An imperialist competitive algorithm for resource constrained project scheduling with activities flotation

Zahra Akeshteh† and Farhad Mardukhi††

Department of Computer Engineering, Faculty of Engineering, Kermanshah Branch, Islamic Azad University, Kermanshah, Iran
Department of Computer Engineering and Information Technology, Faculty of Engineering, Razi University, Kermanshah, Iran

Summary
In this paper an Imperialist competitive algorithm was proposed for Multi-mode resource constrained project scheduling problem (MRCPSP) with considering flotation of activities and discounted cash flow on project cost. This problem is a generalization of the RCPSP problem. In First step mathematical model of problem was defined. Then a solution based on imperialist competitive algorithm was proposed. In next step Taguchi experiments designing applied to tune available parameters of algorithm in order to increase efficiency. Consideration of activities flotation leads to reducing project cost value without changing duration of completing project. To evaluate efficiency of algorithm the result of imperialist competitive algorithm shows low data dispersion and reduction total cost of project compared with genetic and particle swarm optimization algorithm in the same condition.

Key words:
Imperial competitive algorithm, scheduling, resource constrained, activities flotation.

1. Introduction

The role of project management in modern enterprise is significant. Project management defined as the application of techniques, skills, knowledge and tools on activities of project [1]. Through increase of required resources, project managers can usually decrease time of project activities [2]. Common resource constrained project arrangement approaches have been limited to the case where each activity may be carry out in only one predefined method [3]. The essential subject in project scheduling is the resource constrained project scheduling problem (RCPSP) [4]. The main effort of project managers is complete the project through minimum cost. RCPSP includes resource allocation to activities with constrained capacity while there is some priority between activities [5]. In the real world there are numerous ways for execution of each activity. Each way has particular time and needs specific requirement. This type of problems in the literature is defined as multimode resource constrained project scheduling problem (MRCPSP) [2]. The basic RCPSP and MRCPSP suppose that each activity, once started, and then perform until its end [4]. MRCPSP is a generalized description of the RCPSP, that each activity can be accomplished in one of selected ways, with attention to its duration and resource indigence. The resources can be distinguished in three categories: renewable resources such as manpower, machines which limited in unit of time; nonrenewable resources such as budget which are restricted through the project and limited resources such as cash flow per time unit, which are restricted per time unit and for duration of the project. The purpose of the MRCPSP is detecting a mode and a start time for each activity to minimize the makespan and the program being possible with respect to the priority of activities and renewable and nonrenewable resources limitation [4, 6]. In classical RCPSPs and MRCPSPs, renewable resources confines are supposed constant overtime [7-9].

The projects are usually time-consuming and calculating time value of money is necessary to do practical project plan. In order to compute time value of money, decreased cash flow can be applied. Therefore maximizing net present value (NPV) of project total cash flows is one of the project objectives. Set the NVP as a problem objective, converts the problem to MRCPSPD. Russell [10] at 1970 suggested maximizing present value of project cash flow for the first time. He ignored constraint of resource in this problem and proposed a new objective function. In order to conversion of nonlinear to linear model, Russell also used Taylor series as an approximation. Etgar [11] at 1997 assigned cash flow to each event and assumed that cash flow is depended on activity duration with respect to the resources costs in time. Icmeli and Erenguc [12] at 1994 solved MRCPSPD by using two types of Tabu search algorithms which cash flow was assigned to activity and project punished if determined time was exceeded. Mika [13] at 2005 offered a Tabu search algorithm for MRCPSPD that hybrid with simulated annealing. Najafi [14] at 2009 used a genetic algorithm to the resource investment problem with discounted cash flows.

In the last decades, the MRCPSPD is a new problem of the project management literature. In this method a set of modes (time–cost pairs) is given for each activity. The objective of the multimode resource constrained problem with considering NPV leads to select a mode for each

Manuscript received May 5, 2017
Manuscript revised May 20, 2017

activity so the total cost becomes minimized. Many methodologies have been proposed for this problem to decrease the total cost by deciding an optimal combination of carrying out each activity. These methodologies can be classified in to three categories: (a) Exact algorithms such as linear programming, integer programming, dynamic programming, branch and bound algorithms, etc. [15-18]; (b) Heuristic algorithms [19-22]; (c) Meta heuristic algorithms [23-26].

Imperialist competitive algorithm is a Metahuristic algorithm for NP hard problem. This algorithm is introduced by Atashpaz-Gargari et al. in 2007 [27].

In this study, a MRCPSPD is proposed that has some characteristics as follows: (a) Splitting of the activity is not admissible. (b) The durations of activities are deterministic. (c) The renewable resources are constrained in each time period. (d) Each of activity executes only in the selected mode. (e) The direct cost of implementing each activity is dependent on the renewable resource requirements. This assumption is proposed to cover some short comings in realistic existing models of project scheduling and managing. (f) The indirect cost of completing project is considered a constant value in each unit of time due to end of the project. It is apparent that prolonging project will increase this type of cost. (g) For activity i, the required resources in each possible mode, have different quantity in at least one type of renewable resources in comparison with the other modes. This means that in each method, the quantities of the required resources can be changed, but types of required resources cannot be changed. (h) The total cost of completing project consists of direct and indirect costs.

In this paper an imperialist competitive algorithm approach for solving the MRCPSPD introduced based on different components found in literature. Remain of this paper is arranged as follows: Section 2 characterizes the general formulation of the problem. Section 3 describes the steps in the proposed imperialist competitive algorithm in detail, and in Section 4, the manner of tuning parameters and results of computational experiments are reported then efficiency of algorithm is compared with particle swarm optimization and genetic algorithm in the same condition. In final section, the conclusions of this work are presented.

2. Problem definition

The structure of the project was shown by activity-on-node network G=(N,E) where N represent the activities i=1, 2, …, n and E imply the precedence relations between activities. In this paper the finish-start with zero time lags of precedence relations between activities is considered; that means an activity can't be started unless, the set of all its prerequisites were be finished. The first activity of the network shows the start of project and the last activity represents the end of project execution so, these activities do not employ resource and time. Each activity of project has M execution modes with different duration time and resource requirements.

In this problem Rn types of renewable resources are supposed to execute the project. Maximum capacity of each renewable resource k (k=1, 2, …, Rn) is presented by Rkmax which is fixed in each period of time based on resource constrained. In addition executing activity i in mode m needs rimk unit of renewable resource k.

As mentioned consideration of net present value (NPV) on project cash flow will help more realistic project scheduling. Therefore in this study NPV on costs of executing activities were calculated in their finish times. In this part the mathematical formulation of multimode resource constrained project scheduling with considering NPV and activities flotation is presented.

2.1 Model notation, parameter and variable

N: the set of activities

PredList: the set of precedence relations between activities

R_n: the number of renewable resource type

R_{kmax}: maximum capacity of renewable resource k; k= 1…, R_n

$\propto$: discounted rate

M_i: the set of execution mode for activity i; i= 1, …, n

T_{im}: duration of activity i when it is executed in mode m; m=1, …, M_i, i= 1, …, n

C_{im}: direct cost of executing activity i in mode m; m=1, …, M_i, i= 1, …, n

R_{imk}: the number of resource k to execute activity i in mode m; k= 1…, R_n

RR_k: the available quantity of resource type k in unit of time; k= 1…, R_n

Z: summation of direct and indirect costs of project

T_{max}: the total time for completion of project

ST_i: start time of activity i

FT_i: finish time of activity i

C_0: indirect cost of project in each period of time

2.2 Problem modeling

Constrains of this model are represented in equations 1-4.

$$\min \sum_{i=1}^{n}\sum_{j=1}^{M_i} \frac{C_{im}}{(1+\alpha)^{(ST_i+d_{im})}} x_{ij} + \sum_{t=1}^{T_{Max}} \frac{C_0}{(1+\alpha)^t} \quad (1)$$

$$\sum_{j=1}^{M_i} x_{ij} = 1; i = 1...n, j = 1...M_j \quad (2)$$

$$FT_i - FT_j - \sum_{i=1}^{M_i} 0??x_{imt}, 2... \text{ 전는 } 1,2.n \quad j = \quad M_j \quad (3)$$

$$\sum_{i=1}^{n}\sum_{m=1}^{M_i} x_{im} r_{imk} + R_k \sum_{b=ST_i}^{ST_i+d_{im}} x_{im} \le R_k \quad k = 1,2...R \qquad (4)$$

The purpose of this paper is minimizing total direct and indirect costs as the objective function. Consideration of net present value on direct cost causes costly activities lead to the end of project. On the other hand prolonging project causes growth of indirect costs. Therefore the solution for minimizing (1) will determine each activity in which mode and when should be executed. Equation 2 ensures each activity implemented just in one mode. Construction 3 represents technological priority of activity execution. Equation 4 ensures the renewable resource limitation of project scheduling.

3. Imperialist competitive algorithm (ICA)

In this paper an imperialist competitive algorithm is used to solve the multimode resource constrained project scheduling with considering activities flotation and NPV on cost of project. In the next section, methods of coding and decoding problem solutions will be described. Then operation of schedule function is represented to show how it can determine the start time of each activity based on precedence relations and constrained renewable resources.

3.1 Encoding and Decoding of a Solution

Each solution of this problem supposed as two 1×N matrixes that N is the number of project activities. The first part shows the ordered priority of activities to get resources and the second determines the selected mode of activities. Fig. 1 shows an example of the solution in a problem with 10 activities. In this example there are 3 modes for implementing each activity.

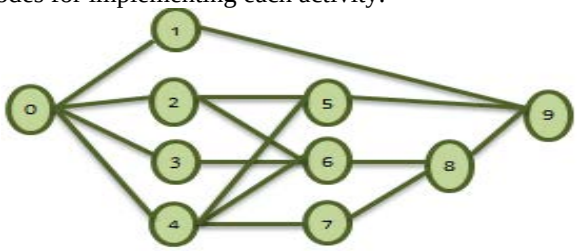

Fig. 1 An activity-on-node network.

According to above arrays, action 8 is the most valuable action of the project to get resource and complete its execution. In other hand an activity can't be started unless all its predecessors were completed.

In this method, the ordered values of actions are modified according to precedence relations between activities. To modify the ordered list of activities, array will be checked from left to right. If all PredLists members of an activity were entered to modify list, that action can be added to new list and omits from array. At first iteration of this example, just activity one can be entered to modify list because all activities have prerequisite except activity 0 (the start imaginary activity). In the next step other members of array from left to right will be checked again and these iterations continue until all actions enter to modified list. Fig. 2 shows the modified list of above example.

Pre Priority of activities to get resource	8	3	6	7	5	1	2	4	0	9
Selected execution mode of activities	3	3	1	2	1	3	1	2	1	1

Fig. 2 An example of problem solution.

Modified Priority of activities	0	3	1	2	4	6	7	8	5	9
Selected execution mode of activities	3	3	1	2	1	3	1	2	1	1

Fig. 3 Modified representation of Fig. 2 solution.

The second part of solutions assigns an execution mode to each activity. For example in Fig. 1 actions {1, 3, 4, 6, and 10} will be implemented in mode 1 and actions {2, 8, and 9} execute in mode 2. Mode 3 is chosen to actions {5, 7}.

3.2 Schedule Generation Algorithm

The objective function of this study is minimization of total direct and indirect cost with considering NPV on it. In order to determine fitness of a solution, the schedule and cost function is executed. In scheduling process three lists are used in each time of project execution;

OL: Ordered list of activities that don't scheduled.

E: List of all executing activities at time unit until they are executing.

ED: List of all executed activities.

In each time unit always the ordered list is checked. If all predecessors of an activity were in executed list, that activity will enter to possible list. Then for each activity of possible list if enough resources exist, those activities can be started and will omit from ordered list. Furthermore those activities enter to executing list until the duration of activities. After completing execution of each activity, it removes from executing list and enters to the executed list. So its succeeding activities can be started.

When all activities entered to the executed list, the time of implementing project is calculated. In the first step of algorithm, all activities exist in the modified ordered list because none of them is scheduled. In Fig. 4 the flowchart of this part is shown.

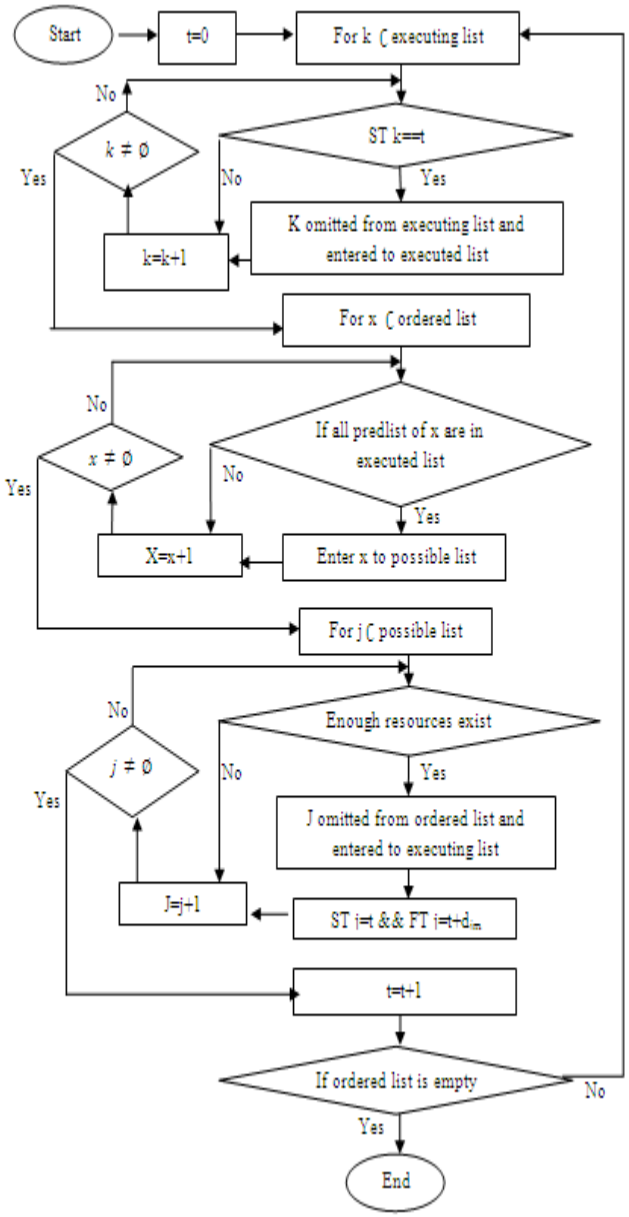

Fig. 4 Flowchart of schedule function.

Then process of recursive scheduling is started. This section uses succeeding relations between activities and starts with end of project time. This piece of function separates critical activities from others. In this part, executions of non-critical activities lead to end of project that decreases the net present value of costs.

3.3 Generation of Initial Empires

In the imperialist competitive algorithm each solution of problem defined as "country". The countries are similar to the concept of chromosome in genetic algorithm (GA) or particle in particle swarm optimization (PSO) algorithm. In the first step of algorithm, the initial generation of random solutions is created. Then each one of these countries enters to the schedule method to determine fitness of that solution.

3.4 Imperialists Selection

According to the number of nEmp from best solutions are chosen as imperialists and each imperialist constructs an empire. The other countries assign to imperialists as its colonies. In addition the possibility of each imperialist to obtain colonies is not equal. So imperialist with best fitness value has more opportunities to take colonies in roulette wheel.

3.5 Total Power of an Empire

As mentioned, each imperialist and its colonies construct an empire. Total power of an empire will be calculated by power of its imperialist and the power of its colonies.

$$\text{total.cost}_n = \left(\quad _n \right) + \text{잨}$$
$$\varepsilon \text{ mean}\left\{\text{cost}\left(\text{colonies of empire}_n\right)\right\} \quad (5)$$

Equation 5 presents that total power of empire contains the power of imperialist and mean power of its colonies. ε is the effect ratio of colonies' power on power of empire and it is a number between 0,1; so colonies have small role on the power of empire.
In first step of algorithm in which the colonies don't assign to the empire, the power of empire is equal to the power of imperialist but at other step equation 5 is used to calculate this value.

3.6 Moving Colonies of Empire toward Their Imperialists (Assimilating)

The colonies of each empire move toward the imperialist through assimilating operation. This operation is used to search the problem space around each imperialist because fitness of imperialist is better than all its colonies. The new position of each colony after assimilating operation is calculated by equation 6.

$$emp.col(i).position = emp.col(i).position + \beta * rand * \left(emp.Imp.position - emp.col(i).position\right) \quad (6)$$

According to above equation, the new position of colony i is calculated from current location and a ratio of its distance from the imperialist. The value of β defines how much a colony will be affected by its imperialist in assimilating operation. The rand function is used to assimilate each colony in different dimensions of its imperialist. The cost and scheduling function executes after specifying new position of each colony. In

assimilating operator both the ordered value list and execution modes of colony will be affected and moving toward imperialist.

3.7 Revolution

This operation is similar to mutation operator in genetic algorithm. According to revolution intensity, some indexes of activities ordered list from a solution are chosen randomly and their value are changed. In addition, these changes will be done on execution modes of that solution. Then new position of problem space is determined that the algorithm should schedule it and calculate its cost.
In this paper revolution operator implements on both the imperialist and colonies but two kinds of this operation are defined. The first type of revolution is employed on all imperialists. The revolution operation on an imperialist is accepted if only the new position achieves better cost else that operation fails and the imperialist remains in its current position. Another kind of revolution operator acts on percentage of countries based on pRevolution. This operation always will be accepted even if new positions gain inappropriate cost.

3.8 Colonies and Imperialist Updating

After implementing assimilation and revolution operators, the cost and scheduling function should be act on new positions. According to the number of colonies (ncol) the best countries are selected and other positions will be omitted. Subsequently, the imperialist of each empire compares with new list of its colonies and the best country will be chosen as the imperialist. So the current imperialist may remain imperialist or a colony may become imperialist and last imperialist entitles colony.

3.9 Imperialists Competition

After updating imperialists and colonies, the cost of each empire will be calculated by Equation 5. The next step in ICA algorithm is decreasing the power of weakest empire. For this purpose the weakest colony of weakest empire is separated from its empire and assigns to another one. The other empires will have opportunity to receive the weakest colony by Equation 7 based on their powers. The probability set of all empires is created by Equation 7 and 8.

$$Pp_n = \frac{N.T.C_n}{\sum_{i=1}^{N_{imp}} N.T.C_i} \quad (7)$$

$$P = \left[Pp_1, \quad _2 \cdots \quad _{n_{imp}} \right] \quad (8)$$

$$R = \left[r_1, \quad _2 \quad _{n_{imp}} \right] \quad (9)$$

$$D = P - R = \left[D_1, \quad _2 \quad _{n_{imp}} \right] \quad (10)$$

In equation 9 an array with Nimp random numbers between [0, 1] is created then equation 10 is calculated and empire with maximum number will be chosen to get the weakest colony of weakest empire. It is important that the most powerful empire has more opportunities to capture the colony but it is not certain to take it.

3.10 Elimination of the Powerless Empire

When an empire loses all its colonies, it should be omitted. It means the weakest empire will collapse during execution of algorithm because its weakest colonies were assigned to other empires. After separating all colonies of an empire, the imperialist will be deleted.
In Fig. 5 the flowchart of algorithm is represented.

4. Experimentations and computational results

At the first part of this section, problem generation and comparison metrics are described. In the next part tuning of the parameters will be specified and performance of proposed algorithm is discussed by comparing with genetic algorithm and particle swarm optimization. This algorithm is simulated in Matlab R2010a and run on a core i5 personal computer with 1.80 GHz Intel (R) CPU and 6 Gig RAM.

4.1 Problem Generation

In order to evaluate the algorithm, a standard problem from the project scheduling library PSPLib is chosen that contains 20 activities with predecessor relations between them. For each algorithm 15 iterations of problem were implemented and mean of the results is calculated. It should be noted that the problem of this library studied the time of completing project but in this study, cost of implementing project based on time is discussed; to this aim, direct cost of executing an activity is determined according to its resource requirement. In other words direct cost is a factor of resource usage and indirect cost of completing project in each period of time is considered 800 units.

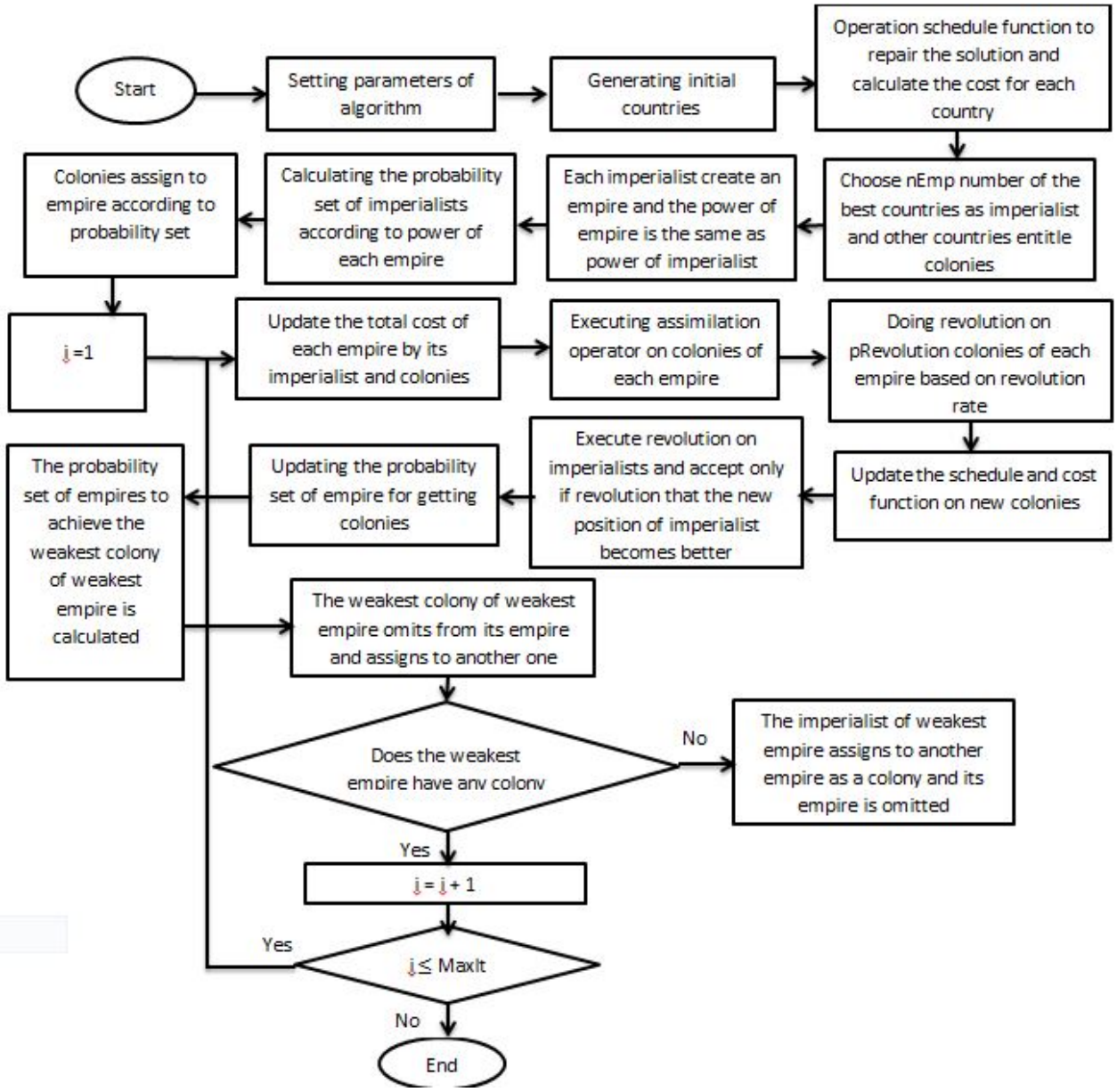

Fig. 5 Flowchart of imperialist competitive algorithm.

4.2 Comparison Metrics

Because of problem complexity and novelty aspects cause to no mathematical models can solve the problem exactly. Therefore to discuss the performance of proposed algorithm, summation of direct and indirect costs is compared with genetic algorithm and particle swarm optimization algorithm.

4.3 Tuning the Parameters

Performance of Meta heuristic algorithm is affected by values of its parameters. Achieving better performance needs parameter setting. There are several ways to set the parameters. In the first step, parameters of each algorithm should be specified as factors. Then some levels determined for each factor. Parameters tuning will determine which level for each factor has better performance on output of algorithm.

One method of calibrating parameters is a full factorial experiment in which each level of a factor is used with all level of other factors; therefore by increasing the number of factors and levels of them, the number of experiment will gain. For example an algorithm with 5 factors and each factor with 3 levels need 125 experiments to specify best levels of parameters. Using orthogonal array is another method that decreases this number and overcomes the shortcoming.

Taguchi design based on orthogonal array reduces the number of experiments while keeping sufficient information. For this purpose it converts solution to ratio of signal to noise and minimize the effect of noise to specify the optimal level of parameters (Eq. 11).

$$\frac{S}{N}\text{ratio} = -10\log\left(\frac{1}{n}\sum_{i=1}^{n} n_i^2\right) \quad (11)$$

This Equation maximizes signal-to-noise ratio and the objective function is kind of "the smaller is better". Taguchi design distinguishes the relative of each factor according to their effect on objective function. In this study some of main parameters of genetic, particle swarm optimization and imperialist competitive algorithm were been tuned.

4.3.1 Parameters tuning for imperialist competitive algorithm

In this study, 7 control factors were determined to be tuned for imperialist competitive algorithm and 3 levels for each factor were considered. These factors are: number of empires (nEmp), population size (nPop), maximum number of iterations (MaxIt), assimilation coefficient (beta), revolution probability (pRevolution), revolution rate (mu) and effect ratio of colonies power on power of empire (zeta) that shown in Table 1. The orthogonal array for 7 factors with 3 levels that designed by Taguchi method is presented in Table 2.

Table 1: Imperialist competitive algorithm factors and levels

Symbol	*Factor*	*Levels*
A	(MaxIt, nPop)	(50,100), (70,50), (100,70)
B	nEmp	4-6-8
C	zeta	0.1, 0.3, 0.5
D	mu	0.3, 0.5, 0.7
E	pRevolution	0.1, 0.2, 0.5
F	beta	0.8, 2, 3

Fig. 6 presented the result of SN ratios for tuning parameters of imperialist competitive algorithm. According to this figure the appropriate levels of factors are respectively 3, 1, 1, 3, 2 and 2. Therefore the proper parameters are nEmp: 4, nPop: 70, MaxIt: 100, beta: 2, pRevolution: 0.2, mu: 0.08 and zeta: 0.1.

Table 2: Orthogonal array L27

Experiment	*A*	*B*	*C*	*D*	*E*	*F*
1	1	1	1	1	1	1
2	1	1	1	1	2	2
3	1	1	1	1	3	3
4	1	2	2	2	1	1
5	1	2	2	2	2	2
6	1	2	2	2	3	3
7	1	3	3	3	1	1
8	1	3	3	3	2	2
9	1	3	3	3	3	3
10	2	1	2	3	1	2
11	2	1	2	3	2	3
12	2	1	2	3	3	1
13	2	2	3	1	1	2
14	2	2	3	1	2	3
15	2	2	3	1	3	1
16	2	3	1	2	1	2
17	2	3	1	2	2	3
18	2	3	1	2	3	1
19	3	1	3	2	1	3
20	3	1	3	2	2	1
21	3	1	3	2	3	2
22	3	2	1	3	1	3
23	3	2	1	3	2	1
24	3	2	1	3	3	2
25	3	3	2	1	1	3
26	3	3	2	1	2	1
27	3	3	2	1	3	2

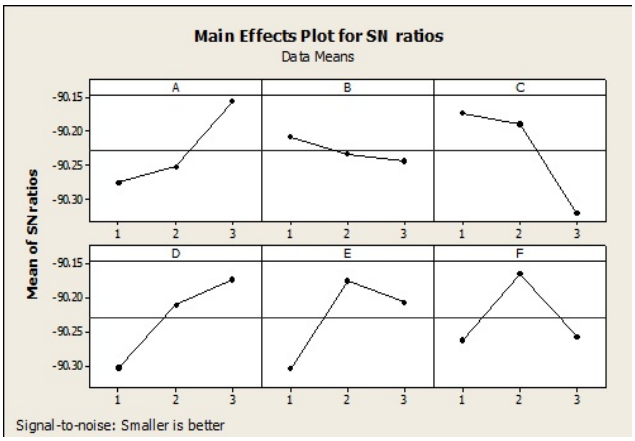

Fig. 6 The Mean S/N Ratio Plot for Each Level of ICA Factors.

4.3.2 Parameters tuning for particle swarm optimization

In this study, 5 factors and 4 levels for each one are considered that shown in Table 3. Through Taguchi method L16 experiments are designed as orthogonal array. The factors of the implemented algorithm are: population size (nPop), maximum number of iterations (MaxIt), inertia weight damping ratio (wdamp), personal learning

coefficient (c1) and global learning coefficient (c2). The levels of each factor are specified according to previous study [28].

Table 3: Particle swarm optimization factors and levels

Symbol	*Factor*	*Levels*
A	(MaxIt, nPop)	(100,100), (110,80), (60,120), (80,120)
B	wdamp	0.1, 0.4, 0.6, 0.8
C	c1	0.1, 0.7, 1.7, 3
D	c2	0.1, 0.7, 1.7, 3

L16 includes 16 experiments which determined by Taghuchi design in Table 4.

Table 4: Orthogonal array L16

Experiment	*A*	*B*	*C*	*D*
1	1	1	1	1
2	1	2	2	2
3	1	3	3	3
4	1	4	4	4
5	2	1	2	3
6	2	2	1	4
7	2	3	4	1
8	2	4	3	2
9	3	1	3	4
10	3	2	4	3
11	3	3	1	2
12	3	4	2	1
13	4	1	4	2
14	4	2	3	1
15	4	3	2	4
16	4	4	1	3

SN ratios of factors and their levels for particle swarm optimization are presented in Fig. 7. Therefore, values of tuned parameters are nPop: 100, MaxIt: 100, w: 0.3, wdamp: o.8, c1: 3 and c2: 1.7 and appropriate levels consist of 1, 4, 4 and 3.

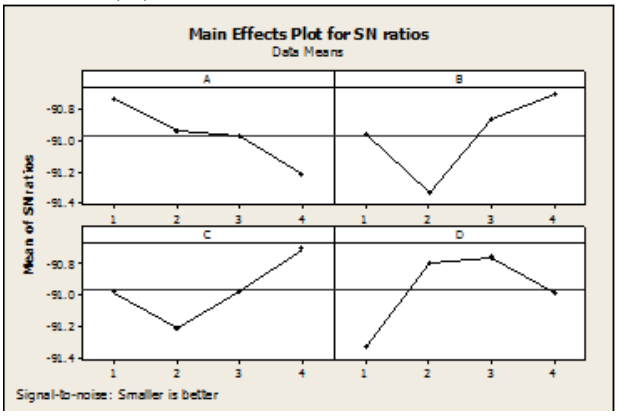

Fig. 7 The Mean S/N Ratio Plot for Each Level of PSO Factors.

4.4 Computational Outcomes

Particle swarm optimization and genetic algorithm are implemented to evaluate the performance of imperialist competitive algorithm. Three algorithms are executed on a standard problem of PSPLib with 20 activities and 3 modes for each activity. The cost of implementing for each mode related to amount of required resources. In Fig. 8 the box plot diagram of the implemented algorithms is shown and value plot is presented in Fig. 9. The result of running each algorithm after 15 iterations represent that imperialist competitive algorithm has minimum average. Moreover the dispersion of its results is less than genetic algorithm and particle swarm optimization.

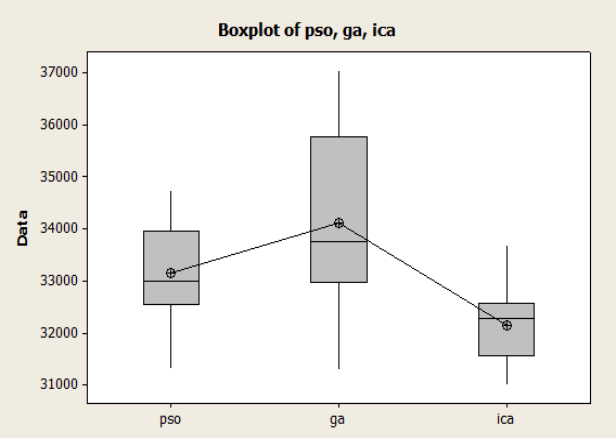

Fig. 8 Box Plot Diagram of ICA, GA and PSO.

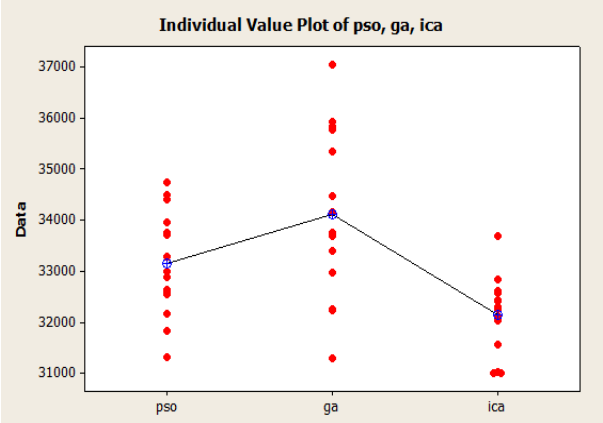

Fig. 9 Value Plot Diagram of ICA, GA and PSO.

Convergence diagram of applied algorithms for summation of direct and indirect costs is shown in Fig. 10. According to this figure, the imperialist competitive algorithm had faster convergence. The convergence diagram for time of project completion is shown in Fig. 11.

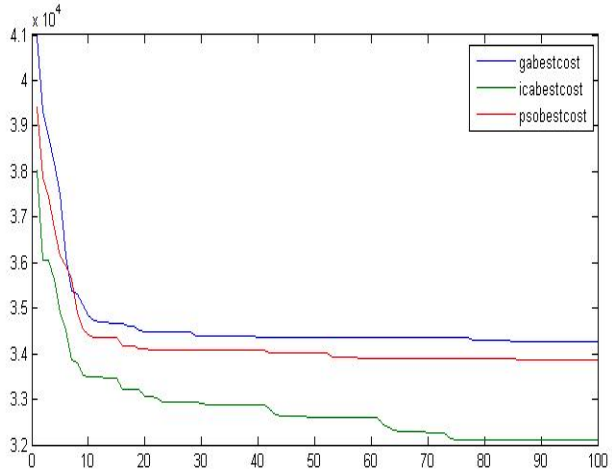

Fig. 10 Convergence Diagram of Project Cost.

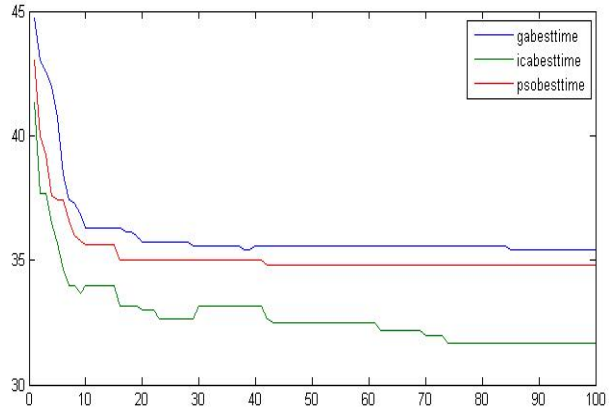

Fig. 11 Convergence Diagram of Project Time.

5. Conclusion

In this study, a MRCPSPD problem introduced as a resource-oriented cost minimization project scheduling problem considering both renewable and nonrenewable resource costs. Subsequently, a Meta heuristic algorithm proposed to tackle this project scheduling problem. An imperialist competitive algorithm including assimilation and revolution on both string of assigning modes and priority of activities utilized to capture resources. In order to generate feasible schedules, the repair function proposed for modifying the ordered list of activities to use resources. In other hand, considering time value of money makes the problem more realistic and consequently making decision is closer to reality. In this paper recursive scheduling codes are implemented to focus on flotation of activities.

For proving efficiency of this algorithm an example from standard test problems (PSPLIB library) was studied and its results were compared with Genetic algorithm and particle swarm optimization algorithm. After tuning parameters of each algorithm, computational results show that imperialist competitive algorithm is more efficient than genetic algorithm and particle swarm optimization. In addition diversion of answer in different iteration of problem is less than other algorithms.

References

[1] Guide, A. Project Management Body of Knowledge (PMBOK® GUIDE). in Project Management Institute. 2001.
[2] Ghamginzadeh, A. and A.A. Najafi, Solving Resource-constrained Discrete Time-cost Trade-off Problem by Memetic Algorithm. 2013.
[3] Salewski, F., A. Schirmer, and A. Drexl, Project scheduling under resource and mode identity constraints: Model, complexity, methods, and application. European Journal of Operational Research, 1997. 102(1): p. 88-110.
[4] Van Peteghem, V. and M. Vanhoucke, A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem. European Journal of Operational Research, 2010. 201(2): p. 409-418.
[5] Afruzi, E.N., et al., A Multi-Objective Imperialist Competitive Algorithm for solving discrete time, cost and quality trade-off problems with mode-identity and resource-constrained situations. Computers & Operations Research, 2014. 50: p. 80-96.
[6] Słowiński, R., B. Soniewicki, and J. Węglarz, DSS for multiobjective project scheduling. European Journal of Operational Research, 1994. 79(2): p. 220-229.
[7] Drexl, A. and J. Gruenewald, Nonpreemptive multi-mode resource-constrained project scheduling. IIE transactions, 1993. 25(5): p. 74-81.
[8] Hartmann, S., Project scheduling under limited resources: models, methods, and applications. Vol. 478. 2012: Springer Science & Business Media.
[9] Klein, R., Project scheduling with time-varying resource constraints. International Journal of Production Research, 2000. 38(16): p. 3937-3952.
[10] Russell, A., Cash flows in networks. Management Science, 1970. 16(5): p. 357-373.
[11] Etgar, R., A. Shtub, and L.J. LeBlanc, Scheduling projects to maximize net present value—the case of time-dependent, contingent cash flows. European Journal of Operational Research, 1997. 96(1): p. 90-96.
[12] Icmeli, O. and S.S. Erenguc, A tabu search procedure for the resource constrained project scheduling problem with discounted cash flows. Computers & Operations Research, 1994. 21(8): p. 841-853.
[13] Mika, M., G. Waligora, and J. Węglarz, Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash

flows and different payment models. European Journal of Operational Research, 2005. 164(3): p. 639-668.
[14] Najafi, A.A., S.T.A. Niaki, and M. Shahsavar, A parameter-tuned genetic algorithm for the resource investment problem with discounted cash flows and generalized precedence relations. Computers & Operations Research, 2009. 36(11): p. 2994-3001.
[15] Liu, L., S.A. Burns, and C.-W. Feng, Construction time-cost trade-off analysis using LP/IP hybrid method. Journal of Construction Engineering and Management, 1995. 121(4): p. 446-454.
[16] Erenguc, S.S., T. Ahn, and D.G. Conway, The resource constrained project scheduling problem with multiple crashable modes: An exact solution method. Naval Research Logistics (NRL), 2001. 48(2): p. 107-127.
[17] Shtub, A., J.F. Bard, and S. Globerson, Project management: engineering, technology, and implementation. 1994: Prentice-Hall, Inc.
[18] Butcher, W.S., Dynamic programming for project cost-time curves. 1967.
[19] Vanhoucke, M. and D. Debels, The discrete time/cost trade-off problem: extensions and heuristic procedures. Journal of Scheduling, 2007. 10(4-5): p. 311-326.
[20] Boctor, F.F., Heuristics for scheduling projects with resource restrictions and several resource-duration modes. The International Journal of Production Research, 1993. 31(11): p. 2547-2558.
[21] Ahn, T. and S.S. Erenguc, The resource constrained project scheduling problem with multiple crashable modes: a heuristic procedure. European Journal of Operational Research, 1998. 107(2): p. 250-259.
[22] Demeulemeester, E., et al., New computational results on the discrete time/cost trade-off problem in project networks. Journal of the Operational Research Society, 1998. 49(11): p. 1153-1163.
[23] Wuliang, P. and W. Chengen, A multi-mode resource-constrained discrete time–cost tradeoff problem and its genetic algorithm based solution. International Journal of Project Management, 2009. 27(6): p. 600-609.
[24] Azaron, A., C. Perkgoz, and M. Sakawa, A genetic algorithm approach for the time-cost trade-off in PERT networks. Applied mathematics and computation, 2005. 168(2): p. 1317-1339.
[25] Chao-Guang, J., et al., Research on the fully fuzzy time-cost trade-off based on genetic algorithms. Journal of Marine Science and Application, 2005. 4(3): p. 18-23.
[26] Afruzi, E.N., et al., A multi-mode resource-constrained discrete time–cost tradeoff problem solving using an adjusted fuzzy dominance genetic algorithm. Scientia Iranica, 2013. 20(3): p. 931-944.
[27] Atashpaz-Gargari, Esmaeil, and Caro Lucas. "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition." Evolutionary computation, 2007. CEC 2007. IEEE Congress on. IEEE, 2007.
[28] Eberhart, Russell, and James Kennedy. "A new optimizer using particle swarm theory." Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on. IEEE, 1995.