Periodica Polytechnica
Civil Engineering

59(3), pp. 361–371, 2015
DOI: 10.3311/PPci.7788

RESEARCH ARTICLE

CBO and CSS Algorithms for Resource Allocation and Time-Cost Trade-Off

Ali Kaveh, *Mostafa* Khanzadi, *Majid* Alipour, *Mohammad* Rajabi Naraky

Received 31-10-2014, accepted 26-03-2015

Abstract

Resource allocation project scheduling problem (RCPSP) has been one of the challenging subjects among researchers in the last decades. Though several methods have been adopted to solve this problem, 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 this problem known as charged system search (CSS) and colliding body optimization (CBO). The results show that both of these algorithms find reasonable solutions, however CBO could find the result in a less computational time having a better quality. Two case studies are conducted to evaluate the performance and applicability of the proposed algorithms.

Keywords

resource allocation · time-cost trade-off · optimization · CBO · CSS

Ali Kaveh
Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran, P.O. Box 16846-13114, Iran
e-mail: alikaveh@iust.ac.ir

Mostafa Khanzadi
Majid Alipour
Mohammad Rajabi Naraky
School of Civil Engineering, Iran University of Science and Technology, Narmak, Tehran, P.O. Box 16846-13114, Iran

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 formed on the basis of precedence constraints. Also each activity of the project can be included in several modes for execution. Each mode has a different effect on the duration of the activity and its resource requirement [2].

Various quantitative types of methods to project management have been suggested since the 1950s. In the first type of methods, such as CPM and PERT, the durations of the activities were the only variables and the resource availability which could affect resource allocation and the entire project scheduling, was not considered so this is one of the major limitations of these methods [3, 4] Therefore, many researchers focused on techniques and optimization methods for project scheduling. The results of these studies in the literature can be classified in four categories: resource constraint scheduling, time cost trade-off, resource leveling and resource allocation [5]

In the case of resource constraint project scheduling problem (RCPSP), the purpose is to minimize the project construction time, considering that each activity must be scheduled according to resource constraints and precedence relationships between activities [6, 7]. The RCPSP is strongly NP-hard [8] and several searching methods including exact methods [9–11] (as dynamic programming, enumeration algorithm, branch and bound algorithms), heuristic [12–14] (as Lagrangian heuristic) and metaheuristic [2, 4, 15–17] (as genetic algorithm, simulated annealing, particle swarm optimization, ant colony algorithm) procedures have been suggested to solve this problem with many different assumptions.

A generalization of the RCPSP is the multi-mode resource-constrained project scheduling problem (MRCPSP) where several performing modes are considered for each activity. In this problem, three basic categories of resources (i.e. renewable, nonrenewable, and doubly constrained ones) are considered. The temporary availability of a renewable resource is constrained at every moment of the planning horizon (as labor, machinery, equipment, etc.). The integral availability of a nonrenewable resource is restricted for the entire project or a specific time interval (as budget). In addition, for a doubly constrained resource, the availability is limited both for the entire project and at every moment. An activity can either consume (as money) or use (as blades) this kind of resources during its accomplishment [3]

Since MRCPSP is a generalization of the RCPSP, this is also NP-hard. Moreover, in the case of more than one nonrenewable resource, the problem of finding a feasible solution (schedule) is already NP-complete [18] Consequently, in large and highly resource-constrained problems, none of the exact algorithms is able to find the best solution in a reasonable time [19]

Among the RCPSPs, the discrete time-cost tradeoff problem (DTCTP) is a well-known problem where the processing time of an activity is a discrete, non-increasing function of the amount of a single nonrenewable resource allocated to this activity. Therefore, according to all possible resource allocations, each activity can be performed in several modes [3] This means that in this case, both the time–cost tradeoff and resource constraint project scheduling problems should be considered simultaneously.

The DTCTP has three sub-problems in which the process is to select activities execution modes depending on following objectives [20] The first is to minimize the project total cost while meeting a given project deadline (deadline problem), the second is minimizing the project total duration for a given non-negative budget (budget problem) and the third is to construct the complete and efficient time–cost profile over the feasible project durations (time-cost curve problem). Though the researchers have studied the DTCTP for many years, however, there are still some defects in considering all the aspects of project management and scheduling. Most of DTCTP researchers had more attention to the nonrenewable resources than renewable ones and the importance of the renewable resources have not been considered adequately [17]. However, total time and cost of real construction projects are affected by many various kinds of renewable resources such as manpower, machines, equipment and etc. [5]

In this paper, a multi-mode resource constrained discrete time-cost tradeoff model (MRC-DTCTP) is developed that considers MRCPSP, DTCTP and resource allocation simultaneously based on recent researches. The main goal is 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 [21] and Colliding Body Optimization (CBO) developed by Kaveh and Mahdavi [22,23] are the methods. Then two case studies have been conducted to evaluate the performance and applicability of the proposed algorithms.

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

2 Problem formulation

2.1 Proposed MRC-DTCTP model

The problem studied in this paper is a multi-mode resource-constrained discrete time–cost tradeoff problem which 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 number of activities in the project network is from 0 to $J + 1$, where activities 0 and $J + 1$ are dummy activities that belong to the start and the end of the project. 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. According to MRCPSP and DTCTP models, each activity $j \in V$ may be executed in one of several different modes of accomplishment given by the set $M_j = 1; \ldots; M_j$. Activity j performed in mode $m \in M_j$, requires r_{jmk} renewable resource $k \in \left\{1, \ldots, M_j\right\}$ for each period of execution, and c_{jm} is the direct cost for the execution in the related mode. The time that activity j is executed in mode m, d_{jm}, is supposed to be a discrete and non-increasing function of both, the amount of resource allocated to it, and the direct cost of executing the activity. When the activity j starts its execution in mode m, any interruption, such as changing the mode is not allowable and it must be continuing in d_{jm} consecutive periods. Moreover the project constrains renewable resources in periods and for each renewable resource $k \in \{1, \ldots, K\}$, its availability per period is constant and given by R_k.

This paper aims at solving MRC-DTCTP optimization model using charged system search (CSS) [21] and colliding body optimization (CBO) [22,23] algorithms introduced by Kaveh et al. (see Kaveh [24] as well). The purpose is to achieve a solution with the minimum total time and cost, considering precedence relations between different activities and resource constrains in one project. The objective functions of the MRC-DTCTP model are formulated to minimize the total project time and cost along with allocation of resources in the entire project makespan, simultaneously.

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

1 The first objective of our MRC-DTCTP model is to minimize duration of the project, which is the finish time of last activity f_j in a project. Therefore the total project duration F_t is:

$$F_t = f_J \tag{1}$$

2 The second objective of our MRC-DTCTP model is to minimize the total project cost. In general, the cost of a project can be divided into two parts: direct cost and indirect cost. The costs directly related to the execution of activities in the project are direct costs. These are mainly dependant on the amount of renewable resources occupied by the activities. It means that for one specific activity, execution mode with a more renewable resource requirement has usually a greater direct cost with a shorter duration. The other part of the project cost is called an indirect cost because it can be related to no execution of activity in the project and it is paid by functional department. In this model, we assume that an indirect cost is a fixed amount in each period of construction time and its amount varies with project duration for the entire project. The direct cost for the entire project depends on the modes that have been selected for the activities and its amount is the sum of activities execution costs. Therefore, the project cost F_c can be formulated as follows [17]:

$$F_C = \sum_{j} \sum_{m \in Mj} \left(x_{jm} \times c_{jm}\right) + f_j \times c_i + y_j \times c_p \times \left(f_j - T_{contract}\right) \quad (2)$$

In this relationship, the first term is the project direct cost $(\sum_{j} \sum_{m \in M_j} (x_{jm} \times c_{jm}))$, where c_{jm} is the direct cost of activity j when executes in mode m. In addition, x_{jm} is a decision variable:

$$x_{jm} = \begin{cases} 1 & \text{if activity } j \text{ executed in mode } m \\ 0 & \text{otherwise} \end{cases}$$

The second term is the project indirect cost $\left(f_j \times c_i\right)$, where c_i is a fixed amount which is considered as indirect cost per period in the project makespan.

The last term in the formula is for considering penalty when the project duration is longer than the project makespan in the contract $\left(y_j \times c_p \times \left(f_j - T_{contract}\right)\right)$. $T_{contract}$ is the project deadline that is mentioned in the project contract and c_p is a penalty in each period of delay. y_j also is a decision variable:

$$y_j = \begin{cases} 1 & f_j > Tcontract \\ 0 & f_j \leq Tcontract \end{cases}$$

2.2 Mathematical model of MRC-DTCTP

According to the method of calculation utilized for project cost and the time mentioned in the former section, the model of MRC-DTCTP is built based on MRCPSP and DTCTP. The MRC-DTCTP has three sub-problems similar to DTCTP: Eq. (1) the deadline problem, minimizing total cost considering project deadline; Eq. (2) the budget problem, minimizing the makespan considering a given non-negative budget; and Eq. (3) the time–cost curve problem, to generate the complete time–cost trade-off profile for a project with constrained resource and discrete time–cost relationship. In this research, normalization is used for multi-objective optimization. In addition, an importance factor for each objective is introduced to enable the decision-maker to control the effect of each objective on the final solution.

The model of the MRC-DTCTP is formulated as follows [17]:

$$minF_t \quad (3)$$

$$minF_c \quad (4)$$

subject to

$$\sum_{m \in Mj} x_{jm} = 1 \quad j \in V \quad (5)$$

$$f_j - \sum_{m \in Mj} (x_{jm} \cdot d_{jm}) \geq f_i \quad \forall (i, j) \in E \quad (6)$$

$$\begin{aligned} &\sum_{j \in A_t} \sum_{m \in M_j} \left(x_{jm} \times r_{jmk}\right) \leq R_k \quad k = 1, \ldots, K, \\ &A_t = \left\{ j | f_j - d_j < t \leq f_j \right\} \end{aligned} \quad (7)$$

In the above formulation, the objective function Eq. (3) minimizes the project time, which is calculated by Eq. (1), and the objective function Eq. (4) minimizes the project cost, which is calculated by Eq. (2). Constraint set Eq. (5) requires every activity to be executed in only one mode. Constraint set Eq. (6) represents the precedence relationships, where d_{jm} is the duration of an activity j when activity j is executed in mode m. Finally Constraint set Eq. (7) 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_{jmk} is the amount of resource k required by activity j if it is executed in mode m.

3 Metahuristic algorithms

The main purpose of this paper is to optimize time–cost tradeoff, which is formulated as a multi-objective optimization problem and search for solutions that minimize the total duration and the total cost simultaneously. In the multi-objective problems, often some of the criteria are in conflict with each other, i.e. for an improvement in one objective, another objective must be sacrificed. Because of this, we used an importance factor for each objective that specify preferences among the objectives.

To search for solutions, two meta-heuristic algorithms (Charged System Search (CSS) and Colliding Body Optimization (CBO)) are designed for implementing the multiobjective optimization. The CSS [21] and CBO [22, 23], developed by Kaveh and Talatahari, and Kaveh and Mahdavi, respectively, are two efficient methods that have not been used for this problem up to now [24].

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. All of the following explanation about this method, including definitions and formulas, are extracted from Kaveh and Talatahari [21].

The Coulomb and Gauss laws provide the magnitude of the electric field $\left(E_{ij}\right)$ at a point inside and outside a charged insulating solid sphere, respectively, as follows [21]:

$$E_{ij} = \begin{cases} \frac{k_e q_i}{a^3} r_{ij} & if \quad r_{ij} < a \\ \frac{k_e q_i}{r_{ij}^2} & if \quad r_{ij} \geq a \end{cases} \tag{8}$$

where k_e is the Coulomb constant, r_{ij} is the separation of the center of sphere and the selected point, q_i is the magnitude of the charge; and a is the radius of the charged sphere. Using the principle of superposition, the resulting electric force due to N charged spheres $\left(F_j\right)$ is as follows [21]:

$$\begin{aligned} & F_j = k_{eq} \sum_{i=1}^{N} \left(\frac{q_i}{a^3} r_{ij} \cdot i_1 + \frac{q_i}{r_{ij}^2} \cdot i_2 \right) \frac{r_i - r_j}{r_i - r_j} \\ & \begin{cases} i_1 = 1, \quad i_2 = 0 \leftrightarrow r_{ij} < a \\ i_1 = 0, \quad i_2 = 1 \leftrightarrow r_{ij} \geq a \end{cases} \end{aligned} \tag{9}$$

Also according to the Newtonian mechanics, we have [21]:

$$\begin{aligned} \Delta r &= r_{new} - r_{old} \\ V &= \frac{r_{new} - r_{old}}{\Delta t} \\ a &= \frac{v_{new} - v_{old}}{\Delta t} \end{aligned} \tag{10}$$

where r_{old} and r_{new} are the initial and final positions of the particle, respectively, v is the velocity of the particle; and a is the acceleration of the particle. 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]:

$$r_{new} = \frac{1}{2} \frac{F}{M} \cdot \Delta t^2 + v_{old} + r_{old} \tag{11}$$

In the CSS method, each solution is considered as a charged particle (CP) in an n-dimensional space, which n is the number of decision variables. The convergence process is carried out through the movements of these particles in the search space. The fore-mentioned electrostatics and mechanics laws govern the forces between these CPs and their movements. The pseudo-code of the CSS algorithm can be 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 CPs are considered as zero. Each CP has a charge of magnitude (q) which its value is calculated as:

$$q_i = \frac{fit\,(i) - fit_{worst}}{fit_{best} - fit_{worst}}; \quad i = 1, 2, \ldots, N \tag{12}$$

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

$$r_{ij} = \frac{\left\| X_i - X_j \right\|}{\left\| \frac{(X_i + X_j)}{2} - X_{best} \right\| + \varepsilon} \tag{13}$$

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:

$$p_{ij} = \begin{cases} 1 & \frac{fit(i) - fit_{best}}{fit(j) - fit(i)} > rand \quad \vee \quad fit(i) > fit(j) \\ 0 & else \end{cases} \tag{14}$$

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

$$\begin{aligned} & F_{ij} = q_j \sum_{i, i \neq j} \left(\frac{q_i}{a^3} r_{ij} \cdot i_1 + \frac{q_i}{r_{ij}^2} \cdot i_2 \right) p_{ij} \left(X_i - X_j\right) \\ & \begin{cases} j = 1, 2, \ldots, N \\ \quad i_1 = 1, \qquad i_2 = 0 \leftrightarrow r_{ij} < a \\ \quad i_1 = 0, \qquad i_2 = 1 \leftrightarrow r_{ij} \geq a \end{cases} \end{aligned} \tag{15}$$

where F_j is the resultant force affecting the j^{th} CP.

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

$$X_{j,new} = rand_{j1} \times k_a \times \frac{F_j}{m_j} \times t_2 + rand_{j2} \times k_v \times V_{j,old} \times t + X_{j,old} \tag{16}$$

$$V_{j,new} = \frac{X_{j,new} - X_{j,old}}{\Delta t} \tag{17}$$

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:

$$k_a = c_1 \left(1 + iter/iter_{max}\right) \tag{18}$$

$$k_v = c_2\,(1 - iter/iter_{max}) \tag{19}$$

where c_1 and c_2 are two constants to control the exploitation and exploration of the algorithm; $iter$ is the iteration number and $iter_{max}$ 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.

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 presented in Ref. [22]. According to the laws of physics, the total momentum and energy of the system after and before the collision are conserved. It can be expressed as:

$$m_1 v_1 + m_2 v_2 = m_1 v'_1 + m_2 v'_2 \tag{20}$$

and

$$\frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 = \frac{1}{2} m_1 {v'_1}^2 + \frac{1}{2} m_2 {v'_2}^2 + Q \tag{21}$$

where v_1 and v_2 are the velocities of the first and second body before collision, respectively; v'_1 and v'_2 are the velocities of the first and second body after collision, respectively; m_1 and m_2 are the masses of the first and second body, respectively; and Q is the loss of kinetic energy due to collision [22]. The velocities of two bodies after a one-dimensional collision can be obtained as:

$$v'_1 = \frac{(m_1 - \varepsilon m_2)\,v_1 + (m_2 + \varepsilon m_2)\,v_2}{m_1 + m_2} \tag{22}$$

$$v'_2 = \frac{(m_2 - \varepsilon m_1)\,v_2 + (m_1 + \varepsilon m_1)\,v_1}{m_1 + m_2}$$

where ε is coefficient of restitution (COR) of two colliding bodies, which defined as:

$$\varepsilon = \frac{|v'_2 - v'_1|}{|v_2 - v_1|} = \frac{v'}{v} \tag{23}$$

For most real objects, ε is between 0 and 1.

In the CBO method, each solution is considered as a colliding body CB that candidate Xi containing a number of variables $\left[i.e., X_i = \left(X_{i,j}\right)\right]$. 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 happened for two purposes: 1) to improve of the moving 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:

$$x_i^0 = x_{min} + rand\,(x_{max} - x_{min}) \quad i = 1, 2, \ldots, 2n \tag{24}$$

where x_i^0 determines the initial value of the $i - thCB$; 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:

$$m_k = \frac{\frac{1}{fit(k)}}{\sum_{i=1}^{n} \frac{1}{fit(i)}} \quad k = 1, 2, \ldots 2n \tag{25}$$

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 equally into two groups.

The lower half of CBs are stationary bodies. These CBs are good agents and velocity of these bodies before collision is zero. Thus:

$$v_i = 0 \quad i = 1, \ldots\ldots, n \tag{26}$$

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 as:

$$v_i = x_i - x_{i-n} \quad i = n + 1, \ldots\ldots, 2n \tag{27}$$

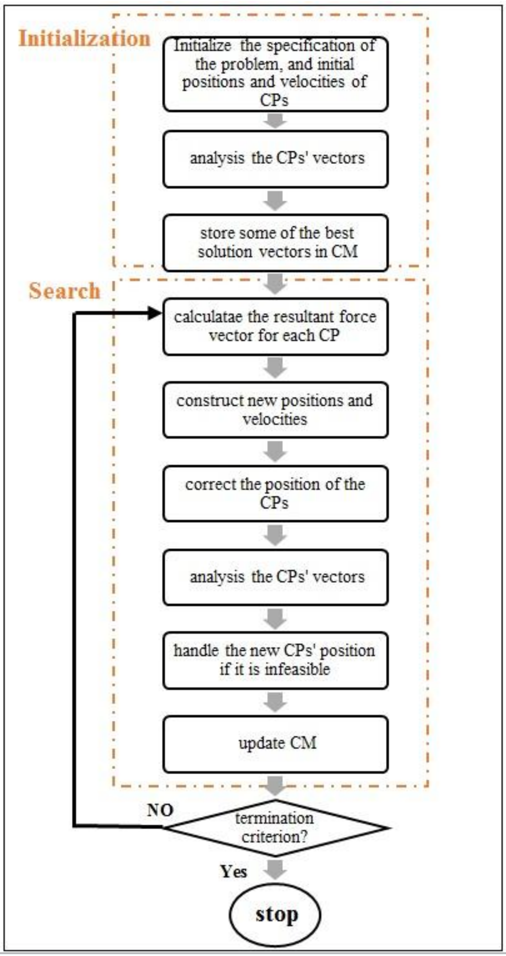
where x_i is position vector of the $i - th\ CB$ in this group and x_{i-n} is $i - th\ CB$ pair position of x_i in the previous group.

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

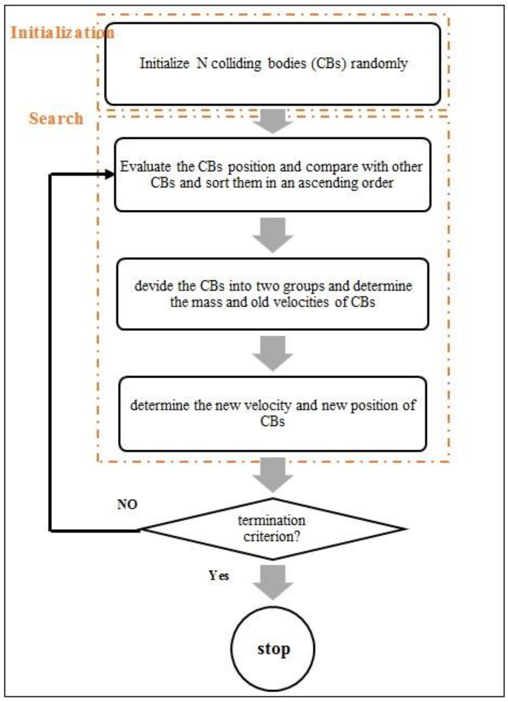
$$v'_i = \frac{(m_i - \varepsilon m_{i-n})\,v_i}{m_i + m_{i-n}} \quad i = n + 1, \ldots, 2n \tag{28}$$

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

$$v'_i = \frac{(m_{i+n} + \varepsilon m_{i+n})\,v_{i+n}}{m_i + m_{i+n}} \quad i = 1, \ldots, n \tag{29}$$



(a) The flowchart of the CSS algorithm [21];



(b) The flowchart of the CBO algorithm [22]

Fig. 1.

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 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:

$$\varepsilon = 1 - \frac{iter}{iter_{max}} \tag{30}$$

where $iter$ is the current iteration number and $iter_{max}$ is the maximum number of iterations, which 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 moving CBs is:

$$x_i^{new} = x_{i-n} + rand^{\circ} v'_i \quad i = n+1, \ldots\ldots, 2n \tag{31}$$

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:

$$x_i^{new} = x_i + rand^{\circ} v'_i \quad i = 1, \ldots\ldots, n \tag{32}$$

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 ^{o} 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.

4 Model application and discussion of the results

Two case studies have been chosen for verification and to show the effectiveness of the proposed MRC-DTCTP model using CSS and CBO. The first case study is a simple project, which is adapted from Hartmann [2] for model verification and the second one is a simplified real warehouse construction project for demonstration of model application. The algorithms have been coded in MATLAB R2013a language and the experiment has been performed on a personal computer with Intel®Core™2

Tab. 1. Activity data of case study 1 adapted from Hartmann [2]

Act ID	Execution mode	Duration (days)	Resource requirement	Direct cost ($1000)
1	1	3	2	5
	2	4	1	1
2	1	2	3	6
	2	4	3	2
3	1	2	4	2
	2	3	2	2
4	1	2	3	6
	2	2	4	4
5	1	3	3	1
	2	3	1	7
6	1	4	2	1
	2	6	1	1

Duo CPU with 4 GB RAM under the windows 7 Ultimate 32-bit operating system. The detailed case studies and the results are as follows:

4.1 Case Study 1: Model Verification

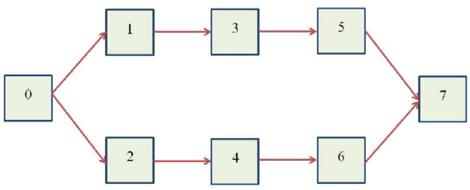


Fig. 2. Activity network of project instance adapted from Hartmann [2]

The network of this project is as in Fig. 2 and the information of the activities including number of modes, durations, resource requirements and direct costs are given in the Table 1. In this case study, there is one renewable resource and its availability is 4 per period. The indirect cost of this project is considered to be $500 per day. Also in the contract mentioned that the deadline of the project is 18 days and the contractor must pay $1000 per day for a delay.

As mentioned in the problem formulation section, there are two objective functions stated in Eqs. (1) and (2) that will form the search space {Time and Cost}. In this simple case, the problem was solved by means of complete enumeration, and entire the search space was checked and all the possible schedules with different fitnesses were compared. The result of examination illustrate that the best solution for this case considering both objective functions, is 12 days and $17000 (Fig. 3).

The presented models, with considering a population size of 200 are solved, the CSS model obtained the best solution in

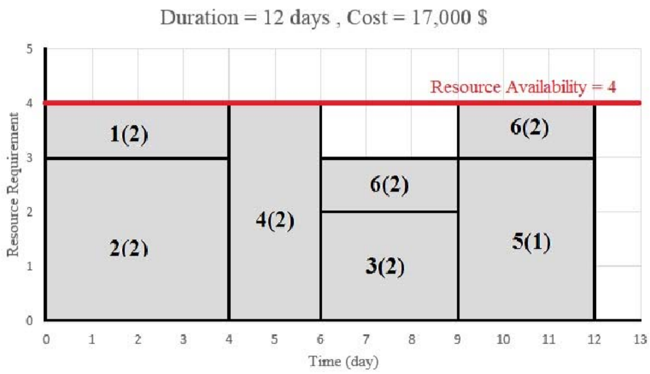


Fig. 3. Schedule of the best Solution

2.9 sec and the CBO model obtained this result in 1.5 sec. The process of optimization shown in Figs. 4 and 5.

Figs. 4 and 5 show that the CBO method has find the best solution in the 7^{th} iteration and the SCC method has find it in the 17^{th} iteration. Also total time that need for finding the best solution in the CBO method is about a half against CSS method. Although both of them could find the best solution.

4.2 Case Study 2: Real Project

This case is a simplified warehouse construction project consists of 37 activities. The case is used to demonstrate the application of the models in real environment. The problem modified according to the model requirements. Activity details of the project are shown in the Table 2. There is one renewable resource in this case and its availability is 12 labors per day. In addition, the indirect cost of the project has been considered $0 per day. The purpose of this case study is to find solutions of CSS and CBO models and make a comparison between models. In both CSS and CBO of this research, the population size and number of iteration were 400 and 100, respectively.

The model has run by CSS and CBO methods for several

Tab. 2. Activity data of case study 2

Act description	Execution mode	Duration (days)	predecessor	Labor requirement (men)	Direct cost ($1000)
Mobilization and site facilities	1	25	-	2	5
Soil test	1	11	-	2	2.2
Excavation work	1	21	1	4	8.4
	2	16		6	9.6
Piling work	1	20	1	5	10
	2	18		6	10.8
Pile loading test	1	15	2	2	3
Backfilling and M&E work	1	9	4	3	2.7
	2	6		5	3
Pile cap work	1	14	2,4	4	5.6
	2	10		6	6
Column rebar and M&E work	1	10	5	5	5
Slab casting	1	12	3,6,7	5	6
	2	11		6	6.6
Column formwork	1	10	8	4	4
Roof beam and slab formwork	1	12	9	5	6
Column casting	1	10	10	4	4
Roof beam and slab rebar	1	10	11,12	5	5
Roof parapet wall casting	1	14	12	5	7
M & E work 1	1	7	12	4	2.8
Door and window frame	1	7	14	3	2.1
M & E work 2	1	7	13,14	4	2.8
Roof slab casting	1	12	15	4	2.4
	2	9		6	5.4
Plastering work	1	10	16,17	4	4
Brick wall laying	1	14	18	4	5.6
	2	10		6	6
Ceiling skimming work	1	7	11	4	2.8
Toilet floor and wall tiling work	1	14	20	3	4.2
	2	10		5	5
Drain work	1	10	19,21	4	4
Apron slab casting	1	9	21,23	5	4.5
Door and window	1	7	22	5	3.5
Painting work	1	14	19,22	4	5.6
Fencing work	1	16	24	5	8
External wall plastering	1	10	25	4	4
	2	9		5	4.5
Electrical final fix	1	6	25	2	1.2
Main gate installation	1	3	24,27	3	0.9
External wall painting	1	12	29	4	4.8
Qualified person inspection	1	5	27,30	2	1
Landscape work	1	10	28,31	2	2
Registered inspector inspection	1	7	32,33	1	0.7
Authority inspection	1	7	34	1	0.7
Defect work	1	14	35	1	1.4
Project handover	1	1	36	1	0.1

Tab. 3. Results of the case study 2

Time Importance Factor	Cost Importance Factor	CSS Model Duration	CSS Model Cost	CBO Model Duration	CBO Model Cost
70	30	187	148.7	182	149.6
60	40	189	147.8	184	149
50	50	192	147	186	148.8
40	60	193	146.4	191	147.2
30	70	205	145.9	193	146.4

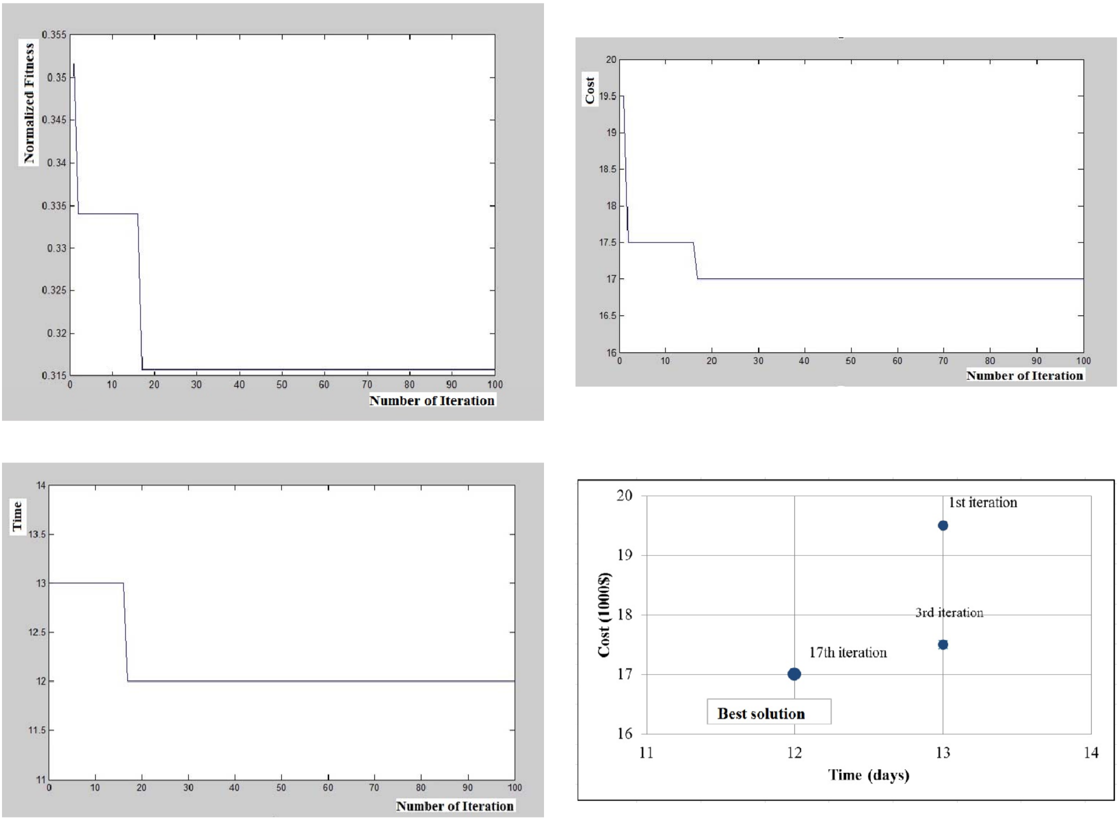


Fig. 4. Optimization process of the CSS model

time. The range of obtained time and cost was 182 - 205 and \$145,900 - \$149,600, for different time/cost ratio factor. Table 3 shows the results of models according to different ratio of time and cost. Also this is important to say that, the CBO model to find the best solution is faster than CSS model.

The Tables 2 and 3 show that when the time importance factor is bigger than the cost ones, the duration of project will be minimized and vice versa. Fig. 6 shows a Pareto front of time-cost of the project during different time/cost ratio.

5 Conclusion

In this study, the application of two meta-heuristic algorithms, namely charged system search (CSS) and colliding body optimization (CBO), are introduced to solve the multi-mode resource constrained project scheduling problem (MRCPSP), the discrete time-cost tradeoff problem (DTCTP), and the resource allocation simultaneously. These problems are well-established scheduling problems.

To validate the models, a simple project adapted from Hartmann [2] is used. The results verified the effectiveness of the models. Then w the models are tested for a larger real construction project. The solutions of this case study show that the CBO model obtains better solutions in a faster process, in comparison to the CSS model. In both case studies, it is assumed that there is no preference on project time and cost in the optimization, but an importance factor is considered for each objective function that the manager can easily make a decision according to the given preferences.

Finding also elaborates that both proposed metahuristics in the considered problems are capable of solving the MRCPSP-DTCTP.

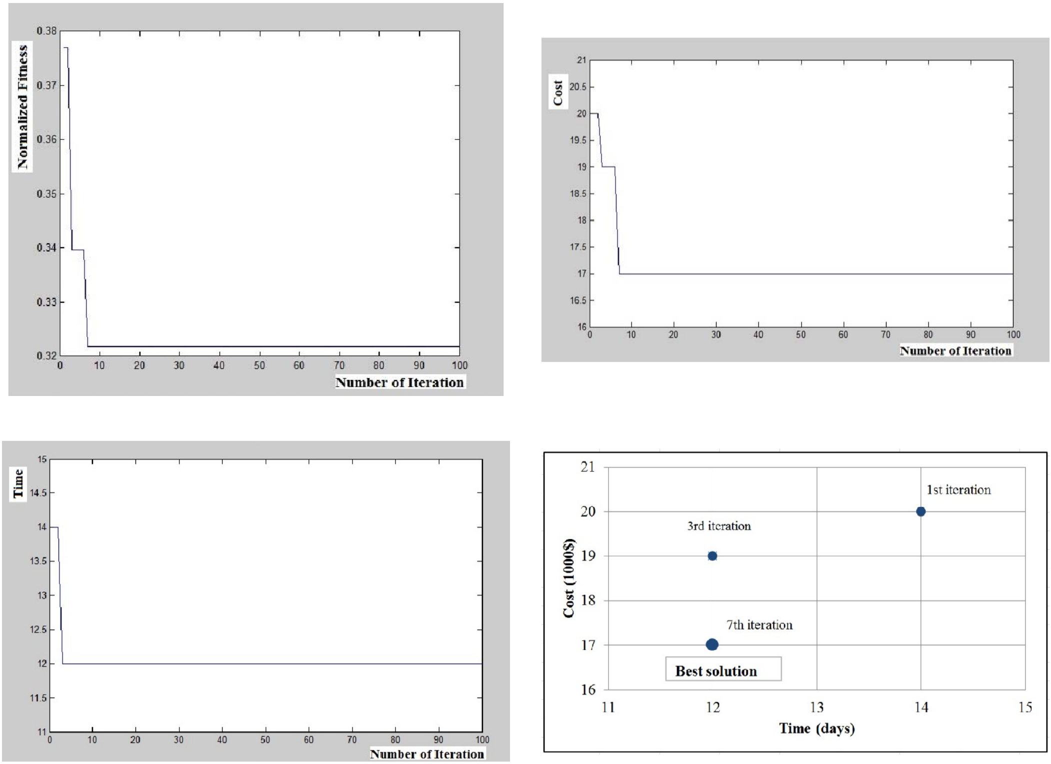


Fig. 5. Optimization process of the CBO model

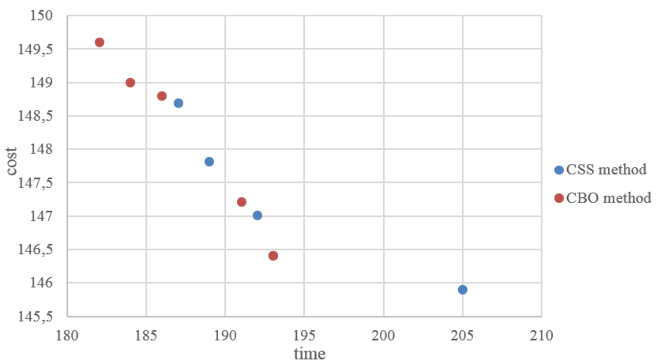


Fig. 6. Pareto front of time-cost of the project during different time/cost ratio

References

1 **PMBOK Guide**, *Project management body of knowledge*, University of Michigan Press; USA, 2013.

2 **Hartmann S**, *Project scheduling with multiple modes: a genetic algorithm*, Annals of Operations Research, **102**, (2001), 111-135, DOI 10.1023/A:1010902015091.

3 **Weglerz J, Jozefowska J, Mika M, Waligora G**, *Project scheduling with finite or infinite number of activity processing modes – A survey*, European Journal of Operational Research, **208**, (2011), 177-205, DOI 10.1016/j.ejor.2010.03.037.

4 **Chen PH, Weng H**, *A two-phase GA model for resource-constrained project Scheduling*, Automation in Construction, **18**, (2008), 485-498, DOI 10.1016/j.autcon.2008.11.003.

5 **Kolisch R, Padman R**, *An integrated survey of deterministic project scheduling*, Omega, **29**, (2001), 249-272, DOI 10.1016/S0305-0483(00)00046-3.

6 **Kolisch R, Hartmann S**, *Heuristic algorithms for solving the resource-constrained project scheduling problem: classification and computational analysis*, In: **Weglarz J** (ed.), Project Scheduling: Recent Models, Algorithms and Applications; Amsterdam, 1999, pp. 147-178, DOI 10.1007/978-1-4615-5533-9_7.

7 **Kolisch R, Hartmann S**, *Experimental investigation of heuristics for resource-constrained project scheduling: an update*, European Journal of Operational Research, **174**(1), (2006), 23-37, DOI 10.1016/j.ejor.2005.01.065.

8 **Blazewicz J, Lenstra JK, Rinnooy Kan AHG**, *Scheduling subject to resource constraints: classification and complexity*, Discrete Applied Mathematics, **5**(1), (1983), 11–24, DOI 10.1016/0166-218X(83)90012-4.

9 **Hindelang TJ, Muth JF**, *A dynamic programming algorithm for decision CPM networks*, Operations Research, **27**(2), (1979), 225-241, DOI 10.1287/opre.27.2.225.

10 **Patterson JH, Harvey RT**, *An implicit enumeration algorithm for the time/cost tradeoff problem in project network analysis*, Control Engineering, **4**(2), (1979), 107-117.

11 **Hartmann S, Drexl A**, *Project scheduling with multiple modes: a comparison of exact algorithms*, Networks, **32**(4), (1998), 283-297, DOI 10.1002/(SICI)1097-0037(199812)32:4<283::AID-NET5>3.0.CO;2-I.

12 **Akkan C**, *A Lagrangian heuristic for the discrete time–cost tradeoff problem for activity-on-arc project networks*; Koc University, Istanbul, Turkey, 1998.

13 **Boctor F**, *Heuristics for scheduling projects with resource restrictions and several resource-duration modes*, International Journal of Production Research, **31**(11), (1993), 2547-2558, DOI 10.1080/00207549308956882.

14 **Kolisch R, Drexl A**, *Local search for non-preemptive multi-mode resource-constrained project scheduling*, IIE Transactions, **29**, (1997), 987-999, DOI 10.1023/A:1018552303415.

15 **Alcaraz J, Maroto C, Ruiz R**, *Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms*, Journal of the Operational Research Society, **54**(6), (2003), 614-626, DOI 10.1057/palgrave.jors.2601563.

16 **Liu SX, Wang MG, Tang LX**, *Genetic algorithm for the discrete time/cost trade-off problem in project network*, Journal of Northeastern University [China], **21**(3), (2000), 81–86.

17 **Wuliang P, Chengen W**, *A multi-mode resource-constrained discrete time–cost tradeoff problem and its genetic algorithm based solution*, International Journal of Project Management, **27**(6), (2008), 600-609, DOI 10.1016/j.ijproman.2008.10.009.

18 **Kolisch R**, *Project Scheduling under Resource Constraints: Efficient Heuristics for Several Problem Classes*, Physisca-Verlag; Heidelberg, 1995, DOI 10.1007/978-3-642-50296-5.

19 **Sprecher A, Drexl A**, *Optimization of 3D trusses with adaptive approach in genetic algorithms*, European Journal of Operational Research, **107**(2), (1998), 431-450.

20 **Vanhoucke M, Demeulemeester E, Herroelen W**, *Discrete time-cost trade-offs in project scheduling with time-switch constraints*, Journal of the Operational Research Society, **53**(7), (2002), 741-751, DOI 10.1057/palgrave.jors.2601351.

21 **Kaveh A, Talatahari S**, *A novel heuristic optimization method: charged system search*, Acta Mechanica, **213**, (2010), 267–289, DOI 10.1007/s00707-009q-0270-4.

22 **Kaveh A, Mahdavi VR**, *Colliding bodies optimization: A novel metaheuristic method*, Computers and Structures, **139**, (2014), 18–27, DOI 10.1016/j.compstruc.2014.04.005.

23 **Kaveh A, Mahdavi V**, *Colliding bodies optimization for discrete optimal design of truss structures*, Computers and Structures, **139**, (2014), 43–53, DOI 10.1016/j.compstruc.2014.04.006.

24 **Kaveh A**, *Advances in metaheuristic algorithms for optimal design of structures*, Springer Verlag; Switzerland, 2014, DOI 10.1007/978-3-319-05549-7.