Workload Balancing for Production Planning With Lot Streaming and Multilevel BOM

Weihao Wang, *Student Member, IEEE*, Chutong Gao, *Student Member, IEEE*, and Leyuan Shi, *Fellow, IEEE*

***Abstract*— This article addresses a real-world tactical production planning problem in which a series of real-world constraints need to be considered, such as no backorder, products with multilevel bills of material (BOMs), and lot streaming production. Under the premise of no backorder, the objective of this plan is to make the workload as balanced as possible throughout the planning horizon. An integer quadratic programming model is first proposed to formulate this problem. Then, based on the analysis of the optimal solution for a common BOM structure, this problem is reformulated to a simplified problem where only one item in BOM needs to be considered. Some optimality properties are further derived to help solve this problem. An enhanced variable neighborhood search algorithm is developed to solve this problem, and a lower bound is put forward to measure the performance of the algorithm. Experimental results show that this algorithm can obtain high-quality solutions in a short time.**

***Note to Practitioners*—Effective and intelligent decisions about production planning could have a significant influence on the performance of the manufacturer. This article is motivated by a tactical production planning problem that we encounter in a factory producing discrete equipment. A production plan needs to be made with the consideration of various realistic characteristics in this factory. Without delaying the orders, the objective of the plan is to balance the workload throughout the planning horizon. Currently, the planners make the production plan manually based on their experience in Microsoft Excel. Due to the complexity of this problem and the increasing number of orders, it is difficult and burdensome for planners to make a high-quality production plan manually. In this article, a mathematical model is proposed to formulate this problem, and some theoretical properties are derived to help solve this problem. An enhanced variable neighborhood search-based algorithm is proposed to solve large-scale problems, which could provide a more detailed plan than the current plan. This study could ease the planner from burdensome planning work, improve the quality of the production plan significantly, and be adapted to solve planning problems with similar production characteristics and objectives. In the future, more production characteristics will be considered to meet the diversified demands of production planning.**

***Index Terms*— Lot streaming, production planning, variable neighborhood search (VNS), workload balancing (WB).**

Manuscript received May 6, 2020; revised June 30, 2020; accepted August 17, 2020. Date of publication September 4, 2020; date of current version January 6, 2021. This article was recommended for publication by Editor J. Li upon evaluation of the reviewers' comments. This work was supported by the National Science Foundation of China under Grant 71690232. This article was presented in part at the IEEE Conference on Automation Science and Engineering (CASE) 2019. *(Corresponding author: Leyuan Shi.)*

Weihao Wang is with the Department of Industrial Engineering and Management, Peking University, Beijing 100871, China (e-mail: wangweihao@pku.edu.cn).

Chutong Gao is with the Department of Mechanics and Engineering Science, Peking University, Beijing 100871, China (e-mail: oliverpku@pku.edu.cn).

Leyuan Shi is with the Department of Industrial and Systems Engineering, University of Wisconsin–Madison, Madison, WI 53706 USA (e-mail: leyuan@engr.wisc.edu).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2020.3018556

I. Introduction

PLANNERS should take various production characteristics and management requirements of the factory into consideration when making a production plan. In this study, we deal with a tactical production planning problem in which an annual production plan needs to be made with the objective of workload balancing (WB) in a complex production scenario. This problem is derived from a factory producing discrete high-end equipment in China. This study could free the planners from the burdensome planning work and improve the quality of the production plan significantly.

A critical production management technique adopted by this factory is the lot streaming. Especially, the demand quantity for each item in each order is split into several sublots for production into consecutive time periods. Once one sublot is finished, it could be used to produce its parent item in the next time period immediately instead of waiting for the completion of the entire lot. This allows the overlapping production of items at different levels in BOM and reduces the production makespan and work-in-process inventory as a result [1]. The number of sublots for each kind of item in each order is specified by the planner based on the availability of the production site and materials. An illustrative example of lot streaming is further provided in Section II.

Products produced in this factory have multilevel bills of materials (BOMs) that illustrate the product structures and the relationships of items. In each time period, the number of parent items that can be produced is limited by the number of its child items in inventory. All products in this factory share the same BOM structure. However, due to some minor differences among the same type of items for different orders, items of different orders cannot be shared, i.e., no common item for different orders exists.

Under such circumstances, given the orders with respective demand quantities and due dates, a production plan needs to be made to specify the production quantity and time of each kind of item for each order over a planning horizon with multiple periods. In other words, the output of the plan is the magnitude and time of each sublot. On the premise that all orders are completed no later than their respective due dates, the objective of this plan is to make the production workload of each kind of item as balanced as possible throughout the planning horizon. This intends to avoid the uneven busyness of the workforce, which would benefit the product quality, workforce management, equipment maintenance, and supply of raw material a lot. In short, the production planning problem in this study could be concluded as a multilevel lot streaming production planning problem with the objective of WB (MLPP).

Some studies have paid attention to similar problems, such as production smoothing problem (PSP), assembly line balancing problem (ALBP), and WB on parallel machine scheduling problems. Production smoothing is a key component of Just-in-Time (JIT) philosophy in the Toyota Production System. The ideal smoothing production plan in JIT expects a steady production rate for each kind of product so that the cumulative production amount of a product increases linearly [2]. The studies of PSP, such as [3]–[5], try to find a production sequence of different final products to reduce the deviation of the actual production schedule from the ideal one. Although PSP also focuses on the steady product output, it is the production sequence of final products that are decided in PSP, and only one product is produced in each stage, whereas, in our problem, a number of products are produced in each time period. We need to decide the production quantity of each item for each order in each time period.

As for the ALBP especially the type-II ALBP, the studies, such as [6]–[8], seek a balanced assignment of operations with different processing times to a given number of workstations along the production line so that certain restrictions (such as precedence constraints) are fulfilled, and a stable workflow with a short cycle time is achieved. Readers could refer to [9] and [10] for related reviews. Compared with ALBP, MLPP ignores the detailed operations of each item and allocates the demand quantity to different time periods to obtain a balanced production workload across the planning horizon.

In the WB on parallel machine scheduling problem, a series of jobs with different processing times need to be scheduled to some parallel machines in order to minimize the dispersion of the completion time of each machine around their average value [11]. Ho *et al.* [11] introduce a square-error-based criterion to measure the workload balance and propose a WB algorithm to solve it. Some studies, such as [12]–[15], put forward some other interchange and local search algorithms to solve this problem. Christ *et al.* [16] propose an iterated min–max procedure for a min–max fairness WB problem on nonidentical parallel machines. In fact, this problem intends to assign a series of numbers into several containers with minimum load difference, whereas, in MLPP, both the division of demand quantities into sublots and the allocation of sublots to time periods need to be decided. Besides, some essential features in the factory that we study, including the lot streaming and BOM hierarchy, should also be considered, which makes it a different problem from the aforementioned problems.

As for the lot streaming, one of the main production characteristics in this problem, it is typically considered with various scheduling problems together, such as blocking flow shop scheduling problem [17], job shop scheduling problem [18], and supply chain scheduling [19]. The magnitude of sublots and other decisions are made to achieve certain objectives. More studies about lot streaming can refer to the survey article [20]. However, none of the studies have paid attention to the lot streaming optimization with the objective of WB.

To the best of our knowledge, despite the abovementioned vast studies, none of the research except our previous work [21] addresses such a production planning problem with so many production characteristics and the objective of workload balance. However, the previous model contains numerous variables and intricate constraints, little theoretical analysis is conducted, and the performance of the proposed algorithm is difficult to evaluate in our previous work. Hence, this problem needs further detailed study.

In this article, we first propose a mathematical programming model for this problem. Then, the reformulation and some optimal properties are derived through the theoretical analysis. We further propose an enhanced variable neighborhood search (EVNS) algorithm to solve this problem. Numerical results show that the proposed algorithm can obtain high-quality solutions efficiently.

The main contributions of this study are concluded as follows.

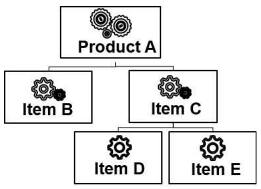
1) As far as we know, MLPP is novel and has never been studied before. We are the first to study such a real-world problem. An integer quadratic programming (IQP) model is proposed to formulate this problem.
2) By exploring the structure of the optimal solutions for a common BOM structure, we reformulate the model with multilevel BOM to a simplified problem where only one item in BOM needs to be considered. Some optimality properties are further derived to help solve the problem.
3) We propose a new local search method based on the features and optimal properties of the problem. This local search method is implemented in the VNS algorithm to enhance its effectiveness in solving MLPP.
4) A lower bound (LB) and a polynomial-time exact algorithm for it is put forward to measure the performance of the EVNS algorithm.

The remainder of this article is organized as follows. Section II introduces the formulation of MLPP. The reformulation of MLPP and some optimality properties are further discussed in Section III. In Section IV, we propose an EVNS algorithm with a local search method to solve this problem. An LB of this problem and the algorithm for it are introduced in Section V. Section VI presents the design and results of computational experiments. The conclusion and future directions are discussed in Section VII.

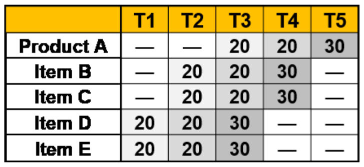
II. Problem Description and Formulation

The description and formulation of MLPP are presented in this section.

In this problem, a total of n orders need to be completed within a T-period planning horizon. Each order $i \in N$ has O_i identical products to be produced and assembled according to the multilevel product BOM. Products of different orders



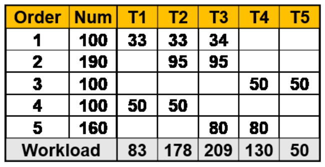
(a)



	T1	T2	T3	T4	T5
Product A	—	—	20	20	30
Item B	—	20	20	30	—
Item C	—	20	20	30	—
Item D	20	20	30	—	—
Item E	20	20	30	—	—

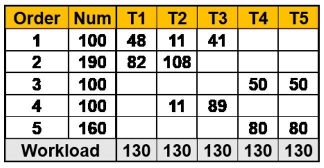
(b)

Fig. 1. Illustration of the lot streaming. (a) BOM of product A. (b) Plan for one order of product A.



Order	Num	T1	T2	T3	T4	T5
1	100	33	33	34		
2	190		95	95		
3	100				50	50
4	100	50	50			
5	160			80	80	
Workload		83	178	209	130	50

(a)



Order	Num	T1	T2	T3	T4	T5
1	100	48	11	41		
2	190	82	108			
3	100				50	50
4	100		11	89		
5	160				80	80
Workload		130	130	130	130	130

(b)

Fig. 2. Illustration of the WB. (a) Production plan for one item before optimization. (b) Production plan for that item after optimization.

share the same BOM structure, but items produced for one order cannot be used by another order. Each order $i \in N$ should be completed no later than its due date d_i, and no tardiness is allowed. The lot streaming is adopted in this factory. Especially, the demand quantity O_{ij} for item j in order i should be split into S_i sublots to be produced in consecutive S_i time periods. In this problem, we need to decide the size and the production time of each sublot x_{ijt} for each kind of item in each order. The objective is to make the production workload $w_{jt} = \sum_{i \in N} x_{ijt}$ of each kind of item $j \in M$ in each time period t as close as possible to the average value $\overline{w_j}$ throughout its production time horizon H_j. The lot streaming and the objective of WB are further explained with the following illustrative examples.

An illustrative example of the lot streaming is shown in Fig. 1. Fig. 1(a) shows the BOM of Product A in which each item requires one unit of each type of its child items. Here, we consider an order for 70 Product A, which requires 70 units of each kind of item according to the BOM. A plan for this order using the lot streaming is shown in Fig. 1(b). In this plan, the lot of each kind of item is split into three sublots that are produced in three consecutive time periods. The first sublots of items D and E at the bottom of the BOM are first produced in T1. Then, their parent item C can start production in T2 without having to wait for all the items D and E to be completed. The production of other items follows a similar pattern, and therefore, the makespan of this order is reduced.

The objective of the WB in this study is illustrated in Fig. 2 in which two production plans for a kind of item are compared. For each kind of item, the workload in each time period refers to the number of items it produces in this time period, no matter for which orders these items are produced. Thus, in the last row, the workload in each time period is calculated by summing up the planned quantity at that time. As shown in 2(a), the workload fluctuates dramatically across these time periods in this plan. After optimizing the production time and sizes of sublots, the workload could become more balanced, as shown in Fig. 2(b).

In this factory, each kind of item is produced using a dedicated production resource. We assume that the production resources are all idle at the beginning of the planning horizon. The capacity of each production resource is sufficient and can, thus, be regarded as unlimited. Due to the make-to-order (MTO) production mode adopted by this factory, the initial inventory of all items at the beginning of the planning horizon is zero.

The notations used in the formulation of MLPP are listed and explained as follows:

Notations:

1) N is the set of orders, $N = \{1, \ldots, n\}$.
2) d_i is the due date of the order i, $i \in N$.
3) M is the set of items in BOM, $M = \{1, \ldots, m\}$.
4) j and k are the indices of items, $j, k \in M$.
5) T is the number of time periods, $T = \max_{i \in N}\{d_i\}$.
6) H is the planning horizon, $H = \{1, \ldots, T\}$.
7) O_{ij} is the required number of item j in order i.
8) $\Gamma(k)$ is the set of direct child items to produce item k in BOM.
9) c_{jk} is the number of item j required to produce one unit of its direct parent item k, $j \in \Gamma(k)$, $O_{ij} = c_{jk} O_{ik}$.
10) LN is the number of levels in BOM.
11) l_j is the level of item j from bottom to top in BOM.
12) S_i is the number of sublots that the lot of each kind of item in order i should be divided into.
13) ET_j is the earliest production start time of item j, $ET_j = l_j$, which sets aside time for the production of its child items.
14) LT_{ij} is the the latest completion time of item j for order i, $LT_{ij} = d_i - (LN - l_j)$, which sets aside time for the production of its parent items, so that this order can be completed no later than its due date d_i.
15) H_{ij} is the production time horizon of item j for order i, $H_{ij} = [ET_j, LT_{ij}]$.
16) LT_j is the latest completion time of item j, $LT_j = T - (LN - l_j) = \max_{i \in N}\{LT_{ij}\}$.
17) H_j is the production time horizon of item j, $H_j = [ET_j, LT_j]$.
18) $\overline{w_j}$ is the average workload of item j during H_j, $\overline{w_j} = (1/(LT_j - ET_j + 1)) \sum_{i=1}^{n} O_{ij}$.
19) G is a large positive number.

Variables:

1) x_{ijt} is the production number of item j of order i in time t, which also represents the size of this sublot.
2) y_{ijt}: $y_{ijt} = 1$ if item j of order i is produced in time t; otherwise, $y_{ijt} = 0$.
3) z_{ijt}: $z_{ijt} = 1$ if two sublots of item j for order i are produced in consecutive periods t and $t + 1$; otherwise, $z_{ijt} = 0$.
4) I_{ijt} is the amount of initial inventory of item j for order i in time period t.
5) w_{jt} is the production workload, namely, the total production quantity, of item j in time period t.

An illustrative example of some notations for item C in Fig. 1(a) is presented in Fig. 3 to help better understand the parameters and their relationship. We consider a planning

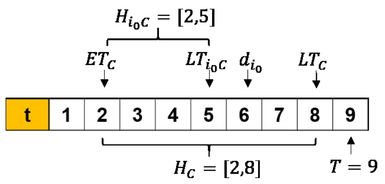


Fig. 3. Illustrative example of the parameters for item C in Fig. 1(a).

horizon with $T = 9$. According to the BOM, the number of levels LN is 3, and the level of item C l_C is 2. Because there are no items D and E available for the production of item C at the beginning, its earliest production start time $ET_C = l_C = 2$ so that the first sublots of its child items D and E could be produced at $t = 1$. Consider an order i_0 that requires 70 product A and should be finished no later than $d_{i_0} = 6$, and the latest completion time of item C for this order LT_{i_0C} is 5 so that the last sublot of item C could be used to produce product A at $t = 6$ without the order being late.

We then use these notations to formulate MLPP. Variance is a commonly used measure for volatility and could penalize both above and below average value, especially the large deviations [11]. Therefore, in this formulation, we use the workload variance $\sum_{t\in H_j}(w_{jt} - \overline{w_j})^2$ of item j in its production time horizon H_j to measure its WB. Thus, the objective function here is to minimize the sum of the workload variances of all items

$$\min \sum_{j\in M}\sum_{t\in H_j}(w_{jt} - \overline{w_j})^2 \tag{1}$$

where the workload of each item in each period is calculated as

$$w_{jt} = \sum_{i\in N} x_{ijt} \quad \forall\ j \in M,\ t \in H. \tag{2}$$

Now, turn to the constraints that the solution should satisfy. Due dates of orders should be met strictly. Thus, the demand constraints ensure that the demand quantity of each type of item in each order should be met exactly within its production time horizon, and no extra items are produced outside the production time horizon

$$\sum_{t\in H_{ij}} x_{ijt} = O_{ij} \quad \forall\ i \in N,\ j \in M \tag{3}$$

$$x_{ijt} = 0 \quad \forall\ i \in N,\ j \in M,\ t \in H \setminus H_{ij}. \tag{4}$$

The sublot number constraints ensure that the number of those sublots divided from one lot of items in one order should be equal to the specified number

$$y_{ijt} \le x_{ijt} \le G \cdot y_{ijt} \quad \forall\ i \in N,\ j \in M,\ t \in H \tag{5}$$

$$\sum_{t\in H_{ij}} y_{ijt} = S_i \quad \forall\ i \in N,\ j \in M. \tag{6}$$

The continuous production constraints are as follows:

$$(y_{ijt} + y_{ij(t+1)}) - 1 \le z_{ijt} \le 0.5(y_{ijt} + y_{ij(t+1)}) \quad \forall\ i \in N,\quad j \in M \quad t \in [ET_j, LT_{ij} - 1] \tag{7}$$

$$\sum_{t\in H_{ij}} z_{ijt} = S_i - 1 \quad \forall\ i \in N,\ j \in M \tag{8}$$

where constraints (7) ensure that $z_{ijt} = 1$ if sublots of item j for order i are produced in time t and $t+1$ ($y_{ijt} = y_{ij(t+1)} = 1$). Constraints (8) ensure that those sublots divided from one lot are produced in consecutive time periods.

The following constraints are BOM-related constraints. The number of items that could be produced is limited by the inventory number of its child items in that time period

$$I_{ijt} \ge c_{jk}x_{ikt} \quad \forall\ i \in N,\ \ j, k \in M,\ \ j \in \Gamma(k),\ \ t \in H \tag{9}$$

in which the inventory quantity of each kind of item for each order at the beginning of each time period equals to the inventory quantity plus the production quantity and minus the consumed quantity for the production of its parent items in the last time period

$$I_{ij(t+1)} = I_{ijt} + x_{ijt} - c_{jk}x_{ikt} \quad \forall\ i \in N,\ \ j, k \in M \quad j \in \Gamma(k),\quad t \in [1, T-1]. \tag{10}$$

Constraints (10) also imply that the items for one order cannot be used to produce other orders. The initial inventory of each item for each order is empty

$$I_{ij1} = 0 \quad \forall\ i \in N,\ \ j \in M. \tag{11}$$

The types and ranges of variables are illustrated as follows:

$$y_{ijt}, z_{ijt} \in \{0, 1\} \quad \forall\ i \in N,\ \ j \in M,\ \ t \in H \tag{12}$$

$$x_{ijt}, I_{ijt}, w_{jt} \in \mathbb{Z}^+ \quad \forall\ i \in N,\ \ j \in M,\ \ t \in H. \tag{13}$$

With the objective function (1) and constraints (2)–(13), we obtain the formulation of MLPP.

III. Reformulation and Optimality Properties

In this study, we deal with the case where $c_{jk} = 1 \forall j, k \in M$, $j \in \Gamma(k)$, which means that the production of each unit of parent item k requires only one unit of each kind of its child item j. This is common in many product BOM structures. For the case where $c_{jk} > 1$, the items' relationship could be transformed into the case $c_{jk} = 1$ by replacing the node representing c_{jk} item j with c_{jk} nodes of item j in BOM, and each of them has the relation $c_{jk} = 1$ with its parent item k. Thus, this study is still in line with the situation where $c_{jk} > 1$.

In this section, we first propose a theorem that transforms MLPP with $c_{jk} = 1$ to a single-item lot streaming production planning problem with the objective of WB (SLPP). Then, we derive some optimality properties to help solve this problem.

A. Reformulation for MLPP With $c_{jk} = 1$

For arbitrary $i \in N, j, k \in M, j \in \Gamma(k)$, if $c_{jk} = 1$, we derive that $O_{ij} = O_{ik}$ and, thus, define $O_i := O_{ij}$, $\forall j \in M$. Furthermore, because H_j for any item $j \in M$ has the same length $LT_j - ET_j + 1 = T - LN + 1$, we have $\overline{w_j} = (1/(LT_j - ET_j + 1))\sum_{i=1}^{n} O_{ij} = (1/(T - LN + 1))\sum_{i=1}^{n} O_i$, which is irrelevant to j. Thus, we define $\overline{w} := \overline{w_j}$, $\forall j \in M$.

In this case, we pay attention to a subproblem in which only an arbitrary item $j \in M$ is considered. We intend to minimize the workload variance of this item across its planning horizon

H_j in ignorance of the BOM-related constraints. This problem is denoted as SLPP for item j (SLPP-j).

Some new notations derived from Section III are introduced for SLPP-j. By omitting those time periods outside H_{ij} in MLPP, we denote the production time horizon of item j for order i in SLPP-j as $H_{\text{sub}_i} := \{1, 2, \ldots, d_{\text{sub}_i}\}$. $d_{\text{sub}_i} = d_i - LN + 1$ represents the latest completion time of item j for order i and is still called due date in SLPP-j for simplicity. H_{sub_i} is a mapping of H_{ij} in MLPP and has the same length for any $j \in M$. Similarly, we denote $H_{\text{sub}} := [1, 2, \ldots, T - LN + 1]$ that corresponds to H_j in MLPP. Let x_{ir} represent the production number of the selected item j for order i in time period $r \in H_{\text{sub}}$, which equals to x_{ijt} for $t = r - 1 + ET_j$ in H_j. Then, w_r represents its total workload in time r. Similar to MLPP, if $x_{ir} > 0$, we set $y_{ir} = 1$; otherwise, $y_{ir} = 0$. If $y_{ir} = 1$ and $y_{i(r+1)} = 1$, $z_{ir} = 1$; otherwise, $z_{ir} = 0$.

Therefore, SLPP-j can be formulated by simplifying the variables and removing constraints (9)–(11) that are related to BOM and inventory of child items in MLPP. The objective function is the workload variance of item j across H_{sub}

$$\min \sum_{r \in H_{\text{sub}}} (w_r - \overline{w})^2 \tag{14}$$

where the workload is calculated as

$$w_r = \sum_{i=1}^{n} x_{ir} \quad \forall r \in H_{\text{sub}}. \tag{15}$$

The demand constraints ensure that the demand quantity of each order should be met exactly within their respective production time horizons

$$\sum_{r \in H_{\text{sub}_i}} x_{ir} = O_i \quad \forall\, i \in N \tag{16}$$

$$x_{ir} = 0 \quad \forall i \in N,\ r \in H_{\text{sub}} \setminus H_{\text{sub}_i}. \tag{17}$$

Similar to MLPP, the sublot number constraints are as follows:

$$y_{ir} \le x_{ir} \le G \cdot y_{ir} \quad \forall\, i \in N,\ r \in H_{\text{sub}} \tag{18}$$

$$\sum_{r \in H_{\text{sub}_i}} y_{ir} = S_i \quad \forall\, i \in N. \tag{19}$$

The continuous production constraints are as follows:

$$(y_{ir} + y_{i(r+1)}) - 1 \le z_{ir} \le 0.5(y_{ir} + y_{i(r+1)}) \quad \forall\, i \in N,\ r \in H_{\text{sub}_i} \tag{20}$$

$$\sum_{r \in H_{\text{sub}_i}} z_{ir} = S_i - 1 \quad \forall\, i \in N. \tag{21}$$

The types and ranges of variables are

$$y_{ir}, z_{ir} \in \{0, 1\} \quad \forall\, i \in N,\ k \in H_{\text{sub}} \tag{22}$$

$$x_{ir}, w_r \in \mathbb{Z}^+ \quad \forall\, i \in N,\ r \in H_{\text{sub}}. \tag{23}$$

With objective function (14) and constraints (15)–(23), we obtain the formulation of SLPP-j. We further denote the solution of SLPP-j as $s_j = \{x_{ir} | i \in N, r \in H_{\text{sub}}\}$ and its objective function value as $p_j = p_j(s_j)$. Similarly, we denote the solution of MLPP as $s = \{x_{ijt}, i \in N, j \in M, t \in H\}$ and its objective function value as $p = p(s)$. Based on this formulation, we derive the relationship of optimal solutions between SLPP-j and MLPP in the following theorem.

Theorem 1: If $c_{jk} = 1 \forall j, k \in M \forall j \in \Gamma(k)$, there exists an optimal solution of MLPP in which the production plan of item k is identical to the plan of its direct child item j with only 1 time period behind, i.e., for $\forall j, k \in M$, $j \in \Gamma(k)$, $t \in H_j$, $x^*_{ik(t+1)} = x^*_{ijt}$.

Proof: We explore the relationship between the solution s_j of SLPP-j and s of MLPP. We first define function $f_s : \mathbb{Z}^{nm(T-LN+1)} \to \mathbb{Z}^{nmT}$ to map the solution of SLPP-j, $j = 1, \ldots, m$, to that of MLPP. In this mapping, the relation of the specific production quantity of each sublot in solutions $x_{ijt} \in s$ and $x_{ir} \in s_j$ for $\forall i \in N, j \in M, r = t + 1 - ET_j$ is given as follows:

$$x_{ijt} = \begin{cases} x_{ir}, & t \in H_j \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

Following (24), we can use the solutions $s_1, s_2, \ldots, s_m$ of SLPP-j for items $j = 1, \ldots, m$ to generate a solution s' of MLPP as $s' = f_s(s_1, s_2, \ldots, s_m)$. Then, s' satisfies all constraints of MLPP except constraints (9)–(11) that reflect the requirement of BOM and are ignored in SLPP-j. Then, we could obtain the relation of their objective values

$$p(s') = \sum_{j=1}^{m} p_j(s_j), \quad \text{where } s' = f_s(s_1, \ldots, s_m). \tag{25}$$

By examining the structure of SLPP-j, we find that the objective function, variables, and constraints of SLPP-j are all irrelevant to j. In other words, the structure of SLPP-j is identical for all items. Thus, there exists optimal solutions $s_1^* = \cdots = s_m^* = s_{\text{sub}}^*$ with

$$p_1^* = \cdots = p_m^* = p_{\text{sub}}^*. \tag{26}$$

Thus, we could further obtain

$$m \cdot p_{\text{sub}}^* = \sum_{j=1}^{m} p_j^* = \sum_{j=1}^{m} \min_{s_j} p_j(s_j) = \min_{s_1,\ldots,s_m} \sum_{j=1}^{m} p_j(s_j). \tag{27}$$

As mentioned earlier, the solution $s' = f_s(s_1, s_2, \ldots, s_m)$ satisfies all the constraints of MLPP except constraints (9)–(11). Therefore, it is a feasible solution for a relaxation of MLPP. As a result, $\min_{s'} p(s')$ is an LB for MLPP. Then, together with (25), we could obtain

$$\min_{s_1,\ldots,s_m} \sum_{j=1}^{m} p_j(s_j) = \min_{s'} p(s') \le \min_{s} p(s) = p^*. \tag{28}$$

On the other hand, we denote s'^* as a solution of MLPP generated from $(s_1^*, \ldots, s_m^*)$, i.e., $s'^* = f_s(s_1^*, \ldots, s_m^*)$, where $s_1^* = \cdots = s_m^* = s_{\text{sub}}^*$. According to the definition of production time horizon in MLPP and the BOM, the H_k of item k has the same length with the H_j of its child item j with 1 time period behind in MLPP, namely, $ET_k = ET_j + 1$. Thus, in the solution s'^*, the parent item k's production plan is identical to that of its direct child item j with only one time period behind. Obtained in this way, s'^* satisfies

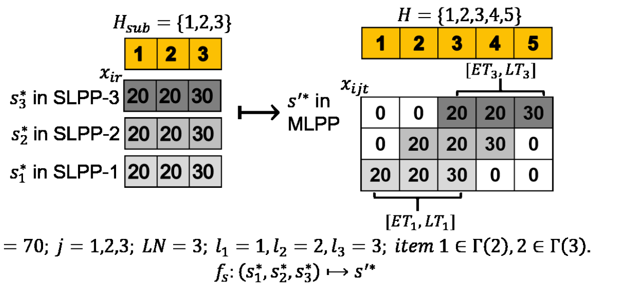

$H_{sub} = \{1,2,3\}$
1 2 3
x_{ir}
s_3^* in SLPP-3: 20 20 30
s_2^* in SLPP-2: 20 20 30
s_1^* in SLPP-1: 20 20 30
s'^* in MLPP
$H = \{1,2,3,4,5\}$
1 2 3 4 5
$[ET_3, LT_3]$
x_{ijt}
0 0 20 20 30
0 20 20 30 0
20 20 30 0 0
$[ET_1, LT_1]$
$O_i = 70$; $j = 1,2,3$; $LN = 3$; $l_1 = 1, l_2 = 2, l_3 = 3$; item $1 \in \Gamma(2), 2 \in \Gamma(3)$.
$f_S: (s_1^*, s_2^*, s_3^*) \mapsto s'^*$

Fig. 4. Example of using solutions of SLPP-j to generate a solution of MLPP with f_s.

constraints (9)–(11) automatically, as illustrated in Fig. 4. Thus, s'^* is a feasible solution of MLPP. Together with (25) and (26), we have

$$p^* \le p(s'^*) = \sum_{j=1}^{m} p_j(s_j^*) = \sum_{j=1}^{m} p_j^* = m \cdot p_{\text{sub}}^*. \tag{29}$$

From (27) to (29), we can derive that $m \cdot p_{\text{sub}}^* = p^*$. This means that the solution of MLPP that we obtain by solving SLPP-j first and adapting its solution to all items according to the way mentioned in Theorem 1 is optimal. This completes Proof of Theorem 1. ■

Theorem 1 simplifies MLPP in which all the items in BOM need to be considered to SLPP-j where only one item needs to be considered. By solving SLPP-j, we obtain an optimal solution s_{sub}^* with objective value p_{sub}^*. Then, according to Theorem 1, we could use this solution to generate an optimal solution s^* with the objective value $p^* = m \cdot p_{\text{sub}}^*$ for MLPP. Thus, SLPP-j is the reformulation for MLPP with $c_{jk} = 1$. Because SLPP-j is actually the same no matter which item it focuses on, for simplicity, we denote SLPP-j as SLPP, its solution s_j as s, and its objective function value $p_j(s_j)$ as $p(s)$ without risk of ambiguity hereafter.

The total number of variables for MLPP with n orders, m items, LN levels of BOM, and T time periods is $(4n+1)mT$, including $2nmT$ binary variables, whereas there are only $(3n+1)\cdot(T-LN+1)$ variables, including $2n(T-LN+1)$ binary variables for SLPP, and a lot of constraints are left out.

As for the complexity of SLPP, if we think of the time periods in H_{sub} as identical parallel machines, the orders N with different demand quantities as the jobs with different processing times, and the workload of each time period as the completion time of each machine, then SLPP with $S_i = 1, H_{\text{sub}_i} = H_{\text{sub}}$ for any order $i \in N$ is equivalent to the WB on parallel machines problem, which is mentioned in Section II and has been proven to be strongly NP-hard by [14]. Thus, SLPP is also strongly NP-hard.

B. Optimality Properties of SLPP

Before we further explore other optimality properties of SLPP, we introduce the changing relationship between the workload and objective function value in Lemma 1.

Lemma 1: For any workload w_{r_1} and w_{r_2}, $r_1, r_2 \in H_{\text{sub}}$, if $|w_{r_2} - w_{r_1}|$ becomes smaller and the workloads at other time periods are unchanged, then $p(s) = \sum_{r\in H_{\text{sub}}}(w_r - \overline{w})^2$ becomes smaller. Similarly, as the former becomes bigger, the latter will become bigger too.

Proof: Without loss of generality, we assume that $w_{r_1} \le w_{r_2}$. Let $w_{r_2} - w_{r_1} = \text{dev}$, and $w_{r_2} + w_{r_1} = C$. Then, we have $w_{r_1} = ((C - \text{dev})/2)$, $w_{r_2} = ((C + \text{dev})/2)$, and

$$\begin{aligned} p(s) &= \left(\sum_{r\neq r_1,r_2}(w_r - \overline{w})^2\right) + (w_{r_2} - \overline{w})^2 + (w_{r_1} - \overline{w})^2 \\ &= \left(\sum_{r\neq r_1,r_2}(w_r - \overline{w})^2\right) + 2\overline{w}^2 - 2\overline{w}C + \frac{1}{2}(C^2 + \text{dev}^2). \end{aligned} \tag{30}$$

Because C does not change in this process, if some quantity is transferred from time period r_2 to r_1 so that $\text{dev} = |w_{r_2} - w_{r_1}|$ becomes smaller, we can derive from (30) that $p(s)$ becomes smaller. Thus, this lemma is proven. ■

In order to explore the optimality property of SLPP, we first consider a sufficiently relaxed problem of SLPP called RSLPP1 and introduce the optimality properties of this relaxed problem in Lemma 2. This helps us to develop the optimality property of SLPP in Theorem 2.

In RSLPP1, we only require that all demands are satisfied eventually. All the other constraints, including the due dates of orders, are relaxed. The decision variable is the workload w_r itself in each time period r, and the object is still to reduce the workload variance. The formulation of RSLPP1 is as follows:

$$\min\ p(w) = \sum_{r\in H_{\text{sub}}}(w_r - \overline{w})^2 \tag{31}$$

$$\text{s.t.}\ \sum_{r\in H_{\text{sub}}} w_r = \sum_{i\in N} O_i \tag{32}$$

$$w_r \in \mathbb{Z}^+ \quad \forall r \in H_{\text{sub}}. \tag{33}$$

We define $w_{\text{inf}} = \min_{w_r\in w}\{w_r\}$ and $w_{\text{sup}} = \max_{w_r\in w}\{w_r\}$. The workload deviation $\text{dev}(w)$ of the solution is calculated as $\text{dev}(w) = w_{\text{sup}} - w_{\text{inf}}$. Based on the formulation, we derive the optimality properties of this relaxed problem in Lemma 2:

Lemma 2: For RSLPP1, the following holds.

1) The deviation of an optimal solution w^* is less than or equal to 1.
2) Any feasible solution whose deviation is less than or equal to 1 is an optimal solution.

Proof: We prove 1) by contradiction. Suppose that there exists an optimal w^* with a deviation more than 1. We set $w_{\text{inf}}^* \leftarrow w_{\text{inf}}^* + \lfloor(\text{dev}(w^*)/2)\rfloor$, $w_{\text{sup}}^* \leftarrow w_{\text{sup}}^* - \lfloor(\text{dev}(w^*)/2)\rfloor$ to generate a new solution $w^{*'}$. This solution still satisfies constraints (32) and (33) and, therefore, is feasible. If its deviation $\text{dev}(w^{*'})$ remains unchanged from w^* (which means there exists more than one w_{inf}^* and one $w_{\text{sup}}^* \in w^*$), repeat this process until the deviation becomes smaller. From Lemma 1, we know that once the deviation $\text{dev}(w^{*'})$ becomes smaller, the objective value $p(w^{*'})$ also becomes smaller, i.e., $p(w^{*'}) < p(w^*)$, which means that w^* is not optimal. This contradiction shows that the deviation of an optimal solution of RSLPP1 must be less than or equal to 1.

Next, we prove 2). Suppose that w^* is an optimal solution. From 1), its deviation is less than or equal to 1. For any feasible solution w whose deviation is less than or equal to 1, $w_r \in w$ can only be selected from $\lfloor \overline{w} \rfloor$ and $\lceil \overline{w} \rceil$. Suppose that w has α periods at which w_r equals to $\lfloor \overline{w} \rfloor$. α can be calculated by the following equation:

$$\alpha \cdot \lfloor \overline{w} \rfloor + [(T - LN + 1) - \alpha] \cdot \lceil \overline{w} \rceil = (T - LN + 1) \cdot \overline{w}. \tag{34}$$

Thus, $p(w) = \alpha \cdot (\lfloor \overline{w} \rfloor - \overline{w})^2 + [(T - LN + 1) - \alpha] \cdot (\lceil \overline{w} \rceil - \overline{w})^2$, which remains the same for any w with a deviation less than 2, including the optimal w^*. Thus, $p(w) = p(w^*)$ and w is also optimal. Proof of lemma 2 is completed. ■

Based on Lemma 2, we further derive an optimality property of SLPP in Theorem 2.

Theorem 2: If there exists a feasible solution s of SLPP subject to $\text{dev}(s) \leq 1$, then s is an optimal solution of SLPP.

Proof: Denote the optimal objective function value of RSLPP1 as p_1^* and the value of SLPP as p_2^*. Because RSLPP1 is a relaxation of SLPP, we have $p_1^* \leq p_2^*$. As s is a feasible solution of SLPP, we could further obtain $p_2^* \leq p(s)$. If the deviation of solution s is less than 2, according to Lemma 2, s is also an optimal solution of RSLPP1 and $p(s) = p_1^*$. From the above, we could obtain

$$p(s) = p_1^* \leq p_2^* \leq p(s). \tag{35}$$

Thus, we know $p(s) = p_2^*$, which that means s is an optimal solution of SLPP. ■

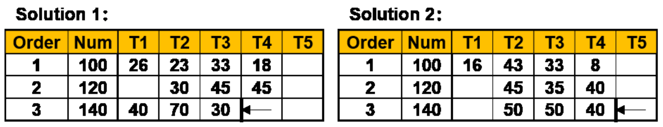
Note, however, that a solution with $\text{dev}(s) > 1$ could also be optimal. This happens in the cases where the due dates of orders are not evenly spread across the planning horizon, e.g., the due dates are clustered in the early phase of the planning horizon. The optimal workload distributions in these cases are stair-shaped with a large deviation from the mean value.

IV. Enhanced Variable Neighborhood Search Algorithm for SLPP

The variable neighborhood search (VNS) algorithm is a metaheuristic algorithm proposed by Mladenovic and Hansen [22]. Its effectiveness has been demonstrated in solving many complex optimization problems, such as flow shop scheduling problem [23], traveling salesman problem [24], lot-sizing problem [25], vehicle routing problem [26], and airline crew rostering problem [27]. Compared with most local search-based heuristic algorithms in which only one neighborhood is fathomed, VNS systematically explores different neighborhoods with a predefined strategy [25]. In this section, we first introduce the design of the neighborhood structure for SLPP and the basic scheme of VNS. Then, a local search method is proposed to enhance the search efficiency of VNS. Finally, we will introduce the EVNS algorithm combined with the local search method that we propose in detail.

A. Neighborhood Structure for SLPP

The design of the neighborhood structure is the basis of the VNS algorithm. We adopt a distance-based neighborhood structure based on the characteristic of the solution of SLPP.


Solution 1:

Order	Num	T1	T2	T3	T4	T5
1	100	26	23	33	18	
2	120		30	45	45	
3	140	40	70	30	←	

Solution 2:

Order	Num	T1	T2	T3	T4	T5
1	100	16	43	33	8	
2	120		45	35	40	
3	140		50	50	40	←

Fig. 5. Example of the neighborhood structure.

Assume that S is a set of feasible solutions for SLPP. We define the distance $D(s, s')$ between two solutions $s, s' \in S$ as the number of orders that have different completion times regardless of the specific production number of each sublot. For instance, the distance between the two solutions in Fig. 5 is 1 because one of the orders (order 3) has a different completion time. The neighborhoods of the incumbent solution from near to far are different sets of feasible solutions that have small to large distances, respectively. A solution s' belongs to the kth neighborhood of incumbent solution s if $D(s, s') = k, k \in \mathbb{Z}^+$.

B. Basic Scheme of VNS

At the beginning of the VNS algorithm, the neighborhood structure N_k, $k = 1, \ldots, K_{\max}$ should be defined, and the exploring sequence and stopping conditions should be specified. An initial solution needs to be generated first as the incumbent solution for the subsequent exploration. Then, a neighborhood search loop is repeated iteratively in which neighborhoods of the incumbent solution are explored from near too far. The better solution found in this process is accepted as the new incumbent solution. Specifically, in a neighborhood search loop for the incumbent solution s, the nearest neighborhood $N_1(s)$ with $k \leftarrow 1$ is explored first. If the number of continuous nonimprovement search $N_{\max}$ within this neighborhood is reached, the VNS algorithm moves to a farther neighborhood with $k \leftarrow k + 1$ until $k = K_{\max}$. If a better solution s' is found in this process, it is accepted as the new incumbent solution and the search loop starts from its nearest neighborhood $N_1(s')$ once again by setting $k \leftarrow 1$.

C. Local Search Method: Maximum Deviation Transfer

In this section, we propose a local search method called maximum deviation transfer (MDT) method to search for better solutions in the neighborhoods. Starting with a solution s' obtained within the kth neighborhood $N_k(s)$ of the incumbent solution s, the MDT method tries to find a better solution s'' in this neighborhood by adjusting the sizes of sublots in solution s' without altering their production time. The main idea of MDT is transferring a number of items of some order from a time period in which the workload has the largest deviation from the mean value to a time period with lower one iteratively without violating the constraints.

Specifically, for an initial solution s', the time r_d in which the workload has the maximum deviation from the average workload $\overline{w}$ is identified first. A certain amount needs to be transferred from this time period to another time period to balance the workload. It is worth noting that because of the constraint that items of an order cannot be shared by another order, the quantity transfer could only be conducted among

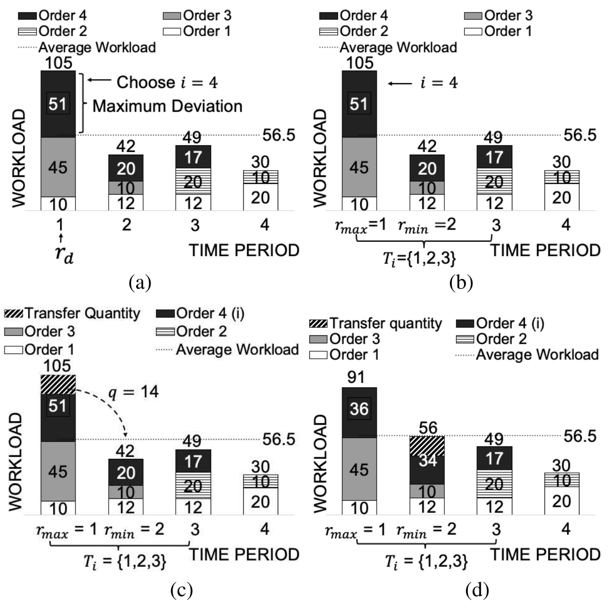


Fig. 6. Illustrative example of the MDT algorithm. (a) Choose time period r_d with maximum deviation and randomly select order i at r_d. (b) Decide $r_{\max}$ and $r_{\min}$ within T_i. (c) Randomly generate the transfer quantity q in time period $r_{\max}$. (d) Transfer q in order i from $r_{\max}$ to $r_{\min}$.

Algorithm 1 MDT Algorithm

```
1:  Input: $L_{max}$, initial solution $\mathbf{s}'$
2:  $l \leftarrow 0$
3:  while $l \leq L_{max}$ do
4:      $r_d \leftarrow \arg\max_{r \in \mathbf{H_{sub}}}\{|w_r - \overline{w}|\}$
5:      Randomly select $i \in \{i|x_{ir_d>0}\}$; $\mathbf{T_i} \leftarrow \{r|x_{ir} > 0\}$
6:      if $w_{r_d} < \overline{w}$ then
7:          $r_{min} \leftarrow r_d$; $r_{max} \leftarrow \arg\max_{r \in \mathbf{T_i}}\{w_r\}$;
8:      else
9:          $r_{max} \leftarrow r_d$; $r_{min} \leftarrow \arg\min_{r \in \mathbf{T_i}}\{w_r\}$;
10:     end if
11:     $up_1 \leftarrow w_{r_{max}} - w_{r_{min}}$; $up_2 \leftarrow x_{ir_{max}} - 1$; randomly
            select an integer $q \in [0, \min\{up_1, up_2\}]$;
12:     $x_{ir_{min}} \leftarrow x_{ir_{min}} + q$; $x_{ir_{max}} \leftarrow x_{ir_{max}} - q$ to obtain $\mathbf{s}''$;
            $\mathbf{s}' \leftarrow \mathbf{s}''$; $l \leftarrow l + 1$;
13: end while
14: Output: $\mathbf{s}'$
```

Algorithm 2 EVNS Algorithm

```
1:  Input: $N_{max}, K_{max}, P, TL$
2:  $p \leftarrow 1$; $t \leftarrow 0$; generate a feasible solution by RTAQ as
        $\mathbf{s_{best}}$;
3:  while $p \leq P$ and $t \leq TL$ do
4:      Generate a solution $\mathbf{s}$ with RTAQ; $k \leftarrow 1$;
5:      while $k \leq K_{max}$ do
6:          $n \leftarrow 1$;
7:          while $n \leq N_{max}$ do
8:              Generate a solution $\mathbf{s}' \in \mathbf{N_k(s)}$; find a $\mathbf{s}''$ around
                    $\mathbf{s}'$ using MDT;
9:              if $\mathbf{s}''$ is better than $\mathbf{s}$ then
10:                 $\mathbf{s} \leftarrow \mathbf{s}''$; $k \leftarrow 1$; $n \leftarrow 1$; continue;
11:             else $n \leftarrow n + 1$;
12:             end if
13:         end while
14:         $k \leftarrow k + 1$;
15:     end while
16:     if $\mathbf{s}$ is better than $\mathbf{s_{best}}$ then
17:         $\mathbf{s_{best}} \leftarrow \mathbf{s}$; $p \leftarrow 1$;
18:         Use Theorem 2 to check whether $\mathbf{s_{best}}$ is optimal; if
                so, return
19:     else $p \leftarrow p + 1$;
20:     end if
21: end while
22: Output: $\mathbf{s_{best}}$
```

those sublots of the same order. Here, we randomly select an order i that is produced in time r_d and denote T_i as the set of time periods in which the sublots for this order i are produced. If workload w_{r_d} is smaller than the average workload $\overline{w}$, we set $r_{\min} = r_d$ and identify the time period $r_{\max}$ with the largest workload $w_{r_{\max}}$ in T_i; otherwise, set $r_{\max} = r_d$ and identify $r_{\min}$ with the minimum workload $w_{r_{\min}}$ in T_i.

After that, the transfer quantity $q, q \in \mathbb{Z}^+$ is generated randomly within the interval $[0, \min\{up_1, up_2\}]$. For the right boundary of this interval, $up_1 = w_{r_{\max}} - w_{r_{\min}}$, and $up_2 = x_{ir_{\max}} - 1$, which ensures that at least one item of order i is produced at $r_{\max}$ so that the constraints about sublots number and continuous production will not be violated. With the transfer quantity q subtracted from $x_{ir_{\max}}$ and added to $x_{ir_{\min}}$, the workload at corresponding time periods is adjusted accordingly. Through the abovementioned steps, we obtain a new solution s''. An illustrative example of these steps is provided in Fig. 6. According to Lemma 1, s'' is not worse than s', so we replace s' with s''. The abovementioned steps repeat until the maximum number of iterations $L_{\max}$ is reached. The procedure of the MDT algorithm is presented in Algorithm 1, and its complexity is $O(L_{\max}T)$.

D. Enhanced Variable Neighborhood Search Algorithm

The EVNS for SLPP is presented in this section. An initial solution needs to be generated at the beginning of EVNS. We first generate the production time for each sublot and then their sizes. Since the lot for order i will be divided into S_i sublots for continuous production and the latest finish time of the last sublot is d_{sub_i}, the latest production time of the first sublot for order i is $d_{\text{sub}_i} - S_i + 1$; otherwise, the last sublot will not be completed on time. Thus, in the initial solution, the production time of the first sublot for order i is randomly selected within the interval $[1, d_{\text{sub}_i} - S_i + 1]$. Then, the remained $S_i - 1$ sublots for order i are assigned to the time periods immediately after the first sublot so that the constraints about continuous production are satisfied. As for the sizes of sublots in the initial solution, we distribute the demand quantity of each order to its sublots evenly and obtain the average value O_i/S_i as the initial size of each sublot for order i. We call this initial solution generation method the random time average quantity (RTAQ) method.

Order	1	2	3	4
O_i	100	50	60	50
d_{sub_i}	3	3	5	6

(a)

k	1	2	3
$O(k)$	150	60	50
$d(k)$	3	5	6

(b)

Fig. 7. Example for the LB. (a) Original order information. (b) Extracted information.

The neighborhood search strategy should also be considered in EVNS. For the search process in the kth neighborhood $N_k(s)$ of the incumbent solution s, we randomly select k orders in s and shift the production times of their sublots arbitrarily within the interval introduced earlier. Then, we reset the size of each sublot in these orders to the average value O_i/S_i as in the initial solution. By this way, a new solution s' in neighborhood $N_k(s)$ is generated. Furthermore, the MDT method is implemented to search for a better solution s'' around it. If s'' is better than the incumbent solution s, it is accepted as the new incumbent solution ($s \leftarrow s''$) immediately, and the neighborhood search loop starts from its nearest neighborhood ($k \leftarrow 1$) again. The search process in each neighborhood continues until the maximum number of consecutive nonimprovement search $N_{\max}$ is reached. After the search in neighborhood $N_k(s)$ is completed, the search moves to a farther neighborhood ($k \leftarrow k+1$) until $k = K_{\max}$.

Moreover, we adopt a perturbation loop to enhance the search ability of EVNS. When the conditions $k = K_{\max}$ and $n = N_{\max}$ are reached, we keep a record of the best solution s_{best} found in this process. Then, we regenerate a new initial solution using the RTAQ method and conduct VNS with MDT introduced earlier again. If a better solution than s_{best} is found after this process, it is recorded as new s_{best}. This perturbation loop continues until the number P of unsuccessful attempts at improving s_{best} is reached. In addition, we define two other stopping conditions for the EVNS algorithm. After we update s_{best} at the end of each perturbation iteration, we use Theorem 2 to check whether this solution has reached optimality. If this solution is already optimal, the algorithm will terminate. Besides, the algorithm will stop when the running time t reaches the time limit TL. The detailed procedure of EVNS for SLPP is introduced in Algorithm 2, and its complexity is $O(PK_{\max}N_{\max}L_{\max}T)$.

V. Lower Bound

In this section, we propose an LB for SLPP and an exact algorithm to calculate it in polynomial time. Since MLPP has not been studied before, there is no benchmark algorithm that could be compared with the algorithm that we proposed. Thus, this LB could help measure the performance of the proposed algorithm.

Here, we consider another relaxed problem of SLPP called RSLPP2. In RSLPP2, the constraints (18)–(21) in SLPP about the sublot number of each order and the continuous production of sublots are relaxed, and the objective is still to minimize the workload variance. What is more, we merge the orders sharing the same due date to a single order and no longer distinguish which orders the workload comes from. This would not change the value of workload variance and could reduce the scale

Algorithm 3 LB Algorithm

```
1: Input: $O(k), d(k), k = 1, \cdots, K$
2: $d(0) \leftarrow 0$; $avg(0) \leftarrow BigNum$;
3: for $k = 1 : K$ do
4:     $p \leftarrow 1$;
5:     $avg(k) \leftarrow \frac{O(k)}{d(k)-d(k-1)}$;
6:     $Avg \leftarrow avg(k)$;
7:     if $avg(k) > avg(k-1)$ then
8:         repeat
9:             $p \leftarrow p+1$;
10:            Sum up the present allocations from stage $k-p+1$
                   to stage $k$ as $Sum \leftarrow Avg \times [d(k)-d(k-p+1)] + avg(k-p+1) \times [d(k-p+1)-d(k-p)]$;
11:            Recalculate the average distribution of the
                   demands in these stages as $Avg \leftarrow \frac{Sum}{d(k)-d(k-p)}$;
12:        until $Avg < avg(k-p)$;
13:        update $avg(m) \leftarrow Avg$ for $k-p+1 \le m \le k$;
14:    end if
15: end for
16: $w_{LB}(r) \leftarrow avg(k),\ d(k-p) < r \le d(k),\ 1 \le k \le K$;
17: Extract $\{d(k_{(l)}) | l = 1, \cdots, K'; k_{(l-1)} < k_{(l)}\}$ in which
        $w_{LB}(r)$ remains the same for $d(k_{(l-1)}) < r \le d(k_{(l)})$;
18: For each $l = 1, \cdots, K'$, calculate $\alpha$ by (36), adjust the real
        number values of the first $\alpha$ $w_{LB}(r)$ in stair $l$ to their
        ceiling integer values and the rest to floor integer values
        to obtain an integer solution $w_{LB}(r)^*, 1 \le r \le d(K)$;
19: $obj(LB) \leftarrow \sum_{r=1}^{d(K)} (w_{LB}(r)^* - \overline{w_{LB}})^2$.
20: Output: $obj(LB)$
```

of orders and simplify the calculation. Especially, we extract every distinct due date $d(k), 1 \le k \le K$ in the chronological order from all orders, where $d(k)$ denotes the time of the kth different due date of orders. The required quantities of those orders with $d_{\text{sub}_i} = d(k), i \in N$ are summed up as $O(k), 1 \le k \le K$. The demand $O(k)$ should be fulfilled before the corresponding distinct due date $d(k)$. We call those time periods $r, d(k-1) < r \le d(k)$ as stage k. In Fig. 7, we present a simple example to illustrate this process. The original demand quantities and due dates of four orders are presented in Fig. 7(a), and the extracted parameters $d(k)$ and $O(k)$ are presented in Fig. 7(b).

We propose an exact LB algorithm to solve RSLPP2 in polynomial time and obtain an LB of SLPP. This algorithm finds the most balanced distribution of demand stage by stage from stage 1 to stage K. At each stage, it finds the optimal real-number demand distribution up to the current stage, which is denoted as avg(k) for stage k. The index p represents the iteration of the adjustment in each stage. After the last stage has been examined, the real-number values of workload $w_{\text{LB}}(r), 1 \le r \le d(K)$ are rounded to integer values to calculate the final lower-bound workload variance obj(LB).

The procedure of LB algorithm is presented in Algorithm 3. Two dummy variables $d(0)$ and avg(0) are initialized in line 2. Lines 3–15 are the main loop of the algorithm. For each stage $k, 1 \le k \le K$, the algorithm examines this stage from

$p = 1$, which means that it attempts to distribute the demand quantity evenly within this stage alone. If the obtained average workload avg(k) is less than the avg($k-1$) of previous stage $k-1$, considering that orders in previous stages cannot be transferred to this stage or else they will miss their deadlines, we cannot find a more balanced demand allocation up to the current stage, so the algorithm goes to the next stage. On the contrary, if avg(k) $>$ avg($k-1$), we could allocate some quantity of stage k to previous stages to balance the workload. Thus, in lines 8–13, we move backward stage by stage ($p \leftarrow p+1$) and reallocate the previously allocation Sum from stage $k-p+1$ to stage k evenly to the time periods in these stages. This iteration continues until a stage $k-p$ that has a larger workload distribution avg($k-p$) than the reallocation Avg of its subsequent stages is met.

After all stages have been fathomed, the workload $w_{\text{LB}}(r)$ in each time period r belonging to stage k is assigned with the real-number avg(k) in line 16. The above procedure generates a real-number solution with a downward stair-shaped structure. In line 17, we further extract a set of time periods $\{d(k_{(l)})|l = 1, \ldots, K'; K' \leq K; k_{(l-1)} < k_{(l)}\}$ such that $w_{\text{LB}}(r)$ remains the same for $d(k_{(l-1)}) < r \leq d(k_{(l)})$. We call all stages between $d(k_{(l-1)})$ and $d(k_{(l)})$ as stair l. The downward stair-shaped structure depicts that $w_{\text{LB}}(d(k_{(l-1)})) > w_{\text{LB}}(d(k_{(l)}))$ for $l = 1, \ldots, K'$. In line 18, for each stair, we round the real values of workload $w_{\text{LB}}(r)$ in the first α periods to ceiling integer values and the remained to floor values on the premise that their sum remains unchanged. α is calculated as follows:

$$\alpha \cdot \lceil w(d(k_{(l)}))\rceil + (d(k_{(l)}) - d(k_{(l-1)}) - \alpha) \cdot \lfloor w(d(k_{(l)}))\rfloor = (d(k_{(l)}) - d(k_{(l-1)})) \cdot w(d(k_{(l)})). \quad (36)$$

Without violating the due date constraint for each stage, this step yields a tighter integer LB solution $w_{\text{LB}}(r)^*, 1 \leq r \leq d(K)$. Finally, we calculate the workload variance in line 19 as the output.

For the computational complexity of the LB algorithm, there are K loops in total for K stages (lines 3–15). In each stage k, the worst case is that the algorithm needs to check from $p = 1$ to $p = k$, each of which takes constant time, so its $O(k)$ for stage k (lines 8–12). Line 13 also takes $O(k)$ for stage k. In total, that is, $\sum_{k=1}^{K} O(k) = O(K^2)$ for lines 3–15. Line 16–19 take $O(T)$, where T is the length of the time horizon. In fact, there are at most n stages for n orders, i.e., $K \leq n$. Thus, the time complexity of the LB algorithm is $O(n^2 + T)$.

We use the same example in Fig. 7 to illustrate the LB algorithm in Fig. 8. According to the LB algorithm, for stage $k = 1$, $O(1)$ is evenly allocated to the three time periods in stage 1 with avg(1) $= (150/3) = 50$, as shown in Fig. 8(a). Since avg(1) $<$ avg(0) $=$ BigNum, the algorithm moves to stage 2. The same situation is encountered in stage 2, and we obtain avg(2) $= (60/2) = 30$ [see Fig. 8(b)]. The algorithm further moves to stage 3. When $p = 1$, the allocation of demand $O(3)$ to stage 3 alone yields avg(3) $= (50/1) = 50$ [see Fig. 8(c)]. Avg $=$ avg(3) $= 50$. The variance up to now is 533.33. Because avg(3) is larger than avg(2), some quantity could be shifted to previous time periods to balance the

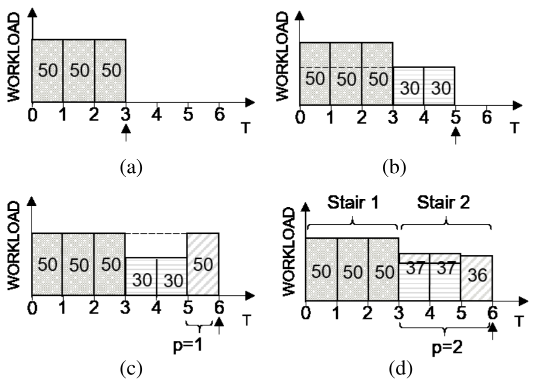


Fig. 8. Illustrative example of the LB algorithm. (a) $k = 1, d(1) = 3$. (b) $k = 2, d(2) = 5$. (c) $k = 3, d(3) = 6, p = 1$. (d) $k = 3, d(3) = 6, p = 2$.

workload. Thus, the algorithm continues in stage 3 with $p = 2$ and moves backward to consider the demand $O(3)$ and the allocation in stage 2 together. The quantity to be reallocated is Sum $=$ Avg $\times (d(3) - d(2)) +$ avg(2) $\times (d(2) - d(1)) = 110$, and the reallocation among the time periods in these two stages is Avg $= (\text{Sum}/(d(3) - d(1))) = 36.67$. Since Avg $<$ avg(1), the backward exploration is terminated, and we update avg(2) $=$ avg(3) $=$ Avg. Since all stages have been fathomed, the workload $w_{\text{LB}}(r)$ in each time period in each stage k is assigned with avg(k), obtaining two stairs with $w_{\text{LB}}(r) = 50$ for $r = 1, 2, 3$, and $w_{\text{LB}}(r) = 36.67$ for $r = 4, 5, 6$. Then, the real numbers in stair 2 are further rounded to integer values 37, 37, and 36, as shown in Fig. 8(d). The final LB obj(LB) is 267.33.

We now prove that the solution obtained by the LB algorithm is optimal for RSLPP2, that is to say, the returned workload distribution among the production time horizon is most balanced, and it could be used as the LB of SLPP therefore.

Theorem 3: For RSLPP2, the LB algorithm obtains the most balanced workload distribution in the planning horizon.

Proof: We conduct the proof based on the downward stair-shaped structure of the solution. Suppose that the final solution we obtained is not optimal. We try to transfer some quantities from certain time periods to others to find a better solution. We randomly select a stair $l \in \{1, \ldots, K'\}$ and divide the time horizon H_{sub} into two subsets: $H_{\text{sub1}} = \{r|1 \leq r \leq d(k_{(l)})\}$ and $H_{\text{sub2}} = \{r|d(k_{(l)}) + 1 < r \leq d(K)\} = H_{\text{sub}} \setminus H_{\text{sub1}}$. Apparently, we could not transfer the workload in H_{sub1} to H_{sub2} because they have to be done no later than $d(k_{(l)})$. On the other hand, according to the algorithm procedure, the workload in H_{sub2} is no larger than that that in H_{sub1}. According to Lemma 1, the transfer from H_{sub2} to H_{sub1} would increase the deviation and increase the variance as a result. Thus, the transfer of workload between H_{sub1} and H_{sub2} will not improve the solution.

Next, we consider the transfer within H_{sub1}. Similarly, we randomly select an l' subject to $d(k_{(l')}) \in H_{\text{sub1}}$. We divide H_{sub1} into two subsets: $H_{\text{sub3}} = \{r|1 \leq r \leq d(k_{l'})\}$ and $H_{\text{sub4}} = H_{\text{sub1}} \setminus H_{\text{sub3}}$. Due to the same reason, we cannot transfer the workload from H_{sub3} to H_{sub4}, and vice versa. We repeat this divide-and-conquer process until one subset of the two becomes an empty set, and we can conclude that no transfer within H_{sub1} will improve the solution. This conclusion can be applied to H_{sub2} for the same reason. Thus, any transfer within H_{sub} is not able to improve the solution. Thus, the final solution that we obtained by the LB algorithm is optimal, and the proof is completed. ■

VI. Computational Experiment

In this section, computational experiments are conducted to verify the reformulation in Theorem 1 and test the performance of the proposed algorithms. The proposed algorithms are coded with C++, and all the experiments are conducted on a 64-bit Windows 10 platform with Intel i7 3.4-GHz CPU and 16.0-GB RAM.

The formulations in Sections II and III are solved using IBM ILOG CPLEX 12.8 under the default configurations within the time limit of 7200 s. Some pilot experiments are conducted to identify the appropriate values of the algorithm parameters in MDT and EVNS. The iteration number $L_{\max}$ in MDT is set to $4 \cdot n$ for instances with n orders. In EVNS, the maximum number of consecutive unsuccessful attempts to search a neighborhood $N_{\max}$ is set to 250. The largest neighborhood $K_{\max}$ and the maximum times of perturbations P are set to 3 and 2, respectively. The time limit TL is also set to 7200 s.

A. Problem Instances Generation

We generate two groups of instances based on their scales.

1) Small-Scale Instances (G_1): A group of instances G_1 with a similar scale to the production plan in the factory that we investigate are generated. The planning horizon in this factory is about one to three years, and there are about ten orders per year. Thus, in G_1, the numbers of orders n are 10, 20, and 30, and the corresponding time periods T are 12, 24, and 36, respectively. Each time period could represent a month in reality. The demand quantity O_i for each order is a multiple of 10 randomly selected from the interval [100, 200]. As for the items number m, we adopt a set of BOM structures with nine items considered by Coleman and McKnew [28]. It contains four BOM structures M1–M4 with three to six layers, respectively. The number of sublots S_i is randomly selected from $\{3, 4\}$.

2) Large-Scale Instances (G_2): In order to meet the challenge of order growth in the future and provide a more detailed production plan, we generate a group of large-scale instances G_2 to test the performance of algorithms with the following parameters: $n = 40, 50$, and 60 with $T = 48$, $n = 50, 60$, and 70 with $T = 72$, and $n = 60, 70$, and 80 with $T = 96$, respectively. Each time period could represent a week in reality. Because we solve SLPP first and use its solution to generate a solution of MLPP according to Theorem 1, the BOM structure has no impact on the test of the performance of SLPP and EVNS, so all the instances in this group are generated with BOM M1 with nine items. Integer S_i is randomly selected from uniform distribution [8, 16]. Other settings are the same as those in small-scale instances.

TABLE I
Optimal Results of Some Instances With $n = 10$, $m = 9$, and $T = 12$ by MLPP and SLPP

BOM	*No.*	*MLPP*	*SLPP*	*m·SLPP*
M1	1	313286.00	34809.56	313286.00
	2	13.50	1.50	13.50
	3	13.50	1.50	13.50
	4	18.00	2.00	18.00
	5	0.00	0.00	0.00
M2	6	0.00	0.00	0.00
	7	15.43	1.71	15.43
	8	7.71	0.86	7.71
	9	86145.40	9571.71	86145.40
	10	15.43	1.71	15.43

We generate ten instances for each combination "*BOM-n-T*" in both groups of instances. A total number of 210 instances are generated to conduct experiments.

B. Numerical Results and Analysis

In this section, we present the results and analysis of the computational experiments. Some notations are used as follows to present the results in tables.

#Sol:	Number of instances for which a feasible solution could be obtained.
#Opt:	Number of instances solved to optimality.
Time:	Average running time of the corresponding algorithm.
R:	Performance ratio of the formulation or algorithm.

The performance ratio R is calculated as follows:

$$R_i = \frac{\text{obj}(\text{alg}_i) - \text{obj}(\text{LB})}{\text{obj}(\text{UB}) - \text{obj}(\text{LB})} \times 100(\%). \quad (37)$$

In this formula, $\text{obj}(\text{alg}_i)$ denotes the objective function value of the corresponding formulation solved by CPLEX or the corresponding algorithm. obj(LB) and obj(UB) denote the value of LB and upper bound, respectively. This performance ratio could reflect the quality of the solution, and the algorithm with the performance ratio closer to 0 is better for this minimization problem. The upper bound solution of the problem is generated by the RTAQ method that is used to generate initial solutions in EVNS. Consequently, the worst possible performance ratio of EVNS is 1. Min.R, Avg.R, and Max.R represent the minimum, average, and maximum performance ratios for the ten instances in each combination, and Avg.R_i < Avg.R_j indicates that the algorithm i outperforms the algorithm j, vice versa.

Table I presents the results of the formulations MLPP and SLPP solved by CPLEX for ten randomly selected instances in G_1 with ten orders, nine items, and 12 time periods that could be solved to optimality. The column "No." presents the ID of the instances. The columns "MLPP" and "SLPP" report the objective function values of the corresponding formulation, respectively. Because in some instances such as 1 and 8, the due dates of orders are not relatively evenly spread across the planning horizon as mentioned at the end of Section III, the optimal workload variances of these instances are very large.

TABLE II
NUMERICAL RESULTS OF MLPP, SLPP, AND EVNS COMPARED WITH LB IN G_1

			MLPP			SLPP			EVNS		
BOM	*n*	*T*	*#Sol*	*#Opt*	*Avg.R(%)*	*#Sol*	*#Opt*	*Avg.R(%)*	*#Sol*	*#Opt*	*Avg.R(%)*
	10	12	10	10	0.04	10	10	0.04	10	9	0.09
M1	20	24	9	0	1899.75	10	10	0.04	10	6	0.22
	30	36	4	0	2789.43	10	9	0.01	10	1	0.67
	10	12	10	10	0.03	10	10	0.03	10	7	0.04
M2	20	24	9	0	1041.42	10	10	0.03	10	3	0.40
	30	36	7	0	3227.66	10	10	0.02	10	1	0.65
	10	12	10	4	414.11	10	10	0.08	10	6	0.15
M3	20	24	9	0	2148.43	10	10	0.03	10	4	0.30
	30	36	7	0	2645.50	10	10	0.02	10	1	0.76
	10	12	10	5	42.08	10	10	0.03	10	8	0.06
M4	20	24	9	0	2282.01	10	10	0.02	10	4	0.22
	30	36	4	0	2327.93	10	9	0.08	10	0	0.61
Total		Total	98/120	29/120		120/120	118/120		120/120	50/120	

TABLE III
NUMERICAL RESULTS OF SLPP AND EVNS COMPARED WITH LB IN G_2

		SLPP				EVNS			
n	*T*	*Min.R(%)*	*Avg.R(%)*	*Max.R(%)*	*Time(s)*	*Min.R(%)*	*Avg.R(%)*	*Max.R(%)*	*Time(s)*
40		0.07	115.00	426.74	7200.00	0.00	0.25	1.52	79.09
50	48	0.29	117.82	328.58	7200.00	0.00	0.60	2.71	113.08
60		77.11	178.52	312.16	7200.00	0.00	0.66	2.90	143.70
50		178.82	1071.94	7069.85	7200.00	0.00	0.58	2.32	142.98
60	72	254.15	765.01	2143.32	7200.00	0.00	0.98	3.44	277.70
70		385.31	1078.33	3268.91	7200.00	0.00	0.66	1.84	433.97
60		331.59	2736.34	10072.46	7200.00	0.00	1.25	3.72	409.55
70	96	511.34	3933.88	11754.59	7200.00	0.00	1.62	2.80	666.05
80		493.08	3044.52	7965.79	7200.00	0.00	0.78	3.05	918.37

According to Theorem 1, for a BOM structure with m items, the minimum workload variance of MLPP should be m times of the minimum workload variance of each individual item corresponding to SLPP. As shown in Table I, for each instance, the objective function value of the optimal solution of MLPP is nine times that of SLPP, which is the number of items in the BOM structures that we investigate. Thus, the results in Table I verify Theorem 1 from numerical experiments. To make the comparison more convenient and obtain the objective function values of the original MLPP problems, the objective function values of SLPP and EVNS and the values of two bounds in Tables II and III are multiplied by m times.

Table II presents the numerical results of the two formulations MLPP and SLPP solved by CPLEX and the EVNS algorithm for small-scale instances in G_1 with a time limit of 2 h. #Sol, #Opt, and Avg.R of each formulation and algorithm are reported in Table II. The IQP problem MLPP is difficult to solve due to the quadratic objective function and large-scale integer variables and constraints. As shown in the subcolumn "#Sol" for MLPP, MLPP could solve the instances with ten orders to obtain feasible solutions, but it fails to solve some instances with more than ten orders. Among the instances in G_1, only 29 instances with ten orders are solved to optimality by MLPP within 2 h, and none of the instances with more than ten orders are solved to optimality.

Due to the simplification from MLPP to SLPP by Theorem 1, all the 120 instances in G_1 could be solved by CPLEX for the formulation SLPP. Besides, nearly all instances could be solved to optimality, which reaches 118 out of 120 instances. From those combinations in which all instances are solved to optimality by SLPP, we could see the performance ratios are all less than 0.1%, which reflects that the LB that we propose is tight.

As for the EVNS algorithm, it solves all instances and obtains the optimal solutions for 50 instances out of 120 instances. For all of the combinations, the Avg.R's obtained by the EVNS algorithm are less than 1%, and the largest Avg.R is only 0.76%, which are small and close to the ratio obtained by SLPP. Although it is inferior to the exact solver of CPLEX for SLPP with regard to the #Opt for the small-scale instances, these results still demonstrate the search ability and effectiveness of the proposed algorithm.

Table III presents the results of the EVNS algorithm and SLPP solved by CPLEX for large-scale instances G_2. As shown in Table III, the Avg.R's of SLPP for all combinations in G_2 are over 100%, which means that, on average, the feasible solutions obtained by CPLEX are even worse than the upper bound solutions that we generate using the RTAQ method. Within a 2-h time limit, even for SLPP that is simplified from MLPP, it is difficult for CPLEX to search for better solutions for these large-scale IQP problems. With the increase in instance sizes, especially the number of time periods that have a great impact on the size of the solution space, all three criteria of SLPP increase significantly. Its Max.R even reaches 11754% appearing in the row with 70 orders and 96 time periods.

Compared with SLPP solved by CPLEX, the largest Avg.R among all combinations obtained by the EVNS algorithm is less than 1.62%, which is close to its performance ratio for small-scale instances and significantly better than the best solutions obtained by SLPP. Particularly, for all of the instances in G_2, the Max.R by the EVNS algorithm is only 3.72%, which is much smaller than most of the Min.R by SLPP. There is only a slight increase in Avg.R with the increase in problem scale, which exhibits its robust ability to solve large-scale problems. Notably, as for the running time of the EVNS algorithm, as shown in the subcolumn "Time" for EVNS, the EVNS algorithm could yield such good solutions in a very short time less than 20 min, which demonstrates its efficiency.

VII. Conclusion

This article deals with a tactical production planning problem with the objective of WB. Many realistic production characteristics are considered, such as no backorder, lot streaming production, and multilevel BOM. We first propose an IQP model to formulate this problem. Then, for a typical BOM structure, this problem is reformulated to a simplified problem where only one item needs to be considered. Some optimality properties are further derived to help solve this problem. We develop an EVNS algorithm in which the MDT local search method is embedded to solve this problem. Furthermore, an LB along with a polynomial-time algorithm for it is proposed to measure the performance of the EVNS algorithm. Computational experiments are conducted to verify the correctness and effectiveness of reformulation. Numerical results show that the proposed algorithm could yield high-quality solutions for large-scale problems in a short time.

Future work on this topic could involve more complicated production environments. The properties of MLPP with more complex BOM structures could be explored. The balance of sublot sizes in each order and the initial inventory of some items may also be considered. The approach can be extended to solve these problems.

References

[1] J. H. Chang* and H. N. Chiu, "A comprehensive review of lot streaming," *Int. J. Prod. Res.*, vol. 43, no. 8, pp. 1515–1536, Apr. 2005.

[2] M. Yavuz and E. Akçali, "Production smoothing in just-in-time manufacturing systems: A review of the models and solution approaches," *Int. J. Prod. Res.*, vol. 45, no. 16, pp. 3579–3597, Aug. 2007.

[3] M. Yavuz, "An iterated beam search algorithm for the multi-level production smoothing problem with workload smoothing goal," *Int. J. Prod. Res.*, vol. 48, no. 20, pp. 6189–6202, Oct. 2010.

[4] P. Chutima and S. Olarnviwatchai, "A multi-objective car sequencing problem on two-sided assembly lines," *J. Intell. Manuf.*, vol. 29, no. 7, pp. 1617–1636, Oct. 2018.

[5] M. Yavuz and H. Ergin, "Advanced constraint propagation for the combined car sequencing and level scheduling problem," *Comput. Oper. Res.*, vol. 100, pp. 128–139, Dec. 2018.

[6] L. Zeltzer, E.-H. Aghezzaf, and V. Limère, "Workload balancing and manufacturing complexity levelling in mixed-model assembly lines," *Int. J. Prod. Res.*, vol. 55, no. 10, pp. 2829–2844, May 2017.

[7] M. Chica, J. Bautista, Ó. Cordón, and S. Damas, "A multiobjective model and evolutionary algorithms for robust time and space assembly line balancing under uncertain demand," *Omega*, vol. 58, pp. 55–68, Jan. 2016.

[8] J. Pereira and E. Álvarez-Miranda, "An exact approach for the robust assembly line balancing problem," *Omega*, vol. 78, pp. 85–98, Jul. 2018.

[9] P. Sivasankaran and P. Shahabudeen, "Literature review of assembly line balancing problems," *Int. J. Adv. Manuf. Technol.*, vol. 73, nos. 9–12, pp. 1665–1694, Aug. 2014.

[10] F. Yang, K. Gao, I. W. Simon, Y. Zhu, and R. Su, "Decomposition methods for manufacturing system scheduling: A survey," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 2, pp. 389–400, Mar. 2018.

[11] J. C. Ho, T.-L. Tseng, A. J. Ruiz-Torres, and F. J. López, "Minimizing the normalized sum of square for workload deviations on m parallel processors," *Comput. Ind. Eng.*, vol. 56, no. 1, pp. 186–192, Feb. 2009.

[12] A. Cossari, J. C. Ho, G. Paletta, and A. J. Ruiz-Torres, "A new heuristic for workload balancing on identical parallel machines and a statistical perspective on the workload balancing criteria," *Comput. Oper. Res.*, vol. 39, no. 7, pp. 1382–1393, Jul. 2012.

[13] A. Cossari, J. C. Ho, G. Paletta, and A. J. Ruiz-Torres, "Minimizing workload balancing criteria on identical parallel machines," *J. Ind. Prod. Eng.*, vol. 30, no. 3, pp. 160–172, Apr. 2013.

[14] S. Schwerdfeger and R. Walter, "A fast and effective subset sum based improvement procedure for workload balancing on identical parallel machines," *Comput. Oper. Res.*, vol. 73, pp. 84–91, Sep. 2016.

[15] S. Schwerdfeger and R. Walter, "Improved algorithms to minimize workload balancing criteria on identical parallel machines," *Comput. Oper. Res.*, vol. 93, pp. 123–134, May 2018.

[16] Q. Christ, S. Dauzere-Peres, and G. Lepelletier, "An iterated min–max procedure for practical workload balancing on non-identical parallel machines in manufacturing systems," *Eur. J. Oper. Res.*, vol. 279, no. 2, pp. 419–428, 2019.

[17] Y. Han, D. Gong, Y. Jin, and Q. Pan, "Evolutionary multiobjective blocking lot-streaming flow shop scheduling with machine breakdowns," *IEEE Trans. Cybern.*, vol. 49, no. 1, pp. 184–197, Jan. 2019.

[18] A. Bożek and F. Werner, "Flexible job shop scheduling with lot streaming and sublot size optimisation," *Int. J. Prod. Res.*, vol. 56, no. 19, pp. 6391–6411, Oct. 2018.

[19] T. Inkaya and M. Akansel, "Coordinated scheduling of the transfer lots in an assembly-type supply chain: A genetic algorithm approach," *J. Intell. Manuf.*, vol. 28, no. 4, pp. 1005–1015, Apr. 2017.

[20] M. Cheng, N. J. Mukherjee, and S. C. Sarin, "A review of lot streaming," *Int. J. Prod. Res.*, vol. 51, nos. 23–24, pp. 7023–7046, Nov. 2013.

[21] W. Wang and L. Shi, "Multi-level lot-streaming production planning with the objective of workload balancing," in *Proc. IEEE 15th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2019, pp. 685–690.

[22] N. Mladenović and P. Hansen, "Variable neighborhood search," *Comput. Oper. Res.*, vol. 24, no. 11, pp. 1097–1100, Nov. 1997.

[23] W. Jomaa, M. Eddaly, and B. Jarboui, "Variable neighborhood search algorithms for the permutation flowshop scheduling problem with the preventive maintenance," *Oper. Res.*, pp. 1–18, Jul. 2019.

[24] X. Xu, J. Li, and M. Zhou, "Delaunay-triangulation-based variable neighborhood search to solve large-scale general colored traveling salesman problems," *IEEE Trans. Intell. Transp. Syst.*, early access, Feb. 21, 2020, doi: 10.1109/TITS.2020.2972389.

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

[26] B. Sarasola, K. F. Doerner, V. Schmid, and E. Alba, "Variable neighborhood search for the stochastic and dynamic vehicle routing problem," *Ann. Oper. Res.*, vol. 236, no. 2, pp. 425–461, Jan. 2016.

[27] Z. Zhang, M. Zhou, and J. Wang, "Construction-based optimization approaches to airline crew rostering problem," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 3, pp. 1399–1409, Jul. 2020.

[28] B. J. Coleman and M. A. McKnew, "An improved heuristic for multilevel lot sizing in material requirements planning," *Decis. Sci.*, vol. 22, no. 1, pp. 136–156, Jan. 1991.

Weihao Wang (Student Member, IEEE) received the B.S. degree in telecommunications engineering from Tianjin University, Tianjin, China, in 2015. He is currently pursuing the Ph.D. degree with the Department of Industrial Engineering and Management, Peking University, Beijing, China.

His research interests include system optimization, operations management, intelligent manufacturing, and supply chain management.

Chutong Gao (Student Member, IEEE) is currently pursuing the bachelor's degree in engineering mechanics with the Department of Mechanics and Engineering Science, Peking University, Beijing, China.

His research interests include the design and analysis of optimization algorithms, machine learning, and supply chain management.

Leyuan Shi (Fellow, IEEE) received the B.S. degree in mathematics from Nanjing Normal University, Nanjing, China, in 1982, the M.S. degree in applied mathematics from Tsinghua University, Beijing, China, in 1985, and the M.S. degree in engineering and the Ph.D. degree in applied mathematics from Harvard University, Cambridge, MA, USA, in 1990 and 1992, respectively.

She is currently a Professor with the Department of Industrial and Systems Engineering, University of Wisconsin–Madison, Madison, WI, USA. She has authored or coauthored two books: *Nested Partitions Method, Theory and Applications* (Springer, 2009) and *Modeling, Control and Optimization of Complex Systems* (Springer, 2003). Her research is devoted to the theory and development of large-scale optimization algorithms, discrete-event simulation methodology, and modeling and analysis of discrete dynamic systems, with applications to complex systems, such as supply chain networks, manufacturing systems, communication networks, and financial engineering.

Dr. Shi is a member of the Institute for Operations Research and the Management Sciences (INFORMS). She is also an Editor of the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING and an Associate Editor of *Discrete Event Dynamic Systems*.