Research Square

Preprints are preliminary reports that have not undergone peer review. They should not be considered conclusive, used to inform clinical practice, or referenced by the media as validated information.

A Resource Flow Based Branch-and-bound Algorithm to Solve Fuzzy Stochastic Resource-constrained Project Scheduling Problem

Yaghoub Alipouri (y.alipouri@kntu.ac.ir)

K N Toosi University of Technology Faculty of Civil Engineering https://orcid.org/0000-0002-4934-5435

Research Article

Keywords: Fuzzy stochastic durations, resource-constrained project scheduling, resource flow, branch-and-bound

Posted Date: July 22nd, 2021

DOI: https://doi.org/10.21203/rs.3.rs-722470/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License

A Resource Flow Based Branch-and-bound Algorithm to Solve Fuzzy Stochastic Resource-constrained Project Scheduling Problem

Yaghoub Alipouri[a,*]

[a] Assistant Professor, Department of Civil Engineering, K.N. Toosi University of Technology, Tehran, Iran, email: y.alipouri@kntu.ac.ir.

Corresponding author: Yaghoub Alipouri

Tel: +98-2188201431.

Full postal address: Civil Engineering Faculty, K.N.Toosi University of Technology, No. 1346, Vali Asr Street, Mirdamad Intersection, Tehran, Iran, **Postal code:** 1996715433

ORCID ID: https://orcid.org/0000-0002-4934-5435

Abstract:

In this paper a resource flow based branch-and-bound procedure is designed to solve the well-known resource constrained project scheduling problem under the mixed uncertainty of fuzziness and randomness (FS-RCPSP). The objective is to minimize the expected makespan of the project subject to precedence and resource constraints. The proposed branch-and-bound can be employed to obtain optimal solutions and also can be truncated in order to find promising near optimal solutions. The depth-first strategy is utilized for constructing the search tree and earliest start time (EST) concept is adopted for selecting a node for further branching while traversing the tree down to the leaves. The performance of developed branch-and-bound is benchmarked against CPLEX and SADESP across an extensive set of 960 problems. The results returned by the proposed algorithm show experimentally its effectiveness to solve the FS-RCPSP.

Keyworks: Fuzzy stochastic durations; resource-constrained project scheduling, resource flow, branch-and-bound.

1. Introduction

One of the important issues when planning a project is the scheduling of some identified activities. Our work is concerned with a well-known problem in the area of scheduling, named Resource-Constrained Project Scheduling Problem (RCPSP), under the uncertain environment. The objective of RCPSP is to minimize the project makespan subject to precedence and resource constraints. Each activity needs a fixed amount of renewable resources in its processing time. The availability of resources and the activity durations are assumed to be known and fixed and the preemption is not allowed for the activities. A real-life example to this problem is a construction project where there are precedence relations between the activities and the availability of some resources, for example human resources, in some time periods are limited. Many attempts have been made to propose exact, heuristic, and meta-heuristic procedures to solve the RCPS problem with deterministic parameters. Examples of such research efforts are Brucker et al. (1998), Damay et al. (2007), Pantouvakis and Manoliadis (2006), Ying et al. (2009), Mahdi Mobini (2009), Zamani (2011), Sebt et al. (2013), Kumar and Vidyarthi (2016), and Sheng et al. (2019).

In most of the real world situations, since the scheduling parameters of a project cannot be exactly estimated, it would be more appropriate if in addition to precedence and resource constraints, uncertainties are also taken into account in the scheduling process. The majority of research efforts to cope with the uncertain information in the project scheduling problems introduce reactive, proactive (robust), stochastic, or fuzzy scheduling approaches (Herroelen and Leus, 2005). In the reactive scheduling, the uncertainty is not considered directly at the time of generating a schedule, but the baseline schedule is repaired or rescheduled when disruptions occur. However, in proactive scheduling, a baseline schedule protected as much as possible against disruptions is generated. Some recently published papers in reactive and/or proactive project scheduling are Sadeh et al. (1993), Van de Vonder et al. (2007), Lambrechts et al. (2008), Leus and Herroelen (2004), Artigues et al. (2013), and Bruni et al. (2017). In contrast to the proactive scheduling, in the stochastic scheduling no baseline schedule is produced and all the scheduling decisions are usually made based on a dynamic decision process, called *policy*, at the time of execution. In the stochastic project scheduling, it is assumed that activities have been performed several times in the previous projects, and therefore there is enough historical data to

provide a probability distribution for each activity. However, in fuzzy project scheduling, due to the lack of historical data, activity durations are estimated by some experts. Since these estimations are vague and imprecise, activity durations are represented by fuzzy numbers. Some papers being published in the area of Stochastic-RCPSP (S-RCPSP) or Fuzzy-RCPSP (F-RCPSP) are as follows: Ashtiani et al. (2011), Fang et al. (2015), Rostami et al. (2018), Atli and Kahraman (2012), and finally Atli and Kahraman (2013), Zoulfaghari et al. (2016). In real world projects, especially in construction projects, it is possible to have both kinds of uncertainties (i.e., randomness and fuzziness) in durations of activities. In these projects, the duration of some activities (activities with enough historical data about the duration) can be modelled as a probability distribution and that of others (activities without enough historical data about the duration) can only be estimated imprecisely by experts. The corresponding scheduling problem for these projects is called fuzzy stochastic project scheduling problem and its resource constrained version (FS-RCPSP) is addressed in the current paper.

Some researchers have coped with the situations where randomness and fuzziness appears simultaneously and have proposed mathematical tools to model this kind of uncertainty. Some of these tools are as follows: probability of a fuzzy event (Zadeh, 1968), probabilistic set (Hirota, 1981), random fuzzy variable (Liu, 2002a; Liu, 2002b), and fuzzy random variable (Puri and Ralescu, 1986; Kruse and Meyer, 1987; Buckley, 2005). Note that the mathematical tools 'random fuzzy variable' and 'fuzzy random variable' are different from each other. Roughly speaking, random fuzzy variables are fuzzy variables taking 'random variable' values and fuzzy random variables are random variables taking 'fuzzy variable' values (Zhu and Liu, 2005). An early application of the concept of a fuzzy random variable was in scheduling. Itoh and Ishii (2005) assumed that uncertainty only exists in activity due-dates and employed fuzzy random variables for modeling these uncertainties in an n-job machine. A fuzzy stochastic project scheduling problem for the first time was addressed by Ke and Liu (2007). Using random fuzzy variables for modeling of uncertainties, they proposed three models for solving the problem under study: expected cost minimization model, (α,β)-cost minimization model, and chance maximization model. They also proposed a hybrid intelligent algorithm for solving these models. Huang et al. (2009) proposed an expected cost model with random fuzzy variables to handle the software project scheduling problem and solved this model using an intelligent genetic algorithm. Nematian et al. (2010) were the first researchers to tackle the Fuzzy Stochastic-

RCPSP (FS-RCPSP) by considering the ready time, duration, and deadline of activities to be fuzzy random variables and converting the resulted model with fuzzy random variables into a mixed-integer linear programming model. Xu and Zhang (2012) studied a more complicated problem, namely resource constrained multiple project scheduling problem when randomness and fuzziness co-exists. The problem was formulated as a multi-objective model with fuzzy random variables for minimizing the total project time and the total tardiness penalty of all projects. This model was solved by a hybrid genetic algorithm with a fuzzy logic controller. Chen and Zhang (2016) presented a proactive strategy for Preemption-RCPSP with fuzzy random duration and resource availabilities. They proposed a mathematical formulation model for this problem with the objective function of maximizing the sum of free slack and then transformed it into an equivalent crisp model. Finally, they demonstrated the efficiency of the model and proactive strategy just by a numerical example. Alipouri et al. (2017) addressed the RCPSP under the fuzzy random environment using the fuzzy probability theory of Buckley (2005). They proposed a mathematical programming model with fuzzy random variables exploiting the concept of the resource flow network. Then, this model was transformed into an MILP model with crisp parameters and variables. They implemented this MILP model in AIMMS (2014) modeling software and solved 960 benchmark problems using the CPLEX 12.6.0.1 solver of this software. Their results showed that the CPLEX solver in AIMMS could work well on most of the benchmark problems; however, this solver could not find any integer solution for about 10% of the problems with 30 activities and 31% of the problems with 60 activities. In addition, the computational efforts required for solving some of the benchmark problems were high. All these drawbacks can prevent practitioners to use CPLEX solver in real world applications for solving FS-RCPSP. More recently, Alipouri et al. (2019) proposed a self-adaptive differential evolution hyper-heuristic algorithm, named SADESP, to solve the same problem and model tackled by Alipouri et al. (2017). They also solved the same 960 benchmark problems as Alipouri et al. (2017) and reported the results for 500, 5000, 50000 and 100000 computed schedules by the algorithm. Unlike the CPLEX, SADESP was able to find integer solutions for all 960 instances, such that with 100000 computed schedules, SADESP worked well (i.e., found better, equal or close objective values) than CPLEX on about 96.88% (i.e., 465 out of 480) of the J30 problems and 95.21% (i.e., 457 out of 480) of the J60 problems.

All the above mentioned drawbacks of CPLEX in solving the MILP model of Alipouri et al. (2017) motivated us to introduce a new exact search methodology based on branch-and-bound (B&B) algorithm which can act better than CPLEX and also SADESP in driving policies for FS-RCPSP. Due to the following two reaons branch-and-bound was chosen for the current research: 1- FS-RCPSP belongs to the set of complex combinatorial problems and very often branch-and-bound is the only exact technique which can optimaly solve these problems within an acceptable computational effort (Demeulemeester and Herroelen, 2002), and 2- Branch-and-bound can be truncated and used as a heauristic to find a near optimal solution within a reasonable time for the large-scale problems having many activities. Branch-and-bound has been used by many resaerchers to solve RCPSP and its variants. For example, Brucker et al. (1998) and Dorndorf et al. (2000) solves RCPSP with a B&B, Davari and Demeulemeester (2018) proposes a B&B for the chance-constrained RCPSP, Kolisch et al. (1995), Sprecher et al. (1997), Sprecher and Drexl (1998), and Altintas and Azizoglu (2020) present a B&B for a multi-mode project scheduling sroblem with a single non-renewable resource, and so on. To the best of our knowledge, there is not any research to propose a B&B method for FS-RCPSP.

Author of the current study believes the contribution of this paper is threefold. First, the problem is modelled as a fuzzy stochastic version of RCPSP (FS-RCPSP) handling the two key sources of uncertainty in a unified way. Second, to the best of the author's knowledge, it is the first effort to propose a B&B algorithm to solve project scheduling problems in an uncertain environment. Third, to the best of the author's knowledge, this is the first time in the literature that the resource flow concept is employed in a B&B procedure for solving a variant of RCPSP.

The remainder of this paper is organized as follows. In the next section, some preliminaries on resource flow and fuzzy probability theories are presented. In section 3, the FS-RCPSP problem description and its model formulations are provided. In section 4, our proposed resource flow based branch-and-bound algorithm for solving FS-RCPSP is presented. In section 5, the results of computational experiments to test the potency of our method in solving the FS-RCPSP are reported. Finally, in section 6, concluding remarks are drawn out and future works are presented.

2. Basic Concepts and Definitions

In this section, some general information is given on resource flow theory. Then, we review some key concepts in Buckley's (2005) approach to modelling uncertainty using fuzzy

probability theory and fuzzy random variables. Hereafter, we place a "bar" and a "tilde" over a letter to denote a fuzzy number and a fuzzy random number, respectively.

2.1. Resource flow theory

In any feasible schedule, upon the completion of an activity *i* its resources must be allocated to other activity/activities. The key idea of resource flow theory is to determine the number of resource units of type *l* that are transferred from the end of one activity to the beginning of another activity. Based on these resource links between the activities a network, named resource flow network, can be constructed for each resource type. Therefore, this network depicts how units of a resource type are passed on between the activities in the schedule. In RCPSP, the problem of resource conflicts existing among the activities of the project can be solved by introducing appropriate resource flow networks.

2.2. Fuzzy probability theory

A discrete probability function P on all subsets of a finite set $U = \{u_1, ..., u_\tau\}$ is defined as $P(\{u_\psi\}) = a_\psi$, $0 < a_\psi < 1$ all ψ and $\sum_{\psi=1}^{\tau} a_\psi = 1$. This probability function is able to handle the uncertainties arising from natural variability (randomness) but cannot deal with the uncertainties due to imprecise information (fuzziness). In order to handle both kinds of uncertainty, Buckley (2005) substitutes a fuzzy number $\bar{a}_\psi = \bar{P}(\{x_\psi\})$ for each crisp number a_ψ and assumes that $0 < \bar{a}_\psi < 1$ for all ψ. The discrete set U together with the $\bar{a}_\psi$ values is a discrete fuzzy probability distribution (Buckley, 2005). Buckley (2005) considers the restriction "there are $a_\psi \in \bar{a}_{\psi\alpha}, \alpha = 1$ so that $\sum_{\psi=1}^{\tau} a_\psi = 1$" on the $\bar{a}_\psi$ values and accordingly defines the fuzzy probability of $\Gamma = \{u_1, .., u_k\}$, $1 \le k < \tau$, and the expected value of a discrete fuzzy probability distribution by their α-cut as follows:

$$\bar{P}_\alpha(\Gamma) = \{\sum_{\psi=1}^{k} a_\psi \,|\, a_\psi \in \bar{a}_{\psi\alpha}, 1 \le \psi \le \tau, \sum_{\psi=1}^{\tau} a_\psi = 1\}, \text{ for } 0 \le \alpha \le 1, \tag{1}$$

$$\bar{\mu}_\alpha = \{\sum_{\psi=1}^{\tau} x_\psi a_\psi \,|\, a_\psi \in \bar{a}_{\psi\alpha}, 1 \le \psi \le \tau, \sum_{\psi=1}^{\tau} a_\psi = 1\}, \text{ for } 0 \le \alpha \le 1. \tag{2}$$

Let $\overset{\circ}{U}$ be a fuzzy random variable and let $f(u,\bar{\theta})$ be a fuzzy probability density, where $u \in IR$ and $\bar{\theta} = \{\bar{\theta}_1,...,\bar{\theta}_Q\}$ is for parameters $\bar{\theta}_q$, $1 \le q \le Q$. The fuzzy probability of $\overset{\circ}{U} \in [z_1, z_2]$ and the expected value of fuzzy probability density $f(u,\bar{\theta})$ are defined by their α-cut as follows:

$$\bar{P}_\alpha(\overset{\circ}{U} \in [z_1, z_2]) = \{\int_{z_1}^{z_2} f(u;\theta)du \,\Big|\, \theta_q \in \bar{\theta}_{q\alpha}, 1 \le q \le Q, \int_{-\infty}^{+\infty} f(u;\theta)du = 1\}, \text{ for } 0 \le \alpha \le 1. \quad (3)$$

$$\bar{\mu}_\alpha = \{\int_{-\infty}^{+\infty} uf(u;\theta)du \,\Big|\, \theta_q \in \bar{\theta}_{q\alpha}, 1 \le q \le Q, \int_{-\infty}^{+\infty} f(u;\theta)du = 1\}, \text{ for } 0 \le \alpha \le 1, \quad (4)$$

In this research, the same as Alipouri et al. (2017, 2019), fuzzy random variables with fuzzy normal density functions are employed to represent the durations of activities. Moreover, all fuzzy parameters are represented by triangular fuzzy numbers (*TFN*s). A fuzzy normal density is shown as $N(\bar{\mu},\bar{\sigma}^2)$, wherein only the values of μ and σ^2 have become fuzzy compared to the crisp normal density $N(\mu,\sigma^2)$. Buckley (2005) provides the result that the fuzzy mean of $N(\bar{\mu},\bar{\sigma}^2)$ equals to $\bar{\mu}$ and its fuzzy variance is $\bar{\sigma}^2$.

3. Problem Description and Formulation

The problem under study is FS-RCPSP, which is a variant of RCPSP considered in an uncertain environment in which randomness and fuzziness exist simultaneously. Here, we just consider the uncertainties existing in durations of activities, owing to the fact that extending the proposed algorithm to another algorithm which considers the uncertainties in other parameters is straightforward.

In FS-RCPSP a single project consisting of n+2 activities is considered. The activities are numbered 0 to $n+1$, where 0th and $(n+1)^{\text{th}}$ activities are start and end dummy activities, respectively. Each activity j cannot be interrupted once in progress (i.e., preemption is not allowed), and has to be started after all its immediate predecessor activities i ($i \in IP_j$) have finished (i.e., precedence constraint). We have L renewable resources (e.g., equipment and human resources), and each resource $l \in L$ has a limited capacity R_l ($1 \le l \le L$) throughout the project duration. Each activity j requires r_{jl} ($0 \le j \le n+1$, $1 \le l \le L$) units of resource l once in progress. The sum of resource requirements for resource l at any time period t cannot exceed R_l (i.e., resource constraint). The processing time of activity j is denoted as $\tilde{d}_j$ ($0 \le j \le n+1$)

which is a fuzzy random variable. The start and finish time of each activity j are respectively shown by $\tilde{s}_j$ and $\tilde{f}_j$ ($0 \le j \le n+1$). The objective is to find precedence and resource feasible completion times for all activities which lead to the minimum expected makespan.

Based on the description above, FS-RCPSP can be formulated as follows:

$$Min \quad \tilde{f}_{n+1} \tag{5}$$

$$s.t.$$

$$\tilde{f}_i \tilde{\le} \tilde{f}_j - \tilde{d}_j \quad for \quad all(i,j) \in E \tag{6}$$

$$\tilde{s}_0 = \tilde{f}_0 = 0 \tag{7}$$

$$\sum_{i \in B(t)} r_{il} \le R_l \quad l = 1,...,L, \quad B(t) = \{i \mid \tilde{f}_i - \tilde{d}_i \le t < \tilde{f}_i\}, \quad t = 1,...,T \tag{8}$$

$$\tilde{f}_i \ge 0 \quad for \quad i = 0,...,n+1 \tag{9}$$

where, E is the set of all precedence relations and $B(t)$ is the set of activities being processed at the time period t. In this formulation, constraints (6) and (8) are respectively used to impose precedence and resource constraints. Due to the lack of a suitable approach for transferring set $B(t)$ to a linear constraint and also due to the presence of fuzzy random parameters and variables, the above mathematical programming model cannot be solved directly. Therefore, Alipouri et al. (2017) made use of assumptions and concepts introduced by Artigues et al. (2003) and introduced a resource flow network model for FS-RCPSP. Based on the fuzzy probability theory of Buckley (2005), they represented the durations of activities by fuzzy random variables with normal probability density $N(\bar{\mu}, \bar{\sigma}^2)$, where $\bar{\mu}$ and $\bar{\sigma}^2$ are triangular fuzzy numbers. Then, adopting the concept of expected value of a fuzzy random variable, they transferred their model with fuzzy random variables to the following mixed integer linear programming (MILP) model with deterministic parameters and variables (Alipouri et al., 2017).

$$Min \quad Z = \omega_1 (f_{n+1,1} - f_{n+1,2}) + \omega_2 (f_{n+1,2}) + \omega_3 (f_{n+1,3} - f_{n+1,2}) \tag{10}$$

$$s.t.$$

$$x_{ij} = 1 \quad \forall (i,j) \in E \tag{11}$$

$$x_{ij} + x_{ji} \le 1 \quad \forall (i,j) \in V^2, i \ne j \tag{12}$$

$$x_{ik} \ge x_{ij} + x_{jk} - 1 \quad \forall (i,j,k) \in V^3, i \ne j, j \ne k, i \ne k \tag{13}$$

$$f_{i,1} \le f_{j,1} - x_{ij}(d_{j,1} + M) + M \qquad \forall (i,j) \in V^2, i \ne j \tag{14}$$

$$f_{i,2} \le f_{j,2} - x_{ij}(d_{j,2} + M) + M \qquad \forall (i,j) \in V^2, i \ne j \tag{15}$$

$$f_{i,3} \le f_{j,3} - x_{ij}(d_{j,3} + M) + M \qquad \forall (i,j) \in V^2, i \ne j \tag{16}$$

$$f_{n+1,1} \ge f_{i,1} \qquad \forall i \in V \tag{17}$$

$$f_{n+1,2} \ge f_{i,2} \qquad \forall i \in V \tag{18}$$

$$f_{n+1,3} \ge f_{i,3} \qquad \forall i \in V \tag{19}$$

$$\sum_{j \in V} C_{ij}^{l} = r_{il} \qquad \forall i \in V, \forall l \in L, i \ne n \tag{20}$$

$$\sum_{i \in V} C_{ij}^{l} = r_{jl} \qquad \forall j \in V, \forall l \in L, j \ne 1 \tag{21}$$

$$0 \le C_{ij}^{l} \le \min(r_{il}, r_{jl}) x_{ij} \qquad \forall (i,j) \in V^2, \forall l \in L, i \ne n, j \ne 1 \tag{22}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in V^2, i \ne j \tag{23}$$

$$f_{0,1} = f_{0,2} = f_{0,3} = 0 \tag{24}$$

where x_{ij} is a binary variable denoting that activity *j* is started immediately after the completion of activity *i* whenever $x_{ij} = 1$, otherwise $x_{ij} = 0$. C_{ij}^{l} denotes the amount of resource *l* directly transferred from activity *i* to activity *j*. *V* and *E* are the set of all activities and the set of all precedence relations, respectively. The objective function (10) minimizes the expected completion time of the end dummy activity and consequently the expected completion time of the project, $\overline{\mu}_{f_{n+1}} = (f_{n+1,1}, f_{n+1,2}, f_{n+1,3})$. To obtain this objective value, Alipouri et al. (2017) adopted the technique described in Buckley and Feuring (2000). First, the problem of minimizing the fuzzy number $\overline{\mu}_{f_{n+1}}$ was converted into a multi-objective problem; then, the multi-objective problem was converted into a single objective problem with the use of weights $\omega_i \ge 0, i = 1,2,3$, $\omega_1 + \omega_2 + \omega_3 = 1$. This is a flexible objective function for project managers since they can vary the values of weights to satisfy their different requirements. Equation (11) introduces the precedence relations between the activities. Constraint (12) is transiting constraint and constraint (13) ensures that no cycles will exist in the network. Constraints (14)-(19) are employed for setting the completion times of activities. Constraints (20)-(22) are resource flow inequalities. By constraint (22) the resource flow values are limited to $\min(r_{il}, r_{jl})$ of arc (i, j) if the arc exists. Constraints (20) and (21) have been devised to ensure that the incoming flow on

node i is equal to the outgoing flow from that node. In the next section, the proposed branch-and-bound procedure for solving this problem is presented.

4. The Proposed Branch-and-bound Algorithm

In what follows, a branch-and-bound procedure is described to solve the FS-RCPSP. This algorithm can be employed to obtain optimal solutions and also can be truncated in order to find promising near optimal solutions. In section 4.1, the branching strategy is presented. Section 4.2 is devoted to the description of five simple and effective rules for pruning the solution tree. The algorithmic description of the proposed B&B for solving FS-RCPSP is summarized in section 4.3. Finally, in section 4.4, an illustrative numerical example is given.

4-1. Branching and node selection

In this subsection, we describe how a solution tree is built by the proposed B&B. In other words, our strategy in creating new nodes of the tree and selecting a node for further branching is explained. Based on the literature, there are three main branching strategies for constructing a search tree.

- *Best-first strategy*, also known as frontier search or skiptracking strategy, which always selects the intermediate node with the lowest bound to branch from next.
- *Breadth-first strategy*, which processes all the nodes at one level of the search tree before going to a higher level.
- *Depth-first strategy*, also known as backtracking strategy, which explores a path by processing node by node down a branch until a node on the path is fathomed or a complete solution is reached. Then, if there exists at least one previously created, unfathomed, intermediate node which has not been fully explored, backtracking is accomplished.

In breadth-first strategy the number of nodes at each level grows exponentially, leading to avoid using this strategy for complex problems. Also, the same as breadth-first strategy, the best-first strategy is unable to find a feasible solution before ending the search and requires more storage capacity than the depth-first strategy. However, a depth-first strategy can find a feasible solution right away leading to have a rather good solution if the algorithm is truncated prematurely. All

the advantages of depth-first strategy against other strategies encouraged us to employ this strategy for constructing the search tree.

While traversing the tree down to the leaves, a decision has to be made at each level of the tree about selecting a node for further branching. To this end, in this study the earliest start time (EST) concept is employed as explained later in this subsection.

In section 3, it was explained that processing time of each activity *i* is represented as fuzzy normal density $N(\overline{\mu}_i, \overline{\sigma}_i^2)$ and the values of $\overline{\mu}_i$ and $\overline{\sigma}_i^2$ are assumed to be triangular fuzzy numbers. Based on the fuzzy probability theory of Buckley (2005), the expected duration of each activity *i* with fuzzy normal density $N(\overline{\mu}_i, \overline{\sigma}_i^2)$ is $\overline{\mu}_i$. Therefore, the expected makespan is also a triangular fuzzy number which can be calculated using the expected durations of activities, $\overline{\mu}_{d_i} = (d_{i,1}, d_{i,2}, d_{i,3})$. Thus, these expected durations will be the base of the calculations in the proposed B&B. Using the expected values of activity durations does not mean to ignore randomness, since these expected values are descriptive quantities calculated based on the probability distributions of fuzzy random variables. It is worthwhile noting that, while traversing on the search tree, if it is needed to compare two triangular fuzzy numbers, their corresponding crisp (*Cr*) values are compared. The corresponding crisp value of the triangular fuzzy number $\overline{B} = (b_1, b_2, b_3)$ is calculated using the following equation:

$$Cr_{\overline{B}} = \omega_1(b_1 - b_2) + \omega_2(b_2) + \omega_3(b_3 - b_2)\,. \tag{25}$$

Each node of the search tree is associated with a partial schedule in which an expected fuzzy finish time is assigned to an activity of the project. Here, based on the concept of resource flow theory, resource precedence relations are defined between the activities that resource transferring has taken place between them. These relations will lead to have feasible partial and/or complete schedules in every node of the search tree (Artigues et al., 2003). To the best of author's knowledge, this is the first time in the literature that the resource flow concept is employed in a B&B procedure for solving a variant of RCPSP.

In the first step of constructing the search tree, the start dummy activity is scheduled at the root node and its expected start and finish times are set to (0,0,0). Moreover, all the available resources are assigned to this dummy activity, and also resource requirements of the end dummy

activity are considered to be equal to the available amount of the resources. The branching process is started by adding eligible activities (i.e., unscheduled activities for which all predecessors have scheduled) of the first level, as children, to the root node, as parent. To this end, at this first level, the algorithm probes for all the activities that the start dummy activity is their only predecessor, and stores them in the eligible activities set of the first level, EAS_1. Then, one node is created for each activity in EAS_1 and this activity is scheduled to start at expected time (0,0,0). The resources required by each of these newly scheduled activities are transferred from the start dummy activity. Among all these newly added nodes, one node has to be selected to branch from it in the next level of search tree. For this selection, an expected EST value and its corresponding crisp value (see Equation (26)) are calculated for all the activities of the project. The competing activity with the lowest Cr_{EST} value (using the lower activity number as a tie breaker) is selected for further branching.

$$Cr_{EST_i} = \omega_1(EST_{i,1} - EST_{i,2}) + \omega_2(EST_{i,2}) + \omega_3(EST_{i,3} - EST_{i,2}), \tag{26}$$

where $EST_{i,1}, EST_{i,2}$, and $EST_{i,3}$ are earliest start times of activity i which are respectively calculated by applying the critical path method on the most optimistic, most likely, and most pessimistic values of the expected durations of project activities.

So far, we could explain the first iteration of the proposed B&B algorithm. From this iteration on until the end of search process the procedure is somehow different, as some strategies for avoiding the node redundancy are employed by the algorithm to reduce the search space of the problem. After selecting a node (we call this node and its corresponding activity as *SN* and *SA*, respectively) from the set EAS_m of level m to add to the tree, based on the resource flow network and precedence relations existing on the current path, activities which have been scheduled so far and are able to provide resources to the activity of the selected node are listed. The activities on this list are sorted based on the non-decreasing order of their expected finish time and from the activity that has the lowest expected finish time it is started to transfer the resources into activity *SA*. Transferring of resources is continued until this activity has collected all its required resources. The resource transferor activities which are not the predecessor of activity *SA* are considered as the resource predecessor of this activity. Then, after taking into account the pruning rules presented in the next subsection, if activity *SA* is not fathomed, this activity is

scheduled at the earliest possible expected start time equal to or greater than the expected time of the activity scheduled at the previous level.

4-2. Pruning Rules

In order to speed up the above proposed branching procedure, it is necessary to introduce some pruning rules which are used to fathom the nodes that branching from them is proved to be unnecessary. Fathoming of a node can happen if it is proved that this node contains non-optimal or infeasible solutions, or by branching from the other intermediate nodes of the search tree a similar or better solution could be obtained (Demeulemeester and Herroelen, 2002). In building our proposed search tree we relied on the following five pruning rules which are simple and effective. These pruning rules are the extended version of the pruning rules which have been proposed in literature for the precedence tree algorithm.

Rule 1. If a child node is scheduled to start in the same expected start time as its parent and it has "smaller" (arbitrarily one can use the word "larger") activity number, this child node is fathomed. This is because a same partial solution can be produced in another branch of the tree, this time by scheduling the smaller numbered activity before the one with the higher number.

Rule 2. Assume that activity $SA \in EAS_m$ is going to be scheduled on expected start time $\bar{\mu}_{s_{SN}}$ at level m of the B&B tree. If this activity can be performed at a previous level with an expected start time less than $\bar{\mu}_{s_{SN}}$ and without violating resource and precedence constraints, the corresponding node of this activity (i.e., SN) is fathomed. This is because of the existence of a node at the previous level that branching from it will result in a same schedule that can be produced by shifting the identified activity/activities.

The other three pruning rules are based on the calculation of lower bounds. The lower bounds calculated for a node are compared with an upper bound. If at least one of the lower bounds of the node exceeds or equals the current upper bound of the search tree, that node is pruned. The initial value of the upper bound is set to the sum of the expected durations of all activities of the project. Each time a complete solution with a better expected makespan is found by the B&B algorithm, the upper bound is changed to be equal to this expected makespan.

Ideally, the best lower bound value for a given node is the value of the best accessible objective value to the problem at hand from this node. However, here obtaining this value is itself NP-hard leading us to relax the problem in order to calculate the lower bounds. Therefore, in this study, after the relaxation of resource constraints, an expected distance matrix, denoted as $\overline{\mu}_{DM}$, that its elements are triangular fuzzy numbers, is calculated as explained in Remark 1 and used in the calculation of lower bounds.

Remark 1. Let $\overline{\mu}_{dm_{ij}} = (dm_{ij,1}, dm_{ij,2}, dm_{ij,3})$ be a triangular fuzzy number denoting the expected length of a longest path (expected distance) from activity *i* to activity *j*. The Floyed-Warshall algorithm, which has the time complexity of $O(n^3)$ (see Lawler (1976)), is used three times to calculate the expected distance matrix $\overline{\mu}_{DM} = (\overline{\mu}_{dm_{ij}})_{i,j \in V}$. This algorithm is applied on the most optimistic values of expected durations of activities to calculate the $dm_{ij,1}$ and on the most likely and most pessimistic values to calculate the $dm_{ij,2}$ and $dm_{ij,3}$, respectively.

Rule 3. If a child node is selected to start in the same expected start time as its parent node, a lower bound (*LB1*) for the child node can be calculated by adding the expected start time of the parent node to the remaining expected distance from the activity of the child node to the end dummy activity.

Rule 4. Let LN_m be the set of unscheduled activities having numbers lower than the activity of node *SN* of the search tree. A lower bound (*LB2*) for this node is calculated as follows.

$$LB2 = \overline{\mu}_{s_{SN}} + (1,1,1) + \max\{\overline{\mu}_{dm_{ij}} \,|\, i \in LN_m, j = n+1\}, \tag{27}$$

where $\overline{\mu}_{s_{SN}}$ denotes the expected start time of the activity of the node *SN*. This rule can be easily justified using Rule 1, where any lower numbered activity prohibited from being scheduled in an expected start time equal to or smaller than that of the currently scheduled, higher numbered activity.

Rule 5. Let *SN* be a given node of the search tree with EAS_m denoting the set of eligible activities to be scheduled after *SN*. A lower bound (*LB3*) for the node *SN* can be calculated as follows.

$$LB3 = \bar{\mu}_{s_{SN}} + \bar{\mu}_{d_{SN}} + \max\{\bar{\mu}_{dm_{ij}} \,|\, i \in EAS_m, j = n+1\}. \tag{28}$$

4-3. Algorithmic description

The algorithmic description of the proposed B&B for solving FS-RCPSP is given in this subsection. The B&B procedure consists of the following eight steps. Also, the flowchart of the proposed B&B is illustrated in Figure 1.

Step1. Read data related to the problem at hand: the number of non-dummy activities to be scheduled (n), expected processing time of each activity i (i=0, …, n+1) as triangular fuzzy number $(d_{i,1}, d_{i,2}, d_{i,3})$, precedence relations, resource requirement of each activity, resource availabilities, truncating criteria.

Step 2. Calculate the expected distance matrix $\bar{\mu}_{DM}$ using expected durations of activities and critical path method calculations as explained in Remark 1.

Step 3. Set the values of ω_1, ω_2, and ω_3 to user-defined values. Set the initial upper bound as $UB = \sum_{i=1}^{n} \bar{\mu}_{d_i}$. Set the level of the B&B tree $m = 0$. Calculate the expected EST value for each activity of the project.

Step 4. Schedule the start dummy activity to start and finish at the expected time (0,0,0) and assign all the available resources to this dummy activity. Assume that the resource requirement of dummy activity n+1 is equal to the availability of resources. Update the partial schedule PS={0}.

Step 5. Increment the level number by one and find the eligible activities of first level, EAS_1. Schedule all the activities of EAS_1 to start at expected time (0,0,0) and transfer the resources required by these activities from start dummy.

Step 6. Select the node with the lowest expected EST value (using lower activity number as a tie breaker) to branch from it at the next level. Update PS with regard to the selected activity. Set m=m+1.

Step 7. WHILE there is at least one unexplored, unfathomed node or truncating criteria are not satisfied, Do

Step 7.1. Find the eligible activities EAS_m of level m, and select the activity with the lowest expected EST value (using lower activity number as a tie breaker), denoted as SA, for further branching.

Step 7.2. Based on the resource flow and precedence relations existing on the current path, list activities being able to send resources to SA and prioritize them

considering their expected finish time such that lowest expected time results in highest priority.

Step 7.3. Starting from the activity with highest priority transfer resources to *SA* until *SA* gets all its required resources, consider the transferor activities which are not the predecessor of activity *SA* as the resource predecessor of this activity.

Step 7.4. Considering five pruning rules presented in section 4.2, if activity *SA* is not fathomed, go to step 7.5; otherwise go to step 7.6.

Step 7.5. Create a node for activity *SA* and schedule it at the earliest possible start time equal to or greater than the expected start time of the activity scheduled at the previous level. Set $m=m+1$, and set *PS*. If *SA* is the end dummy activity, update upper bound.

Step 7.6. If there exists at least one unexplored, unfathomed node, backtrack on the search tree to select another node to branch from next and then update *m* and *PS* with regard to the location of the selected node; otherwise go to step 8.

Step 8. END WHILE

It is necessary to metion that, based on Alcaraz and Maroto (2001), an optimal solution for RCPSP can only be achieved by the exact solution procedure in small projects, usually with fewer than 60 activities, which are not highly resource constrained. For projects with large-scale activities there can be two scenarios to apply the proposed B&B: 1- using more powerful computers to solve the project to optimality, and 2- truncating the algorithm based on a user-defined criteria to find a near optimal solution.

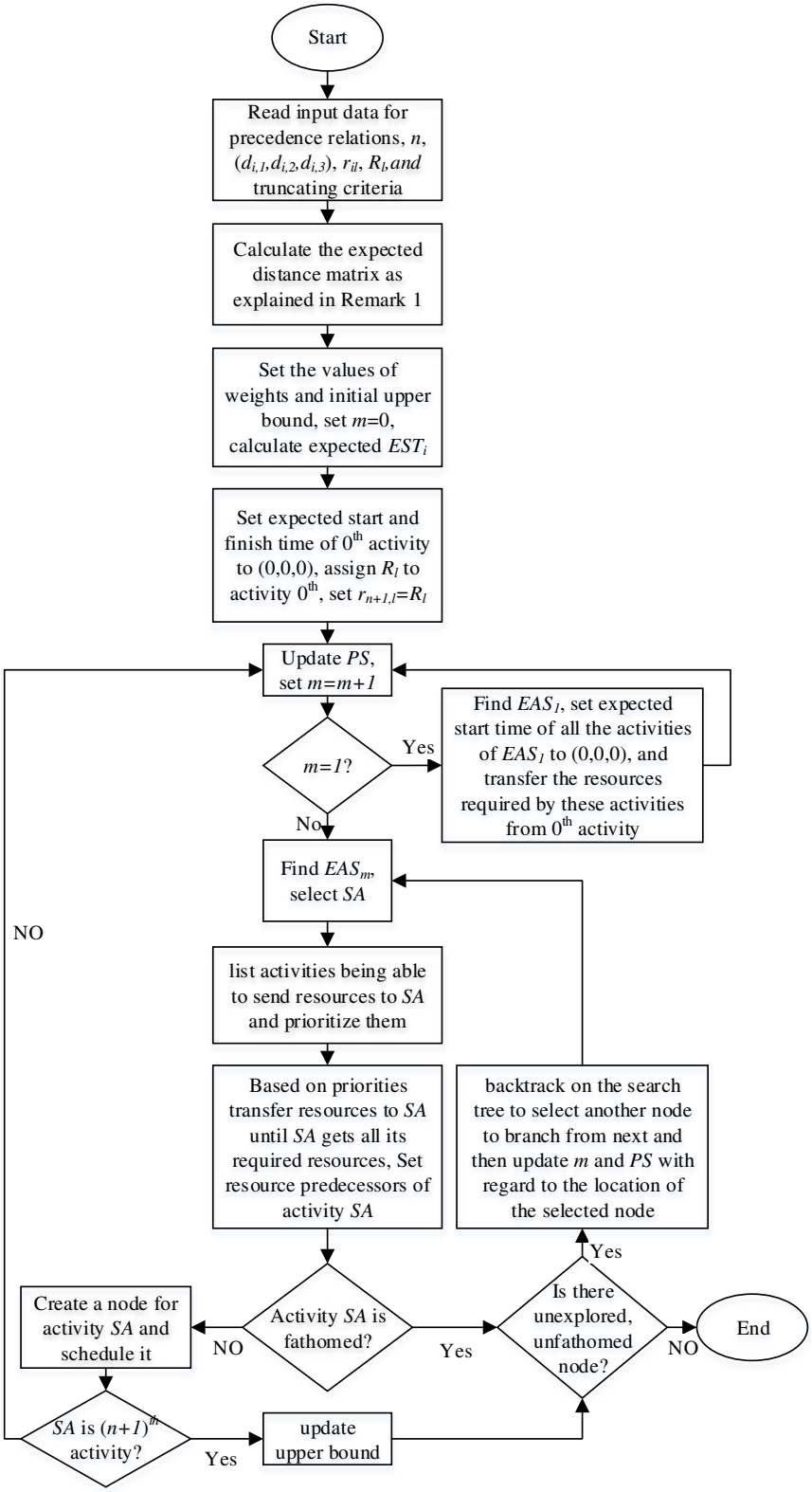

Figure 1. Flowchart of proposed branch-and-bound algorithm

4-4. Numerical example

In this section, our proposed resource flow-based B&B procedure is used to find an optimal solution for the project instance shown in Figure 2. In this activity-on-the-node network, each node is the representative of an activity and the numbers above and below each node denote respectively the expected duration and resource requirement of that activity. The resource availability for this project equals 7. Values of ω_1, ω_2, and ω_3 are set to 0.1, 0.8, and 0.1, respectively. The initial upper bound is calculated as $UB = \sum_{i=1}^{7} \bar{\mu}_{d_i} = (13,27,43)$. Figure 3 depicts how the B&B tree can be built on this instance.

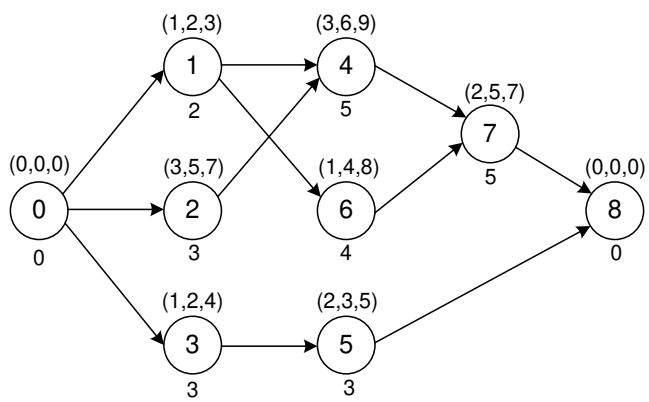

Figure 2. Activity network of the example project

In Figure 3, the first line of every node shows the node number on the left and the activity number of that node on the right. Moreover, on the second line of every node the expected start and finish times for the activity scheduled at that node are indicated. Each pair of numbers on the last line of every node in Figure 3 is assigned to a resource flow. The first and second numbers of a pair correspond respectively to the number of activity sending the resource to the node and the amount of the resource which is sent. In the search tree of Figure 3 the fathomed nodes are depicted in gray or in dotted lines, such that the gray nodes are the nodes pruned by pruning rules 1 and/or 2 and the nodes with dotted lines are the nodes pruned because of the lower bound based pruning rules 3, 4, and/or 5.

The search tree initially (i.e., at level 0) consists of the root node in which start dummy activity is scheduled to start and finish at the expected time (0,0,0). The eligible activities of level 1 are

EAS_1={1,2,3}, which get their required resources from start dummy activity and are scheduled to start at the expected time (0,0,0), see step 5 of the algorithm. The expected EST for all the activities in EAS_1 are equal, so activity 1 which has a lower number than the others is selected to branch from it at level 2. The eligible activities of level 2 are EAS_2={2,3,6}, among them activity 2 is selected as *SA* considering the expected EST values. In regard to the resource flow and precedence relations on the current path, activities 0 (having 5 units of resources) and 1 (having 2 units of resources) can send their resources to activity 2. Considering the expected finish times, since activity 0 has higher priority than activity 1 (see step 7.2 of the algorithm) for sending resources to activity 2, 3 units of resources are transferred from start dummy activity to activity 2. Activity 2 is not fathomed by any of the presented pruning rules, so a node is created for this activity with expected start and finish times equal to (0,0,0) and (3,5,7), respectively. The partial schedule has now three activities *PS*={0,1,2}. At level 3, EAS_3 consists of three activities 3, 4, and 6. The priority of activity 3 in regard to the expected EST values is more than the others; therefore activity 3 is considered as *SA*. Since this activity is not fathomed by any of the pruning rules, it is scheduled to start at expected time (1,2,3) getting its required resources from root node (2 units of resources) and activity 1 (one unit of resources). Activity 1 is a resource transferor to activity 3 but is not its predecessor, thus activity 1 is considered as a resource predecessor of activity 3. It is continued to traverse the tree down until a complete feasible schedule is obtained by scheduling the end dummy activity to start and finish at expected time (8,19,31). The upper bound is now updated by setting its value to the expected finish time of end dummy activity, i.e. *UB*=(8,19,31). Then, since there exist unexplored, unfathomed nodes on the search tree, the backtracking is done and the node 12 on level 5 is selected as *SN*. However, this node is fathomed because its *LB3* equals *UB* and another node, node 13 in Figure 3, is selected for evaluation. This search process finishes after 58 nodes have evaluated and the expected time (8,19,31) is resulted as the minimum value for the expected makespan.

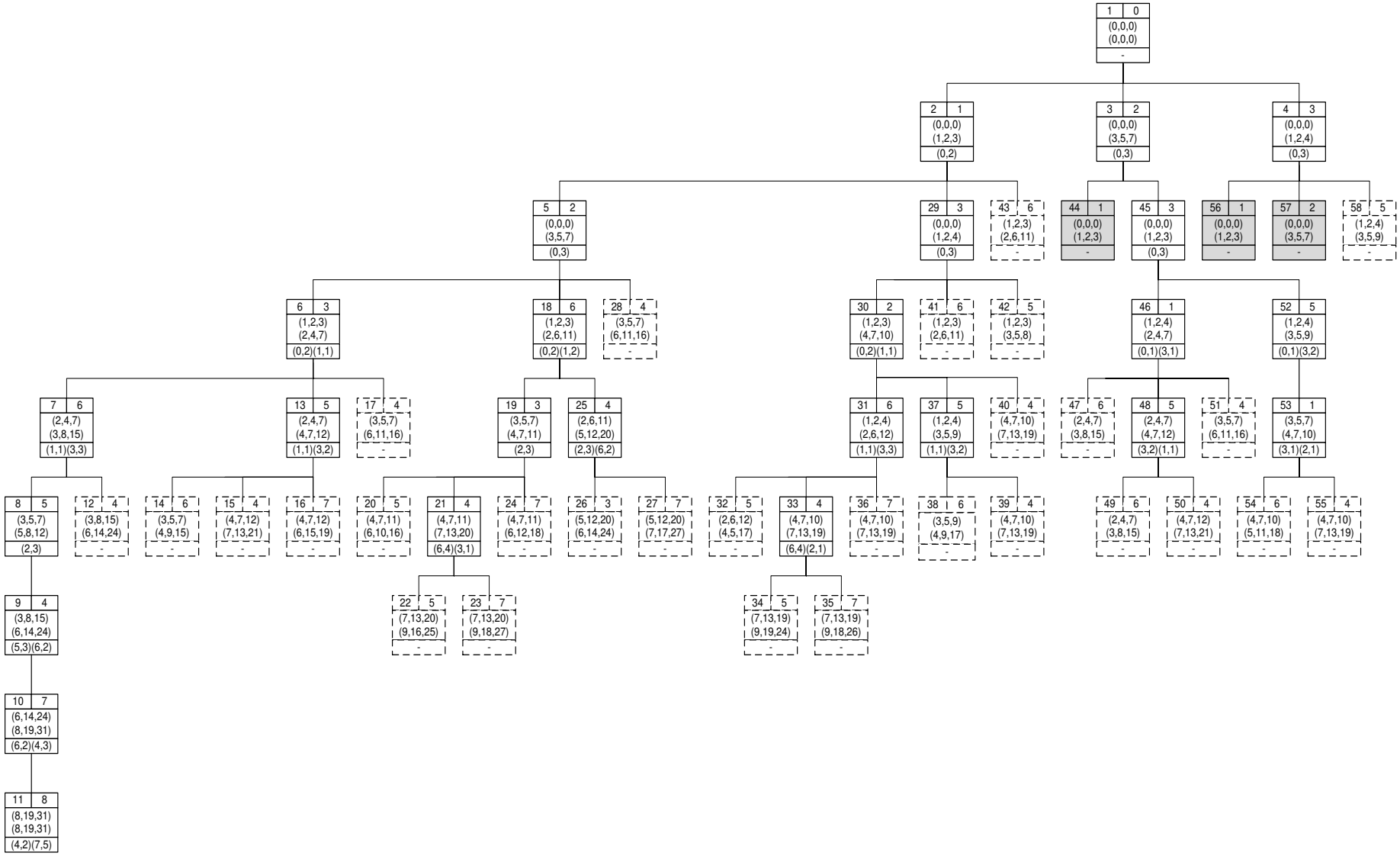

Figure 3. The B&B tree for the numerical example of Figure 2.

5. Computational Experiments

The algorithm proposed in section 4.3 was implemented in Matlab programming language on a laptop with Windows 7 operation system, Intel Core 2 Duo and a CPU at 2.00 GHz. In Section 5.1 the tested benchmark problem instances are described. In Section 5.2 a sensitivity analysis is carried out to investigate the effects of different settings for the weights ω_1, ω_2, and ω_3 on the performance of the B&B. Finally, in Section 5.3 the computational results are provided.

5-1. Test instances

Since there is no benchmark problem set for RCPSP with fuzzy random activity times, we act the same as Alipouri et al. (2017, 2019) and take the J30 and J60 sets of benchmark problems from the PSPLIB (see site: http://www.om-db.wi.tum.de/psplib/data.html) as the base and generate problems with fuzzy random activity durations. Sets J30 and J60 has been systematically generated by the standard project generator ProGen and each set consists of 480 instances.

Here, durations of activities are assumed to be fuzzy normal variables and the calculations are carried out based on the expected values of the durations, $\bar{\mu}_{d_i} = (d_{i,1}, d_{i,2}, d_{i,3})$, $0 \le i \le n+1$, which are triangular fuzzy numbers. Therefore, by generating triangular fuzzy times for the expected durations of activities, with regard to their deterministic durations, suitable benchmark problems for testing the performance of our proposed procedure can be produced. To this end, $d_{i,2}$, $0 < i < n+1$, is set to the deterministic estimation for activity i, $d_{i,1}$ and $d_{i,3}$, $0 < i < n+1$, are calculated as $d_{i,1} = d_{i,2} - 2$ and $d_{i,3} = d_{i,2} + 3$, respectively, and the values of $d_{i,1}$ are bounded by zero. Finally, expected durations of start and end dummy activities are set to (0,0,0).

5-2. Sensitivity analysis

A sensitivity analysis was carried out to investigate the effects of different settings for the weights ω_1, ω_2, and ω_3 on the performance of the B&B. 96 benchmark problems selected randomly from the 96 groups of problems generated in Section 5.1. These were solved to optimality by the proposed B&B based on the 10 scenarios shown in Table 1 for the weights ω_1, ω_2, and ω_3.

Table 1. Different scenarios for the setting of weights ω_1, ω_2, and ω_3.

Weights	Scenario1	Scenario2	Scenario3	Scenario4	Scenario5	Scenario6	Scenario7	Scenario8	Scenario9	Scenario10
ω_1 & ω_3	0.45	0.4	0.35	0.3	0.25	0.2	0.15	0.1	0.05	0
ω_2	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1

For each of the scenarios in Table 1, the mean of the defuzzified makspans for both sets J30 and J60 are reported in Figure 4. The well-known centroid of area (COA) method is employed to defuzzify fuzzy numbers. As seen in Figure 5, an increase in the value of ω_2 (from scenario 1 to 8) leads to a decrease in defuzzified makespans of both sets. However, from scenario 8 to scenario 10, increasing the value of ω_2 increases the defuzzified makespans. Therefore, based on this sensitivity analysis, values 0.1, 0.8, and 0.1 are appropriate settings respectively for the weights ω_1, ω_2, and ω_3 in order to calculate the fitness values in the proposed B&B.

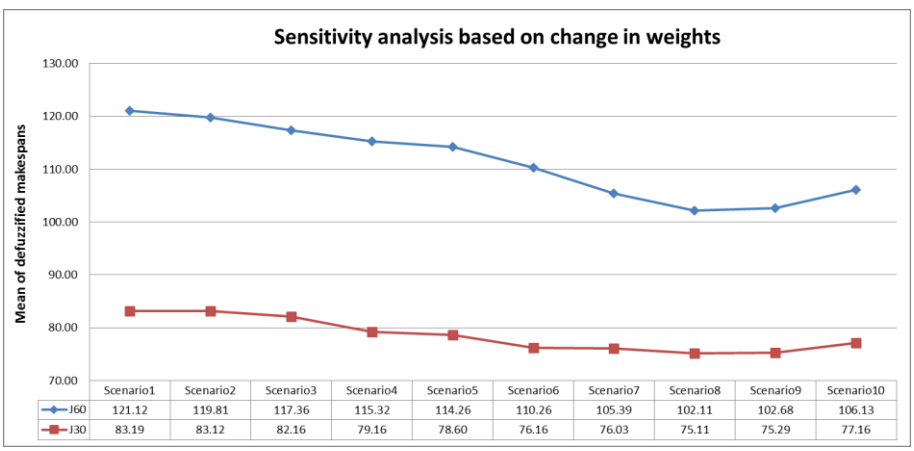

Figure 4. Results of the sensitivity analysis

5-3. Computational results

The literature review of FS-RCPSP indicated that, so far only Nematian et al. (2010), Alipouri et al. (2017), and Alipouri et al. (2019) have proposed a method to solve this problem. Nematian et al. (2010) did not report any results from applying their MILP model on any benchmark problem and just illustrated their proposed procedure by an example with 7 activities. Hence, in this paper, the performance of our proposed B&B algorithm is only compared with the results obtained by Alipouri et al. (2017, 2019) to solve the 960 benchmark problem instances identified

earlier. Comparing against CPLEX and SADESP provides quantitative results that are easily reproducible by other researchers.

Based on the sensitivity analysis carried out in Section 5.2 and also in the same setting as Alipouri et al. (2017, 2019), in all experiments, values of ω_1, ω_2, and ω_3 are considered to be 0.1, 0.8, and 0.1, respectively. Alipouri et al. (2019) showed based on a sensitivity analysis that this setting for the weights results in minimum defuzzified makespans. The results obtained by applying CPLEX, SADESP (with 100000 computed schedules) and our proposed algorithm to solve the project instances generated from J30 and J60 sets are presented in Tables 2 and 3. These instances, based on the input parameters of ProGen, are classified into 96 groups – 48 groups for each set – each one containing 10 problems. The number of problems out of 10 that B&B was able to find an integer solution for them, the mean of objective values (Mean of *Z*), and the mean of computation times (Mean of CPU) are reported for each groups in both tables. In addition, the same results reported in Alipouri et al. (2017) and Alipouri et al. (2019) are presented in these tables under the CPLEX and SADESP columns for comparison. In both tables, for those groups which their results are differentiated in italics from other results, the CPLEX could only find integer solutions for some of the problems. The results of CPLEX for these groups are only for those problem instances for which it was able to find integer solutions. Bold numbers in the columns of Tables 2 and 3 show where either CPLEX, SADESP or B&B has performed better in regard to mean of *Z* and mean of CPU times. Since Alipouri et al. (2019) has compared the results of SADESP with CPLEX, this comparison is not discussed in this paper.

Table 2. CPLEX, SADESP, and proposed B&B results from solving 480 generated problems (48 groups) generated from J30 set

No.	**Group**	**CPLEX**			**SADESP (100000 schedule)**			**Proposed branch-and-bound**		
		Solved No.	**Mean of *Z***	**Mean of CPU (s)**	**Solved No.**	**Mean of *Z***	**Mean of CPU (s)**	**Solved No.**	**Mean of *Z***	**Mean of CPU (s)**
1	j30_1	10	**40.40**	365.57	10	40.75	7568.43	10	**40.4**	**240.67**
2	j30_2	10	**38.62**	13.11	10	**38.62**	8126.23	10	**38.62**	**8.07**
3	j30_3	10	48.84	1.14	10	49.15	8243.15	10	**48.83**	**0.60**
4	j30_4	10	42.06	**0.38**	10	**41.93**	7992.04	10	**41.93**	0.63
5	j30_5	10	58.50	5308.60	10	56.94	8342.16	10	**56.49**	**3705.92**
6	j30_6	10	41.15	3821.49	10	42.09	8265.25	10	**41.14**	**2356.22**
7	j30_7	10	**38.09**	45.18	10	38.39	8326.16	10	**38.09**	**23.87**
8	j30_8	10	40.82	1.99	10	**40.78**	7953.77	10	**40.78**	**1.74**
9	j30_9	0	N/A	4934.15	10	62.9	**8234.62**	10	**60.07**	16553.42
10	j30_10	10	43.64	3767.31	10	42.32	8122.30	10	**42.30**	**2133.73**
11	j30_11	10	45.67	3027.07	10	45.35	8256.12	10	**45.30**	**1285.02**
12	j30_12	10	40.20	8.87	10	40.23	8066.26	10	**40.13**	**4.25**

13	j30_13	0	N/A	3369.70	10	61.88	**8421.26**	10	**60.10**	13562.67
14	j30_14	10	42.89	6517.80	10	42.31	8168.16	10	**40.72**	**3973.02**
15	j30_15	10	43.85	2620.86	10	43.6	8421.11	10	**43.57**	**678.33**
16	j30_16	10	37.27	178.35	10	**37.23**	7939.15	10	**37.23**	**119.80**
17	j30_17	10	**47.82**	**38.83**	10	47.87	8224.40	10	**47.82**	158.63
18	j30_18	10	44.53	**1.43**	10	44.42	8061.22	10	**44.39**	12.25
19	j30_19	10	**42.25**	16.53	10	**42.25**	8155.12	10	**42.25**	**10.94**
20	j30_20	10	**41.43**	3.23	10	41.44	8325.32	10	**41.43**	**2.32**
21	j30_21	10	58.96	13411.07	10	57.87	8065.16	10	**57.3**	**5894.60**
22	j30_22	10	44.54	2702.73	10	44.61	8233.13	10	**44.53**	**2243.70**
23	j30_23	10	**45.84**	385.11	10	45.88	8316.14	10	**45.84**	**168.90**
24	j30_24	10	**42.19**	4.47	10	**42.19**	8398.46	10	**42.19**	**0.381**
25	j30_25	0	N/A	5482.67	10	64.00	**8763.16**	10	**63.46**	21802.88
26	j30_26	10	46.92	4101.51	10	46.68	8536.15	10	**46.67**	**1249.32**
27	j30_27	10	46.57	**141.04**	10	46.43	8011.56	10	**46.32**	370.72
28	j30_28	10	46.15	3196.74	10	**46.14**	8077.82	10	**46.14**	**3.53**
29	j30_29	0	N/A	4596.00	10	72.95	**8043.55**	10	**71.16**	20753.57
30	j30_30	10	48.82	10739.64	10	47.16	8079.88	10	**46.03**	**2750.33**
31	j30_31	10	45.28	4716.09	10	44.89	8012.43	10	**43.73**	**2729.92**
32	j30_32	10	44.90	48.66	10	**44.86**	8623.18	10	**44.86**	**4.17**
33	j30_33	10	**49.54**	**4.65**	10	50.38	8466.58	10	**49.54**	46.35
34	j30_34	10	47.85	1.90	10	**47.83**	8523.55	10	**47.83**	**1.60**
35	j30_35	10	**46.89**	10.30	10	**46.89**	8433.15	10	**46.89**	**2.94**
36	j30_36	10	**46.65**	7.03	10	**46.65**	8442.10	10	**46.65**	**1.88**
37	j30_37	10	64.03	10471.41	10	64.04	8823.16	10	**63.73**	**3725.03**
38	j30_38	10	**49.55**	**964.60**	10	49.83	8522.13	10	**49.55**	1040.17
39	j30_39	10	**47.93**	13.14	10	**47.93**	8321.16	10	**47.93**	**3.68**
40	j30_40	10	**45.69**	1.72	10	**45.69**	8522.88	10	**45.69**	**0.78**
41	j30_41	*8*	*86.41*	*6045.91*	10	75.04	8369.16	10	**74.09**	**5194.53**
42	j30_42	10	51.61	9401.20	10	51.21	8590.20	10	**51.03**	**5326.87**
43	j30_43	10	**47.51**	5734.81	10	47.68	8566.16	10	**47.51**	**1922.11**
44	j30_44	10	44.65	3.22	10	**44.64**	8346.13	10	**44.64**	**1.66**
45	j30_45	*5*	*89.08*	*6211.25*	10	82.77	8324.08	10	**82.09**	**6097.32**
46	j30_46	10	51.78	10143.28	10	**50.17**	8346.50	10	**50.17**	**3539.95**
47	j30_47	10	45.95	3577.96	10	46.00	8003.18	10	**45.85**	**1468.97**
48	j30_48	10	**45.31**	3.13	10	**45.31**	8046.13	10	**45.31**	**2.19**

As it is seen in Table 2, the CPLEX was unable to find an integer solution for all problem instances in 4 groups of J30 set, resulting in a corresponding 'N/A' entry. This was also the case for two and five instances in groups j30_41 and j30_45, respectively. In contrast, SADESP (after 100000 trials) and B&B was able to find integer solutions for all 480 instances of J30. Compared to the CPLEX, proposed algorithm performed better in terms of the mean of Z in 32 of the 48 groups, and both algorithms were tied in 16 groups. When B&B's results for set J30 is compared to those of SADESP, corresponding figures are 32 and 16, respectively. Where B&B beat CPLEX, the average margin between the mean results was 1.24% with some groups showing larger margins than 2%, e.g. j30_5 (3.44%), j30_10 (3.07%), j30_14 (5.06%), j30_21 (2.82%), j30_30 (5.71%), j30_31 (3.42%), and j30_46 (3.11%). The average margin between the results returned by B&B and SADESP was 0.77% in favor of B&B. Where B&B beat SADESP, the average margin between the mean results was 1.02% with some groups showing larger margins

than 2%, e.g. j30_6 (2.26%), j30_9 (4.50%), j30_13 (2.88%), j30_14 (3.76%), j30_29 (2.45%), j30_30 (2.40%), and j30_31 (2.58%). The average margin between the results returned by B&B and SADESP for J30 set was 0.68% in favor of B&B. Based on the results, our algorithm worked well (better or equal) on 100% of the problems in J30 set in terms of the mean of Z.

Table 3. CPLEX, SADESP, and proposed B&B results from solving 480 generated problems (48 groups) generated from J60 set

No.	Group	CPLEX			SADESP (100000 schedule)			Proposed branch-and-bound		
		Solved No.	Mean of Z	Mean of CPU (s)	Solved No.	Mean of Z	Mean of CPU (s)	Solved No.	Mean of Z	Mean of CPU (s)
1	j60_1	10	**61.95**	13712.01	10	62.19	14362.25	10	**61.95**	**11654.12**
2	j60_2	10	**55.09**	327.22	10	55.21	15263.11	10	**55.09**	**168.22**
3	j60_3	10	**57.73**	897.56	10	**57.73**	14900.88	10	**57.73**	**263.15**
4	j60_4	10	56.97	0.87	10	**56.95**	14766.54	10	**56.95**	**0.67**
5	j60_5	0	N/A	8926.37	10	78.21	14908.26	10	**70.29**	**13186.25**
6	j60_6	*8*	*56.89*	*14128.38*	10	56.98	14866.57	10	**56.16**	**8215.11**
7	j60_7	10	**57.76**	230.03	10	57.96	14004.26	10	**57.76**	**103.25**
8	j60_8	10	**56.50**	25.09	10	**56.50**	13994.45	10	**56.50**	**10.03**
9	j60_9	0	N/A	15389.13	10	86.49	13958.22	10	**76.31**	**11263.25**
10	j60_10	*4*	*59.95*	*21836.50*	10	61.06	14696.55	10	**60.00**	**8435.16**
11	j60_11	10	54.91	3721.25	10	53.89	14776.45	10	**53.05**	**2144.20**
12	j60_12	10	53.10	122.26	10	**53.09**	13980.22	10	**53.09**	**74.23**
13	j60_13	0	N/A	3369.70	10	92.36	14378.93	10	**88.27**	**12005.59**
14	j60_14	*6*	*60.48*	*4490.70*	10	57.59	14425.78	10	**56.44**	**2694.14**
15	j60_15	*7*	*60.60*	*1871.2*	10	60.65	14680.23	10	**60.56**	**1309.84**
16	j60_16	10	**53.02**	**253.22**	10	53.09	13457.65	10	**53.02**	316.45
17	j60_17	*6*	*61.42*	*11746.37*	10	68.54	**14030.49**	10	**61.40**	16924.06
18	j60_18	10	**63.37**	**124.58**	10	64.86	14365.32	10	**63.37**	175.23
19	j60_19	10	**59.86**	27.45	10	**59.86**	13908.25	10	**59.86**	**12.12**
20	j60_20	10	**60.87**	0.56	10	**60.87**	13963.24	10	**60.87**	**0.49**
21	j60_21	0	N/A	17342.3	10	81.29	14805.46	10	**81.19**	**12386.52**
22	j60_22	0	N/A	213782.2	10	60.14	**14069.19**	10	**59.61**	16529.32
23	j60_23	10	**59.04**	147.28	10	**59.04**	13904.29	10	**59.04**	**85.26**
24	j60_24	10	**58.30**	22.41	10	**58.30**	14334.45	10	**58.30**	**13.26**
25	j60_25	0	N/A	**8937.23**	10	83.65	14288.71	10	**82.53**	15266.08
26	j60_26	*6*	*52.97*	*4138.00*	10	63.64	14375.12	10	**59.26**	**2483.79**
27	j60_27	10	**61.60**	5026.36	10	61.73	14322.45	10	**61.60**	**1843.32**
28	j60_28	10	61.91	54.86	10	**61.89**	13864.22	10	**61.89**	**46.26**
29	j60_29	0	N/A	16283.70	10	95.33	**14364.66**	10	**92.10**	25328.89
30	j60_30	*5*	*66.46*	*12562.18*	10	67.44	13949.28	10	**64.68**	**6279.25**
31	j60_31	10	58.24	**6221.54**	10	58.29	13761.44	10	**58.09**	7952.36
32	j60_32	10	**65.12**	178.37	10	**65.12**	14076.16	10	**65.12**	**101.03**
33	j60_33	10	73.02	4605.7	10	74.26	14792.13	10	**72.52**	**3201.11**
34	j60_34	10	62.51	1398.67	10	63.48	14041.46	10	**62.09**	**800.26**
35	j60_35	10	**63.10**	13.5	10	64.26	13908.26	10	**63.10**	**11.51**
36	j60_36	10	60.44	0.62	10	60.42	13983.23	10	**60.41**	**0.54**
37	j60_37	0	N/A	11231.45	10	88.49	14720.47	10	**85.74**	**9863.25**
38	j60_38	*7*	*63.59*	*7525.78*	10	65.55	14853.73	10	**61.89**	**5268.04**
39	j60_39	10	64.34	4845.14	10	64.36	14315.29	10	**64.24**	**4325.29**
40	j60_40	10	64.43	**5.25**	10	**64.41**	13952.16	10	**64.41**	16.25
41	j60_41	0	N/A	**14651.5**	10	99.28	14762.11	10	**96.73**	22853.20
42	j60_42	0	N/A	**7638.2**	10	66.16	14236.19	10	**64.89**	13200.58
43	j60_43	10	**65.34**	**7834.66**	10	**65.34**	14729.10	10	**65.34**	8621.48
44	j60_44	10	**62.08**	45.82	10	**62.08**	13942.72	10	**62.08**	**37.12**

45	j60_45	0	N/A	19456.3	10	98.22	14326.30	10	**96.09**	**13265.26**
46	j60_46	0	N/A	6738.4	10	66.28	14235.16	10	**65.82**	**5562.13**
47	j60_47	10	60.67	11851.27	10	60.64	13992.15	10	**59.93**	**3596.25**
48	j60_48	10	64.37	149.32	10	**64.35**	14233.20	10	**64.35**	**99.36**

For J60 set, which has more challenging problems than J30 set, the CPLEX could not find an integer solution for all problem instances in 12 groups having 'N/A' entry in Table 3. In contrast, SADESP (after 100000 trials) and proposed B&B was again able to find integer solutions for all 480 instances of J60. Compared to the CPLEX, in 32 of the 48 groups for J60 sets, B&B performed better in terms of the mean of *Z*, whilst both algorithms were tied in the remaining 16 groups. When B&B's results for set J60 is compared to those of SADESP, corresponding figures are 34 and 14, respectively. Where B&B beat CPLEX, the average margin between the mean results was 0.55% and all the margins were smaller than 2% except for j60_11(3.39%). The average margin between the results returned by the two algorithms was 0.23% in favor of B&B. Where B&B beat SADESP, the average margin between the mean results was 2.59% with 14 groups showing larger margins than 2%. The average margin between the results returned by B&B and SADESP for J60 set was 1.83% in favor of B&B. Based on the results, our algorithm worked well (i.e., better or equal) on 100% of the problems in J60 set in terms of the mean of Z.

CPU time of the algorithms should also be compared to clarify their costs. Since the specifications of the employed operating system for the current study are the same as Alipouri et al. (2017) and Alipouri et al. (2019), a fair comparison can be done easily just by comparing the CPU times reported in Tables 2 and 3. As seen in these tables, in 42 of the 48 groups for J30 and 40 of 48 groups for J60, the computation time taken by CPLEX was higher than B&B, whilst for the remaining 6 groups for J30 and 8 groups for J60, CPLEX performed better in terms of the mean of CPU times. Also, it is observed in Tables 2 and 3 that in 44 of the 48 groups for J30 and 45 of the 48 groups for J60 the mean of the mean of CPU time taken by SADESP (with 100000 produced schedules) was higher than proposed B&B, whilst for the remaining 4 groups for J30 and 3 groups for J60, SADESP performed better in terms of the mean of CPU times. But should it be taken into consideration that, based on the results of Alipouri et al. (2019) the computation time taken by SADESP scales proportionally with the number of schedules evaluated, and the distribution of computation times for SADESP was fairly uniform.

Based on the above numerical experimental results on the project instances generated from J30 and J60 sets from PSPLIB, it can be seen that the proposed B&B outperforms SADESP and

CPLEX. Meanwhile, the comparison of CPU times of algorithms indicates that in average the proposed B&B is faster than SADESP and CPLEX on 89% of the instances. In short, proposed B&B is more effective and efficient than SADESP and CPLEX in solving FS-RCPSP.

6. Conclusion

In this study, we addressed the fuzzy stochastic resource-constrained project scheduling problem (FS-RCPSP), where the objective is to minimize the expected makespan of the project subject to precedence and resource constraints under the mixed uncertainty of fuzziness and randomness. The uncertainties were represented by fuzzy random variables and calculations carried out based on the expected value of the durations. We designed a resource flow based B&B procedure incorporating with five pruning rules to find optimal solutions for this problem. The proposed B&B was successfully tested on 960 benchmark instances generated by ProGen project schedule generator. Our results indicated that the proposed approach can perform better than CPLEX solver and SADESP to solve the FS-RCPSP problems, which allows its use in practice to handle project schedule uncertainties.

Future research should consider testing the proposed B&B algorithm on a wider variety of benchmark problems in order to validate both the generality and robustness of this approach to fuzzy stochastic resource-constrained project scheduling. Currently, only the mean of the activity durations, modeled as fuzzy random variables, is used in the calculations of the proposed algorithm. Future studies can consider modeling other project parameters (e.g. the resource requirements of activities) as fuzzy random variables and include the variance of uncertain parameters along with the mean value in the model. Moreover, future research should try to introduce scenarios and to find a compromise between the solutions related to every scenario. Proposing more effective pruning rules can also be a good subject for future researches.

7. Compliance with ethical standards

Financial Disclosures: The author has no relevant financial disclosures.

Conflict of interest: The author declare that he has no conflict of interest.

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

8. References

Alcaraz J, Maroto C (2001) A robust genetic algorithm for resource allocation in project scheduling. Ann Oper Res 102: 83–109. https://doi.org/10.1023/A:1010949931021.

Alipouri Y, Sebt MH, Ardeshir A, Chan WT (2019) Solving the FS-RCPSP with hyper-heuristics: A policy-driven approach. J Oper Res Soc 70(3):403-419. https://doi.org/10.1080/01605682.2018.1441636.

Alipouri Y, Sebt MH, Ardeshir A, Fazel Zarandi MH (2017) A mixed-integer linear programming model for solving fuzzy stochastic resource constrained project scheduling problem. Oper Res Int J 20:197–217. https://doi.org/10.1007/s12351-017-0321-x.

Altintas C, Azizoglu M (2020) A Branch and Bound Algorithm for a Multi-Mode Project Scheduling Problem With a Single Non-Renewable Resource. Int J Inf Tech Proj Manag (IJITPM) 11(2):1-14. https://doi.org/10.4018/IJITPM.2020040101.

Artigues C, Michelon P, Reusser S (2003) Insertion techniques for static and dynamic resource-constrained project scheduling. Eur J Oper Res 149(2):249–267. https://doi.org/10.1016/S0377-2217(02)00758-0.

Artigues C, Leus R, Talla Nobibon F (2013) Robust optimization for resource-constrained project scheduling with uncertain activity durations. Flex Serv Manuf J 25(1):175-205. https://doi.org/10.1007/s10696-012-9147-2.

Ashtiani B, Leus R, Aryanezhad MB (2011) New competitive results for the stochastic resource-constrained project scheduling problem: exploring the benefits of pre-processing. J Scheduling 4(2):157-171. https://doi.org/10.1007/s10951-009-0143-7.

Atli O, Kahraman C (2012) Fuzzy resource-constrained project scheduling using taboo search algorithm. Int J Intell Syst 27(10):873–907. https://doi.org/10.1002/int.21552.

Atli O, Kahraman C (2013) The MinSlack and Kangaroo algorithm heuristic for fuzzy resource-constrained project scheduling problems. J Mult Valued Logic Soft Comput, 20 (1-2):189-219.

Brucker P, Knust S, Schoo A, Thiele O (1998) A branch and bound algorithm for the resource-constrained project scheduling problem. Eur J Oper Res 107(2):272-288. https://doi.org/10.1016/S0377-2217(97)00335-4.

Bruni ME, Di Puglia Pugliese L, Beraldi P, Guerriero F (2017) An adjustable robust optimization model for the resource-constrained project scheduling problem with uncertain activity durations. Omega-Int J Manage S 71:66-84. https://doi.org/10.1016/j.omega.2016.09.009.

Buckley JJ (2005) Fuzzy Probability: New Approach and Applications. Studies in Fuzziness and Soft Computing Series, Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/3-540-32388-0.

Chen L, Zhang Z (2016) Preemption resource-constrained project scheduling problems with fuzzy random duration and resource availabilities. J Ind Prod Eng 33(6):373-382 https://doi.org/10.1080/21681015.2016.1140089.

Damay J, Quilliot A, Sanlaville E (2007) Linear programming based algorithms for preemptive and non-preemptive RCPSP. Eur J Oper Res 182(3):1012–1022. https://doi.org/10.1016/j.ejor.2006.09.052.

Davari M, Demeulemeester E (2018) A novel branch-andbound algorithm for the chance-constrained resource-constrained project scheduling problem. Int J Prod Res 57(4):1265-1282. https://doi.org/10.1080/00207543.2018.1504245.

Demeulemeester EL, Herroelen WS (2002) Project Scheduling: A Research Handbook. International Series in Operations Research and Management Science Vol. 49, Kluwer Academic Publishers (now Springer). https://doi.org/10.1007/b101924.

Dorndorf U, Pesch E, Phan-Huy T (2000) A branch-and-bound algorithm for the resource-constrained project scheduling problem. Math Method Oper Res 52:413–439. https://doi.org/10.1007/s001860000091.

Fang C, Kolisch R, Wang L, Chundi M (2015) An estimation of distribution algorithm and new computational results for stochastic resource-constrained project scheduling problem. Flex Serv Manuf J 27:585-605. https://doi.org/10.1007/s10696-015-9210-x.

Herroelen W, Leus R, (2005) Project scheduling under uncertainty: survey and research potentials. Eur J Oper Res 165(2):289-306. https://doi.org/10.1016/j.ejor.2004.04.002.

Hirota K (1981) Concepts of probabilistic sets. Fuzzy Set Syst 5(1):31–46. https://doi.org/10.1016/0165-0114(81)90032-4

Huang W, Ding L, Wen B, Cao B (2009) Project Scheduling Problem for Software Development with Random Fuzzy Activity Duration Times, In: Yu W., He H., Zhang N. (eds) Advances in Neural Networks – ISNN 2009. ISNN 2009. Lecture Notes in Computer Science, vol 5552. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-01510-6_8.

Itoh T, Ishii H (2005) One Machine Scheduling Problem with Fuzzy Random Due-Dates. Fuzzy Optim Decis Ma 4:71–78. https://doi.org/10.1007/s10700-004-5571-4.

Ke H, Liu B (2007) Project Scheduling Problem with Mixed Uncertainty of randomness and fuzziness. Eur J Oper Res 183(1):135-147. https://doi.org/10.1016/j.ejor.2006.09.055.

Kolisch R, Sprecher A, Drexl A (1995) Characterization and generation of a general class of resource-constrained project scheduling problems. Manage Sci 41:1693-1703. https://doi.org/10.1287/mnsc.41.10.1693.

Kruse R, Meyer KD (1987) Statistics with Vague Data. Part of the Theory and Decision Library book series (TDLB, volume 6). https://doi.org/10.1007/978-94-009-3943-1.

Kumar N, Vidyarthi DP (2016) A model for resource-constrained project scheduling using adaptive PSO. Soft Comput 20:1565–1580, https://doi.org/10.1007/s00500-015-1606-8.

Lambrechts O, Demeulemeester E, Herroelen W (2008) Proactive and reactive strategies for resource-constrained project scheduling with uncertain resource availabilities. J Scheduling 11:121-136. https://doi.org/10.1007/s10951-007-0021-0.

Lawler EL (1976) Combinatorial Optimization: Networks and Matroids. Holt, Rinehart and Winston, New York.

Leus R, Herroelen W (2004) Stability and resource allocation in project planning. IIE Transactions 36(7):667-682. https://doi.org/10.1080/07408170490447348.

Liu B (2002a) Theory and Practice of Uncertainty Programming. Part of the Studies in Fuzziness and Soft Computing book series (STUDFUZZ, volume 102). Physica-Verlag, Heidelberg. https://doi.org/10.1007/978-3-7908-1781-2.

Liu B (2002b) Random fuzzy dependent-chance programming and its hybrid intelligent algorithm. Inform Sciences, 141(3-4):259–271. https://doi.org/10.1016/S0020-0255(02)00176-7.

Mahdi Mobini MD, Rabbani M, Amalnik MS et al. (2009) Using an enhanced scatter search algorithm for a resource-constrained project scheduling problem. Soft Comput 13:597–610. https://doi.org/10.1007/s00500-008-0337-5.

Nematian J, Eshghi K, Eshragh-Jahromi A (2010) A resource-Constrained Project Scheduling Problem with Fuzzy Random Duration. Journal of Uncertain Systems, 4(2):123-132.

Pantouvakis JP, Manoliadis OG (2006) A practical approach to resource-constrained project scheduling. Oper Res Int J 6(3):299–309. https://doi.org/10.1007/BF02941258.

Puri ML, Ralescu, DA (1986) Fuzzy random variables. J Math Anal Appl 114(2):409–422. https://doi.org/10.1016/0022-247X(86)90093-4.

Rostami S, Creemers S, Leus R (2018) New strategies for stochastic resource-constrained project scheduling. J Scheduling 21(3):349-365. https://doi.org/10.1007/s10951-016-0505-x.

Sadeh N, Otsuka S, Schelback R (1993) Predictive and reactive scheduling with the microboss production scheduling and control system. In: proceedings of the IJCAI-93. Workshop on Knowledge-based Production Planning, Scheduling and Control, 293-306.

Sheng B, Wang H, Xiao Z et al. (2019) A novel heuristic algorithm with activity back-shift response model for resource-constrained project scheduling problem. Soft Comput 23:7805–7819. https://doi.org/10.1007/s00500-018-3410-8.

Sebt MH, Alipouri Y, Alipouri Y (2013) Solving resource-constrained project scheduling problem with evolutionary programming. J Oper Res Soc 64:1327–1335. https://doi.org/10.1057/jors.2012.69.

Sprecher A, Drexl A (1998) Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm. Eur J Oper Res 107(2):431–450. https://doi.org/10.1016/S0377-2217(97)00348-2.

Sprecher A, Hartmann S, Drexl A (1997) An exact algorithm for project scheduling with multiple modes. OR Spectrum 19(3):195–203. https://doi.org/10.1007/BF01545587.

Van de Vonder S, Demeulemeester E, and Herroelen W (2007) A classification of predictive-reactive project scheduling procedures. J Scheduling 10:195-207. https://doi.org/10.1007/s10951-007-0011-2.

Xu J, Zhang Z (2012) A fuzzy random resource-constrained scheduling model with multiple projects and its application to a working procedure in a large-scale water conservancy and hydropower construction project. J Scheduling 15:253-272. https://doi.org/10.1007/s10951-010-0173-1.

Ying KC, Lin SW, Lee ZJ (2009) Hybrid-directional planning: improving improvement heuristics for scheduling resource-constrained projects. Int J Adv Manuf Tech 41(3):358-366. https://doi.org/10.1007/s00170-008-1486-5.

Zadeh LA (1968) Probability measures of fuzzy events. J Math Anal Appl 23:421–427.

Zamani R (2011) A hybrid decomposition procedure for scheduling projects under multiple resource constraints. Oper Res Int J 11:93–111. https://doi.org/10.1007/s12351-009-0073-3.

Zhu Y, Liu B (2005) Some Inequalities of random fuzzy variables with application to moment convergence. Comput Math Appl 50(5-6):719-727. https://doi.org/10.1016/j.camwa.2005.04.015.

Zoulfaghari H, Nematian J, Kanani Nezhad AA (2016) A resource-constrained project scheduling problem with fuzzy activity times. International Journal of Fuzzy System Applications (IJFSA) 5(4):1-15. https://doi.org/10.4018/IJFSA.2016100101.