TABU SEARCH AND SIMULATED ANNEALING FOR RESOURCE-CONSTRAINED MULTI-PROJECT SCHEDULING TO MINIMIZE MAXIMAL CASH FLOW GAP

Yukang He
School of Engineering and Applied Science, Aston University, Birmingham, B4 7ET, United Kingdom
Zhengwen He and Nengmin Wang
School of management, Xi’an Jiaotong University, Xi’an 710049, China

(Communicated by Ruhul Sarker)

ABSTRACT. In reality, a contractor may implement multiple projects simultaneously and in such an environment, how to achieve a positive balance between cash outflow and inflow by scheduling is an important problem for the contractor to tackle. For this fact, this paper investigates a resource-constrained multi-project scheduling problem with the objective of minimizing the contractor’s maximal cash flow gap under the constraint of a project deadline and renewable resource. In the paper, we construct a non-linear integer programming optimization model for the studied problem at first. Then, for the NP-hardness of the problem, we design three metaheuristic algorithms to solve the model: tabu search (TS), simulated annealing (SA), and an algorithm comprising both TS and SA (SA-TS). Finally, we conduct a computational experiment on a data set coming from existing literature to evaluate the performance of the developed algorithms and analyze the effects of key parameters on the objective function. Based on the computational results, the following conclusions are drawn: Among the designed algorithms, the SA-TS with an improvement measure is the most promising for solving the problem under study. Some parameters may exert an important effect on the contractor’s maximal cash flow gap.

1. Introduction. Over the course of a project in reality, the contractor often incurs a series of cash flows occurring in the following two forms ([20]): cash outflow—induced mainly by the execution of activities and the use of resources such as labour, equipment, and materials—and cash inflow—resulting generally from payments for the completion of specified parts of the project according to the contract terms. Based on this, it is easy to understand that during the implementation of the project, keeping a positive balance between cash outflow and inflow is key to effective cash flow management, since, if the outflow cannot be covered by the inflow in a timely...
manner, the contractor may not be able to implement the project smoothly. In such a case, the contractor has to raise money externally to cover the gap between cash outflow and inflow and hence incur an extra financing cost. Moreover, when the gap exceeds the contractor’s financing capacity, the problem may cause project failure or even the bankruptcy of the contractor.

Essentially speaking, the distribution of cash flow over the course of the project is closely related to the arrangement of the project schedule. On the one hand, the magnitude of cash outflow depends on the amount of the resource devoted to activities; and when it occurs is mainly determined by the assigned start time for activities. On the other hand, as the client often pays the contractor based on the progress of the project according to payment terms stipulated in the contract, each cash inflow and its timing rely primarily on the achievement of the project schedule. As a result, through reasonable project scheduling, the contractor can coordinate cash outflow and inflow effectively, thus, minimizing cash flow gap and achieving a positive cash flow balance. In fact, this problem is more meaningful in a multi-project context since, in practice, the contractor may implement multiple projects simultaneously ([9]). In this context, the contractor must arrange schedule for each project and manage all the cash flows for the different projects as a whole. During the implementation of a given project, if the positive balance between cash outflow and inflow falls short, to ensure the project continues smoothly, the contractor needs to extract cash from the other projects or raise money externally to cover the cash flow gap. When this happens, the implementation of the projects could be affected and the contractor’s cost could increase. Thus, in the multi-project context, the contractor needs to arrange the project schedules carefully and try to minimize the cash flow gap for all projects during their implementation.

Based on the discussion above, it is easy to understand that how to schedule projects to minimize the cash flow gap in a multi-project context is a practical and meaningful problem that should be investigated intensively. However, to the best of our knowledge, such a research has not been performed previously, which is demonstrated in the next section through a literature review. For the facts aforementioned, in this paper, we investigate a resource-constrained multi-project scheduling problem with the objective of minimizing the maximal cash flow gap. In the problem, a contractor needs to implement multiple projects concurrently, with each project having its own earliest start time, deadline, and availability of renewable resource. Considering that resources for executing an activity must be procured before starting the activity and the contractor tends to arrange the expense caused by such procurement to occur as late as possible, we assume that over the course of the projects, the cash outflows occur at the start of the activities. The cash inflows, namely, the clients’ payments which occur at the completion of milestone activities, are received based on the project progress, which is measured by the accumulative earned value of the activities finished by the contractor. During the implementation of the projects, the cash flow can be shared among the projects freely and at a certain time, and the cash flow gaps are defined as the accumulative cash outflow minus the accumulative cash inflow. Among the cash flow gaps, the greatest one is the maximal cash flow gap, and our aim is to derive the optimal project schedule to minimize the maximal cash flow gap such that the contractor can implement multiple projects smoothly. We believe that this research provides valuable decision support for a contractor to manage cash flows, especially in the
cases where the contractor’s financing capacity is insufficient for completing the projects.

The remainder of this paper is organized as follows. We give a literature review and discuss the contribution of our research in section 2. Section 3 presents the construction of the optimization model for the studied problem, while section 4 describes the design of the metaheuristic algorithms for solving the model. In section 5, a computational experiment is presented to evaluate the algorithms, and we summarize our conclusions in section 6.

2. Literature review. Project scheduling, a classical problem in project management, has gained increasing attention in the literature over decades ([30]). Within this area of research, project scheduling with the consideration of financial aspects, wherein the classic objective is to maximize the net present value (NPV) of cash flows in projects, is an important branch ([17]). In the branch, a literature stream relevant to our study is the capital-constrained project scheduling problem, which is proposed originally by Doersch and Patterson ([8]). In their study, a zero-one integer programming formulation was constructed to maximize the discounted value of cash flows while the positive cash flow balance was treated as a capital constraint. Following the groundbreaking work of Doersch and Patterson ([8]), Smith-Daniels and Smith-Daniels ([31]) integrated materials constraint into the problem whereas Smith-Daniels et al. ([32]) presented three heuristic procedures and tested their performance in solving relatively large capital-constrained projects. Considering that activities could be performed with one of several alternative modes, Özdamar and Dündar ([28]) proposed a flexible heuristic for scheduling housing projects to maximize financial returns where capital was considered as a limited nonrenewable resource which was reduced by activity expenditures and augmented by the sales of flats. Özdamar ([29]) investigated a multi-mode capital-constrained problem where the contractor had to construct and reconstruct schedules during the progress of the projects to maintain a positive cash balance dynamically. By combining the project payment scheduling problem with the capital-constrained problem, He et al. ([18]) assessed a multi-mode, capital-constrained project payment scheduling problem and developed metaheuristics for the solution.

More recently, to improve the project NPV, Leyman and Vanhoucke ([21]) introduced a novel schedule construction method for the resource-constrained project scheduling problem (RCPSP) with discounted cash flows. The method consisted of the two steps: the first step grouped the activities based on the predecessors and successors in the project network and added these activities to a set based on their finish time and cash flow while the second step did so based on the neighbour activities in the schedule, which might need not to include precedence related activities. The proposed method was implemented in a genetic algorithm metaheuristic and an extensive computational experiment shown the results were competitive with existing work. In addition, Leyman and Vanhoucke ([22]) extended their method to the multi-mode RCPSP with discounted cash flows and three payment models. The method was implemented within a genetic algorithm metaheuristic where two penalty functions, one for deadline feasibility and one for non-renewable resource feasibility, were employed. Using several datasets from literature, the proposed method was tested and the added value of each part of the method was illustrated. Based on the work aforementioned, Leyman and Vanhoucke ([23]) studied the capital- and resource-constrained project scheduling problems with the objective
of maximizing the project NPV, where capital constraints were included to force the project to always have a positive cash balance. In the paper, they proposed three distinct cash flow models that affected the capital availability during the project, and introduced two new schedulers that focused on delaying sets of activities which caused cash outflows to be received at later time.

Different from the researches aforementioned where the NPV of project is maximized under a capital constraint by which the positive cash flow balance is maintained, Elazouni and Gab-Allah ([10]) proposed another relevant problem, namely the finance-based scheduling problem. In the problem, with the objective of minimizing the total project duration, a financially feasible schedule was devised to balance the financing requirements of activities at any period with the cash available during the same period. Following Elazouni and Gab-Allah ([10]), Alghazi et al. ([1]) designed an improved genetic algorithm for the large-size finance-based scheduling problem while Ali and Elazouni ([2]) integrated the line of balance technique with the finance-based scheduling for projects with repetitive non-serial activities. Elazouni et al. ([11]) compared the performance of the genetic algorithm (GA), simulated annealing (SA) and shuffled frog-leaping algorithm (SFLA) in solving discrete versus continuous-variable optimization problems of the finance-based scheduling. The obtained results indicated that the SA outperformed the SFLA and GA for the small-size networks of 30 activities whereas it exhibited the least total duration for the large-size networks of 120 and 210 activities after the prolonged processing time. Assuming that contractors could depend on bank overdrafts to finance their expenses, Al-Shihabi and AlDurgam ([4]) developed max-min ant system algorithms, which used different heuristic information when generating solutions, to solve the finance-based scheduling problem. Further research efforts were made by Fathi and Afshar ([15]) to consider multiple objectives, including the total duration of project, the required credit that had to be procured by bank overdrafts, and the financing cost incurred, in the finance-based scheduling problem.

Some researches have extended the finance-based scheduling problem to a multi-project context where a contractor needs to handle multiple projects simultaneously. For instances, Elazouni ([12]) proposed a heuristic method for the multi-project finance-based scheduling problem and validated its effectiveness by comparing the results with the optimum results obtained by using the integer programming for 15 networks comprising up to 60 activities. Liu and Wang ([24]) considered cash flow and the financial requirements of contractors working in a multiple project environment and proposed a profit optimization model for multiproject scheduling problems using constraint programming. They presented a hypothetical example involving three projects to illustrate the capability of the proposed mode and adopted various constraints, including credit limit and due dates, for scenario analysis. Under cash-constrained conditions, Elazouni and Abido ([13]) developed a strength Pareto evolutionary algorithm to devise Pareto-optimal finance-based schedules of multiple projects, where the problem of multiple projects profit maximization was formulated as a multiobjective optimization problem in which the profit values of the individual projects constituted a set of multiple conflicting objectives. Taking the minimization of the conflicting objectives of financing costs, duration of the group of projects, and the required credit into account, Abido and Elazouni ([3]) employed the modified strength Pareto evolutionary algorithm to obtain the Pareto-optimal fronts for the finance-based scheduling of multiple projects, and used a fuzzy-based technique to help contractors select the best compromise
solution from the Pareto-optimal fronts. El-Abbasy et al. ([14]) presented the development of a multi-objective scheduling optimization model for multiple projects using the fast elitist non-dominated sorting genetic algorithm by which the optimal trade-offs between different projects’ objectives, including total duration, total cost, financing cost, required credit, profit, resource fluctuations, and peak demand, could be obtained.

Besides the capital-constrained project scheduling problem and the finance-based scheduling problem where the positive cash flow balance is considered as a constraint when scheduling projects, in the last two years, there have appeared researches that take the positive cash flow balance as an objective to generate a schedule by which the maximal cash flow gap is minimized in a single-project context. Supposing that activities could be performed with several discrete modes, He et al. ([19]) presented a project scheduling problem where the objective was to minimize the maximal cash flow gap, which was defined as the greatest gap between the accumulative cash inflows and outflows over the course of the project. Due to the NP-hardness of the problem, the mixed and nested versions of variable neighbourhood search, tabu search, and variable neighbourhood search with tabu search were developed and they were evaluated through an extensive computational experiment. Ning et al. ([26]) extended the research of He et al. ([19]) to the uncertain environment where the durations of activities were assumed to be stochastic variables. A robust baseline schedule was generated to minimize the contractor’s maximal cumulative gap by using two metaheuristics, namely simulated annealing and tabu search, which owned different search structures and were equipped with an improvement measure. Moreover, to manage the effect of disruptions caused by activity durations’ randomness on the positive cash flow balance, Ning et al. ([27]) constructed and solved a comprehensive optimization model with time buffers added to the baseline schedule by proactive project scheduling and the schedule adjustment cost determined by reactive project scheduling.

The literatures reviewed above are summarized in Table 1, from which it can be seen clearly that the research gap we want to fill in this paper. The table indicates that there are a lot of researches that have considered the issue of positive cash flow balance in project scheduling, and most of them take this issue as a constraint in single-project or multi-project contexts, where the objective is to maximize project profit, to minimize project duration, or trade-off among multiple objectives. There are also a few papers which take the positive cash flow balance as an objective in a single-project context under the assumptions that the durations of activities are constants or stochastic variables. However, in the multi-project context, we have not found any researches regardless of activity duration being constants or stochastic variables and hence, the problem of how to schedule multiple projects to balance cash outflow and inflow positively has not been tackled so far. The research in this paper, which is marked as bold font with gray background in the table, is aimed at filling this gap under the assumption that the durations of activities are constants. Therefore, the research may have an important implication for the research area.

3. Problem formulation.

3.1. Optimization model. In the project, there are $n^h$ activities of which activities 1 and $n^h$ are the dummy start and end activities, respectively, and the others are all non-dummy activities. The execution of activity $i$ ($i=1,2,\cdots,n^h$) requires $r_{ik}$ units of renewable resource $k$ ($k=1,2,\cdots,K$) per period, and the activity’s duration,
Table 1. Summary of Reviewed Literature

| The positive cash flow balance is taken as a constraint | The positive cash flow balance is taken as an objective |
|--------------------------------------------------------|--------------------------------------------------------|
| The objective is to maximize project profit            | The objective is to minimize project duration          |
|                                                        | The objective is the optimal trade-off among multiple objectives |
|                                                        | The activity durations are constants                   |
|                                                        | The activity durations are stochastic variables        |

A contractor needs to implement a single project

| Doersch and Patterson ([8]); Smith-Daniels and Smith-Daniels ([31]); Smith-Daniels et al. ([32]); Özdamar and Dündar ([28]); Özdamar ([29]); He et al. ([18]); Leyman and Vanhoucke ([21]); Leyman and Vanhoucke ([22]); Leyman and Vanhoucke ([23]) | Elazouni and Gab-Allah ([10]); Alghazi et al. ([11]); Ali and Elazouni ([2]); Elazouni et al. ([11]); Al-Shahabi and AlDurgam ([4]) | Fathi and Afshar ([15]); He et al. ([19]); Ning et al. ([26]); Ning et al. ([27]) |

A contractor needs to implement multiple projects concurrently

| Liu and Wang ([24]) | Elazouni and Abido ([13]); Abido and Elazouni ([3]); El-Abbasy et al. ([14]) | Elazouni and Abido ([13]); Abido and Elazouni ([3]); El-Abbasy et al. ([14]) |

This paper

Cost, and earned value are $d_i$, $c_i$, and $v_i$, respectively. It should be noted that as the two dummy activities do not exist in reality, their $r_k$, $d_i$, $c_i$, and $v_i$, are a constant that equals 0. For project $h$, the earliest start time, deadline, availability of renewable resource $k$, and contract price are $EST^h$, $D^h$, $R_k^h$, and $U^h$ ($U^h = \sum_{i=1}^{n^h} v_i$), respectively.

The cash outflow for the contractor to complete activity $i$, namely, $c_i$, is assumed to occur at the activity’s start time. Let us assume that under the constraints of the earliest start time $EST^h$, project deadline $D^h$, renewable resource availability $R_k^h$, and a precedence relationship among the project’s activities, the start time of activity $i$ is arranged as $s_i$. Then, the start time of all the activities constitutes a schedule for project $h$, which is represented as $S^h=(s_1,s_2,\cdots,s_{n^h})$, and thus, the schedule of the $H$ projects is denoted as $S=(S_1,S_2,\cdots,S^H)$. Among the $H$ projects, we represent the start time of the project that is begun earliest as $s^\text{min}$ and the completion time of the project that is completed last as $s^\text{max}$. Given an $S$, the
contractor’s accumulative cash outflow at time \( t \) \((t \in [s_{1}^{\min}, s_{n}^{\max}])\), which is denoted as \( ACO_{t} \), is calculated using the formula \( ACO_{t} = \sum_{h=1}^{H} \sum_{i=1}^{n_{h}} c_{i} \).

Among the \( n_{h} \) activities, \( M_{h} (M_{h} \leq n_{h}) \) ones are defined as milestone activities on which a payment occurs while others are all non-milestone activities to which no payment is attached. During the implementation of the project, when a milestone activity is finished, the client makes a payment to the contractor. The amount of the \( m \)-th payment, \( p_{m} (m=1,2,\cdots,M_{h}-1) \), equals the product of the contractor’s earned value accumulated from the \((m-1)\)-th payment to this payment and the compensation proportion of project \( h \), \( \theta_{h} (0 \leq \theta_{h} \leq 1) \), i.e., \( p_{m} = \theta_{h} (\sum_{i \in AS_{m}^{h}} v_{i} - \sum_{i \in AS_{m-1}^{h}} v_{i}) \), where \( AS_{m}^{h} \) and \( AS_{m-1}^{h} \) are the sets of the activities that are finished by the \( m \)-th and \((m-1)\)-th payments, respectively. When the project is finished, the sum of payments must equal the contract price of the project, i.e., \( p_{M_{h}} = U^{h} - \sum_{m=1}^{M_{h}-1} p_{m} \). We denote the accumulative cash inflow at time \( t \) as \( ACI_{t} \). Then, given an \( S \), \( ACI_{t} \) is computed using the formula, \( ACI_{t} = H \sum_{h=1}^{H} \sum_{m=1}^{M_{h}} \sum_{i \in O_{m}^{h}} \left( s_{i} + d_{i} \leq s_{j}, \quad (i,j) \in A^{h}, \quad h = 1,2,\cdots,H. \right) \)

\( s_{1} \geq EST^{h}, \quad h = 1,2,\cdots,H. \)

\( s_{i} + d_{i} \leq s_{j}, \quad (i,j) \in A^{h}, \quad h = 1,2,\cdots,H. \)

\( s_{n_{h}} \leq D^{h}, \quad h = 1,2,\cdots,H. \)

\( \sum_{i \in O_{t}^{h}} r_{i,k} \leq R_{k}^{h}, \quad t \in [s_{1}, s_{n_{h}}], \quad h = 1,2,\cdots,H. \)

\( p_{m} = \theta_{h} (\sum_{i \in AS_{m}^{h}} v_{i} - \sum_{i \in AS_{m-1}^{h}} v_{i}) \), \( m = 1,2,\cdots,M_{h}-1, \quad h = 1,2,\cdots,H. \)

\( p_{M_{h}} = U^{h} - \sum_{m=1}^{M_{h}-1} p_{m}, \quad h = 1,2,\cdots,H. \)
\[ ACO_t = \sum_{h=1}^{H} \sum_{i=1}^{n_h} c_{i}, \quad t \in [s_{1h}^{\min}, s_{n_h}^{\max}]. \]  \hspace{1cm} (8)

\[ ACI_t = \sum_{h=1}^{H} \sum_{m=1}^{M_h} p_m, \quad t \in [s_{1h}^{\min}, s_{n_h}^{\max}]. \]  \hspace{1cm} (9)

\[ G_t = ACO_t - ACI_t, \quad t \in [s_{1h}^{\min}, s_{n_h}^{\max}]. \]  \hspace{1cm} (10)

\( s_i \) is a nonnegative integer, \( i = 1, 2, \cdots, n_h, h = 1, 2, \cdots, H. \)  \hspace{1cm} (11)

In the model constructed above, the objective is to minimize the contractor’s maximum cash flow gap in the multi-project context. Constraint (2) ensures project \( h \) cannot start earlier than \( EST_h \). Constraint (3) maintains the precedence feasibility, where \( A^h \) is the set of the precedence relationships among the activities. A deadline is imposed for project \( h \) by constraint (4) while the limitation of the renewable resource availability is represented by constraint (5), in which \( O^h_t \) denotes the set of project \( h \)’s activities that are in progress at time \( t \). Constraints (6) and (7) are the formulae used to determine payments, whereas constraints (8) and (9) are used to calculate the accumulative cash outflow and inflow at time \( t \), respectively. Constraint (10) is used to compute the cash flow gaps, and constraint (11) defines the value scope of the decision variables.

3.2. Properties of the problem. The function of the above model is to minimize \( G_{\text{max}} \) by arranging the start time of the activities under the constraints; hence, obtaining the optimal \( S \) for the multiple projects implemented concurrently. Given an \( S \), the time and amount of cash outflows and inflows in the projects (which depend completely upon the arrangement of the activities’ start time) are determined; hence, \( G_{\text{max}} \) is subsequently obtained. By adjusting \( S \) and comparing the obtained \( G_{\text{max}} \), we can determine the optimal \( S \) that minimizes \( G_{\text{max}} \). We denote the occurrence time of \( G_{\text{max}} \) under the given \( S \) as \( T_{\text{max}} \) and the time of the latest payment occurring no later than \( T_{\text{max}} \) as \( LPT \). It should be noted that if \( G_{\text{max}} \) occurs at several different times, we consider the earliest one as \( T_{\text{max}} \). Based on the aforementioned facts, we propose the following three theorems of the studied problem, which may be helpful for obtaining the solution of the model.

**Theorem 3.1.** Given an \( S \) and its \( G_{\text{max}} \), if the start time of a non-milestone activity \( s_i \) satisfies \( LPT - d_i < s_i \leq T_{\text{max}} \) and can be delayed to a time later than \( T_{\text{max}} \) without any change in the other activities’ start time and any violation of renewable resource constraints, performing such an operation may result in a decrease in \( G_{\text{max}} \).

**Proof.** Given an \( S \), \( ACO_{T_{\text{max}}} \) consists of the cost of the activities having a start time no later than \( T_{\text{max}} \), while \( ACI_{T_{\text{max}}} \) equals the sum of the earned value of the activities having a completion time that is no later than \( LPT \) multiplied by \( \theta^h \). If the start time of the non-milestone activity \( i \), i.e., \( s_i \), satisfies \( LPT - d_i < s_i \leq T_{\text{max}} \), this activity is started no later than \( T_{\text{max}} \) but is completed later than \( LPT \). As a result, the cost of this activity is added to \( ACO_{T_{\text{max}}} \) but the compensation for it is not considered in \( ACI_{T_{\text{max}}} \). Therefore, under the condition that the start time of other activities remains unchanged, delaying the start of this activity to a time later
than $T_{\text{max}}$ may reduce $ACO_{T_{\text{max}}}$ but generate no effect on $ACI_{T_{\text{max}}}$, thus causing $G_{\text{max}}$, which equals $ACO_{T_{\text{max}}} - ACI_{T_{\text{max}}}$, to decrease.

**Theorem 3.2.** Given an $S$ and its $G_{\text{max}}$, if the start time of a non-milestone activity $s_i$ satisfies $LPT - d_i < s_i \leq T_{\text{max}}$ and can be advanced no later than $LPT - d_i$ without any change in other activities’ start time and any violation of renewable resource constraints, performing such an operation may decrease the $G_{\text{max}}$.

**Proof.** The principle behind this theorem is similar to that of theorem 1. For the non-milestone activity with a start time that satisfies $LPT - d_i < s_i \leq T_{\text{max}}$, its cost is accumulated to $ACO_{T_{\text{max}}}$, while the payment for it cannot be calculated in $ACI_{T_{\text{max}}}$. If the start of this activity is advanced no later than $LPT - d_i$, the activity will be completed before or at $LPT$. This causes its earned value to be accumulated to the payment occurring at $LPT$, and thus $ACI_{T_{\text{max}}}$ increases. In this operation, as the other activities’ start times remain unchanged, $ACO_{T_{\text{max}}}$ does not vary, thus leading to the $G_{\text{max}}$ to drop.

**Theorem 3.3.** Given an $S$ and its $G_{\text{max}}$, if the start time of a milestone activity $s_{i_m}$ satisfies $T_{\text{max}} - d_{i_m} < s_{i_m} \leq T_{\text{max}}$ and can be advanced to $T_{\text{max}} - d_{i_m}$ without any change in other activities’ start times and any violation of renewable resource constraints, performing such an operation may result in a decrease in $G_{\text{max}}$.

**Proof.** In a given $S$, if the start time of a milestone activity $s_{i_m}$ meets $T_{\text{max}} - d_{i_m} < s_{i_m} \leq T_{\text{max}}$, this milestone activity will be started no later than $T_{\text{max}}$ and completed later than $T_{\text{max}}$. Hence, its cost will be accumulated to $ACO_{T_{\text{max}}}$, while the client’s payment occurring at the completion of this milestone activity will not be calculated in $ACI_{T_{\text{max}}}$. If the start time of this milestone activity is advanced to $T_{\text{max}} - d_{i_m}$, it will be completed at $T_{\text{max}}$. As a result, a payment that includes the compensation for this milestone activity and other activities finished from the $(m-1)$-th payment to this payment will be made by the client at $T_{\text{max}}$. As the start time of the other activities remains unchanged during this course, the operation will increase $ACI_{T_{\text{max}}}$ but exert no influence on $ACO_{T_{\text{max}}}$, thus resulting in a decrease in $G_{\text{max}}$.

4. Metaheuristic algorithms. The RCPSP has been proven to be an NP-hard problem ([5]). The problem studied in this paper is a RCPSP with the nonregular objective in the multi-project context; hence, it must also be NP-hard. Therefore, it is easy to understand that finding the exact solution is very difficult, even for small problems. Therefore, we use two well-known metaheuristic algorithms (TS and SA), which were originally developed by Glover ([16]) and Metropolis et al. ([25]), respectively, and have been successfully applied to a number of scheduling problems, for the solution of the studied problem. In this section, we present the common features of the algorithms first, then design the TS, SA, and algorithm comprising both the TS and SA, and propose a measure to improve the searching efficiency of the algorithms.

4.1. Common features. The starting solution, denoted as $S^{\text{star}}$, is constructed via the following procedure.

**Step 1** For project $h$, we calculate the start time window of activities using the critical path method without the consideration of renewable resource constraints.

**Step 2** Within the start time window of activities, we arrange the milestone activities, which determine payment times during the implementation of projects,
to start as early as possible while the non-milestone activities with which no payment is connected, to start as late as possible, thus generating a schedule for project \( h \).

Step 3 Under the generated schedule, we check whether the renewable resource constraints are satisfied. If yes, we have obtained a feasible schedule for project \( h \). Otherwise, under the constraints of precedence relationship and project deadline, we adjust the start time of the activities that are in progress at the occurrence time of the resource conflicts until the schedule becomes feasible.

Step 4 For each project, we perform steps 1 to 3 and all the obtained schedules constitute an \( S^{\text{star}} \) for the \( H \) projects.

We use a numerical example depicted in Figure 1 to explain the procedure proposed above, especially Step 2 in it. In the example, the contractor needs to implement two projects, namely project 1 and project 2, concurrently, where there is only one type of renewable resource and in each project, there are two milestone activities marked as the nodes with gray background. The \( d_i, c_i, v_i \) and \( r_{ij} \) of activities are attached with the corresponding nodes and other data of the example are as follows: \( \text{EST}^1=0 \), \( D^1=9 \), \( R^1_1=6 \), \( U^1=17 \), \( \theta^1=0.9 \), \( \text{EST}^2=2 \), \( D^2=10 \), \( R^2_1=9 \), \( U^2=14 \), and \( \theta^2=0.85 \).

Firstly, without the consideration of renewable resource constraints, we calculate the start time window of activities, namely the earliest and latest start times of activities, which are denoted as \( es_i \) and \( ls_i \), respectively, and also attached with nodes in the figure. Then, we arrange the start time of activities according to the rule that the milestone activities start as early as possible while the non-milestone activities as late as possible, and deal with resource conflicts by adjusting some activities’ start time. The generated schedules are shown in Figure 2, which is represented as \( S_{\text{star}}=(S_1,S_2) \) where \( S^1=(0,0,4,3,5,9) \) and \( S^2=(2,2,4,4,10) \). From the figure, it can be observed that in the two projects, the start times of the milestone activities are arranged at their earliest times, namely \( es_i \), while those of the non-milestone activities at their latest times, namely \( ls_i \).

The cash flows under the \( S_{\text{star}} \) are calculated in Table 2 that indicates the \( G_{\max} \) is 8 occurring at time 2. By the way, to illustrate the advantage of the proposed procedure, we also give the cash flows under another solution, \( S_{\text{earl}} \), in Table 2, where notation “/" represents that at the corresponding time, no cash outflow or inflow occur. In the \( S_{\text{earl}} \), except for activity 5 which has to start at time 5 due to the renewable resource constraints, all the activities start at their earliest times, making \( S_{\text{earl}}=(S_1,S_2) \) where \( S_1=(0,0,0,3,5,9) \) and \( S_2=(2,2,2,4,8) \). From the data

![Figure 1. A Numerical Example](image-url)
Figure 2. The Generated Starting Solution

Table 2. Cash Flows under the $S^{\text{star}}$ and $S^{\text{earl}}$

| Project 1 | Project 2 | $ACO_t$ | $ACI_t$ | $G_t$ |
|-----------|-----------|---------|---------|-------|
| $t$       | Cash outflow | Cash inflow | Cash outflow | Cash inflow | $ACO_t$ | $ACI_t$ | $G_t$ |
| 0         | 4          | /        | /        | /        | 4       | 0      | 4     |
| 2         | /          | /        | /        | /        | 8       | 0      | 8     |
| 3         | 2          | 5.4     | /        | /        | 10      | 5.4    | 4.6   |
| 4         | 2          | /        | 5        | 5.1      | 17      | 10.5   | 6.5   |
| 5         | 3          | 5.4     | /        | /        | 20      | 15.9   | 4.1   |
| 7         | /          | /        | /        | 2.55     | 20      | 18.45  | 1.55  |
| 9         | /          | 6.2     | /        | /        | 20      | 24.65  | -4.65 |
| 10        | /          | /        | /        | 6.35     | 20      | 31     | -11   |
| $G_{\text{max}} = 8$ |
| 0         | /          | /        | /        | /        | 6       | 0      | 6     |
| 2         | /          | /        | 7        | /        | 13      | 0      | 13    |
| 3         | 2          | 8.1     | /        | /        | 15      | 8.1    | 6.9   |
| 4         | /          | /        | 2        | 5.1      | 17      | 13.2   | 3.8   |
| 5         | 3          | 2.7     | /        | /        | 20      | 15.9   | 4.1   |
| 7         | /          | /        | /        | 2.55     | 20      | 18.45  | 1.55  |
| 8         | /          | /        | /        | 6.35     | 20      | 24.8   | -4.8  |
| 9         | /          | 6.2     | /        | /        | 20      | 31     | -11   |
| $G_{\text{max}} = 13$ |

in Table 2, we can see that under the $S^{\text{earl}}$, the $G_{\text{max}}$ still appears at time 2 whereas its value grows to 13, greater than that under the $S^{\text{star}}$ by 5.

Based on the current solution $S^{\text{curr}}$, a neighbor solution $S^{\text{neig}}$ is generated according to the following steps.

Step 1 We select a random project from the $H$ projects.

Step 2 In the selected project, we choose an activity arbitrarily and within its start time window, we change its start time to another available one randomly, thus generating an adjusted schedule for the selected project.

Step 3 Under the adjusted schedule, we check whether the renewable resource constraints are satisfied. If yes, we have obtained a feasible adjusted schedule for the
selected project. Else, we tackle the resource conflicts via the method used in step 3 of the procedure for constructing a starting solution, thus causing the adjusted schedule to become feasible.

Step 4  The adjusted schedule of the selected project and the unchanged schedule of other projects constitute a feasible $S_{\text{neig}}$ for $S_{\text{curr}}$.

4.2. Tabu search. The problem that the TS aims to tackle is to find the desirable solution where the $G_{\text{max}}$ is minimized. Beginning from a starting solution, the algorithm updates the current solution through generating its neighbor solution in an iterative way until a stop criterion is satisfied. During this process, a move, which is defined as a quadruple of the number of the selected project, number of the chosen activity in the selected project, original start time of the chosen activity, and new start time of the chosen activity, is used to represent the searching trajectory. In the meantime, a tabu list is employed to prevent the algorithm from exploring the repetitive solutions.

We use an example illustrated in Figure 3 to explain the process described above. In the figure, the $S_{\text{curr}}$ is taken as the $S_{\text{earl}}$ in the numerical example presented in subsection 4.1, and hence $S_{\text{curr}} = (S_1, S_2)$ where $S_1 = (0,0,3,5,9)$ and $S_2 = (2,2,2,4,8)$. By changing the start time of activity 3 in project 1 from 0 to 4, the algorithm generates an $S_{\text{neig}}, S_{\text{neig}} = (S_1, S_2)$ where $S_1 = (0,4,3,5,9)$ and $S_2 = (2,2,2,4,8)$. Then, the $S_{\text{curr}}$ is updated as the $S_{\text{neig}}$ and the move is expressed as $(1,3,0,4)$. In the meantime, the reverse move, which has the form $(1,3,0)$, is added to the tabu list, thus preventing the start time of this activity from being changed back to 0.

![Figure 3. Process for Updating Current Solution by TS](image)

The tabu list is denoted as TL and its length is defined as the maximal number of the reverse moves stored in it. During the searching process of the TS, TL is managed according to the first-in-first-out rule, by which the newest reverse move enters TL from its bottom while the oldest reverse move leaves TL from its top. Corresponding to the current solution being updated as its neighbor solution iteratively, all the reverse moves in TL moves upward step by step until they leave TL. The above operation is also shown in Figure 3.

All the reverse moves in TL are forbidden so that the algorithm can avoid exploring the repetitive solutions. However, if a reverse move in TL can generate a better solution than the best found thus far, which is stored and updated dynamically during the searching process, the algorithm can remove this reverse move from TL directly such that the search can move to the better solution. We still employ the numerical example presented in subsection 4.1 to explain this operation on TL by Figure 4, where $S_{\text{best}}$ represents the best solution found during the searching process. Suppose that for this numerical example, the search of the algorithm proceeds to the $S_{\text{curr}}$ and at current stage, the $S_{\text{best}}$ equals the $S_{\text{curr}}$ whose $G_{\text{max}}$ is 11. We
assume that at the next step, the search wants to move to the $S_{\text{neig}}$ by changing the start time of activity 3 in project 2 from 2 to 4, which is in fact a forbidden move since the reverse move, $(2,3,4)$, is in TL. However, because the $S_{\text{neig}}$ whose $G_{\text{max}}$ equals 8 is better than the $S_{\text{best}}$, the algorithm removes $(2,3,4)$ from TL directly, thus making the search can move to the $S_{\text{neig}}$.

The stop criterion is defined as a given running time of the TS, which is represented as $Time_{\text{stop}}$. In other words, when the operational time of the algorithm reaches $Time_{\text{stop}}$, it terminates and outputs the best solution found as the desirable one. We denote the $G_{\text{max}}$ under $S_{\text{star}}, S_{\text{curr}}, S_{\text{neig}}$, and $S_{\text{best}}$, respectively, as $G_{\text{max}}^{\text{star}}, G_{\text{max}}^{\text{curr}}, G_{\text{max}}^{\text{neig}},$ and $G_{\text{max}}^{\text{best}}$. Then, the implementation steps of the TS, wherein $Time$ represents the running time of the algorithm, are described as follows.

Step 1 We initialize TL, define $Time_{\text{stop}}$, and $Time := 0$.

Step 2 We construct an $S_{\text{star}}$ using the procedure proposed in subsection 4.1 and compute $G_{\text{max}}^{\text{star}}, S_{\text{curr}} := S_{\text{star}}, G_{\text{curr}} := G_{\text{max}}^{\text{star}}, S_{\text{best}} := S_{\text{star}}$, and $G_{\text{best}} := G_{\text{max}}^{\text{star}}$.

Step 3 Based on the $S_{\text{curr}}$, we generate an $S_{\text{neig}}$ using the procedure proposed in subsection 4.1 and compute $G_{\text{max}}^{\text{neig}}$.

Step 4 We assess whether the move from the $S_{\text{curr}}$ to the $S_{\text{neig}}$ is forbidden by TL. If yes, proceed to step 5, else proceed to step 6.

Step 5 We assess whether $G_{\text{max}}^{\text{neig}}$ is less than $G_{\text{max}}^{\text{best}}$. If yes, $S_{\text{curr}} := S_{\text{neig}}, G_{\text{curr}} := G_{\text{max}}^{\text{neig}}, S_{\text{best}} := S_{\text{neig}}, G_{\text{max}}^{\text{best}} := G_{\text{max}}^{\text{neig}}$, update TL, and proceed to step 7. Else, proceed directly to step 7.

Step 6 $S_{\text{curr}} := S_{\text{neig}}$ and $G_{\text{curr}} := G_{\text{max}}^{\text{neig}}$, and if the $G_{\text{max}}^{\text{neig}}$ is less than the $G_{\text{max}}^{\text{best}}$, $S_{\text{best}} := S_{\text{neig}}$ and $G_{\text{best}} := G_{\text{max}}^{\text{neig}}$. We update TL and proceed to step 7.

Step 7 We assess whether $Time$ reaches $Time_{\text{stop}}$. If yes, we proceed to step 8, else we proceed to step 3.

Step 8 We output the desirable results, i.e., $S_{\text{best}}$ and $G_{\text{best}}$ that are finally obtained.

4.3. Simulated annealing. The SA is specified by the cooling scheme, which constitutes the initial temperature, cooling rate, Markov chain length, and stop criterion. The initial temperature $Temp_{\text{init}}$ is calculated as $Temp_{\text{init}} = (G_{\text{max}}^{\text{star}} - G_{\text{max}}^{\text{best}}) / \ln Prob_{\text{init}}$, where $G_{\text{max}}^{\text{max}}$ is the maximal $G_{\text{max}}$ among the 60 neighbor solutions of the starting solution, while $Prob_{\text{init}}$, which is set as 0.9 in this application, is the initial acceptance ratio defined as the number of accepted neighbor solutions divided by that of the proposed neighbor solutions. Beginning from $Temp_{\text{init}}$, the temperature $Temp$ is progressively reduced according to a certain cooling rate $CR$, and under a given $Temp$, the number of transitions is determined by the Markov
chain length $MCL$. Similar to that of the TS, the stop criterion of the SA is defined as a given running time of the algorithm, $Time_{stop}$, as well.

The implementation steps of the SA are as follows, where $TNum$ denotes the number of feasible solutions explored during the searching process under a given temperature.

Step 1 We define $CR$, $MCL$, and $Time_{stop}$ and determine $Temp_{init}$. $Temp := Temp_{init}$, $TNum := 0$, and $Time := 0$.

Step 2 We construct an $S_{star}$ using the procedure proposed in subsection 4.1 and compute $G_{star, \text{max}}$. $S_{\text{curr}} := S_{star}$ and $G_{\text{curr, max}} := G_{\text{star, max}}$.

Step 3 Based on $S_{\text{curr}}$, we generate an $S_{\text{neig}}$ using the procedure proposed in subsection 4.1 and compute $G_{\text{neig, max}}$. $\Delta G_{\text{max}} := G_{\text{curr, max}} - G_{\text{neig, max}}$, and we then assess whether $\Delta G_{\text{max}}$ is greater than 0. If yes, $S_{\text{curr}} := S_{\text{neig}}$ and $G_{\text{curr, max}} := G_{\text{neig, max}}$, and we proceed to step 6; else, we proceed to step 5.

Step 5 We generate a random number from $U[0, 1]$. If this number is not greater than $e^{\Delta G_{\text{max}}/Temp}$, $S_{\text{curr}} := S_{\text{neig}}$ and $G_{\text{curr, max}} := G_{\text{neig, max}}$, and we proceed to step 6; else, we directly proceed to step 6.

Step 6 We assess whether $Time$ reaches $Time_{stop}$. If yes, we proceed to step 9; else, we proceed to step 7.

Step 7 $TNum := TNum + 1$, and we then assess whether $TNum$ reaches $MCL$. If yes, we proceed to step 8; else, we proceed to step 3.

Step 8 $Temp := Temp \cdot CR$ and $TNum := 0$, and we proceed to step 3.

Step 9 We output the desirable results, i.e., $S_{\text{curr}}$ and $G_{\text{curr, max}}$, which are finally obtained.

4.4. Algorithm comprising TS and SA. This algorithm, which is represented as SA-TS, comprises the TS and SA developed above. The SA-TS uses the SA designed in subsection 4.3 as the main algorithm in which the tabu list proposed in subsection 4.2 is adopted to prevent the algorithm from exploring repetitive solutions. In the algorithm, the tabu list is managed according to the same method as that employed in the TS, where the recent reverse moves are stored. Therefore, during the searching process, if a move to a neighbor solution is forbidden by the tabu list, the algorithm does not move to it unless this neighbor solution is better than the current one. In this way, the SA-TS may avoid some invalid iterations and hence enhance the searching efficiency.

4.5. Improvement measure. Based on the theorems presented in subsection 3.2, we propose a measure to improve the quality of the start and neighbor solutions generated during the searching process of the algorithms. This measure is implemented according to the following steps.

Step 1 We input the solution that is required to be improved and denote it as $S_{\text{inpu}}$. We compute $G_{\text{max}}$ under the $S_{\text{inpu}}$ and denote it as $G_{\text{inpu, max}}$.

Step 2 We represent the output solution and its $G_{\text{max}}$ as $S_{\text{outp}}$ and $G_{\text{outp, max}}$, respectively, and $S_{\text{outp, max}} := S_{\text{inpu, max}}$, $G_{\text{outp, max}} := G_{\text{inpu, max}}$. Then, we improve the $S_{\text{outp}}$ by steps 3 to 5 iteratively until it cannot be improved anymore.

Step 3 In the $S_{\text{outp}}$, we assess whether there exists a milestone activity that satisfies the condition described in theorem 3. If yes, we perform the operation presented in the theorem and, hence, generate an adjusted solution represented as $S_{\text{adj}}$; else, we proceed to step 3. We compute $G_{\text{max}}$ under the $S_{\text{adj}}$ and denote
the obtained $G_{\text{max}}$ as $G_{\text{adju}}_{\text{max}}$. If the $G_{\text{adju}}_{\text{max}}$ is less than the $G_{\text{outp}}_{\text{max}}$, we proceed to step 6.

Step 4 In the $S_{\text{outp}}$, we assess whether there is a non-milestone activity that satisfies the condition proposed in theorem 1. If yes, we execute the operation given in the theorem and thus obtain an adjusted solution which is taken as the $S_{\text{adju}}$; else, we proceed to step 4. We calculate $G_{\text{max}}$ under the $S_{\text{adju}}$ and take the obtained $G_{\text{max}}$ as $G_{\text{adju}}_{\text{max}}$. If the $G_{\text{adju}}_{\text{max}}$ is less than the $G_{\text{outp}}_{\text{max}}$, we proceed to step 6.

Step 5 In the $S_{\text{outp}}$, we assess whether there exists a non-milestone activity that matches the condition presented in theorem 2. If the answer is yes, we perform the operation expressed in the theorem and take the adjusted solution as the $S_{\text{adju}}$; else, we proceed to step 7. We compute $G_{\text{max}}$ under the $S_{\text{adju}}$ and take the gotten $G_{\text{max}}$ as $G_{\text{adju}}_{\text{max}}$. If the $G_{\text{adju}}_{\text{max}}$ is less than the $G_{\text{outp}}_{\text{max}}$, we proceed to step 6; else, we proceed to step 7.

Step 6 $S_{\text{outp}} := S_{\text{adju}}$, $G_{\text{outp}}_{\text{max}} := G_{\text{adju}}_{\text{max}}$. We then proceed to step 3.

Step 7 We output the $S_{\text{outp}}$ obtained finally.

5. Computational experiment.

5.1. Experimental design. To validate the performance of the three algorithms and the effect of the improvement measure, we test two versions of the designed three algorithms, namely, the version with the improvement measure and that without the improvement measure, in the experiment. Moreover, to evaluate the influences of some key parameters on the objective function value, we also carry out a one-way analysis of the parameters’ influence and a two-way analysis of the interactions between the parameters. The experiment is conducted on a data set developed by Browning and Yassine ([6, 7]), which is generated for the resource-constrained multi-project scheduling problem and can be downloaded at the website of “http://sbuweb.tcu.edu/tbrowning/RCMPSPinstances.htm” conveniently. The parameter settings of the data set are presented in Table 3, where the $C$, $\text{NARLF}$, $\text{MAUF}$, and $\sigma_{\text{MAUF}}^2$, which are all proposed by Browning and Yassine ([6, 7]), denote the network complexity of multiple projects, normalized average resource loading factor, modified average utilization factor, and variance in the $\text{MAUF}$s of different types of resource, respectively.

As pointed out by Browning and Yassine ([6, 7]), $C$, $\text{NARLF}$, $\text{MAUF}$, and $\sigma_{\text{MAUF}}^2$ may exert an important impact on the solution of the resource-constrained multi-project scheduling problem while in this research, the $M^h$, $\theta^h$, and $D^h$ have a close relationship with cash flow gaps. To explore the effect on the parameters aforementioned on the computational results, in the experiment, we let $C$ have 4 variations, $\sigma_{\text{MAUF}}^2$ takes 2 values, and the values of the $\text{NARLF}$, $\text{MAUF}$, $M^h$, $\theta^h$, and $D^h$ are set at 3 levels. Under a given parameter combination, 20 replications are generated randomly and hence, a full factorial experiment causes $4 \times 2 \times 3^5 \times 20 = 38,880$ instances as a whole.

The performance of the algorithms is evaluated using an index, $\text{ARP}$, which is defined as the average relative per cent below the best solution known, i.e., the best solution found by any of the developed algorithms. To compare the algorithms based on a common computational effort, they use the same start solution and stop criterion. All the algorithms are coded and compiled using Microsoft Visual C++, and the computational experiment is performed on an Intel Core-based personal computer with a 2.60-GHz clock-pulse and 3.88-GB RAM. Based on a preliminary empirical test, their parameters are set as follows: $\text{Time}_{\text{stop}}=98$ seconds, $CR=0.88$, etc.
Table 3. Parameter Settings

| Parameter                              | Setting                                                                 |
|----------------------------------------|-------------------------------------------------------------------------|
| Number of projects, $H$                | 3                                                                      |
| Number of non-dummy activities in projects, $n^h$ | 20                                                                    |
| Network complexity of multiple projects, $C$ | LLL, HLL, HHL, HHH,  where “L” and “H” represent the network complexity of an individual project. “L” means that the network complexity of the project equals 0.14 while “H” implies it is 0.69 |
| Number of resource types, $K$          | 4                                                                      |
| Normalized average resource loading factor, $NARLF$ | $-2, 0, 2                                                             |
| Modified average utilization factor, $MAUF$ | 0.8, 1.0, 1.2                                                          |
| Variance in $MAUF$s of different resource, $\sigma^2_{MAUF}$ | 0, 0.25                                                               |
| Cost of activities, $c_i$              | $\rho_v \cdot c_i$, where $\rho_v$, which is a special parameter defined for generating $v_i$, is randomly selected from $U[1.3, 1.5]$ |
| Earned value of activities, $v_i$       | Randomly selected from $U[1, 9]$                                       |
| Number of milestone activities, $M^h$  | 4, 5, 6, where the dummy end activity must be a milestone activity while other milestone activities are randomly selected from all the non-dummy activities |
| Compensation proportion of projects, $\theta^h$ | 0.7, 0.8, 0.9                                                          |
| Earliest start time of projects, $EST^h$ | Randomly selected from $U[1, 5]$                                      |
| Deadline of projects, $D^h$            | 1.1$\cdot$CPL, 1.3$\cdot$CPL, 1.5$\cdot$CPL, where $CPL$ is the critical path length of the project network without the consideration of renewable resource constraints |

$MCL=3,300$, and the length of TL is set as 6. The experimental design is illustrated in Figure 5, which shows clearly the general framework and main contents of the computational experiment carried out in our research.

5.2. Performance of the algorithms and effects of improvement measure.

The computational results under the different levels of the parameters are shown
Table 4. *ARP(%)* of Algorithms under Different Values of Parameters

| Parameter | Value | TS\textsuperscript{NIM} | TS\textsuperscript{IM} | SA\textsuperscript{NIM} | SA\textsuperscript{IM} | SA-TS\textsuperscript{NIM} | SA-TS\textsuperscript{IM} |
|-----------|-------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| $C$       | LLL   | 8.26            | 3.30            | 7.36            | 2.18            | 6.23            | 1.46            |
|           | HLL   | 7.57            | 2.63            | 7.15            | 2.04            | 6.10            | 1.35            |
|           | HHL   | 7.13            | 2.47            | 6.84            | 1.91            | 5.41            | 1.40            |
|           | HHH   | 6.28            | 1.77            | 5.97            | 1.67            | 5.13            | 1.23            |
| $NARLF$   | –2    | 7.06            | 2.36            | 6.62            | 1.80            | 5.55            | 1.24            |
|           | 0     | 7.38            | 2.61            | 6.70            | 1.90            | 5.75            | 1.39            |
|           | 2     | 7.49            | 2.65            | 7.18            | 2.16            | 5.86            | 1.45            |
| $MAUF$    | 0.8   | 8.20            | 3.00            | 7.57            | 2.46            | 6.23            | 1.52            |
|           | 1.0   | 7.33            | 2.70            | 6.74            | 1.88            | 5.67            | 1.35            |
|           | 1.2   | 6.41            | 1.93            | 6.18            | 1.51            | 5.26            | 1.20            |
| $\sigma_{MAUF}^2$ | 0   | 7.13            | 2.42            | 6.71            | 1.89            | 5.54            | 1.19            |
|           | 0.25  | 7.49            | 2.67            | 6.94            | 2.01            | 5.90            | 1.53            |
| $D^h$    | 1.1-CPL | 6.05           | 1.84            | 5.85            | 1.54            | 5.07            | 1.13            |
|           | 1.3-CPL | 7.25           | 2.45            | 6.76            | 1.96            | 5.61            | 1.35            |
|           | 1.5-CPL | 8.63           | 3.33            | 7.89            | 2.35            | 6.47            | 1.60            |

In Table 4 and Figure 6, where TS\textsuperscript{NIM}, SA\textsuperscript{NIM}, and SA-TS\textsuperscript{NIM} are the version of the algorithms without the improvement measure while TS\textsuperscript{IM}, SA\textsuperscript{IM}, and SA-TS\textsuperscript{IM} the corresponding ones with the improvement measure. In the following of this subsection, we discuss the performance of the algorithms first and then, analyze the effects of the improvement measure.
5.2.1. **Performance of algorithms.** Initially, when comparing the three algorithms designed in this paper, the index, $\text{ARP}$, obtained for both the versions indicate that the SA is superior to the TS whereas inferior to the SA-TS. This may be because the SA is more suitable for the solution of the larger problem owing to its random nature, whereas the TS may have an advantage in tackling the relatively small problem. In our experiment, an instance consists of three projects and in each project, there are 20 activities. Hence, the scale of the problem is generally not very small, making the SA outperform the TS overall. Regarding the SA-TS, it is easy to understand that since exploring repetitive solutions is prevented in this algorithm, the SA-TS generates better results than the SA although they use the same stop criterion. A further interesting phenomenon that can be observed in Figure 6 is the advantage of the SA-TS over the TS and SA shrinks when the improvement measure being adopted. The reason for this may be the quality of the desirable solutions obtained by the SA-TS$^\text{NIM}$ is already at a high level and thus, the algorithm’s potential for improvement is smaller than the TS$^\text{NIM}$ and SA$^\text{NIM}$. As a result, the effects of the improvement measure on the latter two algorithms is more remarkable than that on the former, and accordingly, the superiority of the SA-TS$^\text{IM}$ over the TS$^\text{IM}$ and SA$^\text{IM}$ diminishes.

Secondly, we investigate the influence of the different parameters on the algorithms’ performance. Figure 6 shows that ARPs descend with an increase in $C$ or $\text{MAUF}$ while ascend as $\text{NARLF}$, $\sigma^2_{\text{MAUF}}$, or $D^h$ rise. The reasons for the result are described as follows. The increasing $C$ and $\text{MAUF}$ may lead to the greater precedence and resource constraints respectively. This reduces the size of the solution space that the algorithms need to explore and thus makes the generated results become better. The reverse is true for the case where $D^h$ gets larger. In this case, since the solution space augments with the relaxing deadline constraint, the algorithms obtain worse results under the same stop criterion. As for $\text{NARLF}$ and $\sigma^2_{\text{MAUF}}$, the both parameters can exert an impact on the constraint of renewable resources. A negative $\text{NARLF}$, which implies a front-loading of the resource constraints, has implications for more activities in network than the equal and positive $\text{NARLF}$s. Therefore, when this parameter grows from $-2$ to 0 and 2, the algorithms’ performance declines due to the decreasing effect of the resource constraints. The similar thing may occur when enhancing $\sigma^2_{\text{MAUF}}$. Since the $\sigma^2_{\text{MAUF}}=0$ case means that the problems are equally constrained by all resource types whereas $\sigma^2_{\text{MAUF}}=0.25$ represents cases where the problems are mainly constrained by only one of the resource types, changing $\sigma^2_{\text{MAUF}}$ from 0 to 0.25 reduces the impact of resource constraints and hence the algorithms generate a higher $\text{ARP}$.

5.2.2. **Effects of the improvement measure.** The effects of the improvement measure can be observed clearly from Figure 6, where the ARPs of the TS$^\text{IM}$, SA$^\text{IM}$, and SA-TS$^\text{IM}$ are generally lower than those of the TS$^\text{NIM}$, SA$^\text{NIM}$, and SA-TS$^\text{NIM}$. To explore the effects intensively, we define a new index, $\Delta$ARP, which equals an ARP of the algorithms without the improvement measure minus that of the corresponding algorithms with the improvement measure. The $\Delta$ARP of the three algorithms are shown in Figure 7, which indicates the index goes down with the increase in $C$ or $\text{MAUF}$ whereas goes up as $\text{NARLF}$, $\sigma^2_{\text{MAUF}}$, or $D^h$ grow. This phenomenon is not surprising for the following facts. In the designed algorithms, the improvement measure enhances the quality of the start and neighbor solutions generated during the searching process by adjusting some activities’ start time under the constraints of precedence relationships and renewable resources. When $C$ or $\text{MAUF}$ increase,
the opportunity for such adjustments decreases because the mentioned constraints get stronger. This reduces the effects of the improvement measure and thus causes \( \Delta ARP \) to drop. Since the increasing \( NARLF \) or \( \sigma^2_{MAUF} \) may weaken the renewable resource constraints, \( \Delta ARP \) goes up as the two parameters climb. With regard to \( D^h \), it is easy to understand that a greater deadline may provide more opportunities for adjusting the start times of activities and thereby generate a larger \( \Delta ARP \).

![Figure 7. Effects of Improvement Measure](image)

5.3. **Influences of parameters on objective function value.** The \( G_{max} \) under the different values of the parameters, including \( C, NARLF, MAUF, \sigma^2_{MAUF}, M^h, \theta^h, \) and \( D^h \) are shown in Table 5 and Figure 8. Furthermore, considering the interactions between \( MAUF \) and \( C, NARLF, \sigma^2_{MAUF}, \) or \( D^h \) may exert an influence on the objective function value, we further summarize the \( G_{max} \) under the various combinations of the parameters’ values in Table 6 and Figure 9. Based on the results aforementioned, we conduct a one-way analysis of the parameters’ influence on \( G_{max} \) and a two-way analysis of the interactions between the parameters below.

5.3.1. **One-way analysis of parameters’ influence on \( G_{max} \).** From Figure 8, it can be seen that \( G_{max} \) rises with an increase in \( C \) or \( MAUF \) while drops when \( NARLF, \sigma^2_{MAUF}, M^h, \theta^h, \) or \( D^h \) become larger. The results are in line with expectations because of the facts expressed as follows. As discussed in subsection 5.2, when \( C \) or \( MAUF \) augment, precedence relationships or renewable resources are more constraining. This reduces the space of feasible solutions and thus leads to a higher \( G_{max} \). On the contrary, when \( D^h \) gets greater the project deadline constraints will relax accordingly, hence causing \( G_{max} \) to decline. Since the increasing \( NARLF \) or
Figure 8. Effects of Parameters on Objective Function Value

Figure 9. Interactive Effects of Parameters on Objective Function Value
Table 5. $G_{\text{max}}$ under Different Values of Parameters

| Parameter | Value | $G_{\text{max}}$ | Parameter | Value | $G_{\text{max}}$ |
|-----------|-------|------------------|-----------|-------|------------------|
| $C$       | LLL   | 66.06            | $\sigma^2_{\text{MAUF}}$ | 0     | 70.37            |
|           | HLL   | 67.34            |           | 0.25  | 66.86            |
|           | HHL   | 69.66            | $M^h$    | 4     | 81.48            |
|           | HHH   | 71.43            |           | 5     | 67.12            |
| $NARLF$   | −2    | 70.63            | $\theta^h$ | 0.7   | 83.76            |
|           | 0     | 68.51            |           | 0.8   | 68.73            |
|           | 2     | 66.73            |           | 0.9   | 53.36            |
| $\text{MAUF}$ | 0.8 | 65.57            | $D^h$    | 1.1-CPL | 72.88          |
|           | 1.0   | 68.44            |           | 1.3-CPL | 67.66          |
|           | 1.2   | 71.86            |           | 1.5-CPL | 65.33          |

Table 6. $G_{\text{max}}$ under Combinations of Different Values of Parameters

| MAUF | C  | $G_{\text{max}}$ | MAUF | NARLF | $G_{\text{max}}$ | MAUF | $\sigma^2_{\text{MAUF}}$ | $G_{\text{max}}$ | MAUF | $D^h$ | $G_{\text{max}}$ |
|------|----|------------------|------|-------|------------------|------|--------------------------|------------------|------|-------|------------------|
| 0.8  | LLL| 61.8             | 0.8  | −2    | 66.96            | 0.8  | 0                        | 66.67            | 0.8  | 1.1-CPL| 71.33            |
|      | HLL| 64.29            | 0    | 65.46            | 0.25  | 64.46            | 1.3-CPL| 64.61            |
|      | HHL| 66.6             | 2    | 64.28            | 1.0   | 0                        | 69.99            | 1.5-CPL| 60.78            |
|      | HHH| 69.58            | 1.0  | −2    | 70.3             | 0.25  | 66.88            | 1.0   | 1.1-CPL| 72.2             |
| 1.0  | LLL| 66.08            | 0    | 68.33            | 1.2   | 0                        | 74.46            | 1.3-CPL| 67.46            |
|      | HLL| 67.16            | 2    | 66.7             | 0.25  | 69.25            | 1.5-CPL| 65.65            |
|      | HHL| 69.48            | 1.2  | −2    | 74.62            |           |              | 1.2   | 1.1-CPL| 75.12            |
|      | HHH| 71.05            | 0    | 71.75            |           |              | 1.3-CPL| 70.9             |
| 1.2  | LLL| 70.3             | 2    | 69.21            |           |              | 1.5-CPL| 69.57            |
|      | HLL| 70.58            |      |                |           |              |           |                |
|      | HHL| 72.9             |      |                |           |              |           |                |
|      | HHH| 73.67            |      |                |           |              |           |                |

$\sigma^2_{\text{MAUF}}$ may decrease the effect of renewable resource constraints, $G_{\text{max}}$ goes down as these two parameters go up.

The influences of the parameters, $M^h$ and $\theta^h$, on $G_{\text{max}}$ are more direct than $C$, MAUF, NARLF, $\sigma^2_{\text{MAUF}}$, and $D^h$. First, $M^h$ denotes the number of milestone activities in projects and thus, when it increases, the number of the client’s payments increases correspondingly. This improves the distribution of cash inflows over the course of the project and the contractor’s expense is thus compensated more quickly, which makes $G_{\text{max}}$ decrease. Second, $\theta^h$ is the client’s compensation proportion. Therefore, its increase can enhance each payment amount during the execution of projects and thus cause $G_{\text{max}}$ to decrease.

5.3.2. Two-way analysis of interactions between parameters. The two-way interaction between MAUF and C, NARLF, $\sigma^2_{\text{MAUF}}$, or $D^h$ are shown in Figure 9, which indicates such an interaction may also exert an impact on $G_{\text{max}}$. First, the figure exhibits that the disparity of the $G_{\text{max}}$s under the different MAUFs descends with the increase in C, NARLF, or $\sigma^2_{\text{MAUF}}$ while ascends with the increase in $D^h$. The
above phenomenon is reasonable because when \( C \) is little or \( D_h \) is great, the precedence relationships or project deadlines are less constraining. In such cases, the constraints of renewable resources can have a larger effect on the objective function value, hence, the \( G_{\text{max}} \) under different \( \text{MAUF} \)'s are more disparate, compared with the case with a greater \( C \) or a less \( D_h \). As to \( \text{NARLF} \) or \( \sigma_{\text{MAUF}}^2 \), the outcomes are easier to understand. Since at a lower level of \( \text{NARLF} \) or \( \sigma_{\text{MAUF}}^2 \) (e.g. \( \text{NARLF}=2 \) or \( \sigma_{\text{MAUF}}^2=0.25 \)), the impact of resource constraints is greater than that at a higher level of the two parameters (e.g. \( \text{NARLF}=2 \) or \( \sigma_{\text{MAUF}}^2=0.25 \)), the \( G_{\text{max}} \) disparity goes down as \( \text{NARLF} \) or \( \sigma_{\text{MAUF}}^2 \) go up.

Second, from Figure 9, we can also find that the variation trends of \( G_{\text{max}} \) with \( C \), \( \text{NARLF} \), \( \sigma_{\text{MAUF}}^2 \), or \( D_h \) are identical with the corresponding ones in the one-way analysis, however, the slope of these curves changes with the alteration in \( \text{MAUF} \). To be specific, when \( \text{MAUF} \) increases, the curves for \( C \) and \( D_h \) become flatter whereas those for \( \text{NARLF} \) and \( \sigma_{\text{MAUF}}^2 \) get steeper. The underlying reasons for this phenomenon are as follows. Increasing \( \text{MAUF} \) strengthens the constraints of renewable resources and accordingly, weakens the impact of precedence relationship and project deadline constraints on the objective function value. As a result, \( G_{\text{max}} \) becomes more insensitive to the variation of \( C \) and \( D_h \) and thus, the slope of the curves for \( C \) and \( D_h \) declines as \( \text{MAUF} \) grows. The case for \( \text{NARLF} \) and \( \sigma_{\text{MAUF}}^2 \) is contrary to that for \( C \) and \( D_h \). Because there exists a positive correlation between the influence of \( \text{MAUF} \) and that of the two parameters, the objective function value gets more sensitive to the variation of \( \text{NARLF} \) and \( \sigma_{\text{MAUF}}^2 \) when increasing \( \text{MAUF} \). Therefore, the slope of the curves for \( \text{NARLF} \) and \( \sigma_{\text{MAUF}}^2 \) climbs with an increase in \( \text{MAUF} \).

6. Conclusions. This paper investigates a RCPSP with the objective of minimizing the maximal cash flow gap for cases in which the contractor is required to implement multiple projects concurrently, with each project having its own deadlines and availability of renewable resources. First, based on the formulation of cash flows for the projects, we construct a non-linear integer programming optimization model for the problem and present three theorems of the studied problem. Then, for the NP-hardness of the problem, we design metaheuristic algorithms including the TS, SA, and an algorithm comprising both TS and SA (i.e. the SA-TS) to solve the model. According to the proposed theorems, we then develop an improvement measure to enhance the searching efficiency of the designed algorithms. Finally, we conduct a computational experiment performed on a data set, which comes from existing literature and is developed for the resource-constrained multi-project scheduling problem, to evaluate the performance of the three algorithms and the contribution of the improvement measure. In addition, based on the obtained computational results, the effects of some key parameters on the objective function value are also analyzed.

From the research results, the following conclusions are drawn: For the studied problem, the SA is superior to the TS whereas inferior to the SA-TS and the advantage of the SA-TS over the SA and TS decreases when the improvement measure being employed. With an increase in \( C \) or \( \text{MAUF} \), the algorithms’ performance improves but as \( \text{NARLF} \), \( \sigma_{\text{MAUF}}^2 \), or \( D_h \) rise, it deteriorates. The maximal cash flow gap grows with an increase in \( C \) or \( \text{MAUF} \) while drops when \( \text{NARLF} \), \( \sigma_{\text{MAUF}}^2 \), \( M^h \), \( \theta^h \), or \( D_h \) get larger. Besides, with an increase in \( C \), \( \text{NARLF} \), and \( \sigma_{\text{MAUF}}^2 \) or a decrease in \( D_h \), the disparity of the \( G_{\text{max}} \)s under the different \( \text{MAUF} \)'s descends.
and as MAUF rises, the curves of $G_{\text{max}}$ for $C$ and $D^h$ become flatter whereas those for NARLF and $\sigma_{\text{MAUF}}^2$ get steeper.

It should be noted that in our research, we assumed that an activity can only be executed with a single mode and resources are renewable ones. In the future, we will extend our research to a multi-mode environment in which an activity owns several discrete execution modes and non-renewable resources exert a limitation on selecting an execution mode for an activity. Furthermore, depending on how to procure resources for executing an activity, the contractor’s cash outflows may occur at any time during the activity duration, therefore, the research can also be extended to the case where cash outflows take place in a more general fashion. Ultimately, although the metaheuristic algorithms developed in this paper can solve the studied problem effectively, designing more efficient algorithms is still a direction worthy of further investigation.

Acknowledgments. We would like to express our gratitude to the editor and the anonymous referees for their careful reading of the paper and valuable comments.

REFERENCES

[1] A. Alghazi, A. Elazouni and S. Selim, Improved genetic algorithm for finance-based scheduling, *J. Comput. Civil Engineering*, 27 (2013), 379–394.

[2] M. M. Ali and A. Elazouni, Finance-based CPM/LOB scheduling of projects with repetitive non-serial activities, *Construction Management Economics*, 27 (2009), 839–856.

[3] M. Abido and A. Elazouni, Multiobjective evolutionary finance-based scheduling: Entire projects’ portfolio, *J. Comput. Civil Engineering*, 25 (2011), 85–97.

[4] S. T. Al-Shihabi and M. M. AlDurgam, A max-min ant system for the finance-based scheduling problem, *Comput. Industrial Engineering*, 110 (2017), 264–276.

[5] J. Blazewicz, J. K. Lenstra and K. A. H. G. Rinnooy, Scheduling subject to resource constraints: Classification and complexity, *Discrete Appl. Math.*, 5 (1983), 11–24.

[6] T. R. Browning and A. A. Yassine, A random generator of resource-constrained multi-project network problems, *J. Scheduling*, 13 (2010), 143–161.

[7] T. R. Browning and A. A. Yassine, Resource-constrained multi-project scheduling: Priority rule performance revised, *Internat. J. Production Economics*, 126 (2010), 212–228.

[8] R. H. Doersch and J. H. Patterson, Scheduling a project to maximize its present value: A zero-one programming approach, *Management Science*, 23 (1977), 882–889.

[9] M. Engwall and A. Jerbrant, The resource allocation syndrome: The prime challenge of multi-project management?, *Internat. J. Project Management*, 21 (2003), 403–409.

[10] A. M. Elazouni and A. A. Gab-Allah, Finance-based scheduling of construction projects using integer programming, *J. Construction Engineering Management*, 130 (2004), 15–24.

[11] A. Elazouni, A. Alghazi and S. Selim, Finance-based scheduling using meta-heuristics: Discrete versus continuous optimization problems, *J. Finance Management Property Construction*, 20 (2015), 85–104.

[12] A. Elazouni, Heuristic method for multi-project finance-based scheduling, *Construction Management Economics*, 27 (2009), 199–211.

[13] A. Elazouni and M. Abido, Multiobjective evolutionary finance-based scheduling: Individual projects within a portfolio, *Automat. Construction*, 20 (2011), 755–766.

[14] M. S. El-Abbasy, A. Elazouni and T. Z. F. ASCE, Generic scheduling optimization model for multiple construction projects, *J. Comput. Civil Engineering*, 31 (2017).

[15] H. Fathi and A. Afshar, GA-based multi-objective optimization of finance-based construction project payment scheduling, *KSCE J. Civil Engineering*, 14 (2010), 627–638.

[16] F. Glover, Future path for integer programming and links to artificial intelligence, *Comput. Oper. Res.*, 13 (1986), 533–549.

[17] W. S. Herroelen, P. Dommelen and E. L. Demeulemeester, Project network models with discounted cash flows: A guided tour through recent developments, *European J. Oper. Res.*, 100 (1997), 97–121.

[18] Z. He, R. Liu and T. Jia, Metaheuristics for multi-mode capital-constrained project payment scheduling, *European J. Oper. Res.*, 223 (2012), 605–613.
[19] Z. He, H. He, R. Liu and N. Wang, Variable neighbourhood search and tabu search for a discrete time/cost trade-off problem to minimize the maximal cash flow gap, Comput. Oper. Res., 78 (2017), 564–577.

[20] A. Jiang, R. R. A. Issa and M. Malek, Construction project cash flow planning using the Pareto optimality efficiency network model, J. Construction Engineering Management, 17 (2011), 510–519.

[21] P. Leyman and M. Vanhoucke, A new scheduling technique for the resource-constrained project scheduling problem with discounted cash flows, Internat. J. Prod. Res., 53 (2015), 2771–2786.

[22] P. Leyman and M. Vanhoucke, Payment models and net present value optimization for resource-constrained project scheduling, Comput. Industrial Eng., 91 (2016), 139–153.

[23] P. Leyman and M. Vanhoucke, Capital- and resource-constrained project scheduling with net present value optimization, European J. Oper. Res., 256 (2017), 757–776.

[24] S. S. Liu and C. J. Wang, Profit optimization for multiproject scheduling problems considering cash flow, J. Construction Engineering Management, 136 (2010), 1268–1278.

[25] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, Equation of state calculations by fast computing machines, J. Chemical Physics, 21 (1953), 1087–1092.

[26] M. Ning, Z. He, T. Jia and N. Wang, Metaheuristics for multi-mode cash flow balanced project scheduling with stochastic duration of activities, Automat. Construction, 81 (2017), 224–233.

[27] M. Ning, Z. He, N. Wang and R. Liu, Metaheuristic algorithms for proactive and reactive project scheduling to minimize contractor’s cash flow gap under random activity duration, IEEE Access, 6 (2018), 30547–30558.

[28] L. Özdamar and H. Dündar, A flexible heuristic for a multi-mode capital constrained project scheduling problem with probabilistic cash inflows, Comput. Oper. Res., 24 (1997), 1187–1200.

[29] L. Özdamar, On scheduling project activities with variable expenditure rates, IIE Transactions, 30 (1998), 695–704.

[30] C. Schwindt and J. Zimmermann, Handbook of Project Management and Scheduling, Springer International Publishing AG, Berlin, 2014.

[31] D. E. Smith-Daniels and V. L. Smith-Daniels, Maximizing the net present value of a project subject to materials and capital constraints, J. Oper. Management, 7 (1987), 33–45.

[32] D. E. Smith-Daniels, R. Padman and V. L. Smith-Daniels, Heuristic scheduling of capital constrained projects, J. Oper. Management, 14 (1996), 241–254.

Received May 2019; revised November 2019.

E-mail address: heyukang1994@163.com
E-mail address: zhengwenhe@mail.xjtu.edu.cn
E-mail address: wangnm@mail.xjtu.edu.cn