Int. J. Civ. Eng. (2016) 14:325–337
DOI 10.1007/s40999-016-0031-4

RESEARCH PAPER

Fuzzy Resource Constraint Project Scheduling Problem Using CBO and CSS Algorithms

A. Kaveh[1] · M. Khanzadi[1] · M. Alipour[1]

Received: 5 September 2015 / Revised: 11 November 2015 / Accepted: 30 November 2015 / Published online: 14 June 2016
© Iran University of Science and Technology 2016

Abstract Resource allocation project scheduling problem (RCPSP) has been one of the challenging subjects amongst researchers in the past decades. Most of the researchers in this area have used deterministic variables; however, in a real project, activities are exposed to risks and uncertainties that cause delay in project's duration. There are some researchers that have considered the risks for scheduling; however, new metahuristics are available to solve this problem for finding better solution with less computational time. In this paper, two new metahuristic algorithms are applied for solving fuzzy resource allocation project scheduling problem (FRCPSP), known as charged system search (CSS) and colliding body optimization (CBO). The results show that both of these algorithms find reasonable solutions; however, CBO finds the results in a less computational time, with a better quality. A case study is conducted to evaluate the performance and applicability of the proposed algorithms.

Keywords Resource allocation · Fuzzy logic · Optimization · CBO · CSS

1 Introduction

Project management is the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements [1]. The activity networks of the construction projects are conducted on the basis of precedence relationships. The main project's goals are almost time, cost, and quality. Project managers severely try to achieve the project's goal and finish their projects within minimum duration, minimum cost, and maximum quality. For achieving these goals, any project manager needs a reliable scheduling program considering the project's special circumstances. Therefore, many researchers focused on techniques and optimization methods for project scheduling. The results of these studies in the literature can be classified into four categories: resource constraint scheduling, time cost trade-off, resource leveling, and resource allocation [2]. The main problem type in this study is the well-known resource-constrained project scheduling problem (RCPSP).

This problem type aims at minimizing the total duration or makespan of a project subject to precedence relations between the activities and the limited renewable resource availabilities and is known to be NP-hard [3]. Moreover, if there is more than one nonrenewable resource, the problem of finding a feasible solution for the RCPSP is NP-complete [4]; thus the exact methods could not find the best solution in this kind of problems, especially in the large-scale problems. Several searching methods including exact methods [5–7] (as dynamic programming, enumeration algorithm, branch and bound algorithms), heuristic [8, 9] (as Lagrangian heuristic) and meta-heuristic [11–15] (as genetic algorithm, simulated annealing, particle swarm optimization, and ant colony algorithm) procedures have been suggested to solve this problem with many different assumptions [16]. Also several researchers have tried to improve the results of RCPSP; however, there are few researches about the non-deterministic scheduling problem. Milene et al. [17] showed the results of a research on using the Bellman–Zadeh approach to decision making in a fuzzy

✉ A. Kaveh
alikaveh@iust.ac.ir

[1] Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, P.O. Box 16846-13114, Tehran, Iran

environment for solving multi-objective optimization problems. Wang and Lin [18] developed an intelligent resource allocation model using genetic algorithm and fuzzy inference for reducing lateness of orders with specific due dates. While the genetic algorithm is responsible for arranging and selecting the sequence of orders, the fuzzy inference module conveys how resources are allocated to each order. Zhang and Xing [19] presented a fuzzy-multi-objective particle swarm optimization to solve the fuzzy time–cost–quality trade-off (TCQT) problem. The time, cost and quality are described by fuzzy numbers and a fuzzy multi-attribute utility methodology incorporated with constrained fuzzy arithmetic operations is adopted to evaluate the selected construction methods.

Although limited researches have been conducted in the non-deterministic RCPSP, new metahuristics are available to solve this problem for finding better solution with less computational time.

In this paper, a fuzzy resource constrained project scheduling problem (FRCPSP) is developed that considers RCPSP, risks, and uncertainties simultaneously based on recent researches. For this purpose, two newly developed metahuristics are employed and the findings are compared together and with the previous algorithms. One of the goals of this paper was to utilize two new and efficient algorithms for these problems and compare the quality of the solutions. Charged System Search (CSS) developed by Kaveh and Talatahari [20] and Colliding Body Optimization (CBO) developed by Kaveh and Mahdavi [21] are the proposed models. Then two case studies are conducted to evaluate the performance and applicability of the proposed algorithms.

The remainder of the paper is organized as follows: in Sect. 2, the problem is described briefly and the mathematical model of the problem is presented. In Sect. 3, the algorithms used, CSS and CBO are explained briefly. Section 4 shows the computational results, and finally the concluding remarks are detailed in Sect. 5.

2 Problem Formulation

2.1 RCPSP

The problem studied in this paper is a resource-constrained project scheduling problem which is defined as below [16]:

A project involves the scheduling of $j = 1, \ldots, J$ activities that are described in an activity-on-node (AON) network $G = (V, E)$, where the nodes and arcs represent the set of activities V and finish-to-start precedence relationship (with lag 0) E, respectively. The counter of the activities in the project network is from 0 to $J + 1$, where activities 0 and $J + 1$ are dummy activities specifying the start and finish of the project and these do not take any durations. Precedence relationships between some of the activities in the project necessitate that an activity j cannot be started before all its predecessors P_j are finished due to the technological requirements. Activity j, requires r_j renewable resource k for each period of execution. The time that activity j needs to be executed, d_j, is supposed to be a discrete and non-increasing function of the amount of resource allocated to it. When the activity j starts its execution, any interruption, such as changing the duration or amount of resource cannot happen, and it must be continued in d_j consecutive periods.

Moreover, the availability of the resource k is given by R_k.

The first aim of this paper was to solve RCPSP optimization model using charged system search (CSS) [20] and colliding body optimization (CBO) [21] algorithms introduced by Kaveh et al. (see Kaveh [22] as well). The purpose was to achieve a solution with the minimum total time, considering precedence relations between different activities and resource constraints at the same time. The objective functions of the RCPSP model are formulated to minimize the total project time with allocation of resources in the entire project makespan, simultaneously.

When an activity is selected, the corresponding activity duration and resource requirement will be assigned. Afterwards, a feasible schedule based on activity information and given constraints will be produced. The outcome of the resulting schedule is the determination of the total project time.

The objective of RCPSP model is to minimize the duration of the project, which is the finish time of the last activity f_{J+1} in a project. Therefore, the total project duration F_t is

$$\min F_t \tag{1}$$

In the above formulation, the objective function minimizes the project time F_t.

The constrained is explained as

$$f_j - d_j \geq f_i \quad \forall (i,j) \in E \tag{2}$$

This constraint guarantees the consideration of the precedence relationships. In this formula f_j is the finish time of the activity j, d_j is the duration of activity j and f_i is the finish time of the predecessor of activity j that is called i.

$$\sum_{j \in A_t} r_{jk} \leq R_k \quad k = 1, \ldots, K, \quad A_t = \{ j | f_j - d_j < t \leq f_j \} \tag{3}$$

Constraint set Eq. (3) indicates that for each time instant t and for each resource type k, the renewable resource amounts required by the activities which are currently processed (i.e., A_t) cannot exceed the resource availability, where r_{jk} is the amount of resource k required by the activity j.

$$f_i \geq 0 \quad i = 0,\ 1,\ \ldots,\ j+1 \tag{4}$$

Finally, constraint set Eq. (4) ensures that every output has been positive and the schedule logic will be true.

2.2 Fuzzy Logic

For the time being, little historical data are available that we cannot use for calculation, or when the estimation is not detailed, fuzzy logic will be used increasingly. Conventional sets mainly are kind of sets whose membership is defined on a white/black basis, while in fuzzy set theory; members are not precise phenomena and they take uncertain values that are defined on gray basis. Fuzzy Sets have been designed to deal with a wide range of real-world domains involving linguistic descriptions [23]. Fuzzy set theory has been developed for uncertainties that cannot be quantified due to their qualitative and subjective nature. Fuzzy logic is derived from fuzzy set theory to deal with a set of membership functions that assign to each object a grade of membership ranging between zero (no membership) and unity (full membership). In the classical theory, an element x does or does not belong to a set X. In the fuzzy-set theory, an element may more or less belong to a set: $\mu X(x) \in [0, 1]$ and also it may belong to more than a set. In the fuzzy logic, the values are fuzzy numbers and have a specific distribution [24]. For example fuzzy numbers can be introduced as a single, rectangular, trapezoidal, or triangular number, as shown in Fig. 1.

2.3 FRCPSP

Algebraic operations on real numbers can be extended to fuzzy numbers, i.e., fuzzy variables defined on the real line, by means of the extension principle [24–26]. A method based on the α-cut representation of fuzzy sets and interval analysis is used in extension principle [24]. After determination of the input factors affecting the duration and resource needed by each activity as fuzzy numbers, Extension principle is employed to determine the project makespan. The extension principle state that if $f{:}R{*}R \rightarrow R$ be a binary operation over real numbers, then it can be extended to the operation of fuzzy numbers. The concept of the extension principle is presented as follows:

1. Select a particular α-cut value, where $0 \leq \alpha \leq 1$.
2. The associated crisp values of the input fuzzy numbers corresponding to α is determined as ($a\alpha$, $b\alpha$).
3. Using the values obtained in the previous step, and interval operations, compute the value of the project duration which correspond to those input factors.
4. Steps 1–3 are repeated for as many values of α that are needed to refine the solution. Coverage of the entire range of α-cut makes the output of the model a fuzzy number.

In the case that there is more than one parameter affecting the value of the output (project makespan in this example), different combination of associated crisp values obtained in Step 2 must be considered and the output of the model must be simulated for different

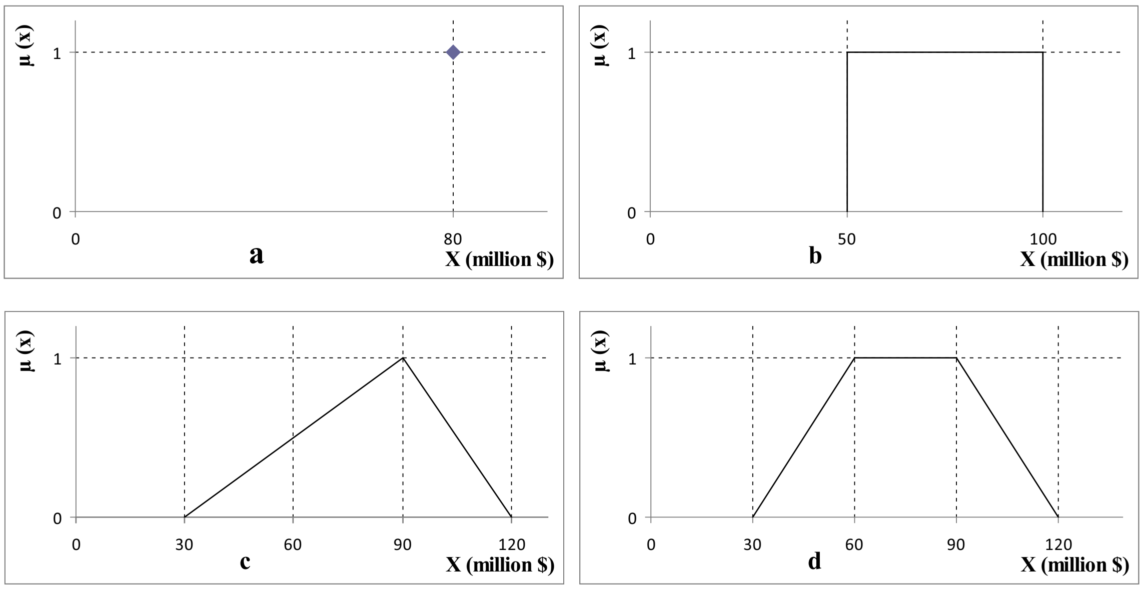


Fig. 1 Different types of fuzzy numbers: **a** single value; **b** rectangular distribution; **c** triangular distribution; **d** trapezoidal distribution Extracted from [24]

combinations of crisp values of this factor at each α-cut. The output obtained from Step 2 is given by: $[x\alpha, y\alpha]$, where $x\alpha$, $y\alpha$ represents the minimum and maximum of outputs resulting from different combination of crisp values at each α-cut, respectively [24, 27].

The proposed model for FRCPSP is explained in the flow diagram of Fig. 2.

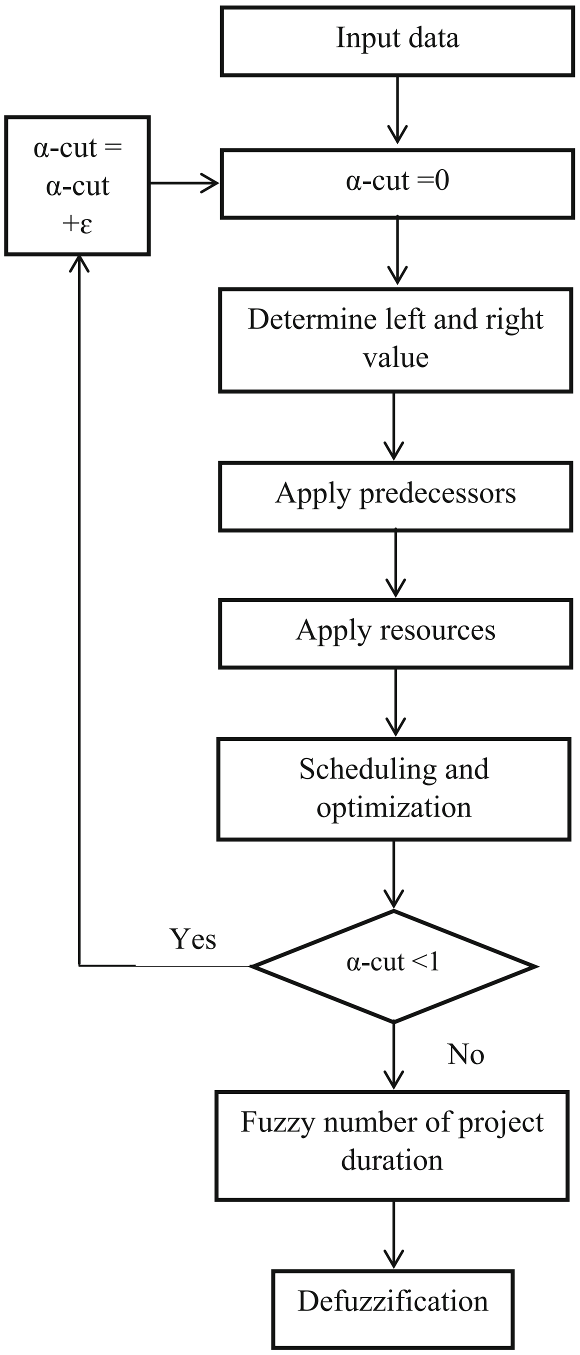


Fig. 2 Flowchart of proposed model of FRCPSP

3 Metahuristic Algorithms

RCPCP aims at minimizing the total duration or makespan of a project subject to precedence relations between the activities and the limited renewable resource availabilities and is known to be NP-hard [3]. Moreover, if there is more than one nonrenewable resource, the problem of finding a feasible solution for the RCPSP is NP-complete [4], so the exact method could not find the best solution in a logical time. For these reasons meta-heuristic algorithms has been used to solve this kind of problems. Although meta-heuristics may not always find the optimal solution, they can find a close to best solution or the best depending on the ability of the utilized meta-heuristic algorithm.

The main purpose of this paper is the optimization of RCPSP and FRCPSP. These two problems deal with scheduling of the projects with resource limitations. The best solution of these problems corresponds to the minimum duration of the project.

To search for solution, two new meta-heuristic algorithms (Charged System Search (CSS) and Colliding Body Optimization (CBO)) are implemented for the optimization. The CSS and CBO, developed by Kaveh and Talatahari [20], and Kaveh and Mahdavi [21], respectively, are two efficient methods that have not been used for this problem up to now [16]. The main algorithms of the proposed meta-heuristics are explained briefly in the following:

3.1 Charged System Search

The charged System Search (CSS) is a population-based meta-heuristic algorithm proposed by Kaveh and Talatahari [21], which is based on laws from electrostatics and Newtonian mechanics laws. This method is extensively used in optimization problem examples which can be found in [28–30]. The following explanation about this method, including definitions and formulas, is extracted from [16, 21].

The Coulomb and Gauss laws provide the magnitude of the electric field at a point inside and outside a charged insulating solid sphere, respectively.

In CSS, each solution is considered as a charged particle (CP)in an n-dimensional space, where n is the number of decision variables. The convergence process is carried out through the movements of these particles in the search space. The electrostatics and mechanics laws govern the forces between these CPs and their movements. The pseudo-code of the CSS algorithm is summarized as follows:

Level 1: initialization.

Step 1 Initialization: in this step, the parameters of the CSS algorithm are initialized as follows: Initialize an array of charged particles (CPs) with random positions. The initial velocities of the CPs are considered as zero. Each CP has a charge of magnitude q_i whose value is calculated as

$$q_i = \frac{\mathrm{fit}(i) - \mathrm{fit}_{\mathrm{worst}}}{\mathrm{fit}_{\mathrm{best}} - \mathrm{fit}_{\mathrm{worst}}}; \quad i = 1, 2, \ldots, N, \tag{5}$$

where $\mathrm{fit}_{\mathrm{best}}$ and $\mathrm{fit}_{\mathrm{worst}}$ are the best and the worst fitness of all the particles; fit(i) represents the fitness of particle i. The separation distance (r_{ij}) between two charged particles is defined as

$$r_{ij} = \frac{\left\| X_i - X_j \right\|}{\left\| \frac{(X_i + X_j)}{2} - X_{\mathrm{best}} \right\| + \varepsilon}, \tag{6}$$

where X_i and X_j are the positions of the i-th and j-th CPs, respectively; X_{best} is the position of the best current CP, and ε is a small positive number to avoid singularities.

Step 2 CP ranking. Evaluate the values of the fitness function for the CPs, compare, and sort them in an increasing order.

Step 3 Charged memory (CM) creation. Store the number of the first CPs equal to the charged memory size (CMs) and their related values of the fitness functions in the (CM).

Level 2: search.

Step 1 Attracting force determination. Determine the probability of moving each CP toward the others considering the following probability function:

$$p_{ij} = \begin{cases} 1 & \frac{\mathrm{fit}(i) - \mathrm{fit}_{\mathrm{best}}}{\mathrm{fit}(j) - \mathrm{fit}(i)} > \mathrm{rand} \vee \mathrm{fit}(i) > \mathrm{fit}(j) \\ 0 & \mathrm{else} \end{cases}, \tag{7}$$

and calculate the attracting force vector for each CP as follows:

$$\mathrm{F_{ij}} = \mathrm{q_j} \sum_{i,i \neq j} \left(\frac{q_i}{a^3} r_{ij} \times i_1 + \frac{q_i}{r_{ij}^2} \times i_2 \right) p_{ij} (X_i - X_j) \times \begin{cases} j = 1, 2, \ldots, N \\ i_1 = 1, i_2 = 0 \leftrightarrow r_{ij} < a \\ i_1 = 0, i_2 = 1 \leftrightarrow r_{ij} \geq a \end{cases}, \tag{8}$$

where F_j is the resultant force affecting the j_{th} CP, and a is the acceleration of the CP after absorption that can be obtained from Eq. (9).

$$V = \frac{r_{\mathrm{new}} - r_{\mathrm{old}}}{\Delta t} \qquad a = \frac{v_{\mathrm{new}} - v_{\mathrm{old}}}{\Delta t} \tag{9}$$

where r_{old} and r_{new} are the initial and final positions of the particle, respectively, V is the velocity of the particle, a is the acceleration of the particle, and Δt is the time step. Combining the above equations and using the Newton's second law, the displacement of any object as a function of time is obtained as [21]:

$$r_{\mathrm{new}} = \frac{1}{2} \frac{F}{M} \times \Delta t^2 + v_{\mathrm{old}} + r_{\mathrm{old}} \tag{10}$$

Step 2 Solution construction. Move each CP to the new position and find its velocity using the following equations:

$$X_{j,\mathrm{new}} = \mathrm{rand}_{j1} \times k_{\mathrm{a}} \times \frac{F_j}{m_j} \times t_2 + \mathrm{rand}_{j2} \times k_{\mathrm{v}} \times V_{j,\mathrm{old}} \times t + X_{j,\mathrm{old}} \tag{11}$$

$$\mathrm{V_{j,new}} = \frac{X_{j,new} - X_{j,old}}{\Delta t}, \tag{12}$$

where rand_{j1} and rand_{j2} are two random numbers uniformly distributed in the range (1, 0); m_j is the mass of the CPs, which is equal to q_j in this paper. Δt is the time step, and it is set to 1. k_{a} is the acceleration coefficient; k_{v} is the velocity coefficient to control the influence of the previous velocity. In this paper, k_{v} and k_{a} are taken as

$$k_{\mathrm{a}} = c_1 (1 + \mathrm{iter}/\mathrm{iter}_{\mathrm{max}}) \tag{13}$$

$$k_{\mathrm{v}} = c_2 (1 - \mathrm{iter}/\mathrm{iter}_{\mathrm{max}}), \tag{14}$$

where c_1 and c_2 are two constants to control the exploitation and exploration of the algorithm, iter is the iteration number, and $\mathrm{iter}_{\mathrm{max}}$ is the maximum number of iterations.

Step 3 CP position correction. If each CP exits from the allowable search space, correct its position.

Step 4 CP ranking. Evaluate and compare the values of the fitness function for the new CPs and sort them in an increasing order.

Step 5 CM updating. If some new CP vectors are better than the worst ones in the CM, in terms of their objective function values, include the better vectors in the CM and exclude the worst ones from the CM.

Level 3: Controlling the terminating criterion. Repeat the search level steps until a terminating criterion is satisfied (Fig. 3a).

3.2 Colliding Body Optimization

The Colliding Body Optimization (CBO) algorithm is developed based on one-dimensional collision laws [22]. Consider two moving bodies with masses of m_1, m_2 and velocities of v_1, v_2. These two bodies collide with one another. All of the following explanation about this method, including definitions and formulas, are those of [16, 22]. According to the laws of physics, the total momentum and energy of the system after and before the collision are conserved.

In CBO, each solution candidate X_i containing a number of variables [i.e., $X_i = (X_{i,\mathrm{j}})$] is considered as a colliding body (CB). The CBs are composed of two equal main groups, namely stationary and moving objects, in which the moving objects move to follow the stationary objects, and a collision occurs between pairs of objects. This is done for two purposes: (1) to improve the moving of the objects positions and (2) to push stationary objects towards better positions. The pseudo-code of the CBO algorithm can be summarized as follows:

1. The initial positions of CBs are determined with random initialization in the search space:

$$x_i^0 = x_{\min} + \mathrm{rand}(x_{\max} - x_{\min}) \quad i = 1,\ 2,\ \ldots,\ 2n, \tag{15}$$

where x_{i}^0 determines the initial value of the i-th CB, $x_{\min}$ and $x_{\max}$ are the minimum and the maximum allowable values vector for the variables, rand is a random number in the interval [0, 1], and $2n$ is the number of CBs.

2. The magnitude of the body mass for each CB is defined as

$$m_k = \frac{\frac{1}{\mathrm{fit}(k)}}{\sum_{i=1}^{n} \frac{1}{\mathrm{fit}(i)}} \quad k = 1,\ 2,\ \ldots,\ 2n, \tag{16}$$

where fit(i) represents the fitness of the i-th agent, and $2n$ is the number of population size. Clearly a CB with good values has a larger mass than the bad ones.

3. The arrangement of the CBs fitness values is performed in an ascending order. The sorted CBs are divided into two equal groups.
The lower half of CBs are stationary bodies. These CBs are good agents and velocity of these bodies before collision is zero. Thus

$$v_i = 0 \quad i = 1,\ \ldots,\ n \tag{17}$$

The upper half of the CBs are moving bodies, which move toward the lower half. The better and worse CBs, i.e., bodies with upper and lower fitness values of each group will collide together. The velocity of these bodies before collision is

$$v_i = x_i - x_{i-n} \quad i = n\ + 1,\ \ldots,\ 2n, \tag{18}$$

where x_i is position vector of the i-th CB in this group and x_{i-n} is the i-th CB pair position of x_i in the previous group.

4. After the collision, the velocity of bodies in each group is calculated using aforementioned equations. The velocity of moving CBs after the collision is

$$v_i' = \frac{(m_i - \varepsilon m_{i-n})v_i}{m_i + m_{i-n}} \quad i = n + 1,\ \ldots,\ 2n, \tag{19}$$

where m_i is the mass of the i-th CB and m_{i-n} is the mass of the i-thCB pair. Also, the velocity of stationary CBs after the collision is

$$v_i' = \frac{(m_{i+n} + \varepsilon m_{i+n})v_{i+n}}{m_i + m_{i+n}} \quad i = n + 1,\ \ldots,\ n, \tag{20}$$

where m_i is the mass of the i-th CB, m_{i+n} is the mass of the i-th moving CB pair, and ε is COR and is defined as the ratio of the separation velocity of two bodies after collision to approach velocity of the two bodies before collision. In this algorithm, the index is defined to control of the exploration and exploitation rates. For this purpose, the COR decreases linearly from unit value to zero. Thus, ε is defined as

$$\varepsilon = 1 - \frac{\mathrm{iter}}{\mathrm{iter}_{\max}}, \tag{21}$$

where iter is the current iteration number and $\mathrm{iter}_{\max}$ is the maximum number of iterations. COR equal to unit and zero represent the global and local search, respectively. In this way a good balance between the global and local search is achieved by increasing the iteration.

5. The new positions of CBs are obtained using the generated velocities after the collision in position of stationary CBs. The new positions of the moving CBs are

$$x_i^{\mathrm{new}} = x_{i-n} + \mathrm{rand}^\circ v_i' \quad i = n + 1,\ \ldots,\ 2n, \tag{22}$$

where x_i^{new} and v'_i are new position and the velocity after the collision of the i-th moving CB, respectively, and x_{i-n} is the old position of the i-th stationary CB pair. Also, the new position of each stationary CB is

$$\mathrm{x}_i^{new} = \mathrm{x}_{i-n} + \mathrm{rand}^\circ v_{\mathrm{i}}' \quad \mathrm{i} = \mathrm{n} + 1, \ldots, \mathrm{n}, \tag{23}$$

where x_j^{new}, x_i, and v'_i are the new position, old position, and the velocity after the collision of the i-th stationary CB, respectively. Here rand is a random vector uniformly distributed in the range (−1, 1) and the sign ° denotes an element-by-element multiplication.

6. The optimization is repeated from Step 2 until a termination criterion, specified as the maximum number of iterations, is fulfilled (Fig. 3b).

4 Model Application and Discussion of the Results

Two case studies are chosen for verification and to show the effectiveness of the proposed RCPSP and FRCPSP models using CSS and CBO. The first case study is a simple project, which is adapted from Kolisch and Sprecher (PSPLIB) [10] for RCPSP model verification and the second one is a simplified real construction project for demonstration of FRCPSP model application. The algorithms are coded in MATLAB R2013a language and the experiment are performed on a personal computer with Intel® Core™ 2 Duo CPU with 4 GB RAM under the windows 7 Ultimate 64-bit operating system. The detailed case studies and the results are as follows:

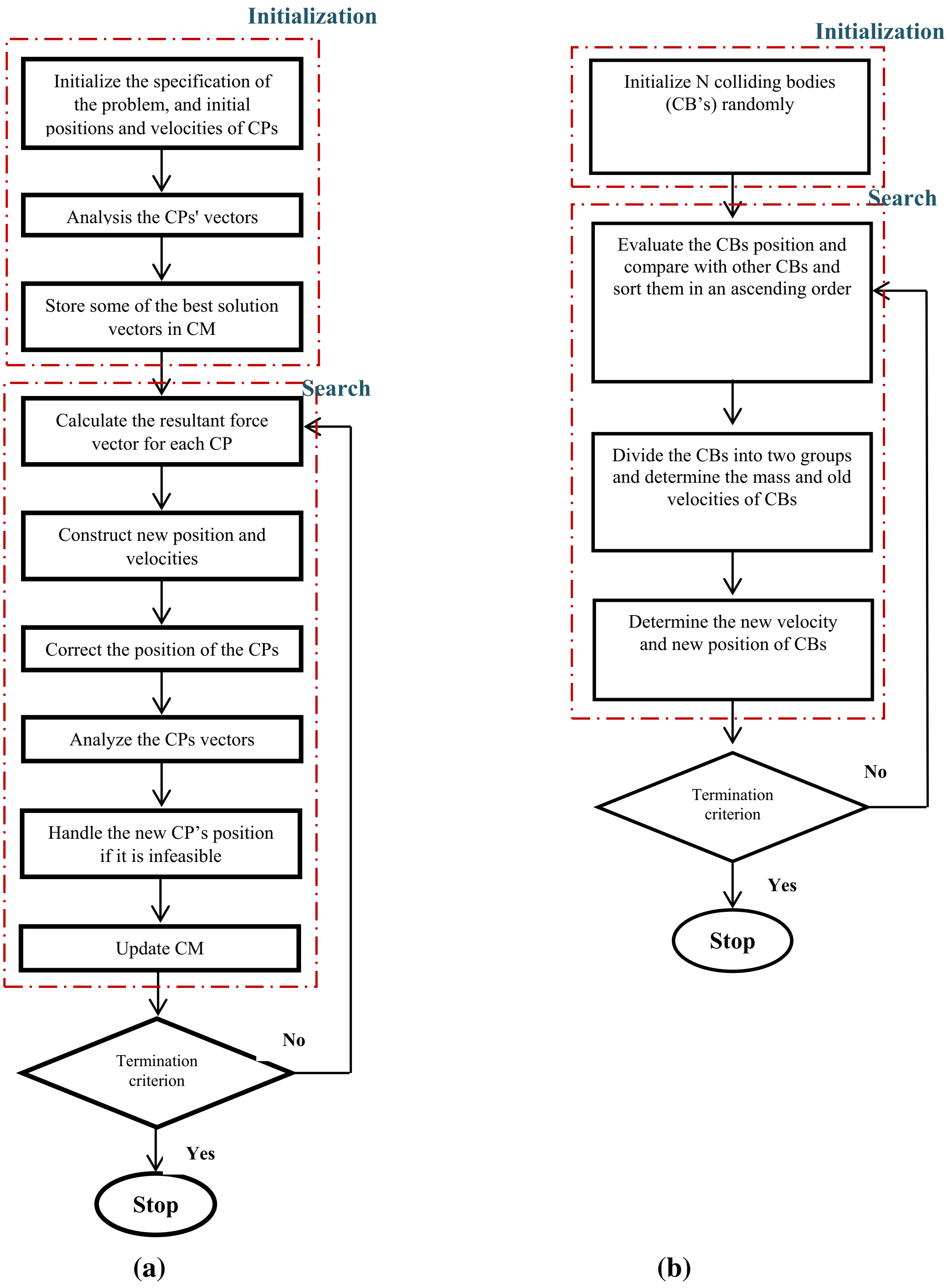

Initialization
Initialize the specification of the problem, and initial positions and velocities of CPs
Analysis the CPs' vectors
Store some of the best solution vectors in CM
Search
Calculate the resultant force vector for each CP
Construct new position and velocities
Correct the position of the CPs
Analyze the CPs vectors
Handle the new CP's position if it is infeasible
Update CM
Termination criterion
No
Yes
Stop
(a)
Initialization
Initialize N colliding bodies (CB's) randomly
Search
Evaluate the CBs position and compare with other CBs and sort them in an ascending order
Divide the CBs into two groups and determine the mass and old velocities of CBs
Determine the new velocity and new position of CBs
Termination criterion
No
Yes
Stop
(b)

Fig. 3 **a** The flowchart of the CSS algorithm [21]; **b** the flowchart of the CBO algorithm [22]

4.1 Case Study 1: Verification of the RCPSP Model

The network of this project is shown in Fig. 4 and the information of the activities including durations, resource requirements in each type of resources, and predecessors are given in Table 1. In this case study, there are four renewable resources and their availabilities are 12, 13, 4, 12, respectively. Due date of the project is 38 days.

As mentioned in the problem formulation section, there is a main objective function stated by Eq. (1) that will model the search space. In this case example, the problem is solved by exact method, and the entire search space is checked. The result of the examination illustrates that the best solution for this case considering both objective functions is 43 days. The schedule of the best solution is shown in Fig. 5.

In the present models, the number of population size is considered as 200 and the number of iteration is set to 50. The CSS model obtained the best solution in 0.2 s and the CBO model obtained this result in 0.1 s. The process of optimization is shown in Fig. 6.

In this figure, it is shown that the CBO model could find the best solution in the 10th iteration and the SCC model has found it in the 19th iteration. Also the total time needed for finding the best solution in the CBO method is about half of that of the CSS approach, although both methods have found the best solutions.

The results of the case example 1 are compared in Table 2. As can be seen in this table, all models can find the best solution; however, the CPU time needed is different from one model to another. Although in small type of example the CPU time is not very important, in the large- and very large-scale projects the CPU time has a meaningful impact on selecting a model.

4.2 Case Study 2: FRCPSP

This case is a small housing construction project consisting of 27 activities. The case is used to show the application of the proposed models in a real environment considering risk and uncertainties using fuzzy logic. The problem is modified according to the model structure. Activity details of the project are shown in the Table 3. In this case example there is one renewable resource and its availability is six persons per day. The purpose of this case example is to solve the RCPSP using CSS and CBO and make a comparison between models. In both proposed CSS and CBO algorithm of this research, the population size and number of iterations are considered 200 and 100, respectively.

The activity network of this project is shown in Fig. 7.

The algorithm is coded in MATLAB R2013a software and the experiment is performed on a personal computer with Intel® Core™ 2 Duo CPU with 4 GB RAM under the windows 7 Ultimate 64-bit operating system.

Table 4 shows the results of models according to different α-cut. Also it is important to mention that the CBO model finds the best solution faster than the CSS model and genetic algorithm.

Table 4 shows that the CBO and Genetic algorithm could find the best solution for this problem; however, CSS result is also close to the optimal solution. Moreover, the CBO model needed a less computational time compared to the CSS model and Genetic algorithm. Figure 8 shows the fuzzy project makespan in CBO and CSS models.

4.3 Defuzzification

Final step of the proposed FRCPSP is defuzzification. Since the project makespan obtained from this research is a fuzzy number, to use this number in real environment, it

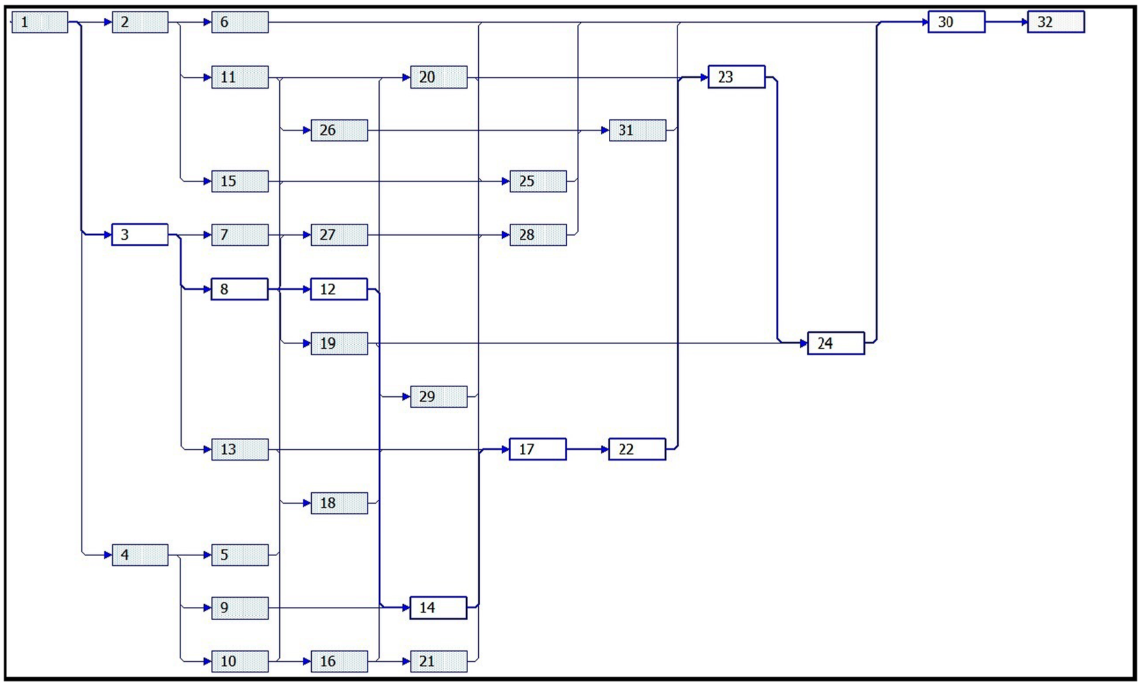


Fig. 4 Activity network of project instance adapted from Kolisch and Sprecher [10]

Table 1 Activity data of case study 1 adapted from Kolisch and Sprecher [10]

Act ID	Duration (days)	Predecessors	Resource requirement Type 1	Resource requirement Type 2	Resource requirement Type 3	Resource requirement Type 4
1	0	–	–	–	–	–
2	8	1	4	–	–	–
3	4	1	10	–	–	–
4	6	1	–	–	–	3
5	3	4	3	–	–	–
6	8	2	–	–	–	8
7	5	3	4	–	–	–
8	9	3	–	1	–	–
9	2	4	6	–	–	–
10	7	4	–	–	–	1
11	9	2	–	5	–	–
12	2	8	–	7	–	–
13	6	3	4	–	–	–
14	3	9, 12	–	8	–	–
15	9	2	3	–	–	–
16	10	10	–	–	–	5
17	6	13, 14	–	–	–	8
18	5	13	–	–	–	7
19	3	8	–	1	–	–
20	7	5, 11, 18	–	10	–	–
21	2	16	–	–	–	6
22	7	16, 17, 18	2	–	–	–
23	2	20, 22	3	–	–	–
24	3	19, 23	–	9	–	–
25	3	10, 15, 20	4	–	–	–
26	7	11	–	–	4	–
27	8	7, 8	–	–	–	7
28	3	21, 27	–	8	–	–
29	7	19	–	7	–	–
30	2	6, 24, 25	–	7	–	–
31	2	26, 28	–	–	2	–
32	0	29, 30, 31	–	–	–	–

should be defuzzified. Defuzzification is a process that can transform the resulting fuzzy values into a crisp value. The center of area (COA), also called center of centroid or center of gravity, is utilized for defuzzification. The center of area method employs the following equation [23, 26]:

$$u^{\mathrm{COA}} = \frac{\int_u u \times \mu^{\mathrm{conseq}}(u)\mathrm{d}u}{\int_u \mu^{\mathrm{conseq}}(u)\mathrm{d}u} \qquad (24)$$

The center of area method is used in this research to convert the fuzzy number of project makespan into a crisp number.

Table 5 shows crisp values for project makespan in different methods.

5 Conclusion

In this study, the application of two meta-heuristic algorithms, namely charged system search (CSS) and colliding body optimization (CBO), are discussed and these are used to solve the resource-constrained project scheduling problem (RCPSP) and fuzzy resource-constrained project scheduling problem (FRCPSP). RCPSP has been one of the challenging problems among the researchers in the past decades.

To validate the models, first a case example adapted from PSPLIB [10] is employed. The results verified the effectiveness of the proposed models. The result shows that

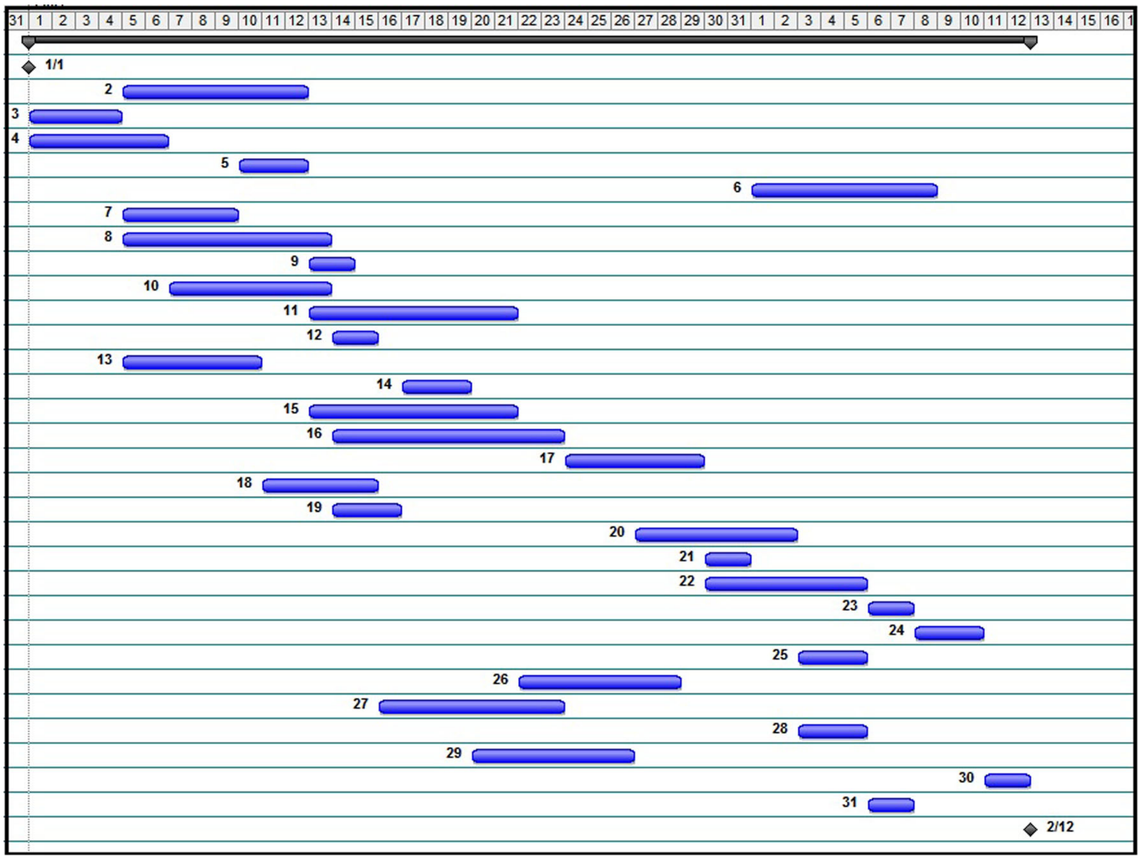


Fig. 5 Schedule of the best solution

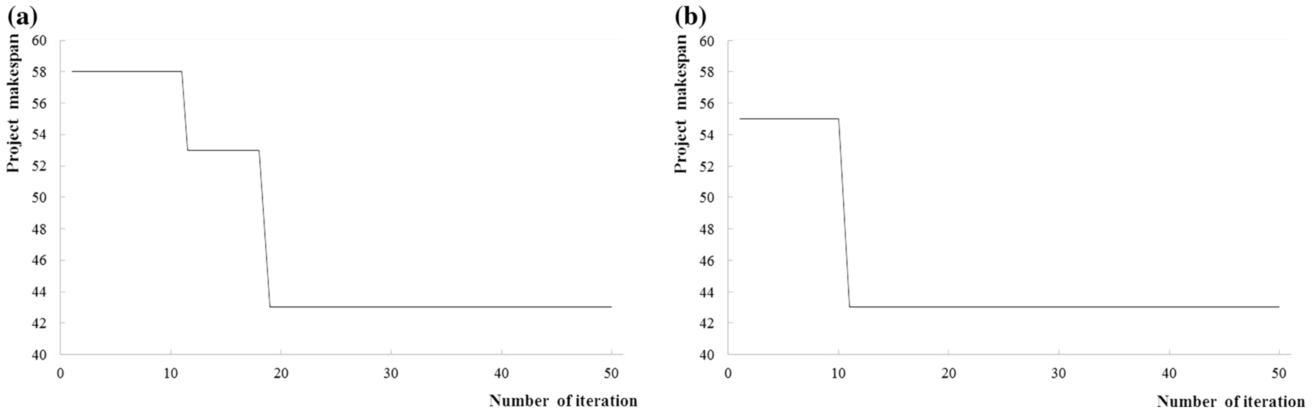


Fig. 6 Optimization process of **a** CSS model, **b** CBO model

Table 2 The average project makespan and the used CPU times by PSPLIB [10], CSS, CBO and MS project software package for the testing the problem instances

Model	Best solution for project makespan (days)	CPU time (s)
PSPLIB [10]	43	0.3
MS project software package	43	1
CSS	43	0.2
CBO	43	0.1

Table 3 Activity data of the case study 2

ID	Act description	Duration (days)			Labor requirement	Predecessor
		Minimum	Normal	Maximum		
1	Planning	22	25	31	2	
2	Testing the site soil	9	11	14	2	
3	Soil loading	12	15	18	2	2
4	Constructing the structure	8	10	14	5	
5	Foundation	9	10	13	4	3
6	Structure assembly	11	12	15	5	4,5
7	Column welding	8	10	13	4	6
8	Roof beam welding	9	10	12	5	7
9	Slab formwork	12	14	17	5	8
10	Slab casting	6	7	8	4	9
11	Construction of the wall	6	7	8	3	10
12	Door and window frame	5	7	8	4	11
13	First plastering work	8	10	12	4	11
14	Ceiling skimming	6	7	8	4	10
15	Drain work	9	10	12	4	13
16	Apron slab casting	8	9	11	5	14
17	Door and window	6	7	9	5	12,15,16
18	Final plastering	12	14	16	4	17
19	Painting	14	16	19	5	18
20	Electrical final fix	5	6	8	2	19
21	Main gate installation	3	3	5	3	18
22	Mechanical installation	10	12	15	4	19
23	Building the view	4	5	7	2	11,17
24	Landscape work	8	10	11	2	23
25	Control and checking	6	7	9	1	20,22,21,23,24
26	Defect work	12	14	17	1	25
27	Project delivery	1	1	1	1	26

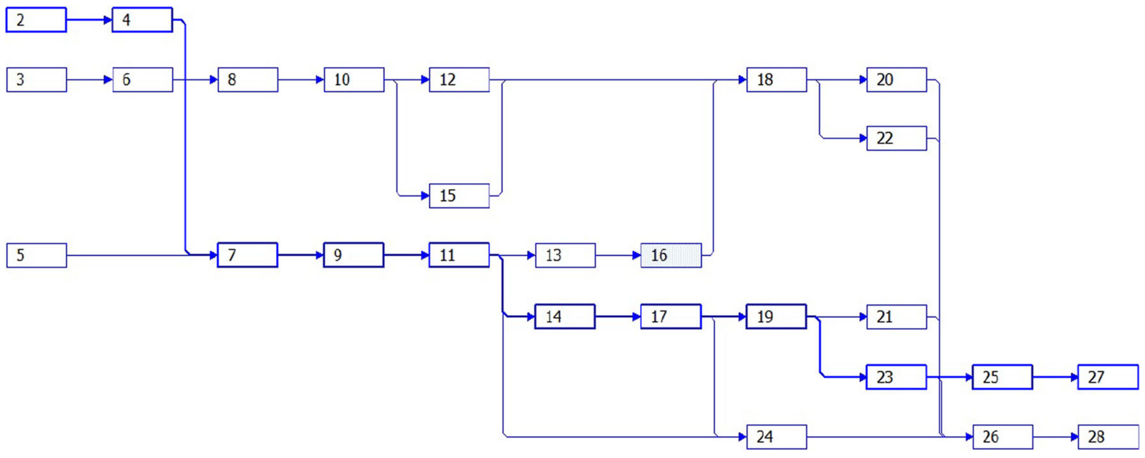
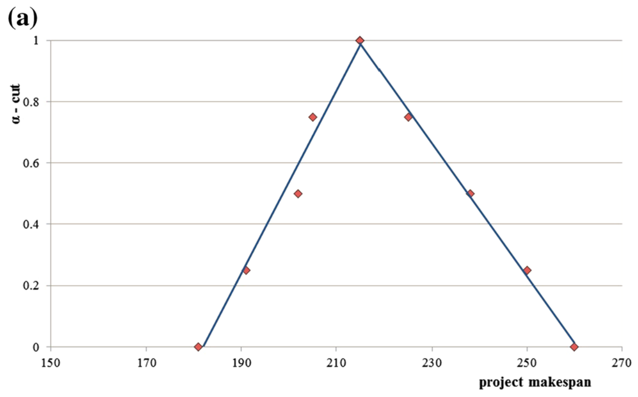


Fig. 7 Activity network of the case study 2

Table 4 Results of the case study 2

α-Cut	0	0.25	0.5	0.75	1	Computational time (s)
Optimal solution						
Minimum duration (days)	179	188	198	201	–	125
Normal duration (days)	–	–	–	–	210	
Maximum duration (days)	254	246	235	221	–	
CSS model						
Minimum duration (days)	181	191	202	205	–	15
Normal duration (days)	–	–	–	–	215	
Maximum duration (days)	260	250	238	225	–	
CBO model						
Minimum duration (days)	179	187	195	201	–	3
Normal duration (days)	–	–	–	–	210	
Maximum duration (days)	254	245	231	223	–	
Genetic algorithm						
Minimum duration (days)	179	188	196	202	–	8
Normal duration (days)	–	–	–	–	210	
Maximum duration (days)	254	246	232	221	–	



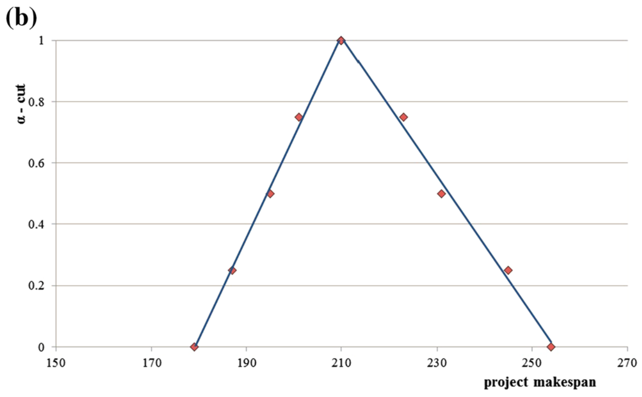


Fig. 8 Fuzzy project make span in **a** CSS and **b** CBO model

Table 5 A crisp value for project makespan in different methods

Method	Solution
Optimal solution	216.5
CSS model	219.5
CBO model	216.5
Genetic algorithm	216.5

in small projects CSS and CBO could find the best solution although CBO model can do it in a less computational time. Then for applying the model on the FRCPSP a case example is used by 27 activities and the activity duration considered as fuzzy numbers. The results of this case study show that the CBO model and Genetic algorithm obtain better solutions in comparison to the CSS model. Also the proposed CBO model can find the best solution in a faster process, in comparison to the genetic algorithm and CSS model.

In summary, findings also elaborate that both proposed metahuristics in the considered problems are capable of solving the RCPSP and FRCPSP.

References

1. Project Management Institute (2013) A guide to the project management body of knowledge (PMBOK), 5th edn. Four Campus Boulevard, Newtown Square, USA
2. Kolisch R, Padman R (2001) An integrated survey of deterministic project scheduling. Omega 29:249–272
3. Blazewicz J, Lenstra J, Rinnooy Kan A (1983) Scheduling subject to resource constraints: classification and complexity. Discret Appl Math 5:11–24
4. Kolisch R, Drexl A (1997) Local search for non-preemptive multi-mode resource constrained project scheduling. IIE Trans 29:987–999
5. Hindelang TJ, Muth JF (1979) A dynamic programming algorithm for decision CPM networks. Oper Res. 27(2):225–241

6. Patterson JH, Harvey RT (1979) An implicit enumeration algorithm for the time/cost tradeoff problem in project network analysis. Control Eng 4(2):107–117
7. Hartmann S, Drexl A (1998) Project scheduling with multiple modes: a comparison of exact algorithms. Networks. 32(4):283–297
8. Akkan C (1998) A Lagrangian heuristic for the discrete time–cost tradeoff problem for activity-on-arc project networks, in Working Paper, Koc University, Istanbul
9. Boctor F (1993) Heuristics for scheduling projects with resource restrictions and several resource-duration modes. Int J Prod Res. 31(11):2547–2558
10. Kolisch R, Sprecher A (1996) PSPLIB—a project scheduling problem library. Eur J Oper Res 96:205–216
11. Hartmann S (2001) Project scheduling with multiple modes: a genetic algorithm. Ann Oper Res 102:111–135
12. Chen PH, Weng H (2008) A two-phase GA model for resource-constrained project scheduling. Autom Constr 18:485–498
13. Alcaraz J, Maroto C, Ruiz R (2003) Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms. J Oper Res Soc 54(6):614–626
14. Liu SX, Wang MG, Tang LX (2000) Genetic algorithm for the discrete time/cost trade-off problem in project network. J Northeast Univ (China) 21(3):257–259
15. Wuliang P, Chengen W (2008) A multi-mode resource-constrained discrete time–cost tradeoff problem and its genetic algorithm based solution. Int J Proj Manag 27(6):600–609
16. Kaveh A, Khanzadi M, Alipour M, Naraki MR (2015) CBO and CSS algorithms for resource allocation and time-cost trade-off. Period Polytech Civ Eng 59(3):361–371. doi:10.3311/PPci.7788
17. Milene B, Carvalho PY, Ekel Carlos APS, Martins Joel G, Pereira Jr (2005) Fuzzy set-based multi-objective allocation of resources: solution algorithms and applications. Nonlinear Anal: Theory, Methods & Applications 63:e715–e724. doi:10.1016/j.na.2005.02.053
18. Wang K-J, Lin Y-S (2007) Resource allocation by genetic algorithm with fuzzy inference. Expert Syst Appl 33:1025–1035
19. Zhang H, Xing F (2010) Fuzzy-multi-objective particle swarm optimization for time–cost–quality tradeoff in construction. Autom Constr 19:1067–1075
20. Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech. 213(3–4):267–286
21. Kaveh A, Mahdavi VR (2014) Colliding bodies optimization: a novel meta-heuristic method. Comput Struct 139:18–27
22. Kaveh A (2014) Advances in metaheuristic algorithms for optimal design of structures. Springer, Switzerland
23. Zimmermann HJ (2001) Fuzzy set theory and its application, 4th edn. Kluwer Academic Publishers, Boston
24. Dong WM, Wong FS (1987) Fuzzy weighted averages and implementation of the extension principle. J Fuzzy Sets Syst 21:183–199
25. Zadeh LA (1975) The concept of a linguistic of a variable and its application to approximate reasoning. Inf Sci 8:199–249
26. Nasirzadeh F, Khanzadi M, Alipour M (2014) Determination of concession period in build-operate-transfer projects using fuzzy logic. Iran J Manag Stud. 7(2):423–442
27. Nasirzadeh F, Afshar A, Khanzadi M (2008) System dynamics approach for construction risk analysis. Int J Civ Eng IUST. 6(2):120–131
28. Kaveh A, Nikaeen M (2013) Optimum design of irregular grillage systems using CSS and ECSS algorithms with different boundary conditions. Int J Civ Eng Trans A Civ Eng. 11(3):143–153
29. Kaveh A, Safari H (2014) Hybrid-enhanced charged system search for solving travelling salesman problem and one of its applications: the single-row facility layout problem. Int J Civ Eng. 12(3):363–370
30. Kaveh A, Maniat M (2014) Damage detection in skeletal structures based on charged system search optimization using incomplete modal data. Int J Civ Eng IUST. 12(2):291–298