To appear in the *International Journal of Production Research*
Vol. 00, No. 00, 00 Month 20XX, 1–15

The integration of resource allocation and time buffering for bi-objective robust project scheduling

Yangyang, Liang[a], Nanfang, Cui[b], Xuejun, Hu[c*] and Erik Demeulemeester[d]

[a]*Hubei University of Economics, Wuhan, Hubei 430205, P.R. China;* [b]*School of Management, Huazhong University of Science and Technology, Wuhan, Hubei 430074, P.R. China;* [c] *Business School, Hunan University, Changsha, Hunan 410082, P.R. China;* [d]*Research Center for Operations Management, Department of Decision Sciences and Information Management, Faculty of Economics and Business, Katholieke Universiteit Leuven, BE-3000 Leuven, Belgium*

(*Received 00 Month 20XX; accepted 00 Month 20XX*)

In the recent decades, the recognition that uncertainty lies at the heart of modern project management has induced considerable research efforts on robust project scheduling for dealing with uncertainty in a scheduling environment. The literature generally provides two main strategies for the development of a robust predictive project schedule, namely robust resource allocation and time buffering. Yet, the previous studies seem to have neglected the potential benefits of an integration between the two. Besides, few efforts have been made to protect simultaneously the project due date and the activity start times against disruptions during execution, which is desperately demanded in practice. In this paper, we aim at constructing a proactive schedule that is not only short in time but also less vulnerable to disruptions. Firstly, a bi-objective optimization model with a proper normalization of the two components is proposed in the presence of activity duration variability. Then a two-stage heuristic algorithm is developed which deals with a robust resource allocation problem in the first stage and optimally determines the position and the size of time buffers using a simulated annealing algorithm in the second stage. Finally, an extensive computational experiment on the PSPLIB network instances demonstrates the superiority of the combination between resource allocation and time buffering as well as the effectiveness of the proposed two-stage algorithm for generating proactive project schedules with composite robustness.

Keywords: Robust project scheduling; Bi-objective optimization; Robust resource allocation; Time buffering; Two-stage algorithm

1. Introduction

The well-known *resource-constrained project scheduling problem* (RCPSP) involves the development of a precedence and resource feasible project schedule (i.e. the so-called *baseline schedule*, BS) under the objective of minimizing the project makespan, assuming a static and deterministic environment (for an extensive discussion we refer to Demeulemeester and Herroelen (2002)). During execution, however, a practical project is subject to high levels of uncertainty related to such factors as activity duration variability, machine breakdowns, resources that arrive behind schedule (Artigues et al., 2013; Hall et al., 2015). These changes can be translated into an increase or decrease of the activity duration, and hence perturb the as-planned execution of the baseline schedule. Moreover, the activity disruptions can propagate throughout the network due to the constraints of both strictly technological precedence relations and resource-driven precedence relations, resulting in a kind of snowball effect and incurring a generally low probability of on-time delivery.

Among the many project scheduling problems dealing with uncertainty, the research on robust project scheduling has received an ever-growing attention (Demeulemeester and Herroelen, 2011;

*Corresponding author. Email: xuejun_hu@hnu.edu.cn

Bruni et al., 2017; Lamas and Demeulemeester, 2017; Ma et al., 2018). A robust project scheduling approach enables the generation of a *proactive* BS that is protected as much as possible against disruptions and deploys a *reactive* policy reasonably whenever a conflict in the ongoing schedule occurs (Artigues et al., 2015; Davari and Demeulemeester, 2017). The literature has distinguished between two types of robustness measures: *quality robustness* and *solution robustness*. The former refers to the insensitivity of the objective value (be it the project makespan, the net present value or the project cost) to disruptions, and the latter refers to the difference between the BS and the realized schedule. When it comes to proactive scheduling for optimizing solution robustness, two main strands of research have been extensively explored in the literature: the research on robust resource allocation (Artigues et al., 2003) and the research on time buffering (Leus, 2003). In the following paragraphs, we briefly discuss the state of the art in these two main strands.

The first body of research aims to generate stable resource flow networks in which the renewable resources are transferred between the activities in an efficient way so that the schedule stability could be maintained. Artigues et al. (2003) introduced a simple method for determining a feasible resource flow on the basis of a parallel schedule generation scheme. Leus and Herroelen (2004) presented a branch-and-bound procedure for allocating a single resource type with the objective of minimizing the expected weighted deviation cost between the planned and realized start times of the projected schedule (i.e. *stability cost*, SC), under the assumption of exponential activity disruption lengths. Policella (2005) proposed to construct a *chained Partial Order Schedule* (POS) through chaining procedures to include a set of additional arcs representing resource flows. Three heuristic solution algorithms based on mixed integer programming (MIP) formulations together with one constructive procedure named MABO (*myopic activity-based optimization*) were developed by Deblaere et al. (2007) for solving the robust resource allocation problem.

The second body of research advocates to insert scattered time buffers in front of the project activities in order to absorb potential disruptions caused by earlier activity delays and to protect the activity start times as well as possible (Herroelen and Leus, 2004; Zheng et al., 2018). Leus (2003) presented an adapted float factor (ADFF) heuristic, which inserts longer time buffers in front of activities that would incur a high deviation cost in the starting times. To make sure that the buffered BS using the ADFF procedure is resource feasible, Van de Vonder et al. (2006) developed a resource flow-dependent float factor (RFDFF) heuristic to construct solution robust schedules by exploiting a feasible resource flow network generated using the procedure of Artigues et al. (2003). Van de Vonder et al. (2008) further proposed the virtual activity duration extension (VADE) heuristic and the starting time criticality (STC) heuristic to solve the proactive RCPSP in the presence of activity duration variability. The above procedures have been developed under the objective of minimizing the stability cost function.

Despite the popularity of robust resource allocation and time buffering for solving the proactive scheduling problem, it is observed that these two research streams were investigated separately. As a matter of fact, Van de Vonder et al. (2008) have long ago addressed that the robustness of a project schedule generated through time buffering largely depends on the corresponding resource flow network that is used, but they had not looked further into this important issue. The very few works that consider resource allocations when buffering an initial schedule merely rely on a feasible resource flow network that is randomly derived regardless of any robustness measures.

In addition, the two aforementioned strands of research mainly focus on generating a solution robust project schedule with minimum stability cost. The significance of quality robustness has been pointed out by many studies (Goldratt, 1997; Hu et al., 2016) and has received high attention of project practitioners. Therefore, it is crucial and practical to set up a bi-objective optimization model that minimizes both quality robustness and solution robustness. Yet, the study on the bi-criterion scheduling problem is comparatively sparse. Al-Fawzan and Haouari (2005) developed a tabu search algorithm for solving a bi-objective model where the total free slack and the project makespan are used as the solution robustness and quality robustness measures, respectively. Abbasi et al. (2006) presented a similar bi-objective model that minimizes a linear function of the makespan and the weighted sum of the float times. Van de Vonder et al. (2005) were among the first to probe into the trade-off between the SC and the makespan. The objective of their work is to address the issue whether to concentrate safety time in project and feeding buffers (Goldratt, 1997) or to insert time buffers that are scattered throughout the BS in order to enhance schedule stability. Van de Vonder et al. (2008) opted for a bi-objective optimization function that maximizes the timely project completion probability (quality robustness) and minimizes the SC simultaneously. Most recently, Ghoddousi et al. (2016) developed a two-stage multi-objective buffer allocation approach for robust project scheduling, in which the Pareto solutions are evaluated in terms of the deviation from the initial start times and due dates.

Different to all these studies, our work adopts the starting time criticality as a surrogate measure of solution robustness and therefore proposes an integrated bi-objective optimization approach for scheduling, robust resource allocation and time buffering. The contributions lie in the novelty of the bi-objective model with a proper normalization of the two components of the objective function, and a two-stage solving algorithm that contains an improved resource allocation procedure as well as a customized simulated annealing (SA) heuristic.

The remainder of this article is organized as follows. The next section provides some basic notations and a statement of the research problem using an illustrative example. In Section 3, the bi-objective model for generating a robust project schedule is established. Section 4 presents the two-stage algorithm for solving the model. Section 5 contains the experimental outcomes of this research and the necessary analysis to justify the effectiveness of the proposed methods. The last section concludes the paper and discusses some future research directions.

2. Definition and problem statement

Consider a project that is represented as an activity-on-the-node network $G = (N, A)$, where a set of nodes N=$\{0, 1, 2...n\}$ denotes the project activities and a set of arcs $A \subseteq N \times N$ the zero-lag finish-start precedence relations between the activities. Activities 0 and n indicate the dummy start and dummy end activities, respectively, both of which have zero durations and zero resource usages.It is assumed that each activity is not allowed to start earlier than its planned starting time in the BS during project execution (i.e. the railway scheduling constraint) (Van de Vonder et al., 2006, 2008; Zheng et al., 2018). Further notations are summarized in Table 1.

"Insert here Table 1"

Figure 1(a) shows a simple project network that will be used as an illustrative example throughout the paper. This project consists of 11 activities, and only one kind of resource is needed with a constant availability of 10 units.The three numbers in the bottom line of each rectangle represents the deterministic (baseline) duration, the per-period resource requirement, and the marginal penalty cost of each activity, respectively. To model the random activity durations we use a right-skewed lognormal distribution with mean equal to the baseline duration.

"Insert here Figure 1"

Figure 1(b) shows a baseline schedule with minimum makespan for the example project by applying a branch-and-bound algorithm for the RCPSP (Demeulemeester and Herroelen, 1992). The starting times of activities are represented by a list $\mathbb{S}_0^B = \{0, 0, 0, 0, 6, 4, 5, 2, 8, 9, 13\}$. It is known that the BS determines the starting time of each activity, as well as sequences the activities that use the same resource unit(s) through certain resource-driven precedence relations. An elegant way to represent those resource-driven relations is a resource flow network, $G' = (N, A_R)$, with N the same set of nodes as in the original project network $G = (N, A)$ and A_R the set of resource flow arcs (Artigues et al., 2003; Herroelen and Leus, 2004). Figure 2(a) shows a feasible resource flow network that is expressed by the resource profile representation. Note that A_R are connecting two nodes i and j if there exists a resource flow $f(i, j, k) > 0$ of any resource type k from activity i (when it finishes) to activity j (when it starts). It should be noted that it is often possible to have different schemes of resource allocations for a given BS. For instance, Figure 2(b) depicts another feasible resource flow network for the example schedule in Figure 1(b). In the context of this paper, the full arcs denote the direct precedence relationships in the original project network G, while the dashed arcs indicate additional precedence constraints imposed by the resource flows in the network G'.

"Insert here Figure 2"

Until now, the BS together with the resource flow networks is constructed in a deterministic environment. In order to protect against anticipated disruptions, researchers have advocated the use of time buffers in front of project activities. The scattered insertion of time buffers can reserve space for the time uncertainty of the activity that is caused by the disruption factors, and can prohibit the propagation of the disruption through the schedule, enhancing the solution robustness of the schedule (Herroelen and Leus, 2004; Pang et al., 2018). Extensive simulation experiments in previous studies have revealed that among the various time buffering approaches, the starting time criticality (STC) heuristic of Van de Vonder et al. (2008) ranks best in providing a solution robust project schedule. As mentioned in the introduction section, the stability cost is generally used as a solution robustness measure, i.e. $SC = \sum\limits_{j \in N} w_j \times E|s_j^R - s_j^B|$. And a fairly reliable approach for estimating this objective value is by using simulation. However, simulation can be very computationally demanding especially for very large projects. Besides, simulation does not deliver the information of resource transfers into our problem structure (Lambrechts, et al., 2011).

In order to measure solution robustness in a more efficient and effective manner, a surrogate measurement, which is inspired by the STC heuristic, is introduced as follows:

$$stc_j = w_j \times Pr\left(s_j^R > s_j^B\right) = w_j \times \sum_{\forall i:(i,j) \in T(A \cup A_R)} Pr\left(d_i^R > s_j^B - s_i^B - LPL\left(i,j\right)\right) \tag{1}$$

where $Pr\left(s_j^R > s_j^B\right)$ denotes the probability that the actual starting time of activity j has to be postponed due to the disruptions of its technologically constrained predecessors as well as extra resource-driven predecessors. $T\left(A \cup A_R\right)$ is defined as the set of all direct and transitive precedence relations in the extended network $G \cup G' = (N, A \cup A_R)$. $LPL(i,j)$ is the sum of the durations of all activities on the longest path between activity i and activity j in the extended network. The activity weight w_j remains the same with the marginal penalty cost w_j in SC. Note that a multiplication of the penalty cost and the probability of delay in Eq. (1) well measures the risk of activity delay and hence reflects the magnitude of the schedule stability or solution robustness.

Based on the two feasible resource flow networks represented in Figure 2(a) and Figure 2(b), the STC heuristic of Van de Vonder et al. (2008) is utilized to generate the corresponding buffered schedules as shown in Figure 3(a) and Figure 3(b). Note that a same project makespan of 16 is maintained in both cases. It is observed in Figure 3 that there is a one-unit buffer (marked by black squares) in front of activity 7 in the schedule I^B, whereas the buffer size in front of activity 7 is twelve units in the schedule II^B. The values of the total starting time criticality, $\sum\limits_{j \in N} stc_j$, for these two buffered schedules are 4.431 and 3.932, respectively. These results indicate that the resource allocation decision directly affects the time buffering process and hence the robustness of the resulted buffered predictive schedule.

"Insert here Figure 3"

3. A bi-objective model with composite robustness

The above-mentioned STC heuristic well protects an activity from disruptions that are propagated throughout the extended project network $G \cup G'$ by inserting time buffers between this activity and its predecessors, thus generating a predictive schedule with high solution robustness. This practice, however, can have a negative impact on the makespan performance. The project makespan, denoted as C_{max} (Stork, 2001), is a commonly-used objective function to measure quality robustness. In this section, the possible trade-off between the two types of robustness will be analyzed when an identical resource allocation shown in Figure 2(a) is maintained throughout the discussion.

Figure 4(a) and Figure 4(b) show two different buffered schedules generated using the STC heuristic when the project due dates are set to 18 and 20, respectively. The corresponding values of $\sum_{j \in N} stc_j$ for measuring the schedule stability are 1.473 and 0.629, respectively. Apparently, the project schedule becomes more solution robust when there are sufficient time buffers to cushion uncertainties, at the expense of a longer project makespan. In other words, the two types of robustness measures are somewhat conflicting.

"Insert here Figure 4"

Therefore, the fundamental research issue in this paper is to construct a bi-objective scheduling model that strikes a balance between quality robustness and solution robustness by the integration of robust resource allocation and time buffering. The first objective function is denoted as $Z_{qual} = C_{max}$, and the second objective function is $Z_{stab} = \sum_{j \in N} stc_j$. The two components of the objective function are of different magnitude and hence have to be normalized. In this paper, the commonly used min-max normalization method is employed (Han, 2005; Gajera et al., 2017). Specifically, the normalization of the makespan objective is computed by the following formula:

$$\tilde{Z}_{qual} = (Z_{qual} - Z_{qual}^{min})/(Z_{qual}^{max} - Z_{qual}^{min}) \tag{2}$$

where Z_{qual}^{max} is the maximum project makespan, which equals the predefined project due date (denoted as δ_n). And Z_{qual}^{min} is the minimum makespan generated by the branch-and-bound algorithm of Demeulemeester and Herroelen (1992).

Similarly, the normalization of the stability objective is calculated by the following equation:

$$\tilde{Z}_{stab} = (Z_{stab} - Z_{stab}^{min})/(Z_{stab}^{max} - Z_{stab}^{min}) \tag{3}$$

in which Z_{stab}^{max} and Z_{stab}^{min} are the maximum and minimum of the total starting time criticality obtained by the STC heuristic, respectively, when the project makespans are set to Z_{qual}^{max} and Z_{qual}^{min}, respectively.

In this way, both the makespan and stability values can be mapped to the range of $[0, 1]$. Next, a weighted sum of the two normalized components is defined as the optimization objective of our model, where a weighting parameter λ $(0 \leq \lambda \leq 1)$ maps the relative importance of the makespan performance versus the schedule stability in a specific project. Using the weighted sum of multiple objectives as the optimization goal can also be found in some other works, see, e.g. Ulusoy and Özdamar, 1995, Nudtasomboon and Randhawa, 1997, Al-Fawzan and Haouari, 2005, Abbasi et al., 2006, Voß and Witt, 2007, and Bomsdorf and Derigs, 2008.

The bi-objective optimization model can now be formulated in the following.

$$\min \quad \tilde{Z}_{comp}(\lambda) = \lambda \tilde{Z}_{qual} + (1 - \lambda)\tilde{Z}_{stab} \tag{4}$$

$$\begin{aligned}
s.t. \quad & s_j^B = \triangle_j + \max_{i \in Rpre_j^T}\left(s_i^B + d_i^B\right), \quad \forall j \in N && (5)\\
& s_j^R = \max\left(s_j^B, \max_{i \in Rpre_j^T}\left(s_i^R + d_i^R\right)\right), \quad \forall j \in N && (6)\\
& \sum_{j \in N} f(i,j,k) = \sum_{j \in N} f(j,i,k) = r_{ik}, \quad \forall i \in N \backslash \{0,n\}, \quad \forall k \in K && (7)\\
& \sum_{j \in N} f(0,j,k) = \sum_{j \in N} f(j,n,k) = R_k, \quad \forall k \in K && (8)\\
& \sum_{j \in s(t)} r_{jk} \leq R_k, \quad \forall k \in K && (9)\\
& \triangle_j \geq 0, \quad \forall j \in N && (10)\\
& s_0^R = s_0^B = 0 && (11)\\
& C_{max} = s_n^B && (12)\\
& s_j^R \in \mathbb{N}, \quad s_j^B \in \mathbb{N}, \quad \forall j \in N && (13)\\
& f(i,j,k) \in \mathbb{N}, \quad \forall i,j \in N, \quad \forall k \in K && (14)
\end{aligned}$$

Eq. (4) aggregates the two normalized components in a linear bi-objective function. Note that $\lambda = 1$ or $\lambda = 0$ makes the problem a single-objective optimization problem with minimum makespan or minimum schedule instability, respectively.

Eq. (5) imposes the precedence constraint, in which $Rpre_j^T$ is the set of all the direct and transitive predecessors of activity j in the extended network $G \cup G' = (N, A \cup A_R)$ and Δ_j is the size of time buffers inserted before activity j.

Eq. (6) specifies the railway scheduling constraint, in which activity j is not allowed to start earlier than its planned starting time in the BS nor before any of its predecessors.

Eq. (7) ensures that for each non-dummy activity j, the sum of the input and output resource flows of this activity must be equal to the resource requirement r_{jk}.

Eq. (8) denotes that for each resource type k, the sum of the output resource flows from the dummy start activity must equal the sum of the resource flows into the dummy end activity, both being equal to the resource availability R_k.

Eq. (9) enforces the renewable resource constraint, in which $s(t)$ is the set of activities in progress during time period t.

Eq. (10) ensures that the size of time buffers assigned for each activity is non-negative.

Eq. (11) specifies that there is no delay in the starting time of the dummy start activity 0.

Eq. (12) calculates the project makespan of the proactive schedule, which equals the start time of the dummy end activity.

Eq. (13) and Eq. (14) impose integrality on the activity start times and the resource flow variables.

The time buffering problem with single-objective has shown to be NP-hard. The problem studied in this paper deals with the two objectives by a weighted sum approach, which is also NP-hard. Therefore, a two-stage heuristic algorithm is designed in the next section for solving the proposed bi-objective model efficiently.

4. A two-stage algorithm

The two-stage algorithm deals with a robust resource allocation problem in the first stage and buffers the extended network with resource flows using a simulated annealing algorithm in the second stage. Note that the resource flows that solve the resource allocation problem for the initial schedule are preserved in the buffered schedule. The working principles of each heuristic approach are illustrated in detail as follows.

4.1 *Robust resource allocation*

According to the literature, the MABO heuristic proposed by Deblaere et al. (2007) has proven to be superior to any existing resource allocation algorithms that attempt to minimize the stability cost, i.e. $SC = \sum_{j \in N} w_j |s_j^R - s_j^B|$. The MABO procedure decides its best possible resource allocation at a minimum SC, computed through simulation of a number of executions of the partial schedule, assuming a fixed (incomplete) resource flow network. As pointed out by Zhang et al. (2011), a simulation-based method generally causes two problems. First, it consumes substantial computation times. Second, chances are that non-unique resource flow arcs will be generated for a certain project schedule due to the uncertain and random nature of simulation itself. In order to avoid these drawbacks, a surrogate evaluation index, $stc_j(A \cup A_R \cup H_j^q)$, is used in replacement of the SC criterion (see below). Besides, several new tie-break rules are proposed that are reasonable and effective as will be demonstrated in our simulation experiment. In view of the improvements based on the original MABO, our resource allocation method is named as the developed MABO, short for D-MABO. The detailed steps of D-MABO are presented in Algorithm 1, which consists of three steps that have to be executed for each activity j $(j \in N)$.

In the initialization step, the set of resource flow arcs A_R is initialized to the set of unavoidable resource arcs A_U, which is originally proposed by Deblaere et al. (2007) to reduce the solving difficulty of the resource allocation problem.

The aim of Step 1 is to calculate the resource units available for activity j, $Avail_{jk}(A \cup A_R)$, which equals the sum of resource units that the current predecessors i can allocate to activity j, i.e., $Avail_{jk}(A \cup A_R) = \sum_{\forall i:(i,j) \in T(A \cup A_R)} alloc_{ik}$. If the current predecessors of activity j have no sufficient resource units available for any resource type k, extra predecessors have to be added in Step 2.

In Step 2, the set H_j is defined as the set of all possible arcs between a potential resource source h of the current activity j and j itself. The set $H_j^i (i = 1, 2, ..., q)$ is a subset of H_j (i.e. $H_j^1, H_j^2, ..., H_j^q \subseteq H_j$), terms as the minimal subset (with q being the total number of minimal subsets), which not only accounts for the missing resource requirements of j for any resource type k, but also has a minimum number of activities. All the minimal subsets are evaluated. The one with the lowest $stc_j(A \cup A_R \cup H_j^q)$ will be selected and added to A_R. This index is calculated as follows:

$$stc_j(A \cup A_R \cup H_j^q) = w_j \times \sum_{\forall i:(i,j) \in T\left(A \cup A_R \cup H_j^q\right)} Pr\left(d_i^R > s_j^B - s_i^B - LPL\,(i,j)\right) \tag{15}$$

with $(A \cup A_R \cup H_j^q)$ the set of all direct and additional precedence arcs in the extended network $G \cup G' = (N, A \cup A_R)$.

The aim of step 3 is to allocate the actual resource flows $f(i,j,k)$ to the predecessors of activity j in $(A \cup A_R)$ and to update the number of units of resource type k allocated to each activity, $alloc_{ik}$.

Algorithm 1: Steps of the D-MABO method

```
Initialize: generate an initial schedule $\mathbb{S}_0^B$, set $A_R = A_U$, and $\forall k \in K$, $alloc_{0k} = R_k$;
Sort all the activities by increasing $s_j^B$ (tie-break: decreasing activity weight $w_j$) to get a sorted list LIST1;
```

```
For every activity $j$ in LIST1:
1. $\forall k$: calculate $Avail_{jk}(A \cup A_R) = \sum_{\forall i:(i,j) \in T(A \cup A_R)} alloc_{ik}$
2. If $\exists k$: $Avail_{jk}(A \cup A_R) < r_{jk}$
    2.1 Define the set of arcs $H_j$:
        $(h, j) \in H_j \Leftrightarrow (h, j) \notin A \cup A_R$, $s_h + d_h \leq s^B_j$, $\exists k : alloc_{hk} > 0$ and $Avail_{jk}(A \cup A_R) < r_{jk}$
    2.2 Determine all the minimal subsets $H^1_j, H^2_j, ..., H^q_j \subseteq H_j$ such that
        $\forall k \in K$: $Avail_{jk}(A \cup A_R \cup H^i_j) \geq r_{jk}, \quad i = 1, ..., q$
    2.3 Identify the subset $H^*_j \in \{H^1_j, H^1_j, ..., H^q_j\}$ such that
        $stc_j(A \cup A_R \cup H^*_j)$ is minimized
    2.4 Add $H^*_j$ to $A_R$
3. For each resource type $k$, allocate resource flows $f(i, j, k)$ to the arcs $(i, j)$ as follows:
    3.1 Sort the predecessors $i$ of activity $j$ by increasing number of the successors $l$ of activity
        $i$ with $s^B_l > s^B_j$ and $r_{lk} > 0$ to get a list LIST2:
        Tie-break 1: Decreasing activity weights $w_i$
        Tie-break 2: Decreasing finish times $s^B_i + d_i$
        Tie-break 3: Decreasing available resources $alloc_{ik}$
        Tie-break 4: If $i \in pred_j$, activity $i$ has the priority to allocate resource for activity $j$
        Tie-break 5: Sort activity $i$ randomly
    3.2 While $alloc_{jk} < r_{jk}$, take the next activity $i$ from LIST2
        $f(i, j, k) = min(alloc_{ik}, r_{jk} - alloc_{jk})$
        $alloc_{jk} = alloc_{jk} + f(i, j, k)$
        $alloc_{ik} = alloc_{ik} - f(i, j, k)$
```

After all this, the three-step procedure will be ultilized for the next activity in LIST1 until a complete feasible resource allocation is obtained at the end of LIST1.

4.2 *Time buffering*

While the D-MABO procedure in Section 4.1 enables the generation of a robust resource flow network, it also represents an un-buffered schedule with the shortest project makespan as well as the approximately lowest $\sum stc_j$ value, which will be adopted as an initial solution of the next phase. More specifically, a simulated annealing (SA) algorithm is designed in the second stage to optimally buffer the initial precedence, resource and deadline feasible schedule generated in the first stage. The neighborhood solutions are generated by changing the size of time buffers in front of an activity or a set of activities in an effort to make a trade-off between quality robustness and solution robustness. The essence of the proposed SA is illustrated below from three aspects: solution representation, neighborhood operators, and control parameters.

Solution representation

A feasible solution can be represented by the following two n-element lists:

- **Activity position list, L_{posi}:** This list defines the order that activities are started. It is a precedence-feasible permutation of activities, in which each activity has to be scheduled after all its predecessors and before all its successors so that no precedence constraints are violated.
- **Time buffer list, B_{posi}:** This list indicates the size of time buffers in front of each activity. A value of 0 means that no time buffer is inserted before the corresponding activity.

Then, a combination of the activity position list and the time buffer list, denoted by a pair of lists (L_{posi}, B_{posi}), can be decoded into a precedence and resource feasible schedule $\mathbb{S}^B = \{s^B_1, s^B_2 ... s^B_n\}$ by exploiting an extended serial schedule generation scheme (Kolisch, 1996). Note that this solution representation does not need to consider the resource feasibility, since the resource flow network generated previously has already resolved the resource conflicts. Obviously, the inclusion of robust resource allocations in the first stage speeds up the time buffering process.

Neighborhood operators

The neighborhoods of the current solution are generated by changing the size of time buffers at each iteration step without violating the precedence relations or resource constraints, as illustrated in the following steps.

Step 1: Randomly choose one activity X from the pair of lists $(L_{posi}^{curr}, B_{posi}^{curr})$;

Step 2: Increase or decrease the buffer size of activity X by a discrete value between $[-\Delta, +\Delta]$. At each iteration step, the neighborhood space of the current solution contains at most (2Δ) solutions. In our implementation Δ is experimentally set to 3.

Step 3: Move forwards or backwards activity X itself and all its direct and transitive successors upon the insertion of time buffers, leading to the corresponding neighbor solutions;

Step 4: If the project completion time is within the deadline, this neighbor solution is regarded as a candidate; otherwise it is deemed to be infeasible. A set $N(L_{posi}^{neig}, B_{posi}^{neig})$ is defined to denote the set of all the immediate feasible neighbor solutions of $(L_{posi}^{curr}, B_{posi}^{curr})$ in this iteration.

In each iteration, the improvement of the composite objective function, $\Delta\tilde{Z}_{comp}^{curr}(\lambda) = \tilde{Z}_{comp}^{neig}(\lambda) - \tilde{Z}_{comp}^{curr}(\lambda)$, is evaluated for each candidate in the set $N(L_{posi}^{neig}, B_{posi}^{neig})$. If $\tilde{Z}_{comp}^{neig^*}(\lambda) = \min \tilde{Z}_{comp}^{neig}(\lambda)$ is less than the current objective function value $\tilde{Z}_{comp}^{curr}(\lambda)$, the current solution $(L_{posi}^{curr}, B_{posi}^{curr})$ will be replaced by the neighbor solution $(L_{posi}^{neig^*}, B_{posi}^{neig^*})$.

Furthermore, the multi-objective problem in this article aims at generating the approximations of the non-dominated solutions $(ANDS)$. Let $ANDS_\lambda$ be the set of the potentially efficient solutions of the single-objective problem defined by λ. At each step of the SA, the generated neighborhoods are scanned and the set $ANDS_\lambda$ is updated through removing dominated solutions.

Control parameters

- **Initial temperature:** The initial value of the temperature T^{init} is calculated by the equation, $T^{init} = \Delta\tilde{Z}_{comp}^{init}(\lambda) \Big/ ln(\chi^{init})$, where $\Delta\tilde{Z}_{comp}^{init}(\lambda)$ is the range of change in the objective function $\tilde{Z}_{comp}^{init}(\lambda)$ after 50 random moves of the initial solution, and the initial acceptance ration χ^{init} is defined as the assumed proportion between accepted moves and all the moves generated for T^{init}.
- **Markov chain length:** The length of the Markov chain determines the number of transitions at a certain temperature level, which is calculated as $L = 10N$ in this implementation where N is the number of activities in a project.
- **Cooling scheme:** In order to make the procedure more selective, we progressively decrease the temperature according to the decreasing function: $T^{curr} := \mu T^{curr}$, in which the cooling rate μ is set to 0.9.
- **Stopping criterion:** The search process terminates when the current temperature T^{curr} drops to a certain threshold, i.e., $T^{curr} \leq T^{stop}$, where the final temperature T^{stop} is set as 0.01 in our implementation.

4.3 *An illustrative example*

The gist of the proposed optimization algorithm can be sketched using still the project instance shown in Figure 1(a). In the first stage, an un-buffered schedule together with its resource flow network by the D-MABO procedure is displayed in Figure 5(a), with a project makespan of $Z_{qual} = 13$ and a stability criterion of $Z_{stab} = \sum stc_j = 11.821$. Recall that the corresponding values of $\sum stc_j$ for the two randomly generated resource allocations shown in Figure 2 are 15.172 and 16.097, respectively (the makespans both being 13). It is obvious in this example that the schedule stability gets improved through proper resource allocations compared to the case where random resource allocations are used.

"Insert here Figure 5"

The SA algorithm is then used in the second stage based on the obtained resource flow network to generate a buffered project schedule. Table 2 displays the optimized results under different values of λ. Note that the two components of the objective function without normalization are listed for practical reasons. Let us take a look at the performance of the proposed two-stage algorithm when $\lambda = 0.7$. Figure 5(b) depicts the corresponding robust schedule with $Z_{qual} = 16$ and $Z_{stab} = 2.469$. Recall that the $\sum stc_j$ values for the two buffered schedules shown in Figure 3 are 4.431 and 3.932, respectively (the makespans both being 16). This result indicates that the buffered schedule obtained by our two-stage algorithm not only further optimizes the schedule stability within a reasonable project makespan compared with the unbuffered schedule generated in the first stage, but also achieves higher solution robustness compared to the buffered schedule with random resource allocations.

"Insert here Table 2"

5. Computational experiments

In this section, the results of an extensive computational experiment are provided to show the capability of the proposed two-stage algorithm to generate proactive project schedules with composite robustness.

5.1 *Experimental layout*

Our computational experimentation was conducted using the 30-activity instances of the well-known PSPLIB data set, with the number of project instances equal to 480 (i.e. NUM=480) (Kolisch and Sprecher, 1997). The stochastic activity durations are assumed to follow a right-skewed lognormal distribution, which is also used by some other works (Herroelen and Leus, 2001; Hu et al., 2015, 2016). More specifically, the realized duration d_j^R for activity j is randomly generated by the lognormal distribution function, allowing us to simulate the project execution with varying levels of uncertainty (represented by the standard deviation, σ) in the activity durations while keeping the mean durations unchanged. Three levels of σ (i.e. $\sigma \in \{0.3, 0.6, 0.9\}$) are used to represent a project uncertainty that is Low (L), Medium (M) or High (H), respectively.

The initial project schedule with minimum makespan, s_n^0, is generated by the branch-and-bound algorithm of Demeulemeester and Herroelen (1992). The due date of every project is set to 130% of the minimum project makespan as was done in most literature (Vonder de Vonder et al., 2006, 2008), i.e. δ_n=$1.3 \times s_n^0$. The activity weight w_j for each non-dummy activity j is drawn from a discrete triangular distribution with $P(w_j = x) = 0.21 - 0.02x$, in which $x \in \{1, 2, .., 10\}$, $\forall j \in N \backslash \{0, n\}$. The weight of the dummy end activity, w_n, denotes the marginal tardiness penalty cost of the project completion beyond the due date δ_n. Van de Vonder et al. (2005) defines a weighting parameter WP to indicate the ratio between w_n and the average of the distribution of all other activity weights w_{aver}, i.e., $w_n = WP \times w_{aver}$, in which $w_{aver} = \sum\limits_{j=1}^{n-1} w_j/(n-1)$. This weighting parameter WP measures the importance of on-time project completion and is set to 5 in our experiment.

In order to evaluate the robustness of the generated proactive schedules against duration variabilities, we need to simulate the real execution of the baseline schedule subject to both technologically constrained and resource flow-based precedence relations. To this aim, the simulation-based solution robustness measure, $SC = \sum\limits_{j \in N} w_j \times E|s_j^R - s_j^B|$, is used as the performance metric for each project schedule rather than the $\sum_{j \in N} stc_j$ measure in the objective function. And $\overline{SC}$ is defined to represent the average stability cost over all J30 instances. The simulated execution of a baseline project schedule uses the parallel scheduling generation scheme (PSGS) (Kolisch, 1996) and follows the railway scheduling policy. For each project instance and for each combination of factor settings, 1,000 simulation replications ($M = 1,000$) were generated and the average performances were calculated for various methods that are to be tested.

5.2 *Comparison results of resource allocation methods*

In this section, a set of experiments was first conducted to verify the efficiency of the proposed D-MABO method as opposed to two other approaches for resource allocation, i.e. a random procedure by Artigues et al. (2003) (termed as RRAS, *random resource allocation scheme*) and the original MABO algorithm by Deblaere et al. (2007).

Apart from the $\overline{SC}$ metric, another four indicators are defined to evaluate the performances of these three resource allocation algorithms, which are denoted in the following:

- *Resource allocation variability, RAV*: For each instance in the J30 data set, each resource allocation method is run 100 trials to generate the corresponding resource flow networks. A parameter Num_{inst} is used to denote the total number of resource flow networks that are different from each other for a given instance network. The RAV metric is calculated as $RAV\% = (\sum_{inst=1}^{NUM} Num_{inst} - NUM) \Big/ NUM * 100$.
- *Average relative deviations of SC, ARD*: $ARD\% = \sum_{m=1}^{M} |SC(m) - SC| / (SC * M) * 100$, where $SC(m)$ denotes the stability cost of the schedule in the m^{th} simulation run.
- *Average computational time, ACT*.
- *Maximum computational time, MCT*.

The comparison results of the three resource allocation procedures are presented in Table 3. It is observed that the proposed D-MABO method always generates lower values of $\overline{SC}$ and $ARD\%$ than RRAS and MABO under three uncertainty levels. This means that the resource flow networks generated by D-MABO are more solution-robust (i.e. more stable) than the other two approaches. Recall that the RAV metric measures the diversity of resource transferring plans for a certain project schedule. In practice, a unique solution is generally preferred in order to better support management decisions, otherwise the project manager will be struggling to change plans once the resource allocation algorithm is applied for a wide variety of scenarios. The D-MABO well guarantees the uniqueness of resource allocations (i.e. $RAV\%$=0) for a given schedule, whereas the random or simulation-based nature of RRAS or MABO might lead to different resource allocations (i.e. $RAV\% >$0) at a time.

As to the computational times, RRAS has the fastest speed. This follows from the fact that this method only aims to generate feasible resource flow networks without any optimization of the robustness performance. Besides, the adapted D-MABO algorithm is faster than the original MABO since it uses a surrogate evaluation index instead of relying on simulation when deciding on the best resource flows that should be added to the existing network. It is worth noting that MABO has proven to be superior to any existing resource allocation algorithms with the objective of minimizing SC (Demeulemeester and Herroelen, 2011). Yet, our method is capable of achieving a lower SC than MABO with a faster solution speed. This result effectively demonstrates that it is quite appropriate to adopt $\sum stc_j$ as a surrogate measure, which cannot only save a lot of computation time, but can also achieve a remarkable stability performance.

"Insert here Table 3"

5.3 *Comparison results of the two-stage algorithm*

This section investigates the capability of the proposed SA algorithm to generate buffered, robust project schedules in combination with the three resource allocation procedures illustrated in the last section. In practice, the project manager is generally interested in the real data of makespan and stability metrics rather than the normalized objective function values. Therefore, Table 4 provides the average project makespan ($\overline{C_{max}}$) and the average stability cost ($\overline{SC}$) over all J30 instances, based on 1,000 simulation runs per project under three levels of duration variability.

First of all, as uncertainty in the project environment increases both $\overline{C_{max}}$ and $\overline{SC}$ increase as predicated in all cases. It is obvious that the higher the duration variability, the more risks the project will be faced with, inevitably causing an adverse effect on the schedule robustness. Secondly, as λ increases, meaning that the project manager is more in favor of quality robustness compared to solution robustness, the project makespan decreases whereas the schedule instability increases in Table 4. This result accords with common sense and hence verifies the feasibility of the SA algorithm for deriving near optimal solutions. Remark that in the case of $\lambda = 1$ the problem aims to minimize the project makespan individually, which is equivalent to the first-phase resource allocation problem (based on the deterministic minimum-makespan schedule). Due to the different dimensions of the makespan and stability metrics, a proper normalization of the two components is done from the start, which can better support the manager's decision to strike a balance between the two robustness measures based on his/her objective predilection.

Last but not the least, the proposed two-stage algorithm D-MABO+SA always achieves lower values of $\overline{C_{max}}$ and $\overline{SC}$ than those of the other two algorithms (RRAS+SA and MABO+SA) in all cases for λ, which further demonstrates the superiority of D-MABO for generating robust resource allocations in the first phase. It is notable that the benefit of a robust buffered schedule generated based on the resource flow network is to avoid the occurrence of resource conflicts. Namely, the change of the random activity duration will not affect the transmission of resources among activities, as long as the transfer relationship of resources is unchanged. Therefore, the integration of resource allocation and time buffering can largely facilitate the generation of a robust project schedule in practice.

"Insert here Table 4"

In what follows, two multi-objective performance metrics are further defined for comparing sets of pareto solutions obtained by the three solving algorithms (i.e. RRAS+SA, MABO+SA, and D-MABO+SA). The first metric is defined by $\eta_h = \sum_{inst=1}^{NUM} \eta_{inst,h}$, where h represents the h^{th} algorithm that is to be evaluated, and NUM is the total number of project instances. $\eta_{inst,h}$ is calculated by the following equation, based on the work of Al-Fawzan and Haouari (2005):

$$\eta_{inst,h} = |ANDS_{inst,h} \cap ANDS_{inst}|/|ANDS_{inst}|, inst = 1, 2, ..., NUM, h = 1, 2, 3 \qquad (16)$$

Specifically, $\eta_{inst,h}$ denotes the contribution of algorithm V_h to the approximate non-dominated set $ANDS_{inst}$ of the $inst^{th}$ instance in the J30 PSPLIB data set. Consequently, if V_h dominates the other algorithms, the value of $\eta_{inst,h}$ would be close to 1.

The second performance metric is the so-called hypervolume value, HV, which represents the size of the objective space covered by the approximations of the non-dominated set (Batt, 2011; Yen and He, 2014). The hypervolume value used in our experiment is obtained by the LebMeasure algorithm (Fleischer, 2003). It calculates the volume that is dominated exclusively by one point in the approximate set, discards that point and then moves on to the subsequent point until all points have been processed and all volumes have been summed.

The results of the approximations of non-dominated sets for the three algorithms are displayed in Table 5. One can well perceive that D-MABO+SA clearly outperforms RRAS+SA and MABO+SA regarding the two multi-objective performance metrics. That is, the value of η_3 ranks highest among all three methods, and the hypervolume value of D-MABO+SA is significantly better than any other algorithms. To sum up, the experiments show that the proposed robust scheduling method generates high-quality solutions when considering the two objectives simultaneously. The algorithm is also able to obtain a set of non-dominated solutions with remarkable performance, showing its practicality and superiority for solving real-world project scheduling problems under uncertainty.

"Insert here Table 5"

6. Conclusion

In this paper, a bi-objective resource-constrained project scheduling problem with objectives of makespan minimization (quality robustness) and stability maximization (solution robustness) is investigated. Our contribution is three-fold. First, the starting time criticality is used as an excellent measure of solution robustness, which is different from the current literature. Second, the impact of different resource flow networks on the robustness of the buffered predicative schedules is investigated for the first time, and an integrated bi-objective optimization model for scheduling, resource allocation and time buffering is therefore proposed. In the model, the two components of the objective function are properly normalized using min-max normalization in order to deal with the different scaling of makespan and stability metrics. Third, a two-stage heuristic procedure is developed for solving the problem with remarkable performance, in which an improved way of generating robust resource allocations is introduced and a customized simulated annealing algorithm is designed to optimally buffer the schedule. Our simulation experiments demonstrate that the model and algorithms are applicable and beneficial to the problem in practice.

While the initial findings are promising, further research is necessary. First, due to the fact that both resources and buffers are allocated to a given initial un-buffered schedule, the impact of different initial schedules on the effectiveness of the resource allocation and time buffering processes can be a topic for future research. Second, although our computational experiment reveals the advantages of the integration of resource allocation and time buffering for generating robust project schedules, whether this integrated approach is beneficial for solving the RCPSP with other optimization criteria (i.e. the net present value) under uncertainty remains to be answered. A third significant area of research would be to consider multi-objective robust scheduling in a multi-project environment.

Funding

This work was supported by the National Natural Science Foundation of China [Grant numbers 71701067, 71801218, and 71271097]. The authors would like to thank the editors and the anonymous reviewers for their invaluable comments and suggestions, which help to improve this paper.

References

[1] Abbasi, B., Shadrokh, S., and Arkat, J. 2006. "Bi-objective resource-constrained project scheduling with robustness and makespan criteria". *Applied Mathematics and Computation* 180(1): 146-152.
[2] Al-Fawzan M. A., and Haouari M. 2005. "A bi-objective model for robust resource-constrained project scheduling". *International Journal of Production Economics* 96(2): 175-187.
[3] Artigues, C., Michelon, P., and Reusser, S. 2003. "Insertion techniques for static and dynamic resource-constrained project scheduling". *European Journal of Operational Research* 149(2): 249-267.
[4] Artigues, C., Leus, R., and Nobibon, F. 2013. "Robust optimization for resource-constrained project

scheduling with uncertain activity durations". *Flexible Services & Manufacturing Journal* 25(1-2): 175-205.

[5] Artigues, C., Leus, R., and Nobibon, F. T. 2015. "Robust optimization for the resource-constrained project scheduling problem with duration uncertainty". Handbook on Project Management and Scheduling Vol. 2. Springer International Publishing.

[6] Batt R. 2011. "Theoretical and practical fundamentals for multi-objective optimisation in resource-constrained project scheduling problems". *Computers & Operations Research* 38(1): 51-62.

[7] Bomsdorf, F., and Derigs, U. 2008. "A model, heuristic procedure and decision support system for solving the movie shoot scheduling problem". *OR Spectrum* 30(4): 751-772.

[8] Bruni M. E., Pugliese L. D. P., Beraldi P., and Guerriero, F. 2017. "An adjustable robust optimization model for the resource-constrained project scheduling problem with uncertain activity durations". *Omega* 71: 66-84.

[9] Davari, M., and Demeulemeester, E. 2017. "The proactive and reactive resource-constrained project scheduling problem". *Journal of Scheduling.* DOI: https://doi.org/10.1007/s10951-017-0553-x.

[10] Deblaere, F., Demeulemeester, E., Herroelen, W., and Van de Vander, S. 2007. "Robust resource allocation decisions in resource constrained projects". *Decision Sciences* 38(1): 5-37.

[11] Demeulemeester, E., and Herroelen W. 1992. "A branch-and-bound procedure for the multiple resource-constrained project scheduling problem". *Management Science* 38(12): 1803-1818.

[12] Demeulemeester, E., and Herroelen, W. 2011. "Robust project scheduling". *Foundations & Trends in Technology Information & Operations Management* 3(3-4): 201-376.

[13] Demeulemeester, E., and Herroelen, W. 2002. "Project scheduling: a research handbook". *Springer Science & Business Media* 102(1-2): 1-685.

[14] Fleischer M. 2003. 'The measure of Pareto optima applications to multi-objective metaheuristics". *International Conference on Evolutionary Multi-Criterion Optimization. Springer-Verlag* 519-533.

[15] Gajera, V., Shubham, Gupta, R., and Jana, P. K. 2017. "An effective multi-objective task scheduling algorithm using min-max normalization in cloud computing". *International Conference on Applied & Theoretical Computing & Communication Technology.* IEEE.

[16] Ghoddousi, P., Ansari, R., and Makui, A. 2016. "An improved robust buffer allocation method for the project scheduling problem". *Engineering Optimization* 49(4):718-731.

[17] Goldratt E.M.1997. "Critical Chain". *New York, The North River Press.*

[18] Hall, N. G., Long, D. Z., Qi, J., and Sim, M. 2015. "Managing underperformance risk in project portfolio selection". *Operations Research* 63(2): 660-675.

[19] Han, J. 2005. "Data Mining: Concepts and Techniques". *USA, Morgan Kaufmann Publishers Inc.*

[20] Herroelen, W., and Leus, R. 2001. "On the merits and pitfalls of critical chain scheduling". *Journal of Operations Management* 19(5): 559-577.

[21] Herroelen, W., and Leus, R. 2004. "Robust and reactive project scheduling: a review and classification of procedures". *International Journal of Production Research* 42(8): 1599-1620.

[22] Hu, X. J., Cui, N. F., Demeulemeester, E., and Bie, L. 2016. "Incorporation of activity sensitivity measures into buffer management to manage project schedule risk". *European Journal of Operational Research* 249(2): 717-727.

[23] Hu, X. J., Cui, N. F., and Demeulemeester, E. 2015. "Effective expediting to improve project due date and cost performance through buffer management". *International Journal of Production Research* 53(5): 1460-1471.

[24] Kolisch, R. 1996. "Serial and parallel resourceconstrained project scheduling methods revisited: Theory and computation". *European Journal of Operational Research* 90: 320-333.

[25] Kolisch, R., and Sprecher, A. 1997. "PSPLIB-a project scheduling problem library: OR software-ORSEP operations research software exchange program". *European Journal of Operational Research* 96(1): 205-216.

[26] Lamas, P., and Demeulemeester, E. 2017. "A purely proactive scheduling procedure for the resource-constrained project scheduling problem with stochastic activity durations". *Journal of Scheduling* 19(4): 409-428.

[27] Lambrechts, O., Demeulemeester, E., and Herroelen, W. 2011. "Time slack-based techniques for robust project scheduling subject to resource uncertainty". *Annals of Operations Research* 186(1): 443-464.

[28] Leus, R. 2003. "The generation of stable project plans". *Quarterly Journal of the Belgian French & Italian Operations Research Societies* 2(3): 251-254.

[29] Leus, R., and Herroelen, W. 2004. "Stability and resource allocation in project planning". *IIE Transactions*, 36(7): 667-682.

[30] Ma, Z., He, Z., Wang, N., Yang Z., and Demeulemeester, E. 2018. "A genetic algorithm for the proactive resource-constrained project scheduling problem with activity splitting". *IEEE Transactions on Engineering Management* (99): 116.

[31] Nudtasomboon, N., and Randhawa, S. U. 1997. "Resource-constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs". *Computers* & *Industrial Engineering* 32(1): 227-242.

[32] Pang, N. , Su, H. , and Shi, Y. 2018. "Project robust scheduling based on the scattered buffer technology". *Applied Sciences*, https://doi.org/10.3390/app8040541.

[33] Policella, N. 2005. "Scheduling with uncertainty: A proactive approach using Partial Order Schedules." *AI Communications* 18(2):165-167.

[34] Stork, F. 2001. "Stochastic resource-constrained project scheduling". phd thesis. *Dissertations* & *Theses - Gradworks* 45(3): 452-454.

[35] Ulusoy, G., and Özdamar, L. (1995). "A heuristic scheduling algorithm for improving the duration and net present value of a project". *International Journal of Operations* & *Production Management* 15(1): 89-98.

[36] Van de Vonder, S., Demeulemeester, E., and Herroelen, W. 2008. "Proactive heuristic procedures for robust project scheduling: an experimental analysis". *European Journal of Operational Research* 189(3): 723-733.

[37] Van de Vonder, S., Demeulemeester, E., Herroelen, W., and Leus, R. 2005. "The use of buffers in project management: the trade-off between stability and makespan". *International Journal of Production Economics* 97(2): 227-240.

[38] Van de Vonder, S., Demeulemeester, E., Herroelen, W., and Leus, R. 2006. "The trade-off between stability and makespan in resource-constrained project scheduling". *International Journal of Production Research* 44(2): 215-236.

[39] Voß, S., and Witt, A. 2007. "Hybrid flow shop scheduling as a multi-mode multi-project scheduling problem with batching requirements: a real-world application". *International Journal of Production Economics* 105(2): 445-458.

[40] Yen, G. G., and He, Z. 2014. "Performance metric ensemble for multiobjective evolutionary algorithms". *IEEE Transactions on Evolutionary Computation* 18(1), 131-144.

[41] Zhang, S. Q., Chen, X. D., Chen, Q. X., and Xin, C. 2011. "Reactive scheduling algorithm for multiple mould and die projects based on optimized resource flow constraints". *Systems Engineering-Theory* & *Practice* 31(8): 1571-1580.

[42] Zheng, W., He, Z., Wang, N., and Jia, T. 2018. "Proactive and reactive resource constrained Max-NPV project scheduling with random activity duration". *Journal of the Operational Research Society* 69(1): 115126.