Redefining Critical Tasks for Responsive and Resilient Scheduling—An Intelligent Fuzzy Heuristic Approach

TAIBA ZAHID¹, MUITABA HASSAN AGHA¹, SALMAN SAGHEER WARSİ², AND USMAN GHAFOOR³
¹NUST Business School, National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan
²Department of Mechanical Engineering, Capital University of Science and Technology, Islamabad 44000, Pakistan
³Department of Mechanical Engineering, Institute of Space Technology, Islamabad 44000, Pakistan
Corresponding author: Usman Ghafoor (usmanghafoor99@gmail.com)

ABSTRACT Rapidly changing market has accentuated the need for developing robust and responsive schedules. Intelligent heuristics techniques have been widely used to provide scheduling alternatives for resource constrained scheduling and identify critical activities essential for monitoring and controlling these processes. We argue that traditional parameters based solely on slack calculation are insufficient to characterize tasks as critical and instead define a fuzzy parameter-based approach, which relies on process and task-specific information. Proposed parameter is used to rank activities on a scale based on their criticality to develop a heuristic. It was observed that the proposed intelligent heuristic performed better on well-known benchmark datasets in terms of both lead times and computation time. An improvement of 1.5-2.5 percent was observed in traditional performance measure of order lead times while for non-traditional criterion measuring robustness, improvement of 30-35 percent was observed. Furthermore, current research provides meaningful insight on critical tasks at the shop-floor.

INDEX TERMS Project scheduling, responsive, heuristics, critical, fuzzy.

I. INTRODUCTION

Resource constrained project scheduling problem (RCPSP) is a well-known problem class in project and operations management. The problem RCPSP was first classified by Conway et al., [1] and later refined by Brucker [2]. Unlike traditional scheduling problems which focus only on task precedence, the RCPSP aim to develop task schedule without violating neither precedence nor resource constraint. The standard resource constrained scheduling problem can be categorized as a combinatorial problem [3] with a limited set of resources and can be termed as NP-Hard [4]. Exhaustive search methods are limited in their capacity for finding optimal solutions of such problems [5]. Hence heuristics which can be defined as an approximate method designed to find a good solution in reasonable amount of time, have been proposed to solve RCPSP. To evaluate performance of a schedule, makespan is a well-established criterion where it is defined as the time required to complete a project [6].

Since it is not possible to define an exact optimal solution for such large-scale combinatorial problems, a lower bound is necessary to compare proposed algorithms in this area. For this purpose, critical path (CP; minimal time required to complete the project with infinite resources) obtained from a baseline schedule is considered as a lower bound [7], [8].

Rapidly changing products and systems have led researchers to focus on flexible solutions for industrial problems [9], [10]. It is imperative to propose dynamic solutions and model uncertainty in our problems for realistic scenarios [11]. Likewise, schedules need to be flexible as well to cope with these industrial requirements. This flexibility and adaptability is termed as robustness in scheduling. Terms such as quality and solution robustness are also used to categorize various forms of robustness [12]. Robustness in a schedule can be defined as an ability to achieve target values (with little or no error) in case of external disruptions [13]. At operational and scheduling level, these external disruptions can be in the form of machine breakdowns, resource discrepancies and/or unexpected orders etc. [14], [15]. Thus, robust schedules need to be responsive in case of such
The objective of such schedule recovery approaches is to generate a schedule that shows minimum deviation from the target value set by the lower bound. Various schedule recovery heuristics have been discussed in literature which use CP information for buffer management. As mentioned earlier, critical path is defined as the minimal time required to complete the project. It divides activities of the project into two major categories which are critical and non-critical by calculating their float or slack value. Critical activities are those which have zero slack/float whereas, float is defined as the time any activity in the project can be delayed or shifted without delaying the project deadline. By definition, float is a naturally inherent buffer in an activity and thus, this buffer is used to control and monitor the project. Schedule recovery algorithms work to insert and arrange buffers in the schedule with information provided by these critical activities. The well-known critical chain method was also developed on this concept and assigned project and feeder buffers based on the critical path of a schedule [18]. Another study [19] proposed a scheduling methodology with buffers assigned to the activities by considering their pessimistic durations. However, the resulted project makespan was far from optimal in this case. Another strategy considered float of an activity to be directly proportional to its impact on project tardiness and assigned instability cost to each task based on this assumption [20].

A similar study proposed by Al-Fawzan and Haouari [21] who measured risk for the project by calculating critical activities of the project. Van de Vonder et al., [22] used starting time criticality (STC) as a parameter for buffer insertion based on dynamic float of activities. Aforementioned studies use critical path knowledge to control, monitor and schedule projects. Activities are ranked according to their critical values in the project and the more critical activities are scheduled first [23]. Various criteria have been proposed to evaluate flexibility and robustness of such schedules. For instance, a measure to compare flexible schedules is the number of critical tasks in a project [24]. Another well-known measure is the use of average or maximum float of the activities in a schedule [25], [26].

In conclusion, it can be observed that information from critical path is used for the development of robust and responsive schedules. Moreover, this analysis which differentiates activities into critical and non-critical activities is significant to control and monitor the project. As mentioned, activities are listed critical or non-critical based on their float value. However, it has been argued in literature that the float measure cannot solely identify critical nature of the project activities and proposes other parameters. For instance, since total float may overestimate a project buffer, free or next float (NF) has been proposed [27]. Next float is defined as the time an activity can be delayed without delaying its immediate successor [28]. However, these parameters only regard float of an individual activity to characterize it as critical whereas, disregard several significant factors such as completion time of the project and processing time of an individual activity. Zahid et al., [29] proposed a float index which considers these parameters and assigns a numeric value to activities. The proposed measure performed well and was observed to be more effective to determine critical nature of an activity.

We argue that in addition to the float-based measures, position of an activity within a project network is also significant. The placement of buffers in a project is also relevant with the position of critical activities in any project [30]. As both float and position are considered vital parameters, it is important to use both of them to define critical nature of activities in a project. Motivated by these observations, main contributions of the proposed work are:

1. Proposal of a fuzzy parameter based on knowledge of float and position of the activity.
2. Development of dynamic heuristic based on fuzzy rules for robust schedules.

We evaluate performance of our proposed heuristic with well-known benchmarks in literature. The results are compared using both static (makespan) and robust measures (sum of float or number of critical tasks). It is observed that the proposed heuristic performed well in both cases. The proposed parameter proved to provide better insight for identifying sensitivity of the project activities.

In the next section we discuss the proposed fuzzy parameter which is used to characterize activities as critical and rank them for priority heuristic scheduling.

II. METHODS

A. CRITICAL INDEX

As discussed in the previous section, critical path knowledge is important for monitoring and evaluation of project schedules. It plays a vital role in distinguishing critical activities from non-critical. Moreover, the information is used to rank and schedule activities for heuristic scheduling [31]. However, merely float is insufficient to mark activities as critical [32], [33]. To consider project and activity specific information, a float index (F.I) was proposed [29] which proved to be a better indicator for defining criticality of the project activities. For each task, F.I was calculated as follows:

$$\text{Float Index} = F.I = 0.5 \times \left[1 - \frac{\text{Activity Duration}}{\text{Makespan}} \right] + 0.5 \times \frac{\text{Next Float}}{\text{Activity Duration}}$$

(1)

However, position of any task in the project is also important while determining its criticality. Literature has proposed a few position based measures in order to rank and schedule activities based on the number of successors [34]) with
studies available to compare these heuristics on test bench instances [35]. Here we define a position index (P.I) to determine place of an activity in the project.

**B. POSITION INDEX**

\[
\text{Position Index} = P.I = \frac{\text{Act Position in project chain}}{\text{Total no. of activities}}
\]

Motivation to use location of an activity as a sensitivity parameter comes from an observation that activities performed at later stages of a project are exposed to more risk. In case of any resource breakdown, an activity which is performed at the initial stages of a project tends to be less perturbing than the one which is performed later. This is because the project manager has more time to reschedule and respond. Based on these observations, it has been suggested to use position of activities for buffer assignment [30].

With these observations, we deduce that in addition to the project characteristics, float and location of an activity are equally important to mark activities as critical. We propose a critical index (C.I) that is calculated by considering all the factors. A fuzzy inference system is used to combine this knowledge and will be discussed in the next lines.

**C. FUZZY INFERENCE SYSTEM**

It was observed that in practical databases, performance changes are more gradual rather than being definitive [36]. Hence, instead of placing project activities in two categories i.e. critical and non-critical) an alternate methodology is required which captures this gradual shift. Fuzzy variables in contrast to Boolean variables enable us to translate input parameters in a more logical and continuous manner [37]. As a result, a fuzzy inference system based is proposed in this study which provides a more sophisticated method to model knowledge-based rules and assign a unique Critical Index (C.I) for each task.

P.I and F.I (discussed above) are used to calculate critical index of each task which is used to rank and schedule activities in the project. The fixed inputs (F.I and P.I) for each task are transformed into fuzzy variables and later on, tested on each of the defined knowledge-based rules.

It would be worth mentioning the difference between a fixed and fuzzy variable here. Based on the amount of float (fixed variable), the activity can be regarded as critical or non-critical. For instance, an activity with zero float is termed as critical while activities with float values of 1 and 10 would both be termed as non-critical, regardless of their difference between the size of floats. In fuzzy mathematics, each activity with a float would be assigned a membership function that will judge criticality of an activity. Membership function defines the truth value of a system on a scale of 0-1. For instance, an activity having a float value of 1 can have a membership function of 0.8 for being critical (due to a small float/slack) and 0.2 for being non-critical. On the other hand, an activity with a float value of 10 may have a membership function for being critical to be 0.2 and 0.8 for being non-critical (due to a larger float). It should be noted that sum of membership functions for all criteria will always be equal to one. In this study, we define three criteria (i.e. low, medium and high) for all variables (i.e. P.I, F.I and C.I) and for every activity, measure a membership function value ranging from 0-1 for each criterion. Figure 1 below show this phenomenon and will be explained in detail here.

1) **STEP I (TRANSFORMATION)**

The constant value of P.I and F.I of each activity is transformed into a fuzzy variable. Three categories were defined for both P.I and F.I to transform an activity. Most commonly used shape functions (L, R & triangular) are used for this conversion [38]. For membership functions, equidistant uniform partition was used to define boundaries; a standard method proposed for such modelling [39]. Figure 2 shows the phenomena in which a constant value of P.I and F.I (say ‘x’ and ‘y’) is assigned a membership function/truth value in the low (\(\mu_l\)), medium (\(\mu_m\)) and high (\(\mu_h\)) range category with standard formulas shown in the figure. Table 1 shows the values obtained with P.I value of 0.2 and F.I value of 0.415. As the input ‘x’ < c where c = 0.3 (see fig. 2) the membership value for P.I low range (\(\mu_l\)) would be 1, and 0 for the other two (i.e. \(\mu_m\) & \(\mu_h\)). Similarly, for a F.I value of 0.415, the input ‘y’ lies between c and d (see Figure 2). Therefore, the membership function for F.I high range (\(\mu_h\)) would be 0 as \(y < d\) while for low range \(\mu_l = 0.62\) and medium range \(\mu_m = 0.38\).

Hence, constant values of P.I and F.I of an activity are transformed into fuzzy membership functions. Activity information (e.g. EST, LST, TF etc.) for first simulation run is obtained through critical path analysis with infinite resources (i.e. baseline schedule).

### TABLE 1. Example P.I and F.I calculation.

| Act | P.I | P.I(\(\mu_l\), \(\mu_m\), \(\mu_h\)) | F.I | F.I(\(\mu_l\), \(\mu_m\), \(\mu_h\)) |
|-----|-----|-----------------------------------|-----|----------------------------------|
| A   | 0.2 | (1,0,0)                           | 0.415 | (0.62,0.38,0)                    |

2) **STEP II (APPLICATION AND IMPLICATION OF RULES)**

Knowledge based approach was applied for the development of rules. As discussed, two activity parameters (i.e. F.I and P.I) were used to define criticality index of an activity in the schedule. With the help of design of experiments, set of rules were established for the calculation of C.I. Based on three fuzzy levels of P.I and F.I, there can be 3 x 3 possible combinations of rules.

If we include the two possible options of AND-OR operator, then a total of 18 possible combinations can be drawn between the two inputs. Output is in the form of C.I which can also take 3 possible states. This concludes our possible set of rules to a total of 18 x 3. In the first set of experiments, purpose was to find most relevant rules that may have an impact on the completion time of any project instance by largely affecting criticality index of the project activities. For this purpose, J30 Instances from PSPLib [40] were selected.
This set of instances contain 480 project instances which differ in their order strength (OS), resource constraints (RC) and resource factor (RF). For the purpose of conducting insightful experiments, we tested all 54 possible combination of fuzzy rules on selected instances. Instances were selected to include all possible combinations of OS, RC and RF. This was done to make sure that reliable results can be obtained with these set of experiments.

A total of nine knowledge based rules were finalized with the help of these design of experiments. The FIS uses fuzzy variables along with these IF-THEN rules to map output in the form of C.I of each activity.

Each rule is applied and tested with inputs obtained from the previous step and inferred accordingly. The rules are shown in Figure 3 and use AND/OR operators to combine F.I and P.I. If OR is used, then maximum value between F.I and P.I of an activity is chosen for the output, otherwise, minimum. For instance, a rule in the table below can be interpreted as:

**IF** F.I is high (i.e. column1) **OR** P.I is high (i.e. row1),

**THEN** C.I is medium for that activity.

The colours indicate different levels of each variable (i.e. red for high, green for medium and yellow for low). As mentioned earlier, these two input activity characteristics (i.e. F.I and P.I) combine to provide a final C.I value for each activity. The benefit of this method is the fact that each activity is not categorized strictly as critical or non-critical but degree of the criticality is defined for each individual activity. This helps in creating a more robust schedule.

The inputs are mapped on fuzzy C.I variable, a process known as implication. Each implication cuts the C.I membership graph (defined with same equidistant partition) at a certain level (decided by the AND/OR operator) to an area. In the process of implication, the edges of
FIGURE 3. Fuzzy Inference rules.

If F.I is low (i.e., column 1) or P.I is high (i.e., row 1), then C.I is medium for that activity.

FIGURE 4. Example for implication of a rule.

Application

If F.I is low, then P.I is high,

\[ \mu_{F.I} = \begin{cases} 0.62 & \text{if } F.I \text{ is low} \\ 0 & \text{otherwise} \end{cases} \]

Implication

The area obtained from step III is used to define a unique and fixed C.I value for each task. Centroid method shown in equation 3 is applied for this purpose which is a standard method.

For instance, after the application of this rule, activity A is assigned an C.I of 54.5.

\[
\text{Act.CI (task } i) = \frac{\sum_{i=1}^{n} \mu_{C.I_i} \cdot C.I_i}{\sum_{i=1}^{n} \mu_{C.I_i}}
\]  

(3)

All activities are assigned a C.I with same procedure. In the next section, methodology for scheduling will be explained.

III. FRAMEWORK

Multimodal resource constrained project scheduling problem (MMRCPSP) has been adopted in the present study. This extension of RCPSP allows an activity to be performed in multiple ways [41] which is more practical to simulate industrial environment. Well-known objective functions for this problem class are defined to evaluate performance of schedules. Equation 4 states a regular performance function in terms of time units. Project makespan or completion time of an individual scheduled project is measured which is then compared with project completion time of the baseline schedule. It should be noted here that completion time or makespan of a baseline schedule is taken as a lower bound (LB) for such comparisons. It is because baseline schedules are the ones with no resource constraints and provide minimal project...
completion time.

\[
\text{Minimize : } (MS_i - MS_{LB}) \quad (4)
\]

where,

\[ j = 1, \ldots, N, N \text{ is the total number of activities in a project.} \]

\[ MS_i = \text{Makespan of schedule } i. \]

\[ MS_{LB} = \text{Makespan of the baseline schedule defining lower bound for the project.} \]

The C.I is used as a priority heuristic for scheduling MMRCPSP [42]. To resolve resource conflicts among multiple activities, C.I is used as a ranking parameter and the activity with highest C.I is scheduled first (HCl rule) at any given time during simulation. This priority heuristic was developed on the knowledge that activities with a higher critical index are the ones with less float and/or high position index in the network. Thus, they have less inherent buffer (due to less float) and less time (due to high P.I) to respond well in time and accommodate uncertainties and should be scheduled first. The proposed framework for the approach was adopted from a previous study and is shown in Figure 6 [43].

Input is given in the form of activity list, resource requirements and precedence constraints. Scheduling is performed using priority heuristics (e.g. FIFO, SPT, HC.I) to assign limited resources in the project. It should be mentioned here that besides resource constraints, no time and budget constraints are used in the present study. A schedule provides information about the sequence of activities in the project along with their resource information. Since it is a multi-mode problem class, mode selection for every task is also done during the simulation run. As depicted in Figure 6, the information from a schedule is used by the evaluator to calculate F.I and P.I of each activity by using equations 1 and 2. Afterwards, C.I for each task in the project is calculated as described in section 2.

It should be noted here that the first simulation run always provides a baseline schedule with infinite resources [44]. Information from this schedule is used in the next simulation runs for priority heuristic scheduling with resource feasible options. Stagnant point is used a stopping criteria in this study. This means that in case of no change in the objective function, the algorithm stops.

**IV. EXPERIMENT DESIGN AND RESULTS**

The experiments were conducted with Python 2.7.10 using Intel core i7 CPU and 8 GB RAM for simulations. We chose well-known benchmark instances used to compare performance of scheduling alternatives. These instances are a part of PSPLib (Project scheduling problems) and MMLib (Multimodal) [40]. Both libraries were created with a different project generator and have multiple datasets where each data set is differentiated with total number of activities.

PSPLib has multiple datasets (e.g. J10, J12 etc) which basically differ in the number of jobs/activities in that particular data set. For instance, a dataset J30 in PSPLib contains projects with 30 number of activities and each activity can be performed with 3 modes. Each dataset has 480-600 number of project instances/networks which differ in network topology and resource parameters defined by various factors such as order strength (OS) and resource factor (RF) [45]. OS varies between 0 and 1 where 1 being a project with a complete series network (i.e. high number of precedence constraints) such as assembly lines. In contrast, 0 is assigned to a network having all parallel activities. RF depicts the tightness of resource availability in any project. In current experiments, we use project instances and record average values of objective function described in equation 4 for each dataset. For our first set of experiments, the proposed priority heuristic was compared with several other on time criterion as defined in equation 4 and are shown in Table 2. It can be seen that the proposed heuristic performed well especially for larger datasets (i.e. a data set with more number of activities). The values shown in table 2 are percentage average deviation measured from the lower bound defined by baseline schedule. Baseline schedules provide the lower bound (LB) which is used as s standard criteria in scheduling for benchmarking. It was noticed that as compared to 48.91%, 76% times optimal results (i.e. showing zero deviation from the baseline schedule) were successfully found for J30 instances [46] with our heuristic. Moreover, it is worth mentioning that as compared to the results obtained in 5000 schedules, the current heuristic was able to provide improved results within schedule range of 500-1000.

We have discussed the performance of proposed fuzzy heuristic on regular performance measure that is makespan. However, several studies suggest that such regular performance measures are limited in their capacity to define a robust schedule [19], [33], [47]. Non-traditional performance measures are suggested in literature to determine robustness of a schedule. Two non-regular performance measures are defined here to determine responsiveness and robustness of any schedule. Equation 5 depicts average next float of a
TABLE 2. Average deviation (measured in percentage) from lower bound (LB).

| Problem Set | Present Study | Bilolikar et al. [46] | Jozezowas et al. [8] | Mendes (2009)[48] | Alcaraz et al. [7] | Lova et al., (2009)[49] | Van Peteghem & Vanhoucke, (2011)[51] |
|-------------|---------------|-----------------------|----------------------|-------------------|------------------|-------------------------|--------------------------------------|
| J10         | 0.06          | 0.16                  | 0.99                 | 0.06              | 0.24             | 0.04                    | 0.0                                  |
| J14         | 0.32          | 0.63                  | -                    | 0.44              | 1.00             | -                       | -                                    |
| J20         | 0.89          | 1.04                  | 6.65                 | 1.21              | 1.96             | 0.89                    | 0.32                                |
| J30         | 10.41         | 11.86                 | 24.44                | -                 | 24.83            | 14.58                   | 13.66                                |

FIGURE 7. Performance comparison of heuristics.

FIGURE 8. Performance comparison of heuristics.

schedule which is considered a good measure for robust schedules [24].

Maximize: \[ \sum_{j=1}^{N} NF_j \]  \( N \) is the total number of activities in a project \( NF_j \) = Next float of activity \( j \)

It can be seen from Figure 7 that proposed heuristic outperforms others in comparison. The proposed heuristic is able to provide better flexibility for schedule recovery in case of any disruptions such as machine breakdowns and errors.

Moreover, in a recent study by Zahid et al. [29], it was concluded that average float index of a schedule can be used to define schedule robustness. The proposed index was concluded to have a higher correlation with schedule tardiness than typical float parameters and thus, making it more suitable to determine schedule robustness. We performed experiments while taking average float index of a schedule (equation 6) as performance measure.

Maximize: \[ \frac{\sum_{j=1}^{N} F.I_j}{N} \]  \( N \) is the total number of activities in a project \( F.I_j \) = Float index of activity \( j \)

It can be seen in Figure 8 that recently proposed heuristic performed best in case of providing highest F.I value. All these experiments with regular and non-traditional performance measures indicate promising robust schedules.

V. CONCLUSION

Scheduling is a major concern and has been established as a NP-Hard problem in literature. Trends in the past two decades have been seen to address this issue with non-traditional techniques. Various intelligent heuristics have been proposed and compared with benchmark instance sets. However, it has been established that no optimization algorithm can be guaranteed to provide optimal solution for every problem. Well-known objective functions include time and cost parameters. As optimal solutions are searched with certain information on project activities and resources, there has been an increasing debate on the applicability of these solutions. Several inside and outside factors can affect the project input parameters at a facility. Moreover, the world has changed after the COVID-19 pandemic and no one can deny the importance of responsiveness and resilience in logistics, supply chain and operations. The current research proposes an intelligent fuzzy heuristic to provide a schedule that is robust. In contrast with previous strategies, the present research was able to provide a criterion that can realistically suggest critical nature of activities based on float and position parameter. Based on these results, heuristics are used to determine schedule that can absorb uncertainties and is thus, robust. Evaluation of proposed heuristic on benchmark libraries provided improved results. Optimal solutions were achieved 76% of the times while 1.5%-2.5% improvement was seen in the regular makespan criterion. In addition to the traditional objective function of
time, the proposed heuristic was compared on non-traditional performance measures. It was observed that the heuristic performed well resulting in the provision of more robust schedules. Further investigations should be carried to use this information of criticality of activities for intelligent buffer management. The parameters can be used to accurately identify probabilistic durations of activities which are all usually assigned normal or beta curves in stochastic scheduling. Such scheduling techniques adopted from PERT do not consider individual activity’s characteristics such as difficulty level, resource competitiveness, position etc. and assign same distribution to every activity. The fuzzy criterion proposed in the current work can be further explored to resolve this problem for stochastic scheduling.

AUTHOR CONTRIBUTIONS
Taiba Zahid conducted the study and wrote the first draft of the manuscript. Mujtaba Hassan Agha designed the methodology. Salman Sagheer Warsi conducted the analysis and participated in the revision. Usman Ghafour supervises the process from the beginning and revised the manuscript. All authors have approved the final manuscript.

NOMENCLATURE
CI Critical Index.
FI Float Index.
FIFO First-In-First-Out.
HC.I Highest Critical Index.
LB Lower Bound.
MS Makespan.
NF Next Float.
OS Order Strength.
PI Position Index.
RC Resource Constrained.
RF Resource Factor.
RCPSP Resource Scheduling Problem.
SPT Shortest Processing Time.

REFERENCES
[1] R. W. Conway, W. L. Maxwell, and L. W. Miller, Theory of Scheduling, Mineola, NY, USA: Dover, 2003.
[2] P. Brucker, Scheduling Algorithms. Berlin, Germany: Springer-Verlag, 2004, ch. 1.
[3] C. Artigues, S. Demassay, and E. Neron, Resource-Constrained Project Scheduling: Models, Algorithms, Extensions and Applications. Hoboken, NJ, USA: Wiley, 2008, ch. 1, p. 21.
[4] J. Blazewicz, J. K. Lenstra, and A. H. G. Rinnooy Kan, Scheduling subject to resource constraints: Classification and complexity, Discrete Appl. Math., vol. 5, no. 1, pp. 11–24, Jan. 1983.
[5] C. Cotta, “Artificial intelligence and knowledge engineering applications: A bioinspired approach,” in Proceedings of the IV Internat. Conf. on Artificial Life. Cambridge, MA, USA: MIT Press, 1995, pp. 39–45.
[6] D. Schatteman, W. Herroelen, S. Van de Vonder, and A. Boone, “Methodology for integrated risk management and proactive scheduling of construction projects,” J. Constr. Eng. Manage., vol. 134, no. 11, pp. 885–893, Nov. 2008.
[7] M. A. Al-Fawzan and M. Haouari, “Dynamic resource constrained project scheduling: An experimental analysis,” Eur. J. Oper. Res., vol. 189, no. 3, pp. 723–733, 2008.
[8] M. Van de Vonder, E. Demeneulemeester, and W. Herroelen, “Proactive heuristic procedures for robust project scheduling: An experimental analysis,” Eur. J. Oper. Res., vol. 189, no. 3, pp. 723–733, Feb. 2008.
[9] M. H. Al-Shabibi, “A bi-objective model for robust resource constrained project scheduling,” Int. J. Prod. Econ., vol. 96, no. 2, pp. 175–178, 2005.
[10] M. A. Al-Fawzan and M. Haouari, “Dynamic resource constrained project scheduling: An experimental analysis,” Eur. J. Oper. Res., vol. 189, no. 3, pp. 723–733, Feb. 2008.
[11] M. Vanhoucke, “On the dynamic use of project performance and schedule risk information during project tracking,” Omega, vol. 39, no. 4, pp. 416–426, Aug. 2011.
[12] M. König, “Generation of robust construction schedules using evolutionary strategies,” in Proc. Eur. Group Intell. Comput. Eng., 2011, pp. 1–8.
[13] E. H. Kutanoglu and S. D. Wu (1998), Improving Schedule Robustness Via Stochastic Analysis and Dynamic Adaptation, Available: http://www.lehigh.edu/inime/documents/improving_schedule98T_001.pdf
[14] M. A. Al-Fawzan and M. Haouari, “A bi-objective model for robust resource constrained project scheduling,” Int. J. Prod. Econ., vol. 96, no. 2, pp. 175–178, 2005.
[15] M. Vanhoucke, “On the dynamic use of project performance and schedule risk information during project tracking,” Omega, vol. 39, no. 4, pp. 416–426, Aug. 2011.
[16] M. König, “Generation of robust construction schedules using evolutionary strategies,” in Proc. Eur. Group Intell. Comput. Eng., 2011, pp. 1–8.
[17] E. H. Kutanoglu and S. D. Wu (1998), Improving Schedule Robustness Via Stochastic Analysis and Dynamic Adaptation, [Online]. Available: http://www.lehigh.edu/inime/documents/improving_schedule98T_001.pdf
[18] M. A. Al-Fawzan and M. Haouari, “A bi-objective model for robust resource constrained project scheduling,” Int. J. Prod. Econ., vol. 96, no. 2, pp. 175–178, 2005.
[19] M. Vanhoucke, “On the dynamic use of project performance and schedule risk information during project tracking,” Omega, vol. 39, no. 4, pp. 416–426, Aug. 2011.
[20] M. König, “Generation of robust construction schedules using evolutionary strategies,” in Proc. Eur. Group Intell. Comput. Eng., 2011, pp. 1–8.
[21] E. H. Kutanoglu and S. D. Wu (1998), Improving Schedule Robustness Via Stochastic Analysis and Dynamic Adaptation, Available: http://www.lehigh.edu/inime/documents/improving_schedule98T_001.pdf
[22] M. A. Al-Fawzan and M. Haouari, “A bi-objective model for robust resource constrained project scheduling,” Int. J. Prod. Econ., vol. 96, no. 2, pp. 175–178, 2005.
[23] M. Vanhoucke, “On the dynamic use of project performance and schedule risk information during project tracking,” Omega, vol. 39, no. 4, pp. 416–426, Aug. 2011.
