Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 6534021, 12 pages
https://doi.org/10.1155/2018/6534021

Research Article

A Biobjective Optimization Model for Deadline Satisfaction in Line-of-Balance Scheduling with Work Interruptions Consideration

Xin Zou,[1] **Lihui Zhang**,[2] **and Qian Zhang**[3]

[1]*Department of Economic Management, North China Electric Power University, Hebei 071003, China*
[2]*School of Economics and Management, North China Electric Power University, Beijing 102206, China*
[3]*State Grid Energy Research Institute, Beijing 102209, China*

Correspondence should be addressed to Xin Zou; zoux788@126.com

Received 11 December 2017; Accepted 25 March 2018; Published 3 May 2018

Academic Editor: Anna Vila

The line-of-balance (LOB) technique has demonstrated many advantages in scheduling repetitive projects, one of which is that it allows more than one crew to be hired by an activity concurrently. The deadline satisfaction problem in LOB scheduling (DSPLOB) aims to find an LOB schedule such that the project is completed within a given deadline and the total number of crews is minimized. Previous studies required a strict application of crew work continuity, which may lead to a decline in the competitiveness of solutions. This paper introduces work interruptions into the DSPLOB and presents a biobjective optimization model that can balance the two conflicting objectives of minimizing the total number of crews and maximizing work continuity. An efficient version of the ϵ-constraint method is customized to find all feasible tradeoff solutions. Then, these solutions are further improved by an automated procedure to reduce the number of interruptions for each activity without deteriorating the performance in both the objectives. The effectiveness and practicability of the proposed model are verified using a considerable number of instances. The results show that introducing work interruptions provides more flexibility in reducing the total number of crews under the LOB framework, especially for serial projects with a tight deadline constraint.

1. Introduction

Repetitive projects are characterized by a number of units to be finished and a set of activities that need to be repeated for each unit in the length of the job [1]. Such projects are frequently encountered in multistage projects (e.g., high-rise buildings and multiple similar houses) and infrastructure networks (e.g., highways, railways, and pipelines). Repetitive projects account for a substantial part of the construction industry [2]. In China, the government has long been concerned with improving the nation's transport system; the "13th Five-Year Plan (2016–2020)" predicted that regional road networks will be increased by 420,000 kilometers and approximately 30,000 kilometers of railways will be built and start operation by 2020. Therefore, efficient planning and scheduling of this type of project is crucial.

In repetitive projects, construction resources (crews) are often required to perform the same or similar work in various units by moving from one to another. Due to this frequent crew movement, an effective schedule is important to maintain work continuity of crews by eliminating their needless interruption days from their movement between units [3]. However, a strict compliance with work continuity by preventing all crews from having interruptions may lead to a longer overall project duration [4]. In recent decades, numerous researches began to focus on repetitive projects scheduling with work interruptions [5–9].

The critical path method (CPM) is the most common and extensively used scheduling technique in construction projects [10]. However, its application in repetitive projects is limited as CPM is incapable of minimizing work interruptions [8]. As such, several scheduling methods applicable to

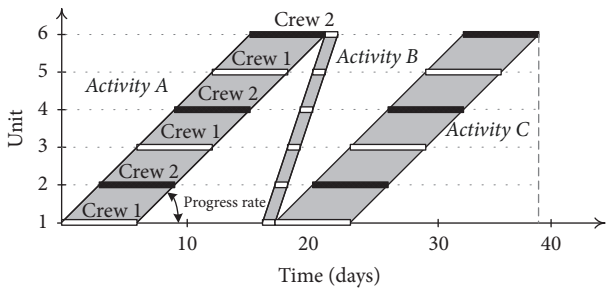


Figure 1: An example of LOB schedule without interruptions.

repetitive projects have been developed since the 1960s [11–13], among which one of the most representative is the line-of-balance (LOB) technique.

Dolabi et al. [14] enumerated several advantages of LOB over the other techniques, one of which is that it allows more than one crew to be employed by an activity concurrently. In fact, if the units of an activity are technically independent (i.e., the precedence relations between them are not mandatory), these units in theory can be performed at the same time by hiring additional crews [15]. Considering multiple crews allows a project to be rescheduled by adjusting the crew distribution of activities, which enhances the practicability of the output schedules. In literature, multiple crew usage has attracted considerable attention and has been considered by many scheduling optimization problems, for example, cost optimization [16], time-cost tradeoff [17], resource leveling [18], deadline satisfaction [19], and profit maximization with cash flow [20]. In this paper, the deadline satisfaction problem under the LOB framework (DSPLOB) will be investigated.

The DSPLOB involves a typical project in which the duration of each activity is assumed to be uniform along all units. The objective is to find an LOB schedule such that the project is completed within a given deadline and the total number of crews is minimized. Available methods for solving this problem can be classified into heuristic procedures and mathematical optimization. Heuristic procedures provide a way to obtain good solutions but do not guarantee optimality, including the integrated CPM and LOB method (CPM/LOB) [21], repetitive unit scheduling system (RUSS) [22], advanced linear scheduling system (ALISS) [23], and search-based heuristic line-of-balance (SHLOB) [14]. Mathematical optimization methods convert the problem into mathematical programming models and then use commercial software to seek the optimal solution. Such works include the mixed-integer linear model developed by Zou et al. [19] and the mixed-integer nonlinear model presented by Dolabi et al. [14]. Since all the above studies required a strict application of work continuity, the optimal schedule constructed by each of them is similar to the one shown in Figure 1, where each activity is represented by a sloped bar and the slope of the bar stands for the progress rate of the activity. A brief review of existing solution methods is as follows:

(i) CPM/LOB starts with a CPM calculation to determine the completion time T_1 of the standard unit and float time TF_j of each activity j. Then, the theoretical number of crews employed for activity j is determined by $C_j = (m-1)d_j/(T_d + \mathrm{TF}_j - T_1)$, where T_d is the given deadline, d_j is the unit duration of activity j, and m is the number of units. However, this method may fail to obtain a feasible LOB schedule that satisfies the deadline constraint if there exists an activity j such that C_j is not an integer.

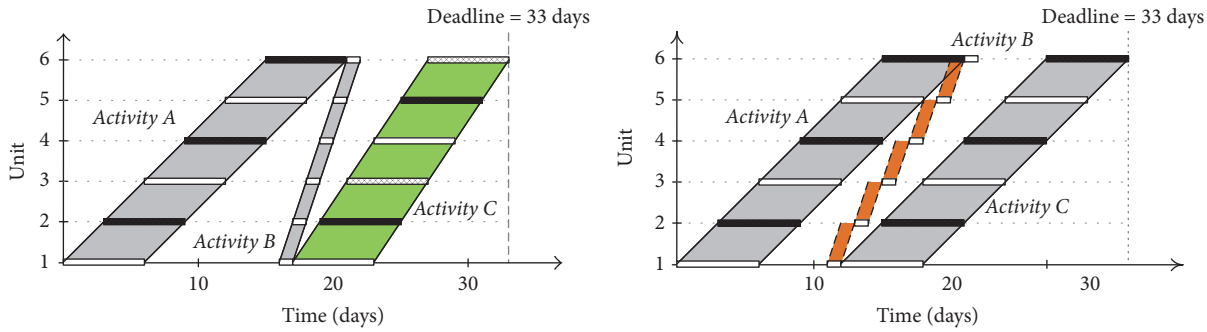
(ii) ALISS (or RUSS) is a heuristic iterative algorithm that initializes an LOB schedule by assigning only one crew to each activity. Then, the schedule is updated iteratively by increasing the number of crews by one for the selected activity. The procedure ends when the given deadline is satisfied or the project duration cannot be shortened by accelerating any one activity, and in the latter case, an infeasible LOB schedule will be obtained.

(iii) SHLOB is also a heuristic iterative algorithm that uses the CPM/LOB solution as the initial LOB schedule. In each iteration, parallel and consecutive activities are considered as a block, and the set of permitted blocks that may lead to a reduced project duration by increasing their progress rates is identified. If this set is null, the procedure fails to obtain a feasible solution; otherwise, one block within the set is selected on the basis of several heuristic rules and the number of crews for each activity in the selected block is increased by one. Then, the procedure goes to the next iteration until the deadline constraint is met. SHLOB is valid only for serial projects in which precedence relations between activities are finish to start and each activity has just one successor.

(iv) The performance of the mixed-integer linear model presented by Zou et al. [19] was investigated using 2700 instances with up to 120 activities. The results revealed that their model can find optimal schedules for all instances within 20 seconds. In comparison, although SHLOB provided each instance with a feasible schedule, only 11.93% were solved to optimality. Of the 2700 stances, 57.59% can be solved by ALISS, of which only 1.7% were found to be optimal. Even worse, CPM/LOB failed to solve any instance. The exact model developed by Dolabi et al. [14] is less practical because it took approximately 31 hours to handle a serial project with 24 activities.

Although many studies have been motivated to solve the DSPLOB, they all suffer from the assumption of preventing all activities from having interruptions. Considering the project shown in Figure 1, if a tighter deadline of 33 days is required, the only way existing solution methods can provide is to increase the number of crews by one for activity C (or A), as shown in Figure 2(a). However, if work interruptions are allowed, as shown in Figure 2(b), the given deadline can also be met by interrupting activity B for 5 days without changing the current crew distribution.

The two LOB schedules shown in Figure 2 indicate two possible tradeoffs between the two important and conflicting



(a) An additional crew is assigned to activity C

(b) Activity B is interrupted for 5 days

Figure 2: Two ways of meeting the given deadline.

objectives of minimizing the total number of crews and maximizing work continuity. As stated earlier, maximizing work continuity can produce significant reductions in the unproductive waiting time of crews that may increase costs, create tensions, and decrease morale [24]. However, it can also lead to a longer project duration that may violate the deadline constraint. If this happens, additional crews have to be employed to compress the project completion time into the given deadline. Therefore, construction planners need to find a plan from all possible tradeoff solutions that strikes an optimal balance between the two conflicting objectives. Unfortunately, existing methods or models are incapable of generating these tradeoffs as the only plan they can produce is a single optimal solution without any idleness, such as the schedule shown in Figure 2(a). There is a pressing need for advanced methods that can help construction planners in generating and evaluating all feasible tradeoff solutions to select an optimal plan that satisfies the specific requirements of the project being considered.

In this paper, we examine the DSPLOB with work interruptions consideration (DSPLOB&WI). Our study has three aspects: (1) establishment of an LOB-based scheduling optimization model (LOB-SOM) in the form of biobjective mixed-integer linear programming to deal with the DSPLOB&WI, (2) proposition of an effective solution method based on the ϵ-constraint multiobjective optimization strategy to find all feasible tradeoff solutions, and (3) implementation of a computational experiment to verify the validity and superiority of the proposed model and to analyze the impact of work interruptions on the planning and scheduling of repetitive projects.

2. Establishment of the LOB-SOM

Suppose that we are given a repetitive project which contains a set $A = \{0, 1, \ldots, n, n+1\}$ of activities and a set $U = \{1, \ldots, m\}$ of units. Each activity $i \in A$ needs to be repeatedly performed in each unit $j \in U$. Without loss of generality, we assume that activities 0 and $n+1$ are dummy activities, where the former represents the beginning and the latter indicates the end of the project. For each activity $i \in A$, denote by $d_i \in R_+$ the unit duration and $\overline{k}_i \in N_+$ the maximal number of available crews. For dummy activities 0 and $n+1$, we set $d_0 = d_{n+1} = 0$ and $\overline{k}_0 = \overline{k}_{n+1} = 1$. Let δ be the prespecified deadline of the project.

2.1. Work Continuity Requirement. According to Hegazy and Wassef [16], an activity i in LOB is thought to be performed without interruptions if $s_{ij} + 1/r_i = s_{i,j+1}$ is true for each unit $j \in U \setminus \{m\}$, as shown in Figure 3, where $r_i \in R_+$ represents the progress rate of activity i and $s_{ij} \in R_+$ denotes the start time of activity i in unit j. Otherwise, the difference $y_{ij} = s_{i,j+1} - (s_{ij} + 1/r_i)$ stands for the interruption duration in activity i after unit j. Although this paper is committed to emphasizing the benefit of allowing work interruptions, we have to understand that there still exist several types of activities that are best performed continuously. Examples of such activities include but are not limited to [3, 25]

(i) activities that are outsourced, or rely on external resources (e.g., hired major equipment, consultants, etc.),

(ii) activities that are performed by resources in which a great loss of cost will be suffered for idle time (e.g., dry docks in a ship yard, shop floor space, or pallets).

Therefore, an effective optimization model must be able to meet the strict work continuity requirement of some special activities.

2.2. Model Formulation. As mentioned earlier, the LOB-SOM aims to optimize the two conflicting objectives of minimizing the total number of crews and maximizing work continuity. Maximizing work continuity is equivalent to minimizing the total duration of interruptions. Therefore, both the objectives can be mathematically described by

$$\min \quad \sum_{i \in A} \sum_{k=1}^{\overline{k}_i} k x_{ik} \tag{1}$$

$$\min \quad \sum_{i \in A} \sum_{j=1}^{m-1} y_{ij}, \tag{2}$$

where binary variables x_{ik} assume a value of one if k crews are employed for activity i and equal zero otherwise and variables

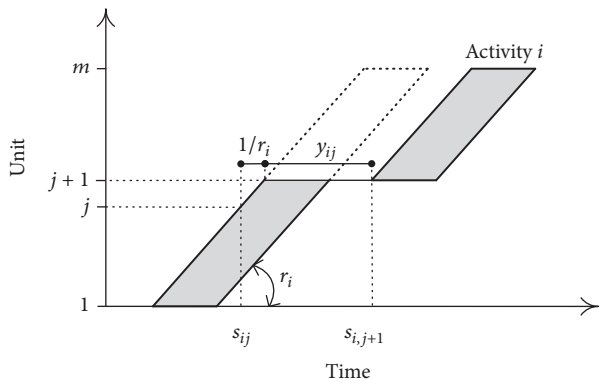


Figure 3: Scheduling an activity with interruptions.

y_{ij} represent the interruption duration in activity i after unit j. To ensure that the number of crews assigned to each activity is definite, variables x_{ik} must satisfy the following constraints:

$$\sum_{k=1}^{\overline{k}_i} x_{ik} = 1, \quad i \in A. \tag{3}$$

For each $i \in A$, let $\overline{y}_i$ be the maximal interruption duration of activity i that can be applied after each unit. Then, the strict work continuity requirement can be guaranteed by (4), where the value of $\overline{y}_i$ equals zero if activity i needs to be prevented from having interruptions; otherwise, it takes a relatively large number (e.g., the deadline δ).

$$0 \le y_{ij} \le \overline{y}_i, \quad i \in A, \ j \in U \setminus \{m\}. \tag{4}$$

The constraint system of the LOB-SOM model consists of precedence constraints, logic constraints, and a deadline constraint. Precedence constraints between activities are expressed by a set $E \subseteq A \times A$ in which each pair of activities (p, i) means that activity p is a predecessor of activity i. With a precedence constraint, activity i cannot start in each unit until the completion of activity p in the same unit; that is,

$$s_{pj} + d_p \le s_{ij}, \quad (p, i) \in E, \ j \in U. \tag{5}$$

The logic constraints are used to clarify the relationship between start times of each activity in adjacent units. As shown in Figure 3, the start time of activity i in unit $j+1$ (i.e., $s_{i,j+1}$) must be equal to $s_{ij} + 1/r_i + y_{ij}$. Because the progress rate r_i of activity i is the ratio of the actual number of crews employed for this activity to its unit duration (i.e., $r_i = \sum_{k=1}^{\overline{k}_i} k x_{ik}/d_i$), the logic constraints finally read as follows:

$$s_{ij} + d_i \sum_{k=1}^{\overline{k}_i} \frac{x_{ik}}{k} + y_{ij} = s_{i,j+1}, \quad i \in A, \ j \in U \setminus \{m\}. \tag{6}$$

The deadline constraint forces the project to be completed within the given deadline δ. Since the project completion time can be evaluated by the start time ($s_{n+1,m}$) of the finish activity $n+1$ in the last unit m, this constraint is given by

$$s_{n+1,m} \le \delta. \tag{7}$$

Integrating all the above objective functions and constraints, the LOB-SOM for solving the DSPLOB&WI is as follows:

$$\begin{aligned} &\min \quad \sum_{i \in A} \sum_{k=1}^{\overline{k}_i} k x_{ik} \\ &\min \quad \sum_{i \in A} \sum_{j=1}^{m-1} y_{ij}. \\ &\text{s.t.} \quad (3)\text{–}(7) \\ &\qquad x_{ik} \in \{0, 1\}; \ s_{ij}, y_{ij} \ge 0 \end{aligned} \tag{8}$$

For the LOB-SOM, we have the following discussions:

(1) If all activities are required to be performed without interruptions (i.e., $y_{ij} = 0$ for each $i \in A$ and $j \in U$), then the model reduces to a single-objective optimization model that was presented by Zou et al. [19] to solve the DSPLOB.

(2) If we replace the objective function (1) with the one stated in (9), where c_{ik} is the direct cost of activity i for employing $k = 1, \ldots, \overline{k}_i$ crews, then the extended model is able to minimize the project direct cost. In real life, the cost of hiring a crew may be different between activities owing to their differences in type of work, crew size, and technical level of workers. In such a situation, the extended model is more suitable for dealing with the DSPLOB&WI. If we choose stubbornly to schedule the project according to the objective of minimizing the total number of crews rather than minimizing the total direct cost of all activities, the obtained LOB schedule may be far from optimal and bring construction companies a loss [19].

$$\min \quad \sum_{i \in A} \sum_{k=1}^{\overline{k}_i} c_{ik} x_{ik}. \tag{9}$$

(3) If we remove the deadline constraint (7) and integrate the objective functions (1) and (2) using a cost function in which the project indirect cost (PIC), early completion incentives (ECI), and liquidated damage (LD) are further considered, then the modified model is the first model that can provide exact solutions for the cost optimization problem in LOB scheduling. As shown in (10), λ_i is the idle resource cost per crew per unit of time of activity i. PIC is a function of the time needed to complete the project; that is, PIC $= s_{n+1,m} \cdot$ ICR, where $s_{n+1,m}$ is used to evaluate the project duration and ICR is the indirect cost rate of the project. ECI is the monetary incentive provided by the owner to the contractor if the project is delivered before the given deadline; that is, ECI $= I \cdot \pi^-$, where I is the incentive per unit of time and π^- is the time when project completion is early. LD is the financial penalty paid by the contractor to the owner if the project is not completed within the desired deadline;

that is, $\mathrm{LD} = L \cdot \pi^+$, where L is the liquidated damage per unit of time and π^+ is the time when project completion is late. The values of π^+ and π^- are evaluated by (11a), (11b), (11c), (11d), (11e), and (11f), where M is a sufficiently large constant and $\omega \in \{0, 1\}$ and $\chi \in R_+$ are auxiliary variables. To solve this problem, Hegazy and Wassef [16] proposed a heuristic method that adopted genetic algorithms to determine the minimum total cost and integrated the commercial scheduling software, *Evolver*, to automate the model implementation. However, the performance of their method was verified only using several small-size projects with up to 10 activities and 19 units.

$$\min \quad \sum_{i \in A} \sum_{k=1}^{\bar{k}_i} c_{ik} x_{ik} + \sum_{i \in A} \lambda_i \sum_{j=1}^{m-1} y_{ij} + \mathrm{PIC} + \mathrm{ECI} - \mathrm{LD} \tag{10}$$

$$\pi^+ \geq \chi - \omega\delta \tag{11a}$$

$$\pi^- \leq \delta(1 - \omega) + \chi - s_{n+1,m} \tag{11b}$$

$$s_{n+1,m} - \delta + M(1 - \omega) \geq 0 \tag{11c}$$

$$\delta - s_{n+1,m} + M\omega \geq 0 \tag{11d}$$

$$\chi \leq M\omega \tag{11e}$$

$$s_{n+1,m} - M(1 - \omega) \leq \chi \leq s_{n+1,m}. \tag{11f}$$

The above discussions highlight the potential of the LOB-SOM model; that is, it can be easily extended or modified to handle other types of LOB-based scheduling optimization problems.

3. Effective Solution Method

Generally, optimization problems with several conflicting objectives have no single solution that can optimize all the objective functions simultaneously. In multiobjective programming, the concept of optimality is replaced with that of Pareto optimality. At this point, construction planners need to find the Pareto front of the problem being considered (i.e., the whole set of efficient solutions), rather than a single optimal solution. The efficient solutions are those that cannot be improved in one objective function without deteriorating their performance in any other.

The LOB-SOM refers to a biobjective programming. In literature, several methods have been proposed to solve this type of problem, for example, ϵ-constraint method [26], biobjective branch and bound [27], two-phase method [28], nondominated sorting genetic algorithm NSGA-II [29], multiobjective differential evolution MODE [30], and multiple objective symbiotic organisms search MOSOS [31], among which NSGA-II, MODE, and MOSOS are heuristic algorithms that provide a way to obtain good solutions but do not guarantee optimality.

In this section, an efficient version of the ε-constraint method presented by Mavrotas [32] is customized to generate the Pareto front of the DSPLOB&WI. The obtained efficient solutions are further improved by an automated procedure to reduce the number of interruptions for each activity. In this process, the total number of crews and total interruption duration corresponding to each solution remain unchanged.

3.1. ϵ-Constraint Method. The idea behind this method is to optimize one of the objective functions by using the rest as constraints. For the DSPLOB&WI, we choose to minimize the total number of crews and use another objective of minimizing the total duration of interruptions as a constraint. In doing so, the LOB-SOM is reformulated as the following augmented model:

$$\begin{aligned} \min \quad & \sum_{i \in A} \sum_{k=1}^{\bar{k}_i} k x_{ik} - \epsilon\eta \\ & \sum_{i \in A} \sum_{j=1}^{m-1} y_{ij} + \eta = e \\ & (3)\text{–}(7) \\ & x_{ik} \in \{0, 1\}; \quad s_{ij}, y_{ij}, \eta \geq 0, \end{aligned} \tag{12}$$

where ϵ is an adequately small number, η is an augmented variable, and e is a upper limit of the total interruption duration.

The augmented model relates to a parametric problem on e that provides efficient solutions by varying e. The range of e is $[0, \mathrm{WI}^+]$, where WI^+ represents the nadir value of the total interruption duration and can be determined using the lexicographical optimization strategy. Practically, this strategy is implemented as follows: we first calculate the minimum TNC^* of the total number of crews by solving the single-objective problem of (1) and (3)–(7). Then, the value of WI^+ is estimated by solving the single-objective problem of (2), (3)–(7), and an additional constraint $\sum_{i \in A} \sum_{k=1}^{\bar{k}_i} k x_{ik} = \mathrm{TNC}^*$, where the additional constraint is used to keep the optimal solution of the first optimization.

Usually, the range of e is divided into q equal intervals using $(q - 1)$ points. Consequently, the total number of runs for the augmented model becomes $(q + 1)$, and the value of e in the uth run is set to $(u - 1)(\mathrm{WI}^+/q)$.

Mavrotas [32] highlighted that the density of the Pareto front (measured in terms of the number of efficient solutions) can be controlled by properly assigning the value to q; that is, the larger the number of points, the more dense the Pareto front but with the cost of higher computation time. Therefore, a careful tradeoff between the computation time and the density of the Pareto front should be made.

3.2. Schedule Improvement. From the practitioners' point of view, frequent work interruptions may increase the complexity of site management and thus the burden on project managers. Inspired by the work of Long and Ohsato [3], this subsection equips the ϵ-constraint method with an automated procedure that can improve each efficient solution by reducing the number of interruptions in all activities without deteriorating the performance in both the objectives. If the

number of crews assigned to an activity under the LOB framework is fixed, the total interruption duration of this activity depends only on its start times at the first and last units. To maintain the Pareto optimality of all efficient schedules obtained by the ϵ-constraint method, only the activities in the intermediate units (i.e., from unit 2 to unit $m-1$) will be adjusted in the process of schedule improvement.

The schedule improvement procedure rearranges activities by an algorithm which includes the following two steps:

(i) *Backward*: this step starts with the end of the project and progresses backward. The goal is to start all activities in the intermediate units as late as possible. Pseudocode 1 lists the pseudocode of the "Backward step." Considering the LOB schedule in Figure 2(b), in which 5 days of interruptions must be added to activity B in order to complete the project within a total of 5 crews and without exceeding the given deadline, as shown in Figure 4(a), using the "Backward step" can reduce the total number of interruptions from 5 to 3.

(ii) *Forward*: this step is performed after the "Backward step" and aims to reduce the number of interruptions by pushing forward each activity in the intermediate units. However, if, after pushing forward, the start time of activity i in unit j is not connected to the value of $s_{i,j-1} + 1/r_i$ (i.e., there is still an interruption in between units $j-1$ and j), then the "Forward step" will give up pushing forward this activity in unit j and proceed to the next unit. The corresponding pseudocode can be found in Pseudocode 2. For the improved schedule in Figure 4(a), the "Forward step" further reduces the total number of interruptions to 1; see Figure 4(b).

Figure 5 exhibits the flowchart of the proposed method, where the augmented model is solved using the CPLEX MIP Solver. As mixed-integer linear programming problems are NP-hard, the solver may not terminate in a reasonable amount of time. For this reason, a maximum running time (e.g., one minute, one hour, or one day) is required. If the augmented model is solved within the time, an exact efficient solution will be obtained; otherwise, both the near-optimal solution and its deviation from optimality will be disclosed.

3.3. An Illustrative Example. To show the application of the present work, a highway project presented by Dolabi et al. [14] is analyzed. This project contains 24 serial activities with no buffers and is divided into 10 units. Activity descriptions, man-hour estimates to finish a unit, crew sizes, daily working hours, and unit durations are given in Table 1. The maximum available crew number is set to 10 for all activities and the deadline is 240 days. For the example project, all activities do not have to maintain strictly work continuity and therefore can be interrupted.

According to the ϵ-constraint method, the nadir value of the total interruption duration is determined by $WI^+ = 294$ days. We divide the range of parameter $e \in [0, 294]$ to 15 equal intervals and then the total number of runs for the augmented model becomes 16. In each run, the values of ϵ and $\overline{y}_i$ are fixed at 0.01 and 240, respectively.

Figure 6 shows the computational results of the example project. The total number of crews calculated from each run of the augmented model is presented in Figure 6(a). Based on this, the Pareto front is extracted and shown in Figure 6(b), which consists of 13 efficient solutions. If work interruptions are disallowed, a total of 63 crews must be employed to complete the project within the given deadline; otherwise, the total number of crews drops to 36 as the interruption duration increases, a saving of 42.86%. Figures 6(d) and 6(e) demonstrate the LOB schedules corresponding to the first and last efficient solutions, respectively. In Figure 6(e), 294 days of interruptions are apportioned among 14 activities, including B, D, E, F, G, H, I, J, L, M, P, T, U, and V. Figure 6(c) indicates that all efficient solutions are exact and can be found in a very short period of time (no more than 0.35 seconds).

Despite the limited scope of the analysis, the above results highlight the advantage of introducing work interruptions into the DPSLOB. Moreover, the constructed Pareto front can help construction planners with different preferences in determining the "most preferred" schedule when additional preference information is provided.

4. Computational Experiment

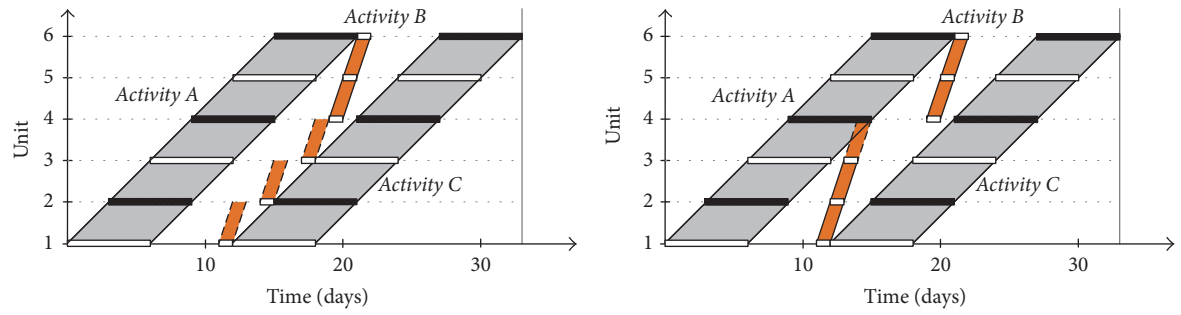
In this section, we design an experiment to analyze the performance of the LOB-SOM in solving the LOBDSP&WI, with the obtained results used to evaluate the impact of work interruptions on the planning and scheduling of repetitive projects. The proposed method is coded with MATLAB (R2015b) and run on a laptop with i7 CPU, 8 GB memory, and Window 10 operating system.

4.1. Test Instances. All instances are generated randomly based on three parameters, including the project size (measured by the number of real activities n and the number of units m), deadline tightness (θ), and coefficient of network complexity (CNC) that represents the average number of predecessors or successors per activity. Our parameter setting is listed in Table 2, where "CNC = 1" stands for serial projects in which each real activity has just one successor and predecessor. Besides that, the maximal number of available crews is set to 10 for each activity. Once the values of n, m, θ, and CNC are fixed, a random instance is generated using the following five steps:

(1) Construct a precedence relation network using the problem generator (ProGen) presented by [33] based on parameters n and CNC. ProGen is adopted because it can avoid producing redundant precedence constraints.

(2) Generate the unit duration d_i of each activity $j = 1, \ldots, n$ from the uniform distribution over the set of $\{1, 2, \ldots, 10\}$.

(3) Calculate the longest project duration (LPD) of the instance by performing each activity with only one crew and without interruptions.



(a) Backward step (b) Forward step

Figure 4: "Backward step" and "Forward step" for an illustrative example.

Table 1: Tabular presentation of project data.

Act	Description	Required man-hours to finish a unit	Number of workers in a crew	Daily working hours	Unit duration (days)
A	Mobilization	192	4	8	6
B	Surveying	64	2	8	4
C	Access road construction	576	6	8	12
D	Clearing grubbing	576	8	8	9
E	Earthmoving	1040	10	8	13
F	Ditch excavation	400	5	8	10
G	Culvert installation	80	5	8	2
H	Peat excavation and swamp backfill	192	6	8	4
I	Embankment	432	6	8	9
J	Utility work	160	4	8	5
K	Fine grading	256	4	8	8
L	Granular base (bulk)	336	6	8	7
M	Barrier walls	576	8	8	9
N	Granular base (fine)	384	6	8	8
O	Signage (foundations)	1600	10	8	20
P	Paving (base)	336	6	8	7
Q	Signage (installation)	288	4	8	9
R	High mast lighting	224	4	8	7
S	Paving (wearing course)	288	6	8	6
T	Shoulder granular	320	5	8	8
U	Lane marking	16	2	8	1
V	Landscaping	72	3	8	3
W	Checkout and acceptance	80	2	8	5
X	Demobilization	160	5	8	4

(4) Estimate approximately the shortest project duration (SPD) of the instance using the algorithm developed by Zou et al. [19], in which all activities are required to be performed continuously.

(5) Set the deadline $\delta = \text{SPD} + (\text{LPD} - \text{SPD})\theta$. Therefore, the smaller deadline tightness θ results in a tighter deadline constraint.

20 random instances are produced for each parameter level combination which leads to in total $3 \times 2 \times 3 \times 20 = 360$ instances. We further divide the range of e to 30 equal intervals and thus the Pareto front corresponding to each instance contains up to 31 efficient solutions. Finally, the experiment involves a total of $360 \times 31 = 11{,}600$ runs for the augmented model, and a maximum running time of 300 seconds is considered for each run.

4.2. Performance Analysis. The performance of the LOB-SOM is assessed using the following three indexes:

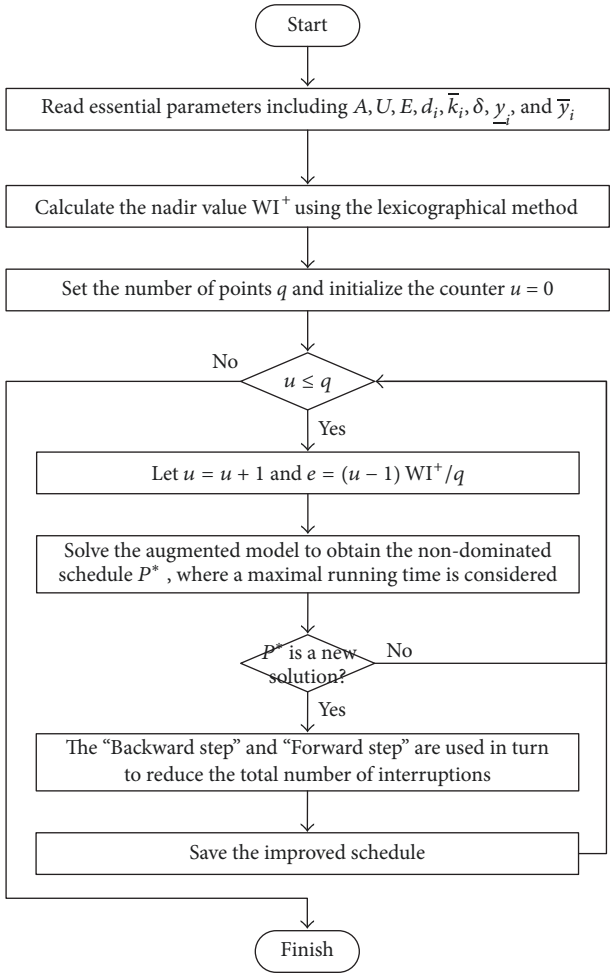


Figure 5: Flowchart of the solution method.

Table 2: Parameters of test problems.

Parameters	Values
Project size (n, m)	(60, 20), (90, 30), and (120, 40)
Coefficient of network complexity (CNC)	1 and 1.5
Deadline tightness (θ)	0.1, 0.3, and 0.5

(1) The average CPU time for the augmented model to complete a single run (ACT)

(2) The average number of unsolved runs of an instance (AUR)

(3) The average percentage of deviation from optimality (AD).

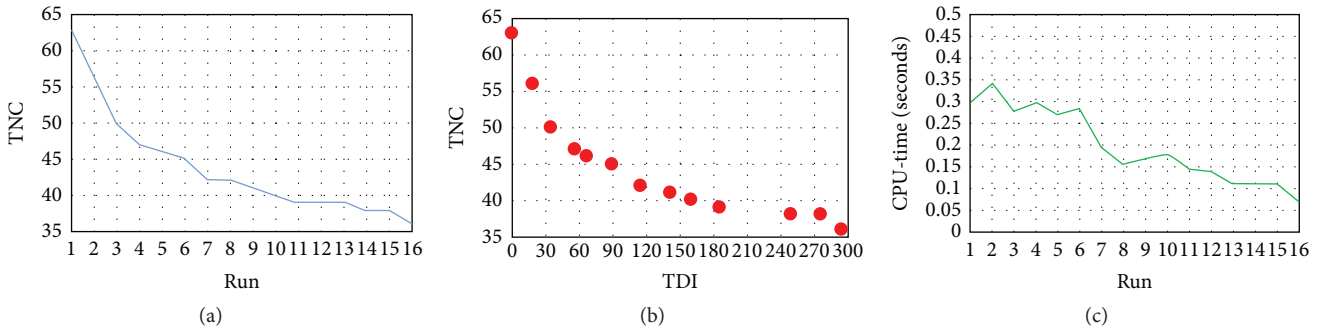
Equations (13) demonstrate formulas for calculating indexes ACT, AUR, and AD. In these equations, t_{uv} is the CPU time spent on the vth run of instance u. Particularly, we assume each unsolved run that fails to find an exact efficient solution contributes to the CPU time by 300 seconds. $w_{uv} = 0$ if the vth run of instance u is completed within 300 seconds, and $w_{uv} = 1$ otherwise. Ω_u is a set that consists of all unsolved runs of instance u. For each $u \in \{1, \ldots, 20\}$ and $v \in \Omega_u$, g_{uv} is the average percentage of deviation between the best solution found by the augmented model and the exact efficient solution, which can be known from the CPLEX MIP Solver.

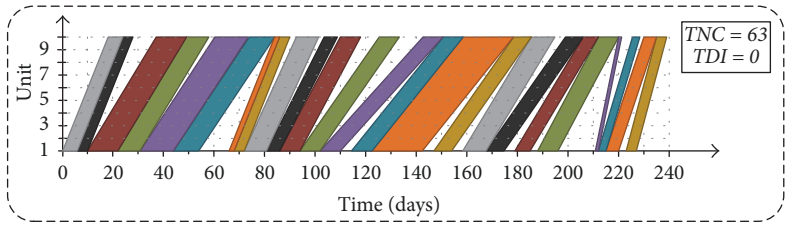
$$\begin{aligned} \text{ACT} &= \frac{1}{20 \times 31} \sum_{u=1}^{20} \sum_{v=1}^{31} t_{uv} \\ \text{AUR} &= \frac{1}{20} \sum_{u=1}^{20} \sum_{v=1}^{31} w_{uv} \\ \text{AD} &= \frac{1}{20} \sum_{u=1}^{20} \left(\frac{1}{|\Omega_u|} \sum_{v \in \Omega_u} g_{uv} \right). \end{aligned} \tag{13}$$

Table 3 details the computational results for different parameter level combinations. For all instances with up to 90 activities and 30 units, we obtain ACT $\le$ 40.62 seconds and AUR = AD = 0. This means that the LOB-SOM can provide small-size and medium-size problems with complete and exact Pareto fronts in a short time. Furthermore, we have ACT $\le$ 105.23 seconds, AUR $\le$ 1.6, and AD $\le$ 0.53% for all instances with 120 activities and 40 units. That is, in the worst-case scenario, an average of 5.16% of runs (1.6 out of 31) of an instance fail to find exact efficient solutions and their average deviation is less than or equal to 0.53%. This indicates that the LOB-SOM can produce high-quality solutions for large-size problems in a reasonable amount of time.

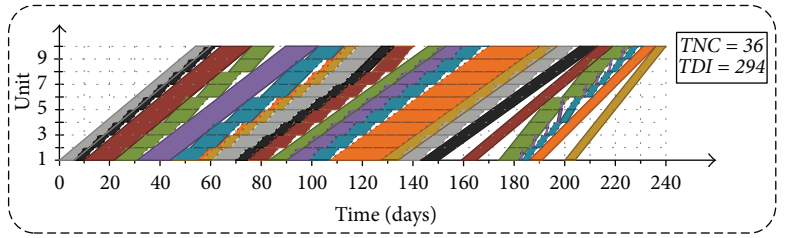
In our experiment, when the values of n and m are fixed, the instances with CNC = 1.5 (i.e., nonserial projects) and the instances with CNC = 1 (i.e., serial projects) have the same number of variables. The difference is that the former have a greater number of (precedence) constraints as they are equipped with a more complex precedence relation network. Observing Table 3 again, an interesting finding is that nonserial projects are earlier to solve than serial projects for any combination of n, m, and θ. For example, when the coefficient of network complexity (CNC) reduces from 1.5 to 1, the ACT corresponding to the instances with $(n, m, \theta) = (90, 30, 0.1)$ rapidly grows from 6.6 to 40.62 seconds. This finding may be a little hard to swallow and a possible explanation is as follows. In the DSPLOB&WI, the total number of crews can be reduced by interrupting an activity, only when this activity is critical and the progress rate of the activity must be faster than its critical predecessors and successors. For simplicity, we call this type of activity interruptible and an example is activity B in Figure 1. If an activity is interruptible, its interruption duration after each unit needs to be calculated carefully to balance the two conflicting objectives; otherwise, it has only to be performed without interruptions. When the number of activities (n) is given, scheduling serials projects will involve more interruptible activities theoretically, which eternally results in an increase in the computational effort of the solver.

Another noteworthy finding is that when the values of n, m, and CNC are fixed (which means that all instances have the same number of variables and constraints), there is a marked drop in all performance indexes, as the deadline





(d) The LOB schedule corresponding to the first nondominated solution



(e) The LOB schedule corresponding to the last nondominated solution

FIGURE 6: Computational results of the example project. *Note.* TNC = the total number of crew; TDI = the total duration of interruption.

(1) Read essential parameters from the optimization module, including the start time s_{ij} of activity i in unit j and the number of crews k_i assigned to activity i.
(2) Calculate the progress rate of each real activity i using the formula $r_i = k_i/d_i$
(3) **FOR** $i = n + 1$ **TO** 1 **DO** // *reschedule all activities in descending order*
(4) **IF** $\sum_{j=1}^{m-1} y_{ij} > 0$ **THEN** // *there exists at least one interruption in activity* i
(5) **FOR** $j = m - 1$ **TO** 2 **DO**
(6) $s_{ij} = \min_{l \in \mathrm{Suc}_i}\{s_{i,j+1} - 1/r_i, s_{lj} - d_i\}$ // Suc_i *consists of all successors of activity* i
(7) **ENDFOR**
(8) **ENDFOR**

PSEUDOCODE 1: The pseudocode of the "Backward step."

(1) Read essential parameters such as s_{ij} and r_i from the "Backward step"
(2) **FOR** $i = 1$ **TO** $n + 1$ **DO** // *reschedule all activities in ascending order*
(3) **IF** $\sum_{j=1}^{m-1} y_{ij} > 0$ **THEN**
(4) **FOR** $j = 2$ **TO** $m - 1$ **DO**
(5) **IF** $s_{i,j-1} + 1/r_i \geq \max_{h \in \mathrm{Pre}_i}\{s_{hj} + d_n\}$ **THEN**
(6) $s_{ij} = s_{i,j-1} + 1/r_i$ // Pre_i is a set of predecessors of activity i
(7) **ENDFOR**
(8) **ENDFOR**

PSEUDOCODE 2: The pseudocode of the "Forward step."

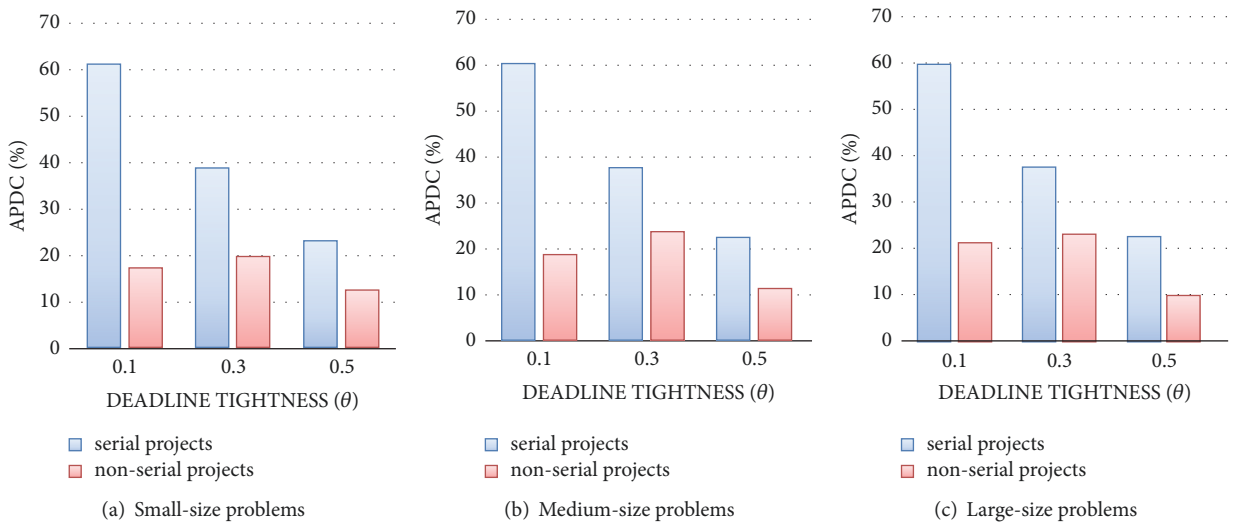


Figure 7: The APDC for different parameter level combinations.

Table 3: Computational results for different parameter level combinations.

(n, m)	θ	Serial projects (CNC = 1)			Nonserial projects (CNC = 1.5)		
		ACT (seconds)	AUR	AD (%)	ACT (seconds)	AUR	AD (%)
(60, 20)	0.1	5.23	0	0	1.14	0	0
	0.3	1.47	0	0	0.66	0	0
	0.5	0.98	0	0	0.46	0	0
(90, 30)	0.1	40.62	0	0	6.60	0	0
	0.3	10.68	0	0	3.29	0	0
	0.5	5.36	0	0	1.34	0	0
(120, 40)	0.1	105.23	1.6	0.42	47.69	1.4	0.53
	0.3	24.85	1	0.15	11.82	0.2	0.21
	0.5	17.88	0.6	0.07	2.61	0	0

tightness (θ) increases. For example, when the value of θ is fixed at 0.1, the ACT, AUR, and AD corresponding to the instances with $(n, m, \text{CNC}) = (120, 40, 1)$ are 105.23, 1.6, and 0.42%, respectively. If the value of θ is increased by 0.5, we obtain ACT = 17.88, AUR = 0.6, and AD = 0.07%. Such a distinction suggests that the LOB-SOM is able to handle lager projects with more than 120 activities and 40 units, if a loose deadline constraint is involved.

4.3. Impact of Work Interruptions. The impact of work interruptions on the planning and scheduling of repetitive projects is quantified by (14), where the index APDC means the average percent decrease in the total number of crews and z_u^f and z_u^l represent the optimal number of crews corresponding to the first and last run of instance u, respectively.

$$\text{APDC} = \frac{1}{20}\sum_{u=1}^{20}\frac{z_u^f - z_u^l}{z_u^f} \times 100\%. \tag{14}$$

Figure 7 shows the computational results of the APDC for different parameter level combinations. Specifically, the APDC corresponding to the instances with CNC = 1.5 is between 9.84% and 23.8%. When the coefficient of network complexity (CNC) reduces to 1, the corresponding APDC rapidly increases to 22.3%–61.15%. Moreover, with the reduction in the deadline tightness (θ), there is a marked increase in the APDC for serial projects of any size. For example, when the value of θ is reduced from 0.5 to 0.1, the APDC corresponding to the small-size problems with $n = 60$ and $m = 20$ rapidly increases from 23.3% to 61.15%. The above results highlight that introducing work interruptions can reduce the total number of crews effectively, especially for serial projects with a tight deadline constraint.

5. Concluding Remarks

This paper presented an exact model (i.e., the LOB-SOM) for solving the deadline satisfaction problem considering work interruptions for typical projects under the LOB framework. The LOB-SOM was implemented on a series of random instances, and the results indicated that construction planners can obtain effective tradeoff solutions for large-size projects within a reasonable amount of time. More importantly, the model is easy for construction planners to

use and does not need them to master any optimization theory.

The contributions of our study can be summarized as follows. First, the proposed LOB-SOM has the capability of balancing the two important and conflicting objectives, which enhances the practicability of solutions. Second, a customized ϵ-constraint method was proposed, which can help construction planners in generating all feasible tradeoff solutions to determine the optimal schedule in line with their preference. Third, we analyzed the impact of work interruptions on the planning and scheduling of repetitive projects by a computational experiment with 360 random instances. Such experiments have never been conducted in literature, and the results presented in our paper may be used as benchmarks to evaluate the performance of other future methods.

The main limitation of the LOB-SOM is that it can work well only for repetitive projects with identical activities such as high-rise building and housing project with identical units. For *nontypical* projects in which the activity duration is variable for different units, the results obtained by the model may be far from optimal. Furthermore, the LOB-SOM only focuses on the nonrenewable resources (e.g., money) and does not take the renewable resources into account. Therefore, future work can focus on improving the performance of the proposed model by introducing renewable resources and their constraints into the DSPLOB&WI.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to acknowledge the Natural Science Foundation of China (71701069) and Fundamental Research Funds for the Central Universities (2017MS175).

References

[1] K. G. Mattila and D. M. Abraham, "Linear scheduling: Past research efforts and future directions," *Engineering, Construction and Architectural Management*, vol. 5, no. 3, pp. 294–303, 1998.

[2] M. A. Ammar, "LOB and CPM integrated method for scheduling repetitive projects," *Journal of Construction Engineering and Management*, vol. 139, no. 1, pp. 44–50, 2013.

[3] L. D. Long and A. Ohsato, "A genetic algorithm-based method for scheduling repetitive construction projects," *Automation in Construction*, vol. 18, no. 4, pp. 499–511, 2009.

[4] S. Selinger, "Construction planning for linear projects," *Journal of the Construction Division*, vol. 106, no. 2, pp. 195–205, 1980.

[5] K. El-Rayes and O. Moselhi, "Optimizing resource utilization for repetitive construction projects," *Journal of Construction Engineering and Management*, vol. 127, no. 1, pp. 18–26, 2001.

[6] K. Hyari and K. El-Rayes, "Optimal planning and scheduling for repetitive construction projects," *Journal of Management in Engineering*, vol. 22, no. 1, pp. 11–19, 2006.

[7] M. Vanhoucke and D. Debels, "The discrete time/cost trade-off problem: Extensions and heuristic procedures," *Journal of Scheduling*, vol. 10, no. 4-5, pp. 311–326, 2007.

[8] K. H. Hyari, K. El-Rayes, and M. El-Mashaleh, "Automated trade-off between time and cost in planning repetitive construction projects," *Construction Management and Economics*, vol. 27, no. 8, pp. 749–761, 2009.

[9] X. Zou, S.-C. Fang, Y.-S. Huang, and L.-H. Zhang, "Mixed-Integer Linear Programming Approach for Scheduling Repetitive Projects with Time-Cost Trade-Off Consideration," *Journal of Computing in Civil Engineering*, vol. 31, no. 3, Article ID 6016003, 2016.

[10] Y. Tang, R. Liu, F. Wang, Q. Sun, and A. A. Kandil, "Scheduling Optimization of Linear Schedule with Constraint Programming," *Computer-Aided Civil and Infrastructure Engineering*, 2017.

[11] D. J. Harmelink and J. E. Rowings, "Linear scheduling model: Development of controlling activity path," *Journal of Construction Engineering and Management*, vol. 124, no. 4, pp. 263–268, 1998.

[12] R. M. Reda, "RPM: Repetitive Project Modeling," *Journal of Construction Engineering and Management*, vol. 116, no. 2, pp. 316–330, 1990.

[13] G. Lucko, "Productivity scheduling method: Linear schedule analysis with singularity functions," *Journal of Construction Engineering and Management*, vol. 135, no. 4, pp. 246–253, 2009.

[14] H. R. Z. Dolabi, A. Afshar, and R. Abbasnia, "CPM/LOB scheduling method for project deadline constraint satisfaction," *Automation in Construction*, vol. 48, pp. 107–118, 2014.

[15] S.-L. Fan, K.-S. Sun, and Y.-R. Wang, "GA optimization model for repetitive projects with soft logic," *Automation in Construction*, vol. 21, no. 1, pp. 253–261, 2012.

[16] T. Hegazy and N. Wassef, "Cost optimization in projects with repetitive nonserial activities," *Journal of Construction Engineering and Management*, vol. 127, no. 3, pp. 183–191, 2001.

[17] T. Hegazy, A. Elhakeem, and E. Elbeltagi, "Distributed scheduling model for infrastructure networks," *Journal of Construction Engineering and Management*, vol. 130, no. 2, pp. 160–167, 2004.

[18] A. Damci, D. Arditi, and G. Polat, "Resource leveling in line-of-balance scheduling," *Computer-Aided Civil and Infrastructure Engineering*, vol. 28, no. 9, pp. 679–692, 2013.

[19] X. Zou, Q. Zhang, and L. Zhang, "Modeling and Solving the Deadline Satisfaction Problem in Line-of-Balance Scheduling," *Journal of Management in Engineering*, vol. 34, no. 1, Article ID 04017044, 2018.

[20] M. M. Ali and A. Elazouni, "Finance-based CPM/LOB scheduling of projects with repetitive non-serial activities," *Construction Management and Economics*, vol. 27, no. 9, pp. 839–856, 2009.

[21] S. A. Suhail and R. H. Neale, "CPM/LOB: New methodology to integrate cpm and line of balance," *Journal of Construction Engineering and Management*, vol. 120, no. 3, pp. 667–684, 1994.

[22] D. Arditi, O. B. Tokdemir, and K. Suh, "Scheduling system for repetitive unit construction using line-of-balance technology," *Engineering, Construction and Architectural Management*, vol. 8, no. 2, pp. 90–103, 2001.

[23] O. B. Tokdemir, D. Arditi, and C. Balcik, "ALISS: advanced linear scheduling system," *Construction Management and Economics*, vol. 24, no. 12, pp. 1253–1267, 2006.

[24] P. G. Ioannou and I.-T. Yang, "Repetitive Scheduling Method: Requirements, Modeling, and Implementation," *Journal of Construction Engineering and Management*, vol. 142, no. 5, Article ID 04016002, 2016.

[25] M. Vanhoucke, "Work continuity constraints in project scheduling," *Journal of Construction Engineering and Management*, vol. 132, no. 1, pp. 14–25, 2006.

[26] M. Laumanns, L. Thiele, and E. Zitzler, "An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method," *European Journal of Operational Research*, vol. 169, no. 3, pp. 932–942, 2006.

[27] T. Stidsen, K. A. Andersen, and B. Dammann, "A branch and bound algorithm for a class of biobjective mixed integer programs," *Management Science*, vol. 60, no. 4, pp. 1009–1032, 2014.

[28] E. L. Ulungu and J. Teghem, "The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems," *Foundations of Computing and Decision Sciences*, vol. 20, no. 2, pp. 149–165, 1995.

[29] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[30] M. Ali, P. Siarry, and M. Pant, "An efficient Differential Evolution based algorithm for solving multi-objective optimization problems," *European Journal of Operational Research*, vol. 217, no. 2, pp. 404–416, 2012.

[31] D.-H. Tran, M.-Y. Cheng, and D. Prayogo, "A novel Multiple Objective Symbiotic Organisms Search (MOSOS) for time-cost-labor utilization tradeoff problem," *Knowledge-Based Systems*, vol. 94, pp. 132–145, 2016.

[32] G. Mavrotas, "Effective implementation of the ϵ-constraint method in multi-objective mathematical programming problems," *Applied Mathematics and Computation*, vol. 213, no. 2, pp. 455–465, 2009.

[33] R. Kolisch, A. Sprecher, and A. Drexl, "Characterization and Generation of a General Class of Resource-Constrained Project Scheduling Problems," *Management Science*, vol. 41, no. 10, pp. 1693–1703, 1995.