\right\} \cup \left\{X_{i't} \mid i, i' \in \mathscr{P}_r\right\}. \tag{30}$$

For any $X_{it} \in \mathscr{S} \subseteq \widetilde{\Omega}$, whereby $\mathscr{S}$ is a subset of all binary setup variables $\widetilde{\Omega}$, the set of binary setup variables that are "1-degree-connected" to $\mathscr{S}$, denoted by $\mathrm{DC}(\mathscr{S})$, is given by

$$\mathrm{DC}\left(\mathscr{S}\right) = \left\{\left\{X_{i,t-1}, X_{it}, X_{i,t+1}\right\} \cup \left\{X_{i't} \mid \mathscr{D}_{i'} = \mathscr{D}_i = j\right\} \cup \left\{X_{i't} \mid i, i' \in \mathscr{P}_r\right\} \mid X_{it} \in \mathscr{S}\right\}. \tag{31}$$

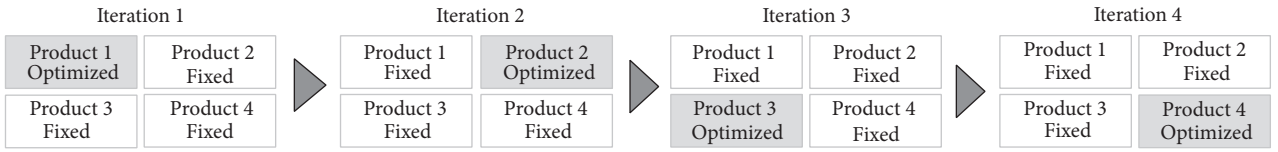

Figure 2: Example of fix-and-optimize with product decomposition.

(1) Set an initial feasible solution of the model, $X_{it} = 1$ for all i and t
(2) $iter = 0$
(3) **repeat**
(4) $iter = iter + 1$
(5) Choose a pair (i, t) from $(\mathcal{N} \times \mathcal{T})$ randomly with the same probability for each element in $\mathcal{N} \times \mathcal{T}$
(6) Solve the subproblem SP_{it}^k: Fix $\overline{DC}^k(\overline{X})$ and re-optimized $DC^k(\overline{X})$.
(7) **if** the solution of SP_{it}^k, $\overline{X}_{SP_{it}^k}$ has lower costs than the current solution $\overline{X}$ **then**
(8) $\overline{X} = \overline{X}_{SP_{it}^k}$
(9) $iter = 0$
(10) **end if**
(11) **until** $iter \geqslant n$

Algorithm 1: Fix-and-Optimize algorithm with parameters (k, n).

Now we can continue to define "2-degree-connection" based on "1-degree-connection." We say two binary setup variables X_{it} and $X_{i't'}$ have "2-degree-connection" or X_{it} is "2-degree-connected" to $X_{i't'}$ if there exists a setup variable $X_{i_1,t_1} \in \widetilde{\Omega}$ such that both X_{it} and $X_{i't'}$ are "1-degree-connected" to X_{i_1,t_1}. Based on the previous definitions, "3-degree-connection," "4-degree-connection",..., "k-degree-connection" can be defined by induction. Then the sets $DC^2(X_{it}), DC^3(X_{it}), \ldots, DC^k(X_{it})$ can be defined.

Similar to "1-degree-connection", we can define the set of binary setup variables that are "k-degree-connected" to $\mathcal{S}$, denoted by $DC(\mathcal{S})$, as follows:

$$DC^k(\mathcal{S}) = DC\left(DC^{k-1}(\mathcal{S})\right), \quad k \geqslant 1. \tag{32}$$

Without ambiguity, we define "0-degree-connection" for completeness as follows, if two binary setup variables X_{it} and $X_{i't'}$ are "0-degree-connected" or have "0-degree-connection" if and only if $i' = i$ and $t' = t$. Hence, the definition of "k-degree-connection" is reflexive, transitive, and symmetric.

3.2. New FO Approach for S-MICLSP. Note that our FO approach solves a series of subproblems iteratively. The key to defining subproblems of our FO approach is to clarify which X_{it} should be reoptimized and which X_{it} should be fixed in each iteration. In the following, we apply the concept of "k-degree-connection" to decompose S-MICLSP and define subproblems of our FO approach. However, we also need some complementary definitions of "k-degree-connection." Recall the set of all binary setup variables, $\widetilde{\Omega} = \{X_{it} \mid i \in \mathcal{P}, t \in \mathcal{T}\}$. For any binary setup variable, $X_{it} \in \widetilde{\Omega}$, we define the complement set of $DC^k(X_{it})$ denoted by $\overline{DC}^k(X_{it}) = \widetilde{\Omega} \setminus DC^k(X_{it})$, which is the set of binary setup variables that are not "k-degree-connected" to X_{it}.

The subproblems of level k associated with X_{it}, denoted by SP_{it}^k, are simple and are defined in the following: $\overline{DC}^k(X_{it})$ is fixed and $DC^k(X_{it})$ is reoptimized. In this definition, we need to point out k is a control parameter and the bigger the level k is, the more binary setup variables are reoptimized in the corresponding subproblem SP_{it}^k. Due to this reason, we limit the maximum number of k to 3 for each subproblem SP_{it}^k in the following numerical experiments of Section 5.

To describe our FO approach, we denote $\overline{X} = \{X_{it}, i \in \mathcal{P}, t \in \mathcal{T}\}$ a setup plan or a setup solution of the model. Also, $\overline{X}$ is called the values of all setup variables at a solution of the model. Note that our FO approach allows the capacity-infeasible solutions in each loop of solving subproblems, while Chen [25] only selected capacity-feasible solutions. We present the pseudocode of our FO approach in Algorithm 1.

Note that, from the pseudocode of Algorithm 1, we only choose a pair (i, t) from $\mathcal{N} \times \mathcal{T}$ in each iteration. Hence, the number of possible subproblems is $N \times T$, where N is the number of the items and T is the number of periods. The number of iterations, n, is another control parameter of the approach. Since $N \times T$ may be very big and it may be too time-consuming for the approach to terminate after $N \times T$ iterations, we need to take an appropriate n that is equal to or smaller than $N \times T$ in the numerical simulation.

3.3. New FO Approach for S-MICLSP-L. Before we propose the new FO approach for S-MICLSP-L, we need to define the "k-degree-connection" of the setup carryover variables Z_{it} in a similar way and extend the scope of "k-degree-connection" of X_{it}. If Z_{it} changes, it may cause the change of the setup carryover variable $Z_{i't}$, whereby $\mathscr{D}_{i'} = \mathscr{D}_i = j$. The change of Z_{it} can also cause the change of the consecutive setup carryover variables $Z_{i-1,t}$ and $Z_{i+1,t}$ by the consecutive setup carryover constraints. From the resource constraints, we can infer that if Z_{it} changes, $Z_{i't}$ may change, whereby $i, i' \in \mathscr{P}_r$.

An observation shows that Z_{it} is restricted by X_{it}. Conversely, if Z_{it} changes, X_{it} may change according to the constraints of the model. The change of Z_{it} can also influence $X_{i',t}$ and $X_{i',t-1}$ whereby $\mathscr{D}_{i'} = \mathscr{D}_i = j$ or $i, i' \in \mathscr{P}_r$. Now we can define $\mathrm{DC}_Z(Z_{it})$ (the subscript Z indicates the approach for S-MICLSP-L), the set of binary variables that are "1-degree-connected" to Z_{it}, or the set of binary variables that have "1-degree-connection" with Z_{it} as follows:

$$\begin{aligned}\mathrm{DC}_Z\left(Z_{it}\right) = {} & \left\{Z_{i,t-1}, Z_{it}, Z_{i,t+1}\right\} \\ & \cup \left\{Z_{i't} \mid \mathscr{D}_{i'} = \mathscr{D}_i = j\right\} \\ & \cup \left\{Z_{i't} \mid i, i' \in \mathscr{P}_r\right\} \cup \left\{X_{i,t-1}, X_{it}\right\} \\ & \cup \left\{X_{i',t-1}, X_{i't} \mid \mathscr{D}_{i'} = \mathscr{D}_i = j\right\} \\ & \cup \left\{X_{i',t-1}, X_{i't} \mid i, i' \in \mathscr{P}_r\right\}.\end{aligned} \tag{33}$$

The definition of "k-degree-connection" for Z_{it}, $\mathrm{DC}_Z^k(Z_{it})$ is similar. We say that Z_{it} and $Z_{i't'}$ have "connection" or Z_{it} is "connected" to $Z_{i't'}$ if there is a finite integer $k \geqslant 1$ such that they are "k-degree-connected."

Respectively, we need to redefine $\mathrm{DC}(X_{it})$ (we use another denotation $\mathrm{DC}_Z(X_{it})$ for S-MICLSP-L), the set of binary variables that is "1-degree-connected" to X_{it}, or the set of binary variables that have "1-degree-connection" with X_{it} as follows:

$$\begin{aligned}\mathrm{DC}_Z\left(X_{it}\right) = {} & \left\{X_{i,t-1}, X_{it}, X_{i,t+1}\right\} \\ & \cup \left\{X_{i't} \mid \mathscr{D}_{i'} = \mathscr{D}_i = j\right\} \\ & \cup \left\{X_{i't} \mid i, i' \in \mathscr{P}_r\right\} \cup \left\{Z_{it}, Z_{i,t+1}\right\} \\ & \cup \left\{Z_{i't}, Z_{i',t+1} \mid \mathscr{D}_{i'} = \mathscr{D}_i = j\right\} \\ & \cup \left\{Z_{i't}, Z_{i',t+1} \mid i, i' \in \mathscr{P}_r\right\}.\end{aligned} \tag{34}$$

With the "1-degree-connection" of X_{it} defined above, we can similarly derive the new definition of "k-degree-connection" for X_{it} as previously described. Based on the new definition of "k-degree-connection" for X_{it}, we can define the subproblems for S-MICLSP-L, with defining the fix binary variables set $\overline{\mathrm{DC}}_Z(X_{it})$ and the reoptimized binary variables set $\mathrm{DC}_Z(X_{it})$, for any binary setup variable X_{it} or pair $(i,t) \in \mathscr{N} \times \mathscr{T}$. The fix-and-optimize approach for S-MICLSP-L is the same as S-MICLSP, except that their definitions of $\mathrm{DC}_Z(X_{it})$ and $\mathrm{DC}_Z(Z_{it})$ are different.

4. Integrative FO and Variable Neighborhood Search (VNS) Approach

We have stated that FO is a local method in Section 1. Since the structure of the feasible-solution set defined by the concept "k-degree-connection" can be relatively large, the solution searched by the FO approach can only be a local optimum in most cases. In order to find a global optimum, or a solution close to the global optimum, Chen [25] proposed an excellent framework which integrates FO and VNS. In his paper, the integrative framework emphasized great performances comparing to his FO approach. In this section, we partly adopt the framework and propose our integrative FO-VNS approach for the S-MICLSP and S-MICLSP-L. The main novelty in contrast to [25] is that our FO-VNS can be extended to models with setup carryovers.

4.1. Integrative FO-VNS for S-MICLSP. To describe our FO-VNS approach, we will use symbols as shown in Notations. To integrate VNS with our FO approach, we need to predefine a finite set of neighborhood structures $G_y(\overline{X})$, $y = 1, 2, \ldots, y_{\max}$, with $G_1(\overline{X}) \subseteq G_2(\overline{X}) \subseteq \cdots \subseteq G_{y_{\max}}(\overline{X})$, where y is the neighborhood index. $G_y(\overline{X})$ can be defined as (35), where q_y is an integer associated with neighborhood index y.

$$\begin{aligned}G_y\left(\overline{X}\right) = \Bigg\{ & X_{it}, i \in \mathscr{P}, t \\ & \in \mathscr{T} \mid \sum_{i=1}^{N}\sum_{t=1}^{T}\left(X_{it}\left(1-\overline{X}_{it}\right) + \overline{X}_{it}\left(1-X_{it}\right)\right) \leqslant q_y\Bigg\}.\end{aligned} \tag{35}$$

We also define model S-MICLSP$(\overline{X}, q_y, k)$, which is a linear relaxation model allowing $0 \leqslant X_{it} \leqslant 1$ derived from the original model by adding constraint (35) to it, where k is the subproblem level described in Section 3. In the FO-VNS approach for S-MICLSP, we obtain the local optimum of S-MICLSP$(\overline{X}, q_y, k)$ by applying our proposed FO approach in Section 3 as *local search* engine.

To enhance the search space, we need to shake the starting solution generated by each local search loop. In the *shaking* procedure, we apply our proposed FO approach as the swapping generator. We define the swapping initial solution X_{it}, which can possibly change from $X_{it} = \overline{X}_{it}$ to $X_{it} = 1 - \overline{X}_{it}$, and use X_{it} to define the subproblems of the FO generator. We also use a tabu list to keep the diversity realized by all previous *shaking*. Our *shaking* procedure is similar to [25] except for the swapping generator. With the above descriptions and notations, our FO-VNS approach for the model can be presented in pseudocode (see Algorithm 2).

4.2. Integrative FO-VNS for S-MICLSP-L. Due to the sophisticated multilevel structure, Chen [25] did not present the

(1) Set an initial feasible solution $\overline{X}_0$ of the model
(2) $\overline{X} = \overline{X}_0$, $\overline{X}^* = \overline{X}_0$, and $y = 1$
(3) **repeat**
(4) *(local search)* Apply the new FO approach as local search engine to solve S-MICLSP($\overline{X}, q_y$), with the neighborhood constraint $G_y(\overline{X})$. Then $\overline{X}'$ is obtained.
(5) **if** $\overline{X}'$ is better than the incumbent $\overline{X}^*$ **then**
(6) $\overline{X}^* = \overline{X}'$, $\overline{X} = \overline{X}'$, and $y = 1$
(7) **else**
(8) $\overline{X} = \overline{X}^*$, and $y = y + 1$
(9) **if** $y \geqslant y_{\max}$ **then**
(10) $y = 1$
(11) **end if**
(12) **end if**
(13) *(shaking)* Use the new FO approach and the tabu list to generate a new starting solution $\overline{X}''$ from the current solution $\overline{X}$. If the solution $\overline{X}''$ is better and does not exist in the tabu list, insert it into the list and let $\overline{X} = \overline{X}''$.
(14) Check the current computation time CT.
(15) **until** CT $\geqslant$ CT$_{\max}$

Algorithm 2: FO-VNS for S-MICLSP.

integrative framework for models with setup carryovers. However, in this paper, our model is single-level and hence we can extend our integrative FO-VNS to S-MICLSP-L. Similar to FO-VNS for S-MICLSP, the effective extension of FO-VNS for S-MICLSP-L also has two main parts, *local search* and *shaking*. Using symbols given in Notations, we now propose the FO-VNS for S-MICLSP-L.

We denote $X_Z = \{X_{it}, Z_{it}\}$ the setup and carryover plan. To construct a local searching engine for S-MICLSP-L, we first define the neighborhood structure of $G_{y_Z}(\overline{X}_Z)$ as

$$G_{y_Z}\left(\overline{X}_Z\right) = \left\{ X_{it}, Z_{it} \mid \sum_{i=1}^{N}\sum_{t=1}^{T}\left[\left(X_{it}\left(1-\overline{X}_{it}\right)+\overline{X}_{it}\left(1-X_{it}\right)\right)+X_{it}X_{i,t-1}Z_{it}\left(1-\overline{X}_{it}\overline{X}_{i,t-1}\overline{Z}_{it}\right)+\left(\overline{X}_{it}\overline{X}_{i,t-1}\overline{Z}_{it}\left(1-X_{it}X_{i,t-1}Z_{it}\right)\right)\right] \leqslant q_{y_Z} \right\}; \tag{36}$$

then the linear relaxation S-MICLSP-L($\overline{X}_Z, q_{y_Z}, k$) can be defined by adding the following constraint to the S-MICLSP-L model where $0 \leqslant X_{it} \leqslant 1$, $0 \leqslant Z_{it} \leqslant X_{it}$ and $0 \leqslant Z_{it} \leqslant X_{i,t-1}$. We predefine that $X_{i,0} = \overline{X}_{i,0} = 1$ for all $i \in \mathcal{P}$.

The swapping of the current setup and carryover plan $\overline{X}_Z$ means that its value is possibly changed from $X_{it} = \overline{X}_{it}$ and $Z_{it} = \overline{Z}_{it}$ to $X_{it} = 1 - \overline{X}_{it}$ and $Z_{it} = X_{it}X_{i,t-1}(1 - \overline{Z}_{it})$. The tabu list contains the setup and carryover plans which will be prevented from being selected by any future swap.

With the above statements, our FO-VNS approach for S-MICLSP-L is similar as Algorithm 2 for S-MICLSP and we neglect the pseudocode.

5. Numerical Experiments

In this section, we evaluate the performances of our proposed FO and FO-VNS approaches.

5.1. Experimental Design. We generate problem instances based on the attributes of real-world steel products. The developed instance generator is documented in Appendix, as well as the instance settings. The experimental design structure is as follows: we generate 100 instances with 10 products, 5 demands (or demand classes), 5 resources, and 50 periods. Both S-MICLSP and S-MICLSP-L are tested on the same instances.

All algorithms are coded in C++ in the environment of Microsoft Visual Studio 2012, and all instances are tested on a PC with Intel Core-i5 3.20 GHz CPU, 4 GB RAM. We compare our approaches with the fix-and-optimize approach of [17]. All LP and MIP subproblems involved are solved by calling the MIP solver of ILOG CPLEX 12.7. All problems and subproblems use a relative MIP gap tolerance of 10^{-4}; the time for ILOG CPLEX 12.7 to solve each subproblem is limited to 2 s for S-MICLSP and 4 s for S-MICLSP-L.

As the results depend on the number of line segments used in the approximated models (see (21) and (28) in Section 2.2) as well as the computational times, we solved each of the 100 problem instances with 5, 10, and 15 line segments. The numerical experiment shows that 10 line segments provide a good compromise between accuracy and

Table 1: Algorithm variants.

Algorithm	Decomposition order	Maximum subproblems n	Decomposition level k
FO1-P	P	—	—
FO1-R	R	—	—
FO1-T	T	—	—
FO1-PR	P → R	—	—
FO1-PT	P → T	—	—
FO1-PRT	P → R → T	—	—
FO2-n_1-L1	—	$N \times T$	1
FO2-n_1-L2	—	$N \times T$	2
FO2-n_1-L3	—	$N \times T$	3
FO2-n_2-L1	—	$0.5 \times N \times T$	1
FO2-n_2-L2	—	$0.5 \times N \times T$	2
FO2-n_2-L3	—	$0.5 \times N \times T$	3
FO2-n_3-L1	—	$0.25 \times N \times T$	1
FO2-n_3-L2	—	$0.25 \times N \times T$	2
FO2-n_3-L3	—	$0.25 \times N \times T$	3
FO-VNS-L1	—	—	1
FO-VNS-L2	—	—	2
FO-VNS-L3	—	—	3
CPX_{std}	—	—	—
CPX_{30std}	—	—	—

computational times. Hence we will choose 10 as the segments number in all other tests and comparisons.

Table 1 lists the algorithm variants compared in the computational experiments. We describe Table 1 in the following four aspects. (1) FO1 is referred to the fix-and-optimize approach proposed by [17], in which they presented three decomposition methods. They defined the subproblems, respectively, by product-, resource-, and time period-oriented decomposition (we refer to P-type, R-type, and T-type decomposition for short in the context). Further, they combined the three decomposition types and presented three more variants: P-type first and then R-type, P-type first and then T-type, and P-type first and then R-type ending with T-type. (2) FO2 is referred to our newly proposed FO approach, in which each variant is entitled with FO2-n-Lk. Recall that n is the control parameter of FO2 introduced in Section 3, and FO2 terminates if n subproblems are consecutively solved without improvement; that is, at most n subproblems are solved in each iteration. Otherwise, the solution of one subproblem is better than the current best solution of the main problem, and FO2 will proceed to a new iteration after the update of the best solution. The decomposition level k is another control parameter described in Section 3. Recall that a bigger k implies that more binary setup variables are reoptimized in one subproblem. Because considering subproblems of level larger than 3 is too time-consuming, we only test FO2 with subproblems of level $k = 1, 2, 3$ (see Section 3.2). (3) FO-VNS is referred to our integrative FO and VNS method. We vary the decomposition level k and obtain three variants of FO-VNS. Overall, absolute time limit for the algorithms is set to 10 minutes for S-MICLSP and 20 minutes for S-MICLSP-L, which can be slightly exceeded by finalizing code. (4) CPX_{std} and CPX_{30std} are referred to standard software (CPLEX, Branch & Cut) as comparisons. The absolute time limit for CPX_{std} is set to be same as FO-VNS, while the time limit of CPX_{30std} is more than CPX_{std} by 30 times.

5.2. Results and Interpretation. Table 2 describes the notations used to measure the solution quality.

5.2.1. Computational Results on S-MICLSP. Table 3 illustrates the computational results on S-MICLSP model. From this table, we can see that FO2 outperforms FO1 in terms of the solution quality. Among the six variants of FO1, FO1-PRT obtains the lowest $\overline{\text{Cost}}$. It can be observed from Table 3 that the variants of FO2 with bigger n and bigger k consume more computational time than FO1. However, FO2 can reduce $\overline{\text{Cost}}$ significantly compared with FO1 in general. 7 out of 9 FO2 variants perform a better $\overline{\text{Cost}}$ with respect to FO1 on average, meanwhile 5 out of 9 FO2 variants are better with respect to FO1-PRT. Also, another observation from this table is that FO2 is more efficient than FO1 in general. Taking FO2-n_2-L2 as an example, this FO2 variant can obtain both reduced $\overline{\text{Cost}}$ and $\overline{\text{Time}}$ compared either with FO1 on average or the best FO1 variant. Among all the variants of FO2, it seems that FO2-n_2-L2 makes the best trade-off between $\overline{\text{Cost}}$ and $\overline{\text{Time}}$ since it can provide better solutions with lower computational time.

For other measurements of the solution quality, FO2 still performs better than FO1 under most circumstances. Considering $\overline{\text{rate}}_{\text{ot}}$, which describes the proportion of periods with overtime, all of the FO1 variants are not able to provide

Table 2: Notations for measuring solution quality.

Notation	Definition	Quality measure
$\overline{\text{Cost}}$	Average total cost	Lower is better
$\overline{\text{Time}}$	Average computational time	Lower is better
$\overline{\text{rate}}_{\text{ot}}$	Average proportion of periods with overtime (in %)	Lower is better
$\overline{\delta - SL}$	Average δ-service level (in %) (see [29])	Higher is better
rate_{NOB}	Proportion of instances for which a feasible solution with neither overtime nor backlogging has been found by the algorithm (in %)	Higher is better
$\text{dev}^{\overline{\text{Cost}}}_{\text{Average}}$	Deviation from the average $\overline{\text{Cost}}$ of FO1 (in %)	Lower is better
$\text{dev}^{\overline{\text{Cost}}}_{\text{Best}}$	Deviation from the best $\overline{\text{Cost}}$ of FO1 (in %)	Lower is better
$\text{dev}^{\overline{\text{Time}}}_{\text{Average}}$	Deviation from the average $\overline{\text{Time}}$ of FO1 (in %)	Lower is better
$\text{dev}^{\overline{\text{Time}}}_{\text{Best}}$	Deviation from the $\overline{\text{Time}}$ of FO1 variants with best $\overline{\text{Cost}}$ (in %)	Lower is better
$\text{dev}^{\overline{\text{Cost}}}$	Deviation from the compared $\overline{\text{Cost}}$ (in %)	Lower is better
$\text{dev}^{\overline{\text{Time}}}$	Deviation from the compared $\overline{\text{Time}}$ (in %)	Lower is better

Table 3: Computational results on S-MICLSP.

Algorithm	$\overline{\text{Cost}}$		$\overline{\text{Time}}$		$\overline{\text{rate}}_{\text{ot}}$	$\overline{\delta - SL}$	rate_{NOB}
FO1-P	705103.74		212.41		6.78	96.58	53.00
FO1-R	715294.16		205.67		6.92	95.71	54.00
FO1-T	696034.43		222.19		5.08	97.80	65.00
FO1-PR	700148.81		218.97		5.00	96.41	59.00
FO1-PT	690659.08		232.80		3.20	100.00	80.00
FO1-PRT	689240.75		234.44		3.56	100.00	79.00
	$\text{dev}^{\overline{\text{Cost}}}_{\text{Average}}$	$\text{dev}^{\overline{\text{Cost}}}_{\text{Best}}$	$\text{dev}^{\overline{\text{Time}}}_{\text{Average}}$	$\text{dev}^{\overline{\text{Time}}}_{\text{Best}}$			
FO2-n_1-L1	−1.65	−0.20	2.13	−3.69	0.00	100.00	100.00
FO2-n_1-L2	−2.27	−0.83	11.40	5.05	0.00	100.00	100.00
FO2-n_1-L3	−3.20	−1.77	18.31	11.57	0.00	100.00	100.00
FO2-n_2-L1	−0.77	0.70	−8.80	−14.00	4.00	97.75	84.00
FO2-n_2-L2	−1.93	−0.48	−0.13	−5.82	3.86	100.00	86.00
FO2-n_2-L3	−2.74	−1.30	10.99	4.67	3.92	100.00	86.00
FO2-n_3-L1	2.38	3.90	−23.21	−27.58	5.00	95.24	51.00
FO2-n_3-L2	0.69	2.18	−13.47	−18.40	5.00	95.15	55.00
FO2-n_3-L3	−0.55	0.92	5.87	−0.16	5.22	97.90	52.00
FO-VNS-L1	−4.95	−3.55	172.32	156.81	0.00	100.00	100.00
FO-VNS-L2	−5.62	−4.23	173.06	157.50	0.00	100.00	100.00
FO-VNS-L3	−5.99	−4.60	172.84	157.29	0.00	100.00	100.00
CPX_{std}	4.08	5.62	171.50	156.03	4.62	95.59	81.00
$\text{CPX}_{\text{30std}}$	−7.31	−5.95	8141.17	7671.53	0.00	100.00	100.00

solutions completely without overtime. However, when $n = n_1 = N \times T$, FO2 can make $\overline{\text{rate}}_{\text{ot}}$ at a zero level, which implies lower overtime cost than FO1. For other FO2 variants, they can also obtain competitive results against FO1. Considering $\overline{\delta - SL}$, only 2 out of 6 FO1 variants obtain 100 percent δ-service level while 5 out of 9 FO2 variants can achieve this goal. We find that, for bigger n and k, FO2 tends to achieve a high δ-service level from the column $\overline{\delta - SL}$ of Table 3. Consider rate_{NOB}, which indicates the solution quality by combining $\overline{\text{rate}}_{\text{ot}}$ and $\overline{\delta - SL}$ in a statistical sense. The results of FO2-n_1-Lk ($k = 1, 2, 3$) imply that there exist solutions without overtime and backlogging for all tested instances, while FO1 obtain solutions at a relative lower rate_{NOB} proportion. We can also observe that for a lower $n = n_3$, FO2-n_3-L2 performs better than FO1-P and FO1-R, while other FO2-n_3-Lk ($k = 1, 3$) perform worse than all the FO1 variants.

The three rows entitled "FO-VNS-Lk" of Table 3 report the solution quality of FO-VNS variants with different k. From the results, the three variants of FO-VNS obtain around 4.95%~5.99% better solutions compared with FO1

Table 4: Computational results on S-MICLSP-L.

Algorithm	$\overline{\text{Cost}}$		$\overline{\text{Time}}$		$\overline{\text{rate}_{ot}}$	$\overline{\delta - \text{SL}}$	rate_{NOB}
FO1-P	727790.95		436.31		15.36	82.41	37.00
FO1-R	741102.74		414.03		20.48	73.96	23.00
FO1-T	717907.32		429.78		14.52	83.69	33.00
FO1-PR	723629.97		442.17		13.00	80.94	45.00
FO1-PT	716389.74		452.53		13.32	87.00	42.00
FO1-PRT	712618.87		474.03		13.00	91.04	46.00
	$\text{dev}^{\overline{\text{Cost}}}_{\text{Average}}$	$\text{dev}^{\overline{\text{Cost}}}_{\text{Best}}$	$\text{dev}^{\overline{\text{Time}}}_{\text{Average}}$	$\text{dev}^{\overline{\text{Time}}}_{\text{Best}}$			
FO2-n_1-L1	−2.40	−0.94	−5.27	−11.77	4.42	96.59	85.00
FO2-n_1-L2	−3.17	−1.73	5.20	−2.02	4.20	96.31	87.00
FO2-n_1-L3	−3.93	−2.50	13.38	5.59	3.94	95.80	88.00
FO2-n_2-L1	−1.10	0.37	−8.28	−14.58	8.90	94.39	63.00
FO2-n_2-L2	−2.36	−0.91	0.06	−6.81	8.38	96.03	62.00
FO2-n_2-L3	−2.81	−1.36	9.21	1.71	8.00	98.80	68.00
FO2-n_3-L1	1.95	3.47	−10.49	−16.63	10.64	85.49	53.00
FO2-n_3-L2	0.71	2.22	−3.92	−10.52	9.68	89.13	50.00
FO2-n_3-L3	−0.63	0.85	4.70	−2.49	9.00	92.03	56.00
FO-VNS-L1	−6.56	−5.17	172.67	153.95	4.94	100.00	83.00
FO-VNS-L2	−7.32	−5.94	172.74	154.01	4.54	100.00	84.00
FO-VNS-L3	−7.54	−6.16	173.38	154.61	4.02	100.00	84.00
CPX_{std}	4.14	5.69	172.56	153.84	7.84	93.76	71.00
$\text{CPX}_{\text{30std}}$	−8.61	−7.25	8187.93	7618.80	0.00	100.00	90.00

on the average, and around 3.55%~4.60% compared with the best FO1 significantly, while the computational time can be considerably larger by 172.32%~173.06% and 156.81%~157.50% compared with the average FO1 and the best FO1. However, the S-MICLSP (also S-MICLSP-L) considered in this paper is usually solved weekly or monthly in a tactical decision for a factory of one steel enterprise. It is thus worth spending more but reasonable time to obtain a significantly better production plan by our FO-VNS.

The last two rows of Table 3 provide comparisons between CPLEX and our proposed algorithms. CPX_{std} runs the same time limit as FO-VNS but it performs worse than all other algorithms. Running a much longer time limit, $\text{CPX}_{\text{30std}}$ performs much better. However compared to FO-VNS, the solution quality of $\text{CPX}_{\text{30std}}$ is slightly improved with an extremely increased computational effort.

5.2.2. Computational Results on S-MICLSP-L. The computational results on the S-MICLSP-L model are given in Table 4. By the similar observations to Table 3, FO2 still outperforms FO1 in terms of the solution quality. Particularly, FO2-n_1-L1 and FO2-n_2-L2 outperform all the FO1 variants when considering $\overline{\text{Cost}}$ and $\overline{\text{Time}}$, which implies they make the best trade-off between $\overline{\text{Cost}}$ and $\overline{\text{Time}}$ in all FO2 variants. Similar to Table 3, FO-VNS gives quite outstanding results compared with FO1 and FO2. FO-VNS performs a better $\overline{\text{Cost}}$ by reducing 5.17%~6.16% against FO1-PRT. All of the FO-VNS variants provide $\overline{\delta - SL}$ of 100%, yet FO1 and FO2 fail to do so. The results of the last two rows in Table 4 are similar to Table 3. CPX_{std} reports its drawbacks in terms of solution quality and computational effort, while $\text{CPX}_{\text{30std}}$ reports its better solution quality with much longer computational time.

According to Tables 3 and 4, it can be confirmed that the advantages of FO2 and FO-VNS against FO1 for S-MICLSP-L grow compared with S-MICLSP. For example, FO2-n_1-L3 has −3.93% of $\text{dev}^{\overline{\text{Cost}}}_{\text{Average}}$ while it has −3.20% in the S-MICLSP case from Table 3. Also, this FO2 variant can obtain −2.50% of $\text{dev}^{\overline{\text{Cost}}}_{\text{Best}}$ while it can obtain 1.77% in the S-MICLSP case from Table 3. This observation can also be found for all variants of FO-VNS.

5.2.3. Comparison of Various Cases of Parameter Settings. To check the effectiveness and efficiency of our proposed approaches under various cases of parameter settings, we implement FO1-PRT, FO2-n_1-L3, and FO-VNS-L3 (best variants of FO1, FO2, and FO-VNS tested ever) again using test cases of different TBO (time between orders) and ST (setup times). TBO can be computationally defined as the ratio between setup costs and inventory costs (see [28]). Varying different parameters, the results of Table 5 with measurements of $\overline{\text{Cost}}$ and $\overline{\text{Time}}$ are illustrated by groups of combinations of TBOs (low with TBO $<$ 2.0, medium with 2.0 $\leqslant$ TBO $<$ 3.0, and high with 3.0 $<$ TBO) and STs (low with ST $<$ 25, and high with ST $\geqslant$ 25).

We compare the results in terms of FO1-PRT for the benchmark. From Table 5, we can see that for high TBO cases, FO2-n_1-L3 and FO-VNS-L3 significantly outperform FO1-PRT in terms of solution quality. These outstanding performances of FO2-n_1-L3 and FO-VNS-L3 can also be found in cases of high ST, compared to cases of low ST.

Table 5: Computational results with various parameters settings.

Algorithm model	TBO	ST	FO1-PRT		FO2-n_1-L3		FO-VNS-L3	
			$\overline{\text{Cost}}$	$\overline{\text{Time}}$	$\text{dev}^{\overline{\text{Cost}}}$	$\text{dev}^{\overline{\text{Time}}}$	$\text{dev}^{\overline{\text{Cost}}}$	$\text{dev}^{\overline{\text{Time}}}$
S-MICLSP	High	High	795664.08	245.51	−2.23	9.96	−4.94	145.95
		Low	689240.75	234.44	−1.77	11.57	−4.60	157.29
	Medium	High	575783.97	244.29	−1.71	8.03	−4.29	146.78
		Low	489240.75	230.71	−1.07	9.63	−2.39	161.63
	Low	High	336501.13	237.52	−0.38	2.35	−3.69	152.33
		Low	230349.23	229.79	2.19	−4.34	−1.83	162.06
S-MICLSP-L	High	High	817073.11	490.18	−3.08	5.44	−6.39	145.87
		Low	712618.87	474.03	−2.50	5.59	−6.16	154.60
	Medium	High	576455.47	486.10	−2.00	3.71	−3.45	146.66
		Low	502618.87	478.28	−1.16	4.26	−2.77	151.84
	Low	High	337218.37	486.34	−0.13	−1.50	−2.50	147.05
		Low	251979.47	466.17	0.18	0.57	−1.10	158.21

Considering cases of high or medium TBO, FO2-n_1-L3 and FO-VNS-L3 obtain lower $\text{dev}^{\overline{\text{Cost}}}$ in S-MICLSP-L than that in S-MICLSP. It implies that for high or medium TBO cases, the advantages of FO2 and FO-VNS in S-MICLSP-L grow compared to that in S-MICLSP, which is similar to the previous observations in Section 5.2.2. All of the above results imply that FO2 and FO-VNS are much more effective than FO1 in cases of high or medium TBO and high ST.

5.2.4. Comparison with Scenario Methods. Typically, because of the continuous nature of demand uncertainty and the dynamic structure of the problems, multistage stochastic modeling methods can be applied and hence scenario methods are used to approximate the models. This methodology is distinguished from ours and here we numerically compare the difference between them. The methodology using scenario method is denoted by SCNm-BC, where Branch & Cut approach is used to solve the models approximated by scenario method. We generate m realizations of random demand for each period and hence the number of scenarios is m^T, where T is the number of periods. We choose $m = 5, 10$ for small test cases. Our methodologies are denoted by PWL-FO2 and PWL-FO-VNS, whereby FO2-n_1-L3 and FO-VNS-L3 are, respectively, applied.

We compare the results in terms of the methodology of PWL-FO2 for the benchmark. From Table 6, we can generally see that the methodology of PWL-FO-VNS is competitive with SCN5-BC and SCN10-BC in terms of $\overline{\text{Cost}}$. The entry "$****$" indicates that there are no computational results. Since the number of scenarios is growing in size exponentially, it is impossible to compute finalized results in reasonable time even when T is relatively small. Both PWL-FO2 and PWL-FO-VNS are more splendid than SCN5-BC in terms of solution quality. SCN10-BC can obtain better solutions when $T = 2, 3$; however the computational time is unacceptable when $T = 6, 10$. Our proposed PWL-FO-VNS obtains competitive solution quality against SCN10-BC while the computational efforts are much less. All of the results imply that our proposed approximation method and solving approaches are efficient.

6. Conclusion

The key contributions and findings of this paper are as follows:

(1) We formulate dynamic S-MICLSP and S-MICLSP-L models mapped to a realistic problem in steel production. We also propose a piecewise linear approximation method to reformulate models that can be solved tractably. This method is novel and can balance the approximated accuracy and computational times. This method can also be extended to other cases such as the lot-sizing problems with substitutions (see [18]) and safety stocks (see [29]).

(2) We present a new fix-and-optimize (FO) approach for both S-MICLSP and S-MICLSP-L which possesses a novel way of decomposing. Our FO decomposes the problems based on the concept of "k-degree-connection" described in Section 3. This decomposition method combines all the information about products, resources, demands, and time periods. Hence, our presented approach is more effective in each iteration. The computational experiments show that our proposed approach outperforms the recent one.

(3) We develop an integrative FO-VNS approach, based on diversifying search space by VNS. The FO-VNS explores more promising regions in each iteration. This approach extends the scope of [25] and can be applied to models with setup carryovers. From numerical results, FO-VNS can obtain solutions with quite high quality by consuming reasonable time in a tactical planning decision, especially in the testing on S-MICLSP-L model.

Table 6: Computational results compared with scenarios methods.

Methodology model	Periods	PWL-FO2		PWL-FO-VNS		SCN5-BC		SCN10-BC	
		$\overline{\text{Cost}}$	$\overline{\text{Time}}$	$\text{dev}^{\overline{\text{Cost}}}$	$\text{dev}^{\overline{\text{Time}}}$	$\text{dev}^{\overline{\text{Cost}}}$	$\text{dev}^{\overline{\text{Time}}}$	$\text{dev}^{\overline{\text{Cost}}}$	$\text{dev}^{\overline{\text{Time}}}$
S-MICLSP	$T = 2$	31610.95	6.05	0.65	133.29	4.52	−19.82	−1.81	246.14
	$T = 3$	52896.83	9.84	−2.67	120.10	1.34	121.28	−2.44	1709.16
	$T = 6$	106087.66	16.48	−3.03	117.48	−4.13	15134.03	****	****
	$T = 10$	180137.95	27.52	−3.33	116.87	****	****	****	****
S-MICLSP-L	$T = 2$	35294.55	11.06	−2.15	161.47	3.20	−13.05	−1.61	236.42
	$T = 3$	55840.08	17.84	−2.86	144.40	3.73	207.43	−3.43	2439.68
	$T = 6$	113090.88	32.39	−3.54	137.08	−4.31	17906.94	****	****
	$T = 10$	195927.06	52.22	−4.02	135.99	****	****	****	****

Table 7: Instance generator settings.

Symbol	Definition	Assumptions and settings
Indices and sets:		
$\mathscr{P}_j$	Products used to fulfill demand j	Set up using clustering algorithm such as K-means clustering and hierarchical clustering after every attribute of products is realized
$\mathscr{P}_r$	Products that are produced by resource r	Set up for randomly chosen product
Deterministic parameters:		
f_i	Setup cost	See (A.4)
p_i	Unit production cost	See (A.3)
h_i	Unit holding cost	2% of the unit production cost
c_i	Unit capacity	See (A.2)
st_i	Setup time capacity	See (A.1)
k_r	Available capacities	See (A.6)
b_j	Unit backlogging Penalty cost	$\geqslant 50 \times \sum_{i \in \mathscr{P}_j} p_i$
oc_r	Overtime cost	See (A.7)
e_{ij}	Demand coefficient	Set $e_{ij} = 1$ for all i and j.
I_{i0}	Initial inventory	$\sim U^{\text{int}}(1.5\xi_i, 4\xi_i)$, with $\xi_i = (p_i / \sum_{k \in \mathscr{P}_j} p_k)(1/T) \sum_{t \in \mathscr{T}, j \in \mathscr{D}_i} E[D_{jt}]$
Random parameters:		
D_{jt}	Demand	Normally distributed, stationary across period horizons, but different mean μ and variation coefficient σ/μ for each demand: $\sim N^{\geqslant 0, \leqslant 5\mu}(\mu, \sigma)$ (with values < 0 cut-off), with $\mu \sim N^{\geqslant 0}(50, 20)$ and $\sigma/\mu \sim N^{\geqslant 0}(0.2, 0.1)$

This paper can explore several avenues in future research; for example:

(i) Extend the S-MICLSP and S-MICLSP-L models to the multilevel versions.

(ii) Find a more effective way to construct decomposition frameworks in each iteration.

(iii) Develop a more efficient local search engine for the integrative metaheuristics.

Appendix

Generating Problem Instances

To solve the real S-MICLSP and S-MICLSP-L applications, we need to gather data in steel production. However, due to the reasons of confidentiality, we have to generate sufficient problem instances, in which structure is as realistic as possible and capacities are rather tight. To ensure that the optimal ordering decisions are nontrivial, we propose a method for generating problem instances based on the realistic attributes of products. The symbols used for describing the instance generator are given in Notations. To generate instances, we also delineate the settings and assumptions of our method which are contained in Table 7.

Each product can be described by values of several attributes $a \in \mathscr{F}$. We assume that all attribute values are integers and $\geqslant 0$. Suppose that we have known the number of elements in demands (or demand classes) $\mathscr{D}$, products $\mathscr{P}$, and resources $\mathscr{R}$, the next question is how to determine $\mathscr{P}_j$ and $\mathscr{P}_r$. From the context, it is obvious that M, the number of elements in $\mathscr{D}$, is not greater than N, which is the number

Table 8: An example of product attributes used in steel production.

Attribute number	Example (name)	Distribution	Characteristic
1	Mixture	$\sim U^{\text{int}}(0,1)$	Property
2	Thickness	$\sim U^{\text{int}}(0,2)$	Measurement
3	Width	$\sim U^{\text{int}}(70,100)$	Measurement
...	...	...	...

of elements in $\mathscr{P}$. Then we can use clustering algorithms such as K-means clustering and hierarchical clustering to cluster the N elements into M classes after the attribute values are realized (see [30] for more details about clustering algorithms). Thus $\mathscr{P}_j$ is determined. As for $\mathscr{P}_r$, we are building it by assigning the resources randomly to each product.

The following two groups of attributes distinguished by real-world steel production can help us generate the instances further.

(1) Property. Attributes that are in products themselves. The products are significantly different from other products by this kind of attributes. These attributes are also the key factors for generating the setup parameters such as st_i and f_i. The set containing this kind of attributes is denoted by $\mathscr{F}_P$.

(2) Measurement. Attributes for specification of some kinds of standards, such as width and length. The variation of these values can also fluctuate the marginal parameters such as c_i, p_i. The set containing this kind of attributes is denoted by $\mathscr{F}_M$.

Table 8 lists an example of product attributes in steel production. The underlying motivation of this classification of the attributes is the fact that the occupied resource of one product is mainly dependent on its "Property" attribute. Hence, the idea for generating the resource parameters, st_i and c_i, is as follows: we assume that both st_i and c_i of one product are linear functions of its attributes value v_{ai}. Particularly, for st_i, we assume that it is linear function of v_{ai} only whose attributes are "Property." For all attributes, the weights w_a^c and w_a^{st} (which could also be negative) are multiplied with the attribute value v_{ai}. We also assume the two base values, c_{0i} and st_{0i}, for generating st_i and c_i, where $c_{0i} \sim N^{\geqslant 0}(3, 0.5)$ and $\text{st}_{0i} \sim N^{\geqslant 0}(2, 0.5)$ (with values < 0 cut-off). Thus, we can generate st_i and c_i as follows:

$$\text{st}_i = \text{st}_{0i} + \sum_{a \in \mathscr{F}_P} w_a^{\text{st}} \cdot v_{ai} \tag{A.1}$$

$$c_i = c_{0i} + \sum_{a \in \mathscr{F}} w_a^c \cdot v_{ai}. \tag{A.2}$$

An observation from production viewpoint is that the unit production cost p_i is strongly correlated to c_i while the setup cost f_i is strongly correlated to st_i. In order to generate p_i and f_i, we first sample random variable U from a normal distribution $N^{\geqslant 0}(1, 0.2)$. Using these random numbers, both p_i and f_i are calculated using same random value, so that they are correlated. We also introduce m_r^c and m_r^{st} as multipliers for physical unit standardization since we assume that U is dimensionless:

$$p_i = 50 \cdot c_i \cdot m_r^c \cdot U \tag{A.3}$$

$$f_i = 50 \cdot \text{st}_i \cdot m_r^{\text{st}} \cdot U. \tag{A.4}$$

The procedure for generating k_r and oc_r is as follows: we first calculate the minimum amount of the resource capacity required for producing sufficient quantities of each product to fulfill the expected demand. We denote this amount $\overline{K}$.

$$\overline{K} = \min\{c_i \mid i \in \mathscr{P}\} \cdot \left(\sum_{t \in \mathscr{T}, j \in \mathscr{D}} E\left[D_{jt}\right] - \sum_{i \in \mathscr{P}} I_{i0} \right). \tag{A.5}$$

The corresponding capacity of resource r, k_r, is estimated by the unit capacity consumption c_i and the setup capacity st_i, while the overtime cost, oc_r, can be also estimated by the unit product cost p_i and the setup cost f_i. Both k_r and oc_r are affected by α^c and α^{st} so they can be correlated with each other.

$$k_r = \beta^k \cdot \frac{\sum_{i \in \mathscr{P}_r} \left(\alpha^c c_i + \alpha^{\text{st}} \text{st}_i\right)}{\sum_{i \in \mathscr{P}} \left(\alpha^c c_i + \alpha^{\text{st}} \text{st}_i\right)} \cdot \frac{\overline{K}}{T} \tag{A.6}$$

$$\text{oc}_r = \beta^{\text{oc}} \cdot \sum_{i \in \mathscr{P}_r} \left(\alpha^c p_i + \alpha^{\text{st}} f_i\right). \tag{A.7}$$

Note that the concept of the instance generator can be extended to the aspects of other products in the production management. This methodology can make the structure of the generating instances as realistic as possible since all the parameters are based on the attributes of the products.

Notations

Notations Used in Model Formulations

Indices and Index Sets

$r \in \mathscr{R}$: Set of resources ($\mathscr{R} = \{1, \ldots, R\}$)
$i \in \mathscr{P}$: Set of products ($\mathscr{P} = \{1, \ldots, N\}$)
$j \in \mathscr{D}$: Set of demands (or demand classes) ($\mathscr{D} = \{1, \ldots, M\}$)
$t \in \mathscr{T}$: Set of periods ($\mathscr{T} = \{1, \ldots, T\}$)
$(i, j) \in \mathscr{A}$: $(i, j) \in \mathscr{A}$ If and only if product i can fulfill demand (class) j

$\mathscr{P}_j$: Set of products that can fulfill demand (class) j
$\mathscr{P}_r$: Set of products that produced by resource r
$\mathscr{D}_i$: Set of demands (or demand classes) whose demand can be fulfilled by product i (only one element under many-to-one structure assumption).

Deterministic Parameters

f_i: Incurred setup cost when production for product i is ready
p_i: Unit production cost of product i
h_i: Holding cost for storing product i to next period per unit and period
b_j: Backlogging penalty cost for demand (class) j per unit and period
oc_r: Overtime cost of resource r per unit of overtime
c_i: Capacity required for manufacturing one unit of product i
st_i: Setup time capacity required for manufacturing product i
k_r: Available capacity of resource r
m_i: Large number, required for setup forcing constraint
e_{ij}: Number of units of product i that satisfies one unit of demand (class) j for any $(i, j) \in \mathscr{A}$.

Random Variables

B_{jt}: Backlog number of demand (class) j during period t
I_{it}: Physical inventory level of product i at the end of period t.

Random Parameters

D_{jt}: Demand (class) j in period t.

Decision Variables

S_{ijt}: Quantity of product i used to fulfill demand (class) j at period t
Q_{it}: Production quantity of product i at period t
X_{it}: Binary variable that indicates whether production of product i occurs at period t
Z_{it}: Binary setup carryover variable for item i at the beginning of period t
O_{rt}: Additional capacity of resource r at period t.

Additional Notations for Approximation

Indices and Index Sets

$l \in \mathscr{L}$: Set of segments ($\mathscr{L} = \{1, \ldots, L\}$).

Random Variables

y_{jt}: Cumulated demand of demand j from period 1 up to period t
I_{jt}: Physical inventory level of demand j at the end of period t
I_{jt}^{prod}: Backlog number of demand j after production at period t, but before demand occurrence
I_{jt}^{end}: Backlog number of demand j at the end of period t.

Approximation Variables

$a_{I_{jt}}^{l}$: Slope value of the on hand inventory for demand j in period t associated with segment l
$a_{B_{jt}}^{l}$: Slope value of the backlog for demand j in period t associated with segment l
$P_{y_{jt}}(x)$: Expected loss function or the failure function of the random variable y_{jt} with respect to the quantity x.

Decision Variables

w_{ijt}^{l}: Cumulated production quantity of product i for demand j in period t associated with interval l
q_{ijt}: Production quantity of product i for demand j in period t.

Notations Used in FO-VNS for S-MICLSP

$\overline{X}$: Setup plan for the current solution
$\overline{X}^*$: Setup plan for the incumbent (the current best solution)
CT: Computation time so far
$\mathrm{CT}_{\max}$: Maximum computation time allowed
y: Index of neighborhood structure $G_y(\overline{X})$
$y_{\max}$: Maximum number of neighborhood considered.

Notations Used in FO-VNS for S-MICLSP-L

$\overline{Z}$: Carryover plan for the current solution
$\overline{Z}^*$: Carryover plan for the incumbent (the current best solution)
$\widehat{Z}$: Carryover plan for linear relaxation model
$\overline{X}_Z = \{(\overline{X}_{it}, \overline{Z}_{it})\}$: Setup and carryover plan for the current solution
$\overline{X}_Z^* = \{(\overline{X}_{it}^*, \overline{Z}_{it}^*)\}$: Setup and carryover plan for the incumbent (the current best solution)
y_Z: Index of neighborhood structure G_{y_Z}
$y_{Z,\max}$: Maximum number of neighborhood considered

$G_{y_Z}(\overline{X}_Z)$: A finite set of neighborhood structures, with $G_1(\overline{X}_Z) \subseteq G_2(\overline{X}_Z) \subseteq \cdots \subseteq G_{y_{Z,\max}}(\overline{X}_Z)$

q_{y_Z}: Maximum distance between two setup and carryover plans, where both of them are in the same neighborhood structure G_{y_Z}.

Auxiliary Notations for Instance Generator

Constants

n_a: Number of attributes.

Indices and Sets

$a \in \mathcal{F} = \{1, 2, \ldots, n_a\}$: Attributes
$\mathcal{F}_P \subseteq \mathcal{F}$: Set of "Property" attributes
$\mathcal{F}_M \subseteq \mathcal{F}$: Set of "Measurement" attributes.

Parameters

w_a^c: Production capacity weight for attribute a
w_a^{st}: Setup capacity weight for attribute a
m_r^c: Dimensional multiplier for calculating unit production cost
m_r^{st}: Dimensional multiplier for calculating setup cost
$\alpha^c, \alpha^{\text{st}}$: Dimensional multipliers for calculating parameters of resource capacity
$\beta^k, \beta^{\text{oc}}$: Dimensionless multipliers for calculating parameters of resource capacity.

Random Values

v_{ai}: Value for attribute a of product i
c_{0i}: Base value of production capacity for attribute a of product i
st_{0i}: Base value of setup capacity for attribute a of product i
U: Random variable, used as a multiplier (dimensionless).

Derived Values

$\overline{K}$: Total resource capacities required for producing the expected demand for all products.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research has been supported by the National Natural Science Foundation of China under Grants 61273233 and U1660202.

References

[1] K. Haase, "Capacitated lot-sizing with sequence dependent setup costs," *OR Spektrum. Quantitative Approaches in Management*, vol. 18, no. 1, pp. 51–59, 1996.

[2] K. Haase, "Capacitated lot-sizing with linked production quantities of adjacent periods," in *Beyond Manufacturing Resource Planning (MRP II)*, pp. 127–146, Springer, 1998.

[3] T. Wu, K. Akartunalı, J. Song, and L. Shi, "Mixed integer programming in production planning with backlogging and setup carryover: modeling and algorithms," *Discrete Event Dynamic Systems*, vol. 23, no. 2, pp. 211–239, 2013.

[4] M. A. F. Belo-Filho, F. M. B. Toledo, and B. Almada-Lobo, "Models for capacitated lot-sizing problem with backlogging, setup carryover and crossover," *Journal of the Operational Research Society*, vol. 65, no. 11, pp. 1735–1747, 2014.

[5] B. Karimi and S. M. T. Fatemi Ghomi, "A new heuristic for the CLSP problem with backlogging and set-up carryover," *International Journal of Advance Manufacturing Systems*, vol. 5, no. 2, pp. 66–77, 2002.

[6] B. Karimi, S. M. T. F. Ghomi, and J. M. Wilson, "A tabu search heuristic for solving the CLSP with backlogging and set-up carry-over," *Journal of the Operational Research Society*, vol. 57, no. 2, pp. 140–147, 2006.

[7] D. Quadt and H. Kuhn, "Capacitated lot-sizing and scheduling with parallel machines, back-orders, and setup carry-over," *Naval Research Logistics*, vol. 56, no. 4, pp. 366–384, 2009.

[8] C. R. Sox and Y. Gao, "The capacitated lot sizing problem with setup carry-over," *IIE Transactions*, vol. 31, no. 2, pp. 173–181, 1999.

[9] K. K. Haugen, A. Løkketangen, and D. L. Woodruff, "Progressive hedging as a meta-heuristic applied to stochastic lot-sizing," *European Journal of Operational Research*, vol. 132, no. 1, pp. 116–122, 2001.

[10] P. Brandimarte, "Multi-item capacitated lot-sizing with demand uncertainty," *International Journal of Production Research*, vol. 44, no. 15, pp. 2997–3022, 2006.

[11] P. Beraldi, G. Ghiani, A. Grieco, and E. Guerriero, "Fix and relax heuristic for a stochastic lot-sizing problem," *Computational Optimization and Applications*, vol. 33, no. 2-3, pp. 303–318, 2006.

[12] Y. Guan, "Stochastic lot-sizing with backlogging: computational complexity analysis," *Journal of Global Optimization*, vol. 49, no. 4, pp. 651–678, 2011.

[13] R. Ramezanian and M. Saidi-Mehrabad, "Hybrid simulated annealing and MIP-based heuristics for stochastic lot-sizing and scheduling problem in capacitated multi-stage production system," *Applied Mathematical Modelling*, vol. 37, no. 7, pp. 5134–5147, 2013.

[14] R. Levi and C. Shi, "Approximation algorithms for the stochastic lot-sizing problem with order lead times," *Operations Research*, vol. 61, no. 3, pp. 593–602, 2013.

[15] D. Mietzner and G. Reger, "Advantages and disadvantages of scenario approaches for strategic foresight," *International Journal of Technology Intelligence and Planning*, vol. 1, no. 2, pp. 220–239, 2005.

[16] B. Karimi, S. M. T. Fatemi Ghomi, and J. M. Wilson, "The capacitated lot sizing problem: a review of models and algorithms," *Omega*, vol. 31, no. 5, pp. 365–378, 2003.

[17] F. Sahling, L. Buschkühl, H. Tempelmeier, and S. Helber, "Solving a multi-level capacitated lot sizing problem with

multi-period setup carry-over via a fix-and-optimize heuristic," *Computers and Operations Research*, vol. 36, no. 9, pp. 2546–2553, 2009.

[18] J. C. Lang and Z.-J. M. Shen, "Fix-and-optimize heuristics for capacitated lot-sizing with sequence-dependent setups and substitutions," *European Journal of Operational Research*, vol. 214, no. 3, pp. 595–605, 2011.

[19] C. F. M. Toledo, M. da Silva Arantes, M. Y. B. Hossomi, P. M. França, and K. Akartunalı, "A relax-and-fix with fix-and-optimize heuristic applied to multi-level lot-sizing problems," *Journal of Heuristics*, vol. 21, no. 5, pp. 687–717, 2015.

[20] J. Xiao, C. Zhang, L. Zheng, and J. N. D. Gupta, "MIP-based fix-and-optimise algorithms for the parallel machine capacitated lot-sizing and scheduling problem," *International Journal of Production Research*, vol. 51, no. 16, pp. 5011–5028, 2013.

[21] P. Hansen and N. Mladenović, "Variable neighborhood search: principles and applications," *European Journal of Operational Research*, vol. 130, no. 3, pp. 449–467, 2001.

[22] K. S. Hindi, K. Fleszar, and C. Charalambous, "An effective heuristic for the CLSP with set-up times," *Journal of the Operational Research Society*, vol. 54, no. 5, pp. 490–498, 2003.

[23] Q. Zhao, C. Xie, and Y. Xiao, "A variable neighborhood decomposition search algorithm for multilevel capacitated lot-sizing problems," *Electronic Notes in Discrete Mathematics*, vol. 39, pp. 129–135, 2012.

[24] F. Seeanner, B. Almada-Lobo, and H. Meyr, "Combining the principles of variable neighborhood decomposition search and the fix&optimize heuristic to solve multi-level lot-sizing and scheduling problems," *Computers and Operations Research*, vol. 40, no. 1, pp. 303–317, 2013.

[25] H. Chen, "Fix-and-optimize and variable neighborhood search approaches for multi-level capacitated lot sizing problems," *Omega*, vol. 56, pp. 25–36, 2015.

[26] G. D. Eppen and R. K. Martin, "Solving multi-item capacitated lot-sizing problems using variable redefinition," *Operations Research*, vol. 35, no. 6, pp. 832–848, 1987.

[27] M. Caserta and S. Voß, "A MIP-based framework and its application on a lot sizing problem with setup carryover," *Journal of Heuristics*, vol. 19, no. 2, pp. 295–316, 2013.

[28] W. W. Trigeiro, L. J. Thomas, and J. O. McClain, "Capacitated lot sizing with setup times," *Management Science*, vol. 35, no. 3, pp. 353–366, 1989.

[29] S. Helber, F. Sahling, and K. Schimmelpfeng, "Dynamic capacitated lot sizing with random demand and dynamic safety stocks," *OR Spectrum. Quantitative Approaches in Management*, vol. 35, no. 1, pp. 75–105, 2013.

[30] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.