Journal of Optimization in Industrial Engineering
Vol.12, Issue 2, Summer & Autumn 2019, 155-178
DOI: 10.22094/JOIE.2018.556347.1531

Research Article

An Evolutionary Algorithm Based on a Hybrid Multi-Attribute Decision Making Method for the Multi-Mode Multi-Skilled Resource-Constrained Project Scheduling Problem

Amir Hossein Hosseinian [a], Vahid Baradaran [b,*]

[a] *Department of Industrial Engineering, Faculty of Engineering, Islamic Azad University, Tehran North Branch, Tehran, Iran*
[b] *Department of Industrial Engineering, Faculty of Engineering, Islamic Azad University, Tehran North Branch, Tehran, Iran*
Received 16 December 2017; Revised 11 December 2018; Accepted 16 December 2018

Abstract

This paper addresses the multi-mode multi-skilled resource-constrained project scheduling problem. Activities of real world projects often require more than one skill to be accomplished. Besides, in many real-world situations, the resources are multi-skilled workforces. In presence of multi-skilled resources, it is required to determine the combination of workforces assigned to each activity. Hence, in this paper, a mixed-integer formulation called the MMSRCPSP is proposed to minimize the completion time of project. Since the MMSRCPSP is strongly NP-hard, a new genetic algorithm is developed to find optimal or near-optimal solutions in a reasonable computation time. The proposed genetic algorithm (PGA) employs two new strategies to explore the solution space in order to find diverse and high-quality individuals. Furthermore, the PGA uses a hybrid multi-attribute decision making (MADM) approach consisting of the Shannon's entropy method and the VIKOR method to select the candidate individuals for reproduction. The effectiveness of the PGA is evaluated by conducting numerical experiments on several test instances. The outputs of the proposed algorithm is compared to the results obtained by the classical genetic algorithm, harmony search algorithm, and Neurogenetic algorithm. The results show the superiority of the PGA over the other three methods. To test the efficiency of the PGA in finding optimal solutions, the make-span of small size benchmark problems are compared to the optimal solutions obtained by the GAMS software. The outputs show that the proposed genetic algorithm has obtained optimal solutions for 70% of test problems.

***Keywords*:** RCPSP; Multi-skilled resources; Optimization; Meta-heuristics; MADM.

1. Introduction

A classical resource-constrained project scheduling problem (RCPSP) deals with scheduling a set of interrelated activities with respect to precedence relations and resource limitations (Hosseini et al., 2014; Naderi, 2013). The RCPSP aims to schedule tasks such that the project completion time is minimized (Afshar-Nadjafi et al., 2012). Blazewicz et al. (1983) proved that the RCPSP belongs to the class of NP-Hard problems and it is difficult to solve it to optimality. The RCPSP and its extensions have been widely investigated in the literature (Hosseini et al., 2014). The multi-mode RCPSP (MRCPSP) is a variant of the standard RCPSP, where there are several modes for performing an activity and each activity can be executed in one out of multiple modes. The modes of activities will not change during their processing times. Each mode represents a certain combination of duration and resource requirements (Afruzi, et al., 2014). The multi-skilled RCPSP (MSRCPSP) is another extension which has been the subject of many studies in the literature. In the multi-skilled RCPSP, resources are workforces that are able to perform at least one skill. Each activity may need one or more skills. In presence of multi-skilled workforces, it has to be determined that which workers should be assigned to each task (Maghsoudlou et al., 2017).

In this paper, a multi-mode multi-skilled RCPSP (MMSRCPSP) model is proposed to minimize the project completion time. Thus, the make-span of the project is considered as the fitness value of a schedule. Considering multi-modal activities and multi-skilled human resources is a step forward to provide more realistic schedules for real-world projects. Since the standard RCPSP is an NP-Hard problem, the MMSRCPSP belongs to the class of NP-hard problems as well. Due to the NP-Hard nature of RCPSP, finding optimal solutions by means of exact methods for large size problems, becomes computationally impractical. Hence, meta-heuristics are used to provide near optimal solutions within reasonable computation times (Agarwal et al., 2015). Therefore, we propose a genetic algorithm (GA) to tackle the MMSRCPSP. This method is called the proposed genetic algorithm (PGA) which uses a hybrid multi-attribute decision making (MADM) approach consisting of the Shannon's entropy method and the VIKOR[1] method to choose the best solutions as parents in order to produce high quality and diverse solutions for the next generation.

1. Vise Kriterijumska Optimizacija I Kompromisno Resenje (VIKOR)

*Corresponding author Email address: ah_hosseinian@iau-tnb.ac.ir

Many researchers employ the tournament selection operator for their genetic algorithm to select candidate solutions for reproduction. The basic tournament selection operator chooses a number of individuals from the population. The chosen individuals compete with each other based on their fitness values to determine the winner. Therefore, the solution with the least fitness value always fails to win the competition. Thus, it can be concluded that the chance of selection for all solutions is not equal. This drawback may prevent the algorithm from finding diverse solutions (Miller and Goldberg, 1996). To tackle this issue, we propose a tournament selection operator which considers two criteria for each solution. These criteria are: (1) the fitness value of an individual, and (2) probability of selection. The probability of selection is a real random number which is generated on the interval [0, 1] for each individual. The higher the value of this random number, the greater the chance of the solution to be selected. The proposed tournament selection operator chooses several random chromosomes from the population. Hence, a decision matrix is created, where the rows and columns represent the solutions and criteria, respectively. Afterwards, the Shannon's entropy (Lotfi and Fallahnejad, 2010) method, which is one of the most renowned approaches for discovering relative importance of criteria, is employed to determine the weights of criteria based on the characteristics of the chosen solutions in each iteration. Having determined the weights of criteria, the VIKOR method (Opricovic and Tzeng, 2004), as a compromise MADM method, is hired to select the best solution in terms of both criteria. This procedure gives opportunities to all individuals to be selected as parents. As another contribution, two novel strategies are proposed for the PGA that strengthen the ability of this method in exploring the solution space.

Furthermore, we propose new crossover and mutation operators for the PGA that always generate feasible solutions. The effectiveness of the proposed method is evaluated in comparison to the harmony search (HS) algorithm, classical genetic algorithm, and a state-of-the-art algorithm known as Neurogenetic method based on several test problems. The Taguchi method is employed to calibrate the input parameters of these three meta-heuristics. The outputs of optimizers for small size problems are compared to the optimal solutions obtained by the GAMS software.

The rest of the paper is organized as follows: Section 2 is devoted to previous studies on the multi-skilled RCPSP and the multi-skilled staff assignment problem (MSSAP). The description of the proposed model and its mathematical formulation come in Section 3. Section 4 is dedicated to a full explanation of the proposed algorithm. Section 5 provides the experimental results obtained by implementing the algorithms. Finally, Section 6 gives conclusions and suggests some research opportunities for further studies.

2. Literature Review

Models of the multi-skilled RCPSP (MSRCPSP) and the multi-skilled staff assignment problem (MSSAP) can be used in many industries such as software development industry, where employees have one or more skills such as designing, analysis, and programming. In software development field, employees with different skills work together so as to develop various products. The MSRCPSP can also be used to provide schedules for training operators of call centers. Trainers of a call center have different training skills. The MSRCPSP model can be applied to scheduling problems, where there are a group of staff members with specific skills required by activities (Bellenguez and Néron, 2007). Large number of studies exist in the realm of the MSRCPSP and the MSSAP. Hegazy et al. (2000) proposed some modifications to resource scheduling heuristics in order to decrease project delays occurred because of resource limitations. A linear programming model was proposed by Gomar et al. (2002) to allocate proper multi-skilled resources to activities of a construction project. Bellenguez and Néron (2005) presented a mathematical formulation for MSRCPSP, where workforces have different efficiencies in performing each of their skills. Corominas et al. (2005) proposed a multi-objective non-linear mixed-integer model for the MSRCPSP. This model was solved as a minimum cost flow problem. Wu and Sun (2006) investigated the effect of learning on efficiency of staff members for the project scheduling problem and staff assignment problem. They developed a genetic algorithm to solve their proposed model. Kadrou and Najid (2006) formulated a multi-mode version of the MSRCPSP. Their formulation includes resource limitation, labor skills and multiple execution modes for activities. A Tabu search (TS) algorithm with a new neighborhood function based on a flow graph scheme was developed to solve the problem. Azaron et al. (2006) developed a new multi-objective model for the resource allocation problem in the PERT networks, where the durations of activities follow Erlang distribution. Gutjahr et al. (2008) presented a non-linear mixed-integer model for the project portfolio selection problem considering competence development of workforces. They developed two meta-heuristics based on the ant colony optimization (ACO) and genetic algorithm combined with a greedy method as well. A hybrid benders decomposition (HBD) method was used by Li and Womer (2009) to solve the MSRCPSP. The researchers also developed a cut-generating scheme to eliminate resource conflicts. In another research by Li and Womer (2009), they modeled the manpower scheduling problem as a multi-purpose RCPSP to assign multi-skilled sailors to execute onboard activities. Valls et al. (2009) proposed a multi-skilled project scheduling model with deadlines for the start and the finish times of tasks. A genetic algorithm consisting of classical scheduling methods and local search algorithms were developed. A mixed-integer linear mathematical formulation was introduced by Heimerl and Kolisch (2010) to schedule multiple projects in presence of multi-skilled manpower. Al-Anzi et al. (2010) developed a

weighted multi-skilled RCPSP model. They considered that workforces have different efficiencies in performing their skills. A construction heuristic and an adaptive large scale neighborhood search method were developed by Cordeau et al. (2010) to schedule multi-skilled human resources and tasks in Telecommunications Company. Gutjahr et al. (2010) proposed a multi-objective model for the project portfolio selection problem. They decomposed the main problem into a portfolio selection problem and a slave problem which involves allocating manpower to projects. Yaghoubi et al. (2011) modeled the resource allocation problem for dynamic PERT networks with limited capacity of concurrent projects. The proposed model assumes that activity durations follow exponential distribution and new projects follow a Poisson distribution. Firat and Hurkens (2012) proposed a flexible matching model (FMM) to determine the group of workforces assigned to activities and produce large number of schedules. In their formulation, the workforces can use all their skills when they are assigned to an activity. Kazemipoor et al. (2013) presented a goal programming formulation for the multi-skilled project portfolio scheduling problem. The researchers considered infinite number of modes for each task. A differential evolution (DE) algorithm was developed to solve the model. Myszkowski et al. (2013) proposed some heuristics for the multi-skilled RCPSP to be employed in evolutionary algorithms in order to increase their robustness. Myszkowski and Skowronski (2013) presented new crossover and mutation operators based on domain knowledge for the genetic algorithm. Mehmanchi and Shadrokh (2013) investigated the impact of learning and forgetting on efficiency of workforces. They proposed an exponential learning function to tackle dynamic competence of manpower. In another study, Kazemipoor et al. (2013) presented a mixed-integer mathematical formulation for the multi-mode multi-skilled RCPSP. The researchers developed a scatter search (SS) algorithm and a Tabu search method as well. A two-phase algorithm consisting of genetic operators and a path relinking method was proposed by Tabrizi et al. (2014) to maximize the net present value (NPV). Correia and Gama (2014) proposed a cost-oriented formulation of MSRCPSP. A branch and price (B&P) method including a column generation approach was developed by Montoya et al. (2014) for MSRCPSP. Myszkowski et al. (2015) introduced a hybrid ant colony optimization consisting of priority rules and ant colony optimization. A teaching-learning-based optimization method (TLBO) was hired by Zheng et al. (2015) to solve the MSRCPSP. Yaghoubi et al. (2015) proposed a multi-objective model for the resource allocation problem. The proposed model controls the resources assigned to servers in a multi-class dynamic PERT network. They also developed a simulated annealing (SA) algorithm to solve the model. Walter and Zimmerman (2016) introduced a mathematical formulation for scheduling and staffing concurrent projects, where workforces can be allocated to several projects. Almeida et al. (2016) utilized a parallel scheduling scheme for the multi-skilled RCPSP. Javanmard et al. (2016) integrated the multi-skilled project scheduling problem with the resource investment problem (RIP). To solve the proposed mixed-integer mathematical model, a genetic-based and a particle-swarm-based algorithm were developed. A multi-objective mixed-integer model was proposed by Maghsoudlou et al. (2016) for the multi-mode multi-skilled RCPSP to optimize make-span, total costs and quality, simultaneously. A new multi-objective invasive weeds optimization algorithm (MOIWO) was developed to solve the proposed model. Maghsoudlou et al. (2017) presented a bi-objective formulation for the MSRCPSP. The objective functions of the model include minimization of totals costs and minimization of reworking risks of tasks. Three multi-objective cuckoo search algorithms were proposed to solve the model. A multi-project multi-skilled project scheduling model was proposed by Chen et al. (2017). They investigated the impact of learning and forgetting on efficiency of workers.

According to the previous researches reviewed in this section, the multi-mode MSRCPSP has not been studied sufficiently. Hence, we propose a new formulation for multi-mode multi-skilled RCPSP called MMSRCPSP. Genetic algorithms are highly efficient for combinatorial optimization problems. Hence, due to high efficacy of genetic algorithms, a modified version of the genetic algorithm has been developed to solve the proposed model. To the best of the authors' knowledge, this is the first time that an evolutionary algorithm based on the MADM methods is presented for the multi-skilled RCPSP. Therefore, it is interesting to investigate the impact of an MADM method on performance of an evolutionary algorithm in solving the MSRCPSP.

3. Problem Formulation

In this section, a multi-mode multi-skilled RCPSP model (MMSRCPSP) is proposed to: (1) determine start times of project activities, (2) determine the skills by which the resources contribute to activities, and (3) determine the modes by which the activities are performed. In this respect, a mixed-integer formulation is proposed to minimize the make-span of the project. In the following, the assumptions, sets, parameters, decision variables, and mathematical formulation are defined.

3.1. Assumptions

- The project activities and the precedence relations between them have been depicted by the activity-on-node (AON) network. Let $G(V, E)$ be the AON network, where V denotes the project activities and E represents logical dependencies between tasks.
- The activities are numbered as $j = 0, 1, 2, ..., N+1$. The activities 0 and $N+1$ are dummy activities which mark the start and the end of the project, respectively. Duration and

resource requirement of a dummy activity is zero.

- It is assumed that there are finish-to-start (FS) precedence relations between tasks without lead and lag (*FS*=0). These relations shape a directed acyclic graph.
- Each activity can be accomplished in one out of multiple modes.
- It is not possible to change the assigned mode of an activity during its execution.
- All resources are multi-skilled human resources. These workers are always available.
- Each activity may need one or multiple skills.
- Each activity may need one or more workers in each period to execute its required skills.
- Standard durations of activities are non-negative integer numbers.
- Each worker is able to perform one or more skills.
- It is not possible to assign a worker to more than one skill of an activity at the same period.
- All required skills of an activity must be started simultaneously.

3.2. Sets

V: Set of activity nodes, $(j, j' = 0,1,2,...,N+1)$,

Λ: Set of workforces, $(s, s' = 1,2,...,S)$,

κ: Set of skills, $(k, k' = 1,2,...,K)$,

Ψ: Set of time periods, $(t, t' = 0,1,2,...,T)$,

Π: Set of execution modes, $(m = 1,2,...,M)$,

E: Set of finish-to-start precedence relations between activities,

$pred_j$: Set of immediate predecessors of activity j,

H_j: Set of skills required by activity j,

g_k: Set of workforces who have skill k,

3.3. Parameters

d_{jm}: Duration of activity j in mode m,

r_{jkm}: The required number of workers to perform skill k of activity j in mode m,

b_k: The number of available workforces who have skill k,

λ_{sk}: $\begin{cases} 1 & \text{, if workforce } s \text{ has skill } k \\ 0 & \text{, Otherwise} \end{cases}$

3.4. Decision Variables

X_{jmt}: $\begin{cases} 1 & \text{, if activity } j \text{ is started in mode } m \text{ at period } t \\ 0 & \text{, Otherwise} \end{cases}$

O_{jm}: $\begin{cases} 1 & \text{, if activity } j \text{ is being performed in mode } m \\ 0 & \text{, Otherwise} \end{cases}$

φ_{jmst}: $\begin{cases} 1 & \text{, if activity } j \text{ is started in mode } m \text{ by worker } s \text{ at period } t \\ 0 & \text{, Otherwise} \end{cases}$

ρ_{jmsk}: $\begin{cases} 1 & \text{, if workforce } s \text{ is performing skill } k \text{ of activity } j \text{ in mode } m \\ 0 & \text{, Otherwise} \end{cases}$

δ_{jkt}: $\begin{cases} 1 & \text{, if the execution of skill } k \text{ of activity } j \text{ is in progress at period } t \\ 0 & \text{, Otherwise} \end{cases}$

ST_j: Start time of activity j

FT_j: Finish time of activity j

U_{jk}: Start time of execution of skill k for activity j

F_{jk}: Finish time of execution of skill k for activity j

3.5. Problem Modeling

$$MinZ = \sum_{t=0}^{T} \sum_{m=1}^{M} t.X_{(N+1)mt} \qquad (1)$$

s.t.

$$\sum_{m=1}^{M}\sum_{t=0}^{T} X_{jmt} = 1 \quad ; \quad \forall j \qquad (2)$$

$$\sum_{m=1}^{M}\sum_{t=0}^{T} t.X_{jmt} + \sum_{m=1}^{M} d_{jm}.O_{jm} \le \sum_{m=1}^{M}\sum_{t=0}^{T} t.X_{j'mt} \quad ; \quad \forall (j,j') \in E \qquad (3)$$

$$\rho_{jmsk} \le \lambda_{sk} \quad ; \quad \forall j, \forall s, \forall k, \forall m \qquad (4)$$

$$X_{jmt} \le O_{jm} \quad ; \quad \forall j, \forall m, \forall t \qquad (5)$$

$$\varphi_{jmst} \le O_{jm} \quad ; \quad \forall j, \forall m, \forall s, \forall t \qquad (6)$$

$$\rho_{jmst} \le O_{jm} \quad ; \quad \forall j, \forall m, \forall s, \forall t \qquad (7)$$

$$r_{jkm}.\delta_{jkt} \le b_k \quad ; \quad \forall j, \forall k \in H_j, \forall t \qquad (8)$$

$$ST_j \ge \max_{j' \in pred_j} (FT_{j'}) \quad ; \quad \forall j \qquad (9)$$

$$U_{jk} = \sum_{m=1}^{M}\sum_{t=0}^{T} t.X_{jmt} \quad ; \quad \forall j, \forall k \in H_j \qquad (10)$$

$$ST_j = U_{jk} \quad ; \quad \forall j, \forall k \in H_j \qquad (11)$$

$$ST_{(N+1)} \le T \qquad (12)$$

$$U_{jk} = U_{jk'} \quad ; \quad \forall j, \forall k \ne k' \in H_j \qquad (13)$$

$$F_{jk} = U_{jk} + d_{jm} \quad ; \quad \forall j, \forall k \in H_j \qquad (14)$$

$$FT_j = ST_j + d_j \quad ; \quad \forall j \qquad (15)$$

$$FT_j \ge F_{jk} \quad ; \quad \forall j, \forall k \in H_j \qquad (16)$$

$$\sum_{m=1}^{M}\sum_{t=0}^{T} \varphi_{jmst} \le 1 \quad ; \quad \forall j, \forall s \qquad (17)$$

$$\sum_{m=1}^{M}\sum_{j=0}^{N+1}\sum_{t'=t-d_{jm}+1}^{t} \varphi_{jmst'} \le 1 \quad ; \quad \forall s, \forall t \qquad (18)$$

$$\varphi_{jmst} \le X_{jmt} \quad ; \quad \forall j, \forall m, \forall s, \forall t \qquad (19)$$

$$\varphi_{jmst} + 1 \ge X_{jmt} + \sum_{k=1}^{K} \rho_{jmsk} \quad ; \quad \forall j, \forall m, \forall s, \forall t \qquad (20)$$

$$\sum_{s=1}^{S} \rho_{jmsk} = r_{jmk}.O_{jm} \quad ; \quad \forall j, \forall m, \forall k \qquad (21)$$

$$X_{jmt},\ \varphi_{jmst},\ \rho_{jmsk},\ O_{jm},\ \delta_{jkt} \in \{0,1\} \quad ; \quad \forall j, \forall s, \forall m, \forall k, \forall t \qquad (22)$$

$$ST_j,\ FT_j,\ U_{jk},\ F_{jk},\ Z \ge 0 \quad ; \quad \forall j, \forall k \qquad (23)$$

3.6. Model Description

Equation (1) is the objective function of the MMSRCPSP which aims to minimize the completion time of project. Constraint (2) guarantees that each activity is performed in one mode and it starts exactly once. Constraint (3) takes care of precedence relations. Constraint (4) secures that workforces assigned to skill k of activity j are able to perform this skill. Constraint (5) implies the logical relation between X_{jmt} and O_{jm}. Moreover, the logical relation between φ_{jmst} and O_{jm} is reflected in Constraint (6), while the relation between ρ_{jmst} and O_{jm} is

preserved in Constraint (7). Constraint (8) satisfies resource limitations. Equation (9) determines the start time of activity j . Equation (10) computes the start time of execution of each needed skill. Equation (11) indicates that start time of each task is equal to the start time of its skills. Constraint (12) implies that the project completion time must be less than the project planning horizon. Equation (13) ensures that all required skills of an activity start concurrently. The completion time of a required skill is computed in Equation (14), whereas the finish time of an activity is determined in Equation (15). Constraint (16) states the logical relation between FT_j and F_{jk} . Constraint (17) secures that at most one start time is determined for workers allocated to an activity. Constraint (18) guarantees that workforces assigned to each mode of an activity will not change during the execution of the activity. Constraints (19) and (20) secure that workers allocated to an activity start their work at once. Constraint (21) forces that the number of workers assigned to a task is equal to the number of required workforces to execute the activity. Ultimately, Constraints (22) and (23) indicate the type of decision variables.

4. Solution Approach

4.1. Solution representation

In this paper, each chromosome is represented as a $3\times N$ vector, where N is the number of positions in each row. Each position is associated to a project activity. The first row of each chromosome is devoted to a precedence-feasible activity list. The first row which is known as schedule representation shows the priority structure between the project activities. Two of the most important schedule representations for the RCPSP are (Hartmann and Kolisch, 2000): (1) the random-key (RK) representation, and (2) the activity-list (AL) representation. According to various experimental tests conducted by Hartmann and Kolisch (2000), the procedures based on activity-list representations are superior to other approaches. Hence, we use the AL representations to encode a project schedule. An activity list is a sequence of activities which represents the relative priority of each activity in comparison to other tasks. Each AL is a vector $U=(j_0, j_1, ..., j_N, j_{N+1})$ in which the priority of the activity at the $l-th$ position is higher than the priority of any activity at $l'-th$ position for which $l'>l$. We assume that j_0 and j_{N+1} belong to the activities 0 and $N+1$, respectively. In a precedence-feasible activity list, each activity is positioned after all its immediate predecessors (Ranjbar et al., 2008). The second row determines the processing modes of activities, while the third row shows the workforces assigned to each task. A sample chromosome is depicted in Figure 1. Suppose that activity "2" requires skills "1" and "4". This activity needs two workers in each period to perform skill "1" and it needs one worker to execute skill "4" in each period. Based on Figure 1, the workers "2" and "4" are assigned to activity "2" to perform skill "1", while the worker "1" is allocated to this activity to perform skill "4".

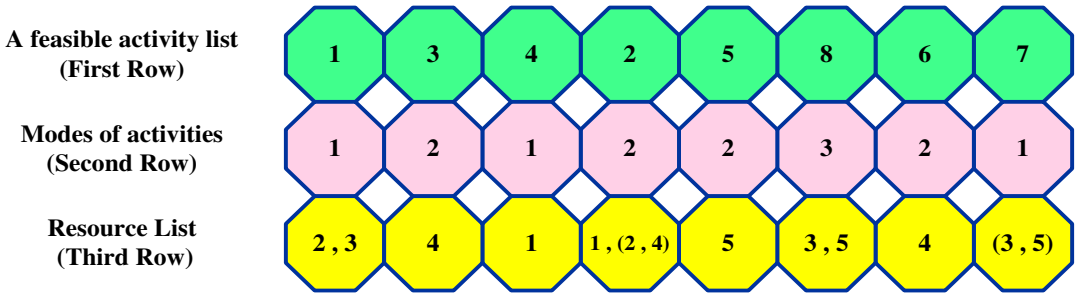

Fig.1. Example of chromosome representation

4.2. The proposed genetic algorithm (PGA)

In this section, a meta-heuristic algorithm is proposed to solve the MMSRCPSP. The structure of the proposed meta-heuristic is based on the genetic algorithm. The genetic algorithm has always been one of the most successful algorithm for optimization problems (Najafi and Salimi, 2018). For the proposed method, we present a novel tournament selection operator based on a hybrid MADM method to select the best parents for generating new individuals. Moreover, two new strategies are presented to explore the solution space more accurately. For the proposed algorithm, new crossover and mutation operators are designed to produce feasible solutions. Figure 2 shows the overall structure of the proposed algorithm. The details of the developed genetic algorithm are described as follows.

Algorithm: The proposed genetic algorithm (PGA)	
1.	**Initialization:**
1.1.	Set the input parameters and generate a random initial population with size of $Npop$;
2.	**Evaluation:**
2.2.	Evaluate individuals of initial population;
3.	**Selection (Proposed tournament selection operator):**
3.1.	Select several solutions randomly;
3.2.	Use Shannon's entropy method to calculate weights of criteria described in Section 4.3;
3.3.	Employ the VIKOR method to prioritize the selected solutions;
3.4.	Keep the best ranked solution and repeat the selection procedure to choose another individual;
4.	**Crossover:**
4.1.	Apply crossover operator on chosen parents to produce offspring;
5.	**Mutation:**
5.1.	Apply mutation operator on the chosen individual;
6.	**Combination:**
6.1.	Merge the population with the individuals generated by crossover and mutation operators;
7.	**Elitism:**
7.1.	Select the best $Npop$ individuals of the combined population;
8.	**Proposed strategies:**
8.1.	Generate an integer random number (IRN) on the interval [1, 2];
8.2.	if $IRN = 1$
8.3.	Use FPES described in Section 4.6 to explore solution space deeper;
8.4.	elseif $IRN = 2$
8.5.	Use SPES described in Section 4.7 to explore solution space deeper;
8.6.	endif
9.	**Termination:**
9.1.	Repeat the above procedure until the stopping criterion is met;

Fig. 2. Structure of the proposed genetic algorithm

4.3. Proposed tournament selection

In this section, a new tournament selection operator is proposed to choose appropriate individuals for reproduction. The basic tournament selection operator selects a predefined number of random chromosomes of the population. The selected chromosomes compete with each other based on their fitness values. The solution with the best fitness value will be chosen as a parent. This procedure repeats to find another parent (Miller and Goldberg, 1996). The drawback of this method is that the chromosome with the least fitness value will never find a chance to be chosen. The selection procedure should consider equal opportunities for all individuals to be chosen as participants in the reproduction. The proposed tournament selection operator gives chance to all individuals to be selected as parents. Hence, the probability of finding diverse solutions is increased substantially. Based on the proposed method, several chromosomes are chosen randomly from the population. For the chosen chromosomes, we consider two criteria as follows:

1. The fitness value of solutions. Since the make-span of the project is considered as the fitness value of each schedule, lower fitness values are preferred (negative criterion).
2. For each selected chromosome, a real random number on the interval [0, 1] is generated. These random numbers indicate the probability of selection for reproduction. The higher the probability of a solution, the higher the chance of selection for reproduction (positive criterion).

Based on the description given above, a decision matrix is created, where the rows and columns indicate the chosen chromosomes and criteria, respectively. In the next step, the chromosomes are ranked based on a hybrid multi-attribute decision making technique called the Entropy-VIKOR. This hybrid MADM technique employs the Shannon's entropy (Lotfi and Fallahnejad, 2010) method to determine the weights of criteria. Since different decision matrices are obtained in each iteration, the Shannon's entropy method is utilized to determine the weights of criteria with respect to the chosen chromosomes. The advantage of Shannon's entropy method is that the weights of criteria are calculated based on characteristics of alternatives rather than biased judgments. Thereafter, the VIKOR method (Opricovic and Tzeng, 2004) is used to prioritize the alternatives. The VIKOR method enables the proposed tournament selection operator to choose the best individual in terms of both criteria in each iteration.

The Shannon's entropy method is one of the renowned methods to acquire the weights of criteria and it consists of the following steps (Lotfi and Fallahnejad, 2010):

Step 1. Create a decision matrix, where the rows and the columns represent the chromosomes and criteria, respectively. Suppose that the decision matrix has $\Gamma(\gamma = 1,...,\Gamma)$ rows and $CR(c = 1,...,CR)$ columns, respectively. The element in the intersection of γ^{th} row and c^{th} column is denoted as $a_{\gamma c}$.

Step 2. Normalize the decision matrix. This step deals with normalizing raw elements of decision matrix when elements have different measurement units and scales. Normalization process makes the variables comparable to each other. In this process, the elements of decision matrix measured on different scales are transformed into a common scale. The normalized decision matrix is obtained using the following formula (Lotfi and Fallahnejad, 2010):

$$\vartheta_{\gamma c} = \frac{a_{\gamma c}}{\sum_{\gamma=1}^{\Gamma} a_{\gamma c}} \qquad \forall c = 1,2,...,CR \tag{24}$$

Where, $\vartheta_{\gamma c}$ is the normalized element in the intersection of γ^{th} row and c^{th} column.

Step 3. Compute the entropy of each criterion. The entropy of c^{th} criterion is denoted as Ent_c calculated by Eq. 25 (Lotfi and Fallahnejad, 2010):

$$Ent_c = -Ent_0 \sum_{\gamma=1}^{\Gamma} \vartheta_{\gamma c}.Ln\left(\vartheta_{\gamma c}\right) \quad \forall c = 1,2,...,CR \tag{25}$$

Where, Ent_0 denotes the entropy constant and can be computed by Eq. 26 [95]:

$$Ent_0 = (Ln(\Gamma))^{-1} \tag{26}$$

Step 4. Compute the degree of diversification for each criterion. The degree of diversification for c^{th} criterion is represented as τ_c, which can be computed as follows (Lotfi and Fallahnejad, 2010):

$$\tau_c = 1 - Ent_c \qquad \forall c = 1,2,...,CR \tag{27}$$

Step 5. Calculate the importance of each criterion using the following formula (Lotfi and Fallahnejad, 2010):

$$w_c = \frac{\tau_c}{\sum_{c=1}^{CR} \tau_c} \qquad \forall c = 1,2,...,CR \tag{28}$$

Where, w_c is the weight of c^{th} criterion. Having obtained the normalized weights of criteria by the Shannon's entropy method, the VIKOR method is applied to prioritize chromosomes based on their fitness value and corresponding probability of selection. The VIKOR method gives a compromise-ranking list based on the weights of criteria (Opricovic and Tzeng, 2004). According to this method, the alternatives are ranked based on the measure of closeness to the ideal solution. The VIKOR method consists of the following steps to rank the chromosomes (Opricovic and Tzeng, 2004; Opricovic and Tzeng, 2007):

Step 1. Determine the best (f_c^*) and the worst (f_c^-) values of all criteria among solutions. f_c^* and f_c^- are obtained by Eqs. 29 and 30, respectively if the c^{th} metric represents a benefit (Opricovic and Tzeng, 2004):

$$f_c^* = \max_{\gamma} f_{\gamma c} \qquad \forall \gamma = 1,2,...,\Gamma \qquad \forall c = 1,2,...,CR \tag{29}$$

$$f_c^- = \min_{\gamma} f_{\gamma c} \qquad \forall \gamma = 1,2,...,\Gamma \qquad \forall c = 1,2,...,CR \tag{30}$$

On the other hand, if the c^{th} metric represents a cost, f_c^* and f_c^- are acquired by Eqs. 31 and 32, respectively (Opricovic and Tzeng, 2004):

$$f_c^* = \min_{\gamma} f_{\gamma c} \qquad \forall \gamma = 1,2,...,\Gamma \qquad \forall c = 1,2,...,CR \tag{31}$$

$$f_c^- = \max_{\gamma} f_{\gamma c} \qquad \forall \gamma = 1,2,...,\Gamma \qquad \forall c = 1,2,...,CR \tag{32}$$

Step 2. Calculate the maximum group utility (ξ_γ) and the minimum individual regret of the opponent (R_γ) values by the following formulas (Opricovic and Tzeng, 2004):

$$\xi_\gamma = \sum_{c=1}^{CR} w_c \left(f_c^* - f_{\gamma c}\right) / \left(f_c^* - f_c^-\right) \qquad \forall \gamma = 1,2,...,\Gamma \tag{33}$$

$$R_\gamma = \max_{c} \left[w_c \left(f_c^* - f_{\gamma c}\right) / \left(f_c^* - f_c^-\right) \right] \qquad \forall \gamma = 1,2,...,\Gamma \tag{34}$$

Step 3. Compute the VIKOR index for each alternative (Q_γ) ($\forall \gamma = 1,...,\Gamma$) using Eq. 35 (Opricovic and Tzeng, 2004):

$$Q_\gamma = \upsilon\left(\xi_\gamma - \xi^*\right)/\left(\xi^- - \xi^*\right) + (1-\upsilon)\left(R_\gamma - R^*\right)/\left(R^- - R^*\right) \quad \forall \gamma = 1,2,...,\Gamma \tag{35}$$

Where, υ and $(1-\upsilon)$ represent the weights for the strategies of maximum group utility and individual regret, respectively. In this paper, υ is set to 0.5. In Eq. 35, $\xi^* = \min \xi_\gamma$, $\xi^- = \max \xi_\gamma$, $R^* = \min R_\gamma$, and $R^- = \max R_\gamma$.

Step 4. Rank the alternatives (chromosomes) by sorting the values ξ, R and Q in ascending order. Three ranking lists are obtained by sorting these values.

Step 5. Propose the alternative with the minimum VIKOR index as a compromise solution. The solution with the least VIKOR index is the best-ranked solution if the following conditions are satisfied (Opricovic and Tzeng, 2007):

Condition 1. The first condition is known as "Acceptable advantage":

$$Q(a'') - Q(a') \geq DQ \tag{36}$$

Where, a' and a'' are the first and second alternatives in the ranking list of Q. DQ is computed using the following formula:

$$DQ = 1/\left(\Gamma - 1\right) \tag{37}$$

Condition 2. The second condition is called "Acceptance stability in decision-making". To meet this condition, the alternative a' must have the best rank in ranking lists of ξ and R as well. This compromise solution is consistent within a decision-making procedure. If one of the conditions is not met, a set of compromise solutions is presented as follows:

1. When the second condition is not satisfied, the alternatives a' and a'' are proposed.
2. When the first condition is not met, the alternatives $a', a'', ..., a^{(P)}$ are proposed. $a^{(P)}$ is acquired by the following relation:

$$Q(a^{(P)}) - Q(a') < DQ \tag{38}$$

Suppose that the third, the fifth, the tenth, the fifteenth and the twentieth solutions have been chosen randomly from the population for tournament selection. Table 1 shows the decision matrix and the normalized decision matrix, where the rows and columns represent the chromosomes and criteria, respectively.

Table 1
The decision matrix and the normalized decision matrix of the example

Solutions	Decision matrix		Normalized decision matrix	
	First criterion (-)	Second criterion (+)	First criterion (-)	Second criterion (+)
3	59	0.4375	0.1578	0.1503
5	65	0.5270	0.1738	0.1810
10	71	0.6547	0.1898	0.2249
15	87	0.7134	0.2326	0.2451
20	92	0.5785	0.2460	0.1987

Based on the normalized decision matrix, the weights of the first and the second criteria have been obtained as $w_1 = 0.5071$ and $w_2 = 0.4929$, respectively. Table 2 reports the weighted normalized decision matrix.

Table 2
The weighted normalized decision matrix of the example

Solutions	Weighted normalized decision matrix	
	First criterion (-)	Second criterion (+)
3	0.0800	0.0741
5	0.0881	0.0892
10	0.0962	0.1109
15	0.1180	0.1208
20	0.1247	0.0979

Thereafter, the VIKOR method is used to rank the chromosomes. Table 3 shows the values of ξ, R and Q for all chromosomes. The results show that the tenth solution has achieved the best rank among all chromosomes. The tenth solution is selected as one of the parents for reproduction and other solutions will return to the population. This process continues to find another parent. The structure of the proposed tournament selection operator is depicted in Figure 3.

Table 3
The ranking of solutions

	ξ, R and Q values of solutions					Ranking of solutions				
	3	**5**	**10**	**15**	**20**	**3**	**5**	**10**	**15**	**20**
ξ	0.4929	0.4252	0.2893	0.4303	0.7481	4	2	1	3	5
R	0.4929	0.3330	0.1844	0.4303	0.5071	4	2	1	3	5
Q	0.6999	0.3784	0.0000	0.5346	1.0000	4	2	1	3	5

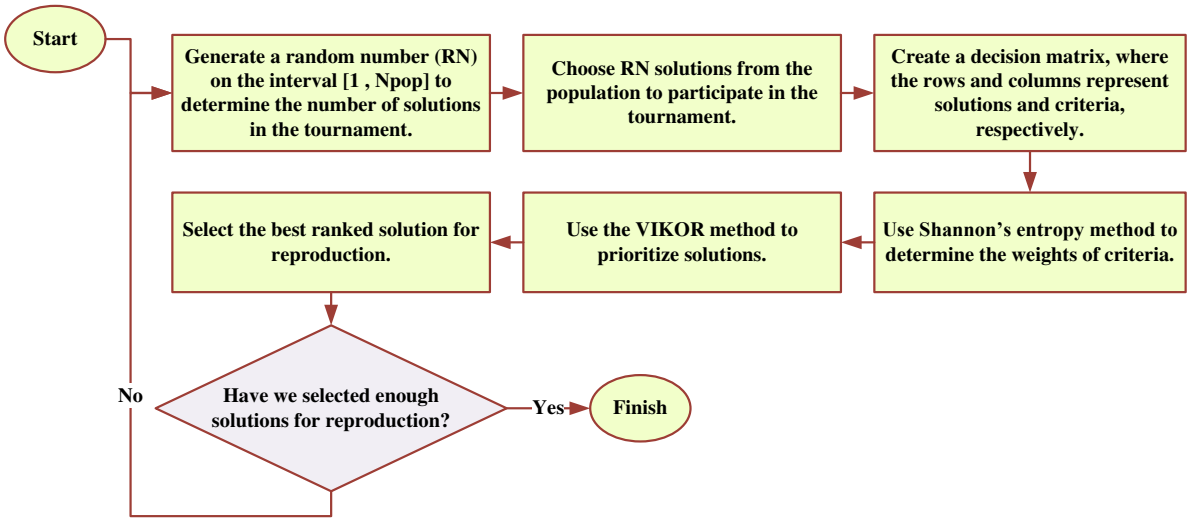

Fig. 3. Procedure of the proposed tournament selection

4.4. Crossover operator

The crossover operator uses two parents selected by the procedure described in Section 4.3. These parents are denoted as $P1$ and $P2$ to generate two offspring represented as $CH1$ and $CH2$. Having selected $P1$ and $P2$, the following phases are required to produce $CH1$ and $CH2$.

- **First Phase (Generating feasible activity lists)**

Two random integer numbers are generated on the interval [1, N] to select two activities from $P1$ and $P2$, respectively. The selected activities are denoted as $SA1$ and $SA2$. To generate precedence feasible activity lists, the chosen activities can be placed anywhere between their nearest predecessor and nearest successor (Bouleimen and Lecocq, 2003). In this respect, the positions of the nearest predecessor and the nearest successor of the selected tasks are identified on activity lists of $P1$ and $P2$. The feasible activity list of $CH1$ is created by locating $SA1$ in a random position between its nearest predecessor and nearest successor on $P2$. The feasible activity list of $CH2$ is produced in a similar way. The $SA2$ is placed in a random position between its nearest predecessor and successor on $P1$. Figure 4 shows activity lists of two solutions as $P1$ and $P2$. Let $SA1$ and $SA2$ be "4" and "5", respectively. The highlighted genes on $P1$ show the positions between the nearest predecessor and the nearest successor of activity "5". Similarly, the highlighted genes on $P2$ depict the positions between the nearest predecessor and the nearest successor of activity "4". An activity list is created for $CH1$ by placing the activity 4 anywhere within the highlighted genes on the second parent. The activity list of $CH2$ is produced by locating the activity 5 within the highlighted genes on the first parent. Assume that the sixth position of $P2$ has been chosen for activity 4 to move. Besides, the activity 5 moves to the fifth gene of $P1$. Figure 4 demonstrates the precedence-feasible activity lists of $CH1$ and $CH2$.

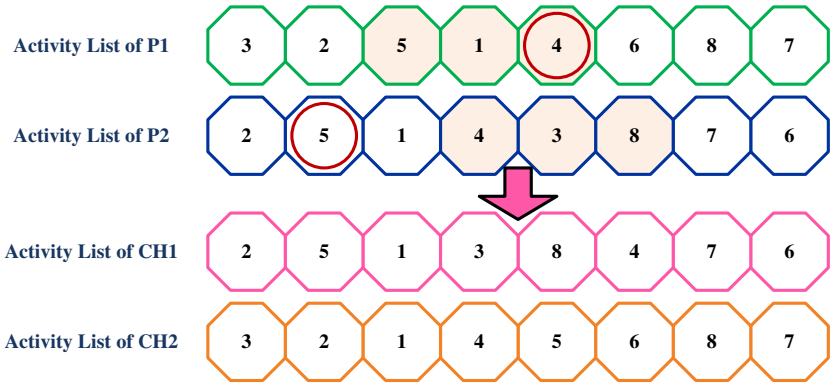

Fig. 4. Generating feasible precedence activity lists for CH_1 and CH_2

- **Second Phase (Generating lists of assigned modes)**

The proposed crossover chooses a predetermined number of random activities. The modes assigned to selected activities on $P1$ will be allocated to these activities on $CH2$. The remaining activities on $CH2$ take their execution modes from $P2$. On the other hand, the execution modes of the selected activities on $P2$ will be assigned to these tasks on $CH1$. The remaining activities on $CH1$ take their processing modes from $P1$. Suppose that activities "2", "3" and "4" have been chosen randomly to change their processing modes on $CH1$ and $CH2$. The modes assigned to activities "2", "3" and "4" on $P1$ will be the modes of these tasks on $CH2$. On the opposite side, the modes of selected activities on $P2$ will be the modes of these activities on $CH1$. The procedure of determining the modes of activities on $CH1$ and $CH2$ is illustrated in Figure 5.

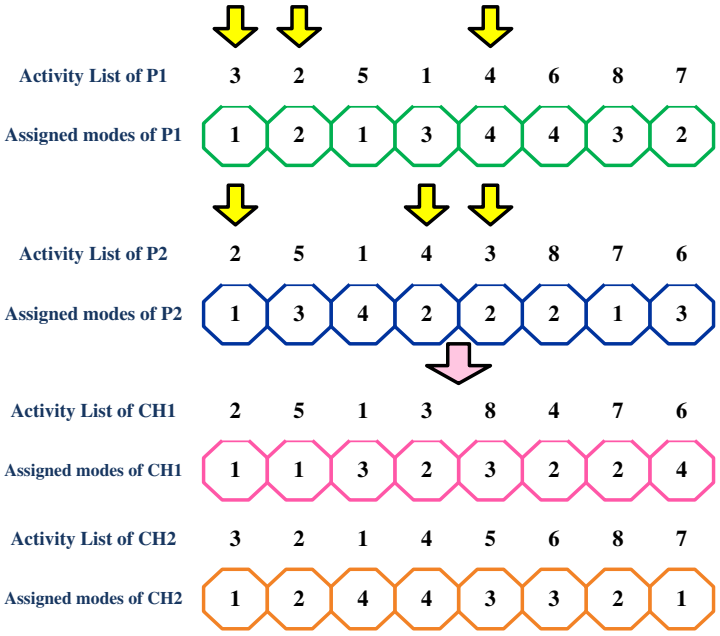

Fig. 5. Determining modes of activities on the offspring

- **Third phase (Generating resource lists)**

The selected activities in the second phase are considered in this phase as well. The workforces allocated to selected tasks on $P1$ will be assigned to these activities on $CH2$. The remaining activities on $CH2$ take their required workers from $P2$. On the opposite side, the workforces assigned to selected activities on $P2$ will be allocated to these tasks on $CH1$. The remaining activities take their required workers from $P1$. Figure 6 illustrates the third row of $CH1$ and $CH2$.

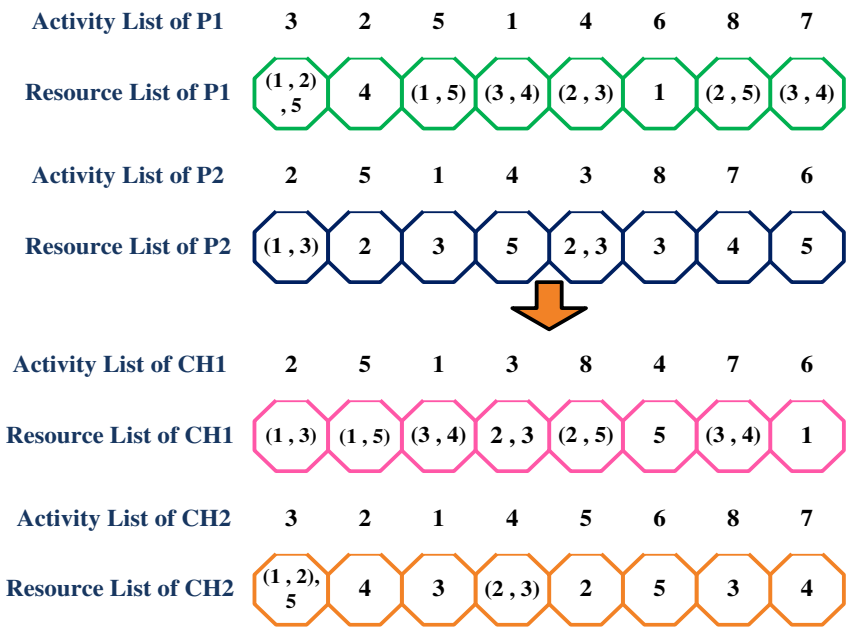

Fig.6. Procedure used to generate resource lists of offspring

4.5. Mutation operator

Mutation is an operator used in the genetic algorithm that changes one or several genes of a chromosome to generate new individuals. This operator prevents the algorithm from converging to a local optima (Deb et al., 2002). To use the mutation operator, a solution is selected randomly from the population as a candidate to be mutated. Since the solution representation in this paper consists of three parts, the proposed mutation operator in this paper generates a new individual in three phases. These phases are described as follows.

- **First Phase (Generating feasible activity lists)**

In this section, a heuristic is proposed to produce a feasible activity list from a candidate solution. This method selects a random activity from the activity list of the candidate solution at the initial step. Then, the positions of the nearest predecessor and the nearest successor of the selected activity are determined. These positions form a set called FP [2]. FPj is a set consisting of the positions between the nearest predecessor and the nearest successor of the activity j. To determine the new position of the activity j, the length of the FPj is multiplied by a real random number on the interval [0, 1]. The outcome will be rounded up which is called q. Thereafter, the $q-th$ member of FPj is chosen as the new position of the activity j. Figure 7 shows an activity-on-node network of a sample project, where the activities 0 and 9 are the dummy start and finish tasks, respectively.

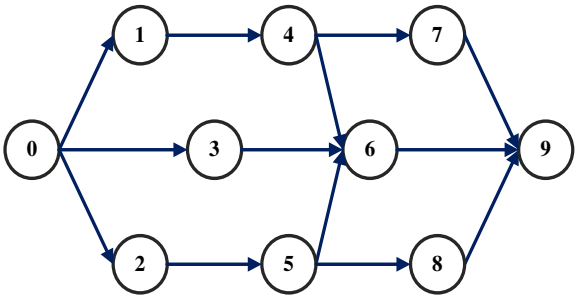

Fig.7. AON network of the example

Figure 8 shows an activity list for the project illustrated in Figure 7. Suppose that activity "4" has been chosen to change its position in order to generate a new precedence-feasible activity list. Hence, the $FP4$ will be $\{2,3,4,5,6\}$. For $q=0.93$, we have $\lceil 5\times 0.93 \rceil = 5$. Thus, the fifth member of the $FP4$ determines the new position of activity "4". The activity "4" takes the sixth position and a new precedence feasible activity list is created. Figure 8 shows the new activity list.

2. Feasible positions (FP)

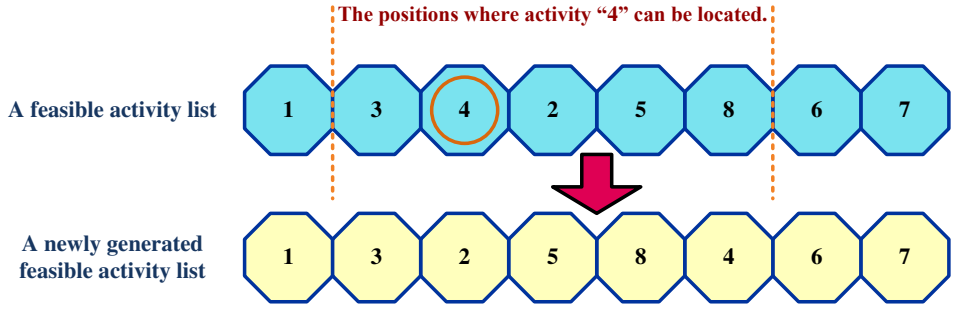

Fig. 8. Procedure used to produce feasible activity list of mutant

- **Second phase(Generating lists of assigned modes)**

A new procedure is presented to determine the modes assigned to activities on a mutated solution. In this procedure, several activities are selected randomly from the candidate solution to change their corresponding execution modes. The new mode of the selected activity j (NMj) on the mutated solution is determined using the following formula:

$$NM_j = \lceil M - CM_j + rand() \rceil \quad (39)$$

.

Where, M denotes the number of available modes and CMj represents the current mode of activity j on the candidate solution. $rand()$ is a random number from the interval [0,1]. Suppose that activities "1", "4" and "5" have been selected randomly from the candidate solution to change their corresponding modes on the mutated solution. Figure 9 shows the assigned modes of the activities on the mutated solution

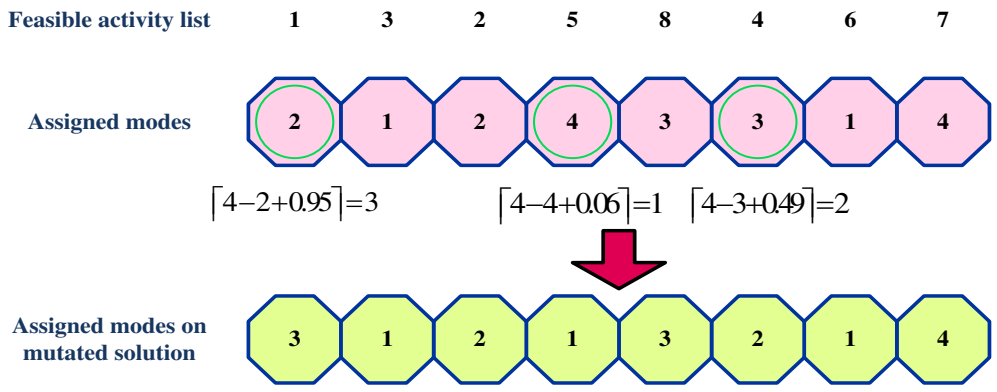

Fig. 9. Procedure used to determine modes of activities on mutant

- **Third phase (Generating resource lists)**

Several activities are chosen randomly to change their corresponding workforces. Due to the modes assigned to activities on mutated solution, the workers are altered. The workforces of chosen tasks will be substituted with other workers who are capable to perform required skills.

4.6. First Proposed Exploring Strategy (FPES)

The procedure initiates with generating a random number (RN) on the interval $[1, Npop]$. For RN number of solutions in the population, a hierarchical neighborhood structure is generated. In this structure, for each solution, a specific number of neighboring solutions are generated. The mutation operator is used to produce neighboring solutions. Each hierarchical neighborhood structure consists of multiple levels. Suppose that the first and the tenth solutions have been selected from the population. Figure 10 shows the hierarchical neighborhood structures for these solutions.

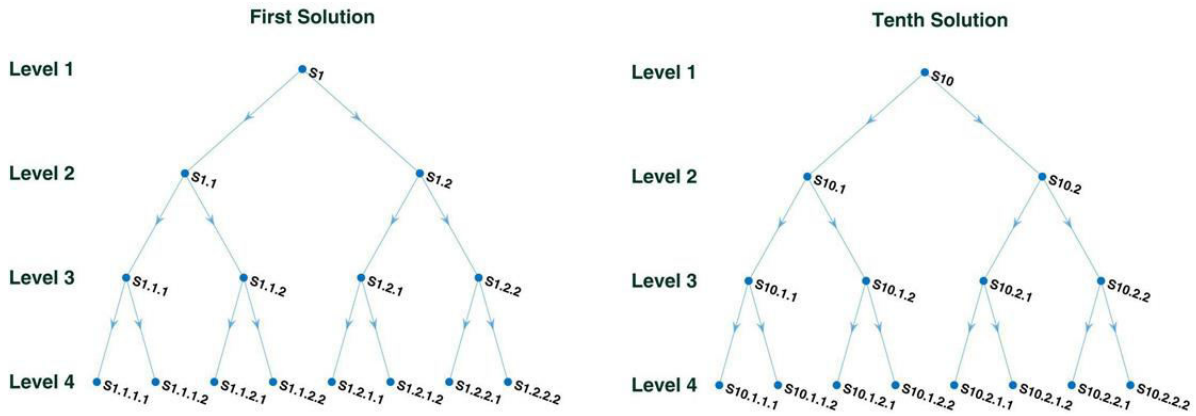

Fig. 10. Hierarchical neighborhood structures for the first and the tenth solutions

Neighboring solutions are compared to each other in terms of fitness value to determine the best individual. Then, the best neighboring solution is compared to the chromosome which belongs to the upper level. If the best neighboring individual has better fitness value, it wins the competition against the solution of the upper level. Otherwise, the best neighboring solution will be the winner with a probability $\Pr(\Pr \in [0,1])$. This approach helps to explore the solution space to find diverse chromosomes. The number of levels (NL) and the number of solutions in each level (NSL) are two of the most significant parameters that affect the quality of exploring the solution space. These comparisons are performed from the lowest level to the first level, continuously. Figure 11 illustrates the tournament held between the neighboring solutions of the tenth solution of the population. As shown in Figure 11, the S10.1.1.1 has won the competition against S10. Thus, the S10.1.1.1 will take the place of S10 in the population and it will be transferred to the next iteration as a member of population.

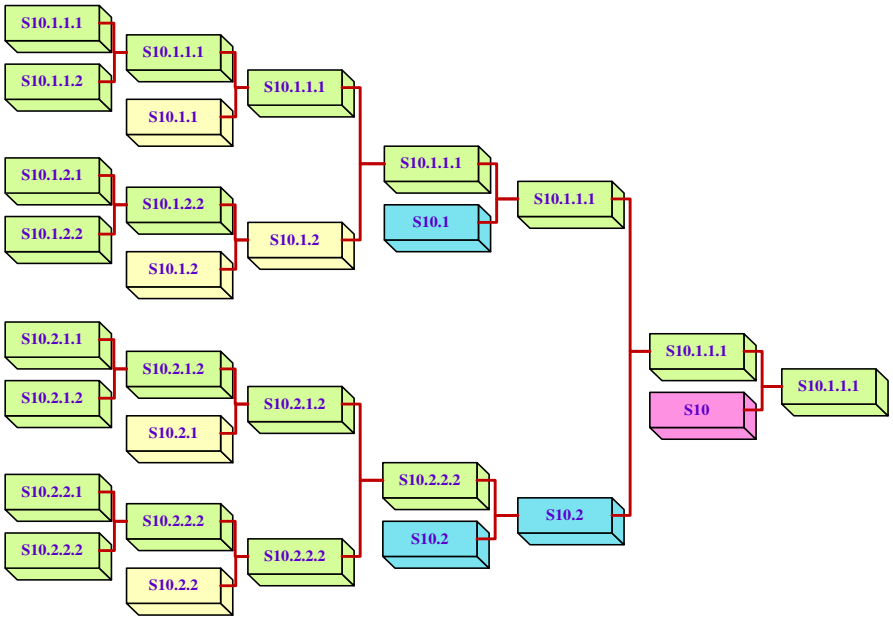

Fig. 11. Tournament held between neighboring solutions of the tenth solution

The higher the values of NL and NSL, the higher the chance of finding high quality and diverse solutions. However, higher values of NL and NSL will increase the computational time, significantly. Hence, it is crucial to set these parameters in such a way that not only diverse and high quality solutions are found, but also the computational time is not substantially increased.

4.7. Second Proposed Exploring Strategy (SPES)

This heuristic requires NL ($NL > 1$) and NSL parameters as well. At the initial step, for each solution in the population, we generate NSL neighboring solutions using the mutation operator. The newly generated solutions are compared to each other to determine the individual with the best fitness value. If the fitness values of both neighboring solutions are the same, one of them is chosen randomly. The next neighboring solutions for the next

level are generated from the chosen solution in the upper level. This procedure continues for a predefined number of levels (NL). Having reached the $NL-th$ level, the process stops and the chosen solutions of all levels are compared to each other in order to detect the best individual. In case of prevailing the current solution in the population, the best neighboring solution will replace the current one in the population. Let $NL=4$ and $NSL=2$. Figure 12 depicts this procedure for the tenth solution of the population.

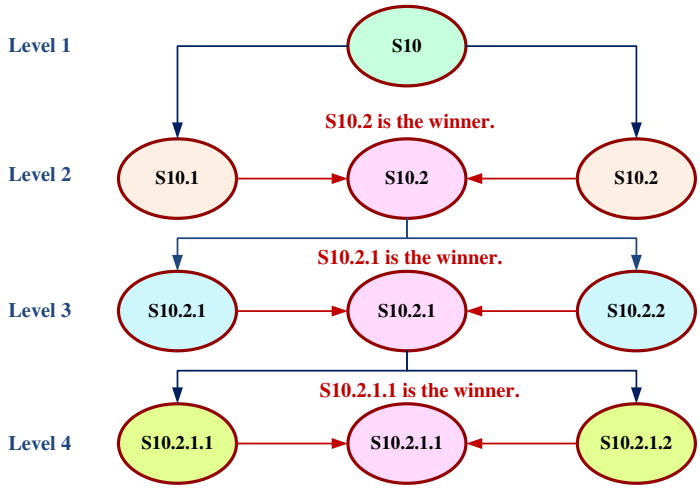

Fig.12. Competitions between solutions in SPES

As illustrated in Figure 13, the S10.2.1.1 has not only won the competition against its predecessors, but also it defeats the S10 as well. Thus, the S10.2.1.1 will take the place of the S10 in the population.

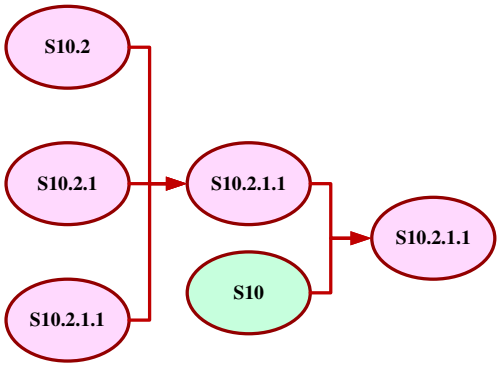

Fig.13. Ultimate winner in SPES

5. Experimental Results

We calibrate the parameters of the MMSRCPSP in Section 5.1, while the input parameters of algorithms are tuned via the Taguchi method in Section 5.2. As mentioned previous sections, the proposed algorithm is compared with the harmony search method (Mehdizadeh and Kivi, 2014), classical genetic algorithm (Hassanpour et al., 2017), and the state-of-the-art neurogenetic algorithm (Agarwal et al., 2015). The performance measures used to compare the algorithms are described in Section 5.3. In Section 5.4, the algorithms are compared to each other in terms of performance measures. All optimizers were coded in the MATLAB R2015b software and run on a personal computer with a 2.33 GHz Intel Quad Core CPU and 4-GB RAM.

5.1. Tuning parameters of MMSRCPSP

To evaluate the effectiveness of the algorithms, 40 test instances have been generated using ProGen software. Since there are no specific test problems for the MMSRCPSP, the chosen test problems have to be tuned by adding some additional data. Table 4 shows the required data for test problems.

Table 4
The values of parameters

Parameters	Value
The number of required skills (K) for the whole project	$U[3,10]$
The required number of skills by an activity	$U[1,K]$

The required number of workers to perform each skill in a determined mode	$U[1,3]$
The available number capable workforces to perform skill k	$U[1,5]$

We classify test problems into small size and large size instances. Test problems 1 to 20 are projects with 30 non-dummy activities considered as small size benchmark instances, while test problems 21 to 40 are projects with 120 non-dummy activities considered as large size benchmark problems. Table 5 shows the characteristics of test problems.

Table 5
Characteristics of test instances

Problem No.	N	M	S	K	Problem No.	N	M	S	K
1	32	4	10	3	21	122	6	13	7
2	32	4	10	3	22	122	6	13	7
3	32	4	10	3	23	122	6	13	7
4	32	4	10	3	24	122	7	13	7
5	32	4	10	3	25	122	7	13	7
6	32	4	10	4	26	122	7	14	8
7	32	4	10	4	27	122	7	14	8
8	32	4	10	4	28	122	7	14	8
9	32	5	10	4	29	122	7	15	8
10	32	5	12	4	30	122	7	15	8
11	32	5	12	5	31	122	7	15	9
12	32	5	12	5	32	122	8	15	9
13	32	5	12	5	33	122	8	15	9
14	32	6	12	5	34	122	8	15	9
15	32	6	13	5	35	122	8	15	9
16	32	6	13	6	36	122	8	16	10
17	32	6	13	6	37	122	8	16	10
18	32	6	13	6	38	122	9	16	10
19	32	6	13	6	39	122	9	16	10
20	32	6	13	6	40	122	9	16	10

5.2. Parameter tuning of algorithms

Since the performance of an algorithm depends excessively on the value of its input parameters, it is crucial to utilize an effective tool to find an appropriate set of values for parameters. In this paper, we use the Taguchi method as one of the design of experiments (DOE) techniques to set the parameters of algorithms. Based on the Taguchi method, there are two types of factors affecting a process: (1) control factors, and (2) noise factors. Control factors can be set by the designer during experimentations, while noise factors are uncontrollable factors that affect the performance of a process by causing variability. The aim of the Taguchi method is to find the best possible setting of control factors in order to minimize the effects of noise factors (Afruzi, et al., 2014). To tune control factors, the Taguchi method presents a set of experiments based on Orthogonal Arrays (OA). The Taguchi method proposes a statistic called signal-to-noise (S/N) ratio to assess the performance of a process. "Signal" and "Noise" represent mean response variable and standard deviation, respectively. In this respect, the higher the values of signal-to-noise ratio, the better the performance of a process (Afruzi, et al., 2014). There are three main types of signal to noise ratios: (1) smaller is better, (2) larger is better, and (3) nominal is better. In this study, we use "smaller is better" signal-to-noise ratio due to our response variable. The S/N ratio for "smaller is better" type of response variables can be calculated using Eq. 40 (Afruzi, et al., 2014):

$$S/N = -10 \times \log\left[\frac{S\left(Y^2\right)}{n}\right] \tag{40}$$

Where, Y and n represent the response variable and the number of orthogonal arrays, respectively. In the following, the large size test problems generated by the ProGen software has been used to conduct the Taguchi experiments. Table 6 shows five levels considered for parameters of algorithms as control factors.

Table 6
Parameters and levels for Taguchi trials

Algorithms	Parameters	Symbols	Parameter Levels				
			Level 1	Level 2	Level 3	Level 4	Level 5
PGA	Crossover rate	*pc*	0.65	0.70	0.75	0.80	0.85
	Mutation rate	*pm*	0.10	0.15	0.20	0.25	0.30
	Number of solutions in population	*Npop*	20	50	100	200	300
	Maximum number of iterations	*MaxIt*	100	200	300	400	500
	Number of levels	*NL*	2	3	4	5	6
	Number of solutions in each level	*NSL*	2	3	4	5	6
HS	Harmony memory size	*HMS*	20	50	100	200	300
	Maximum number of iterations	*MaxIt*	100	200	300	400	500
	Harmony Memory Consideration Rate	*HMCR*	0.75	0.80	0.85	0.90	0.95
	Pitch Adjustment Rate	*PAR*	0.10	0.30	0.50	0.70	0.90
GA	Crossover rate	*pc*	0.65	0.70	0.75	0.80	0.85
	Mutation rate	*pm*	0.10	0.15	0.20	0.25	0.30
	Number of solutions in population	*Npop*	20	50	100	200	300
	Maximum number of iterations	*MaxIt*	100	200	300	400	500
Neurogenetic	Crossover rate	*pc*	0.65	0.70	0.75	0.80	0.85
	Mutation rate	*pm*	0.10	0.15	0.20	0.25	0.30
	Number of solutions in population	*Npop*	20	50	100	200	300
	Maximum number of iterations	*MaxIt*	100	200	300	400	500
	Number of inter-leavings	*NOI*	2	3	4	5	6

Five test instances are chosen randomly and each test problem is run for three times. Consequently, 15 results are obtained for each experiment. The best obtained result among three runs is considered as the result of each test problem. The project completion time of each test instance is transformed into the relative percentage deviation (RPD). Eq. 41 can be used to compute the RPD (Gao, et al., 2013):

$$RPD = \left| \frac{Best_{sol} - Method_{sol}}{Best_{sol}} \right| \times 100 \quad 0 \le RPD \le 100 \tag{41}$$

In the above equation, $Best_{sol}$ is the best objective function value among all acquired values. $Method_{sol}$ is the objective function value obtained by an algorithm. The Minitiab 13 software has been used to depict S/N plots of algorithms. Figure 14 illustrates the corresponding S/N plots of all three algorithms. Based on the Figure 14, the optimal levels of parameters have been depicted in Table 7.

Table 7
Optimal values for input parameters of algorithms

Algorithms	Optimal levels
PGA	$pc = 0.65$, $pm = 0.30$, $Npop = 300$, $MaxIt = 500$, $NL = 6$, $NSL = 6$
HS	$HMS = 300$, $MaxIt = 500$, $HMCR = 0.90$, $PAR = 0.30$
GA	$pc = 0.85$, $pm = 0.30$, $Npop = 300$, $MaxIt = 500$
Neurogenetic	$pc = 0.65$, $pm = 0.30$, $Npop = 300$, $MaxIt = 500$, $NOI = 6$

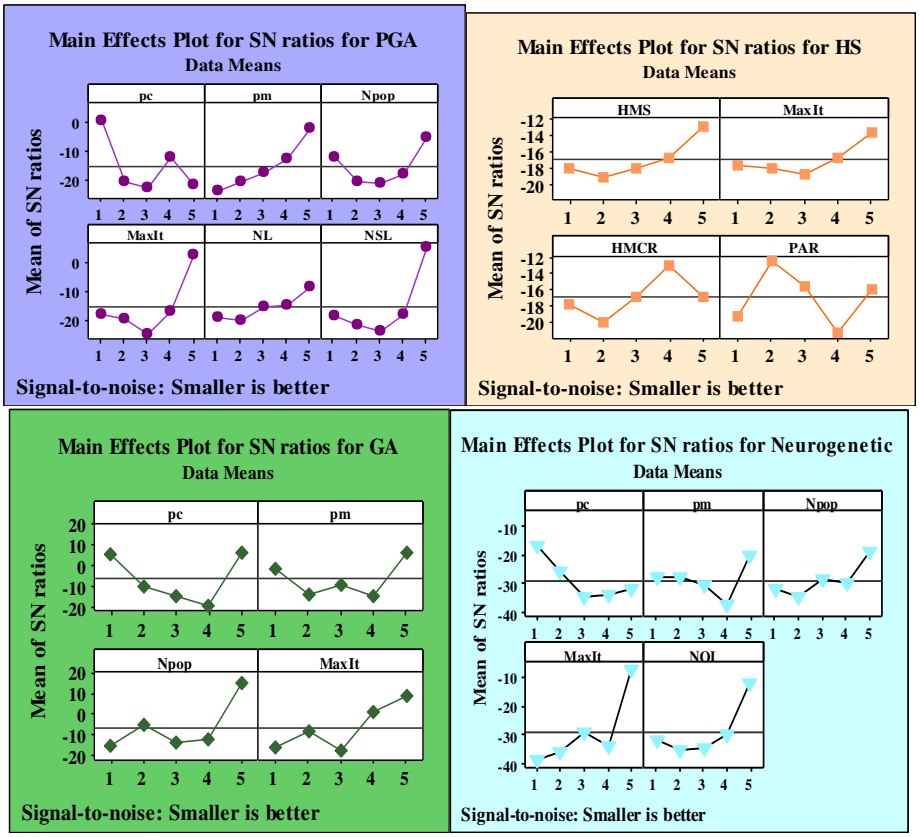

Fig. 14. *S/N* ratio plots for the PGA, HS, GA, and Neurogenetic

Table 8
Comparisons between algorithms for small size problems in terms of ARPD, SDRPD, RPD* and WRPD

Prob. No	Performance Measures															
	ARPD				SDRPD				RPD*				WRPD			
	PGA	HS	GA	NeuroGA*	PGA	HS	GA	NeuroGA*	PGA	HS	GA	NeuroGA*	PGA	HS	GA	NeuroGA*
1	0.00	11.60	0.00	0.00	0.00	10.00	0.00	0.00	0.00	0.83	0.00	0.00	0.00	27.75	0.00	0.00
2	0.00	8.30	15.09	7.45	0.00	4.68	8.29	4.30	0.00	1.55	1.19	4.44	0.00	13.67	21.51	13.85
3	0.00	13.28	11.74	0.00	0.00	6.57	12.05	0.00	0.00	7.05	1.16	0.00	0.00	24.13	28.38	0.00
4	0.00	16.05	14.42	9.84	0.00	8.48	3.69	4.41	0.00	3.16	9.08	5.52	0.00	25.78	18.71	14.13
5	0.00	0.00	16.18	8.20	0.00	0.00	10.38	2.51	0.00	0.00	5.56	6.53	0.00	0.00	31.74	11.92
6	0.00	18.18	0.00	0.00	0.00	8.79	0.00	0.00	0.00	7.57	0.00	0.00	0.00	24.70	0.00	0.00
7	5.41	0.00	16.46	7.93	1.74	0.00	3.76	3.86	3.23	0.00	12.15	3.41	7.54	0.00	21.86	12.26
8	0.00	16.59	0.00	0.80	0.00	8.56	0.00	0.37	0.00	8.00	0.00	0.45	0.00	28.44	0.00	1.28
9	4.43	19.58	14.20	11.17	2.75	10.88	9.45	3.65	1.61	4.27	0.56	6.48	8.54	31.74	23.77	14.68
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	0.00	18.89	21.82	12.37	0.00	13.27	12.88	4.84	0.00	0.19	0.99	5.30	0.00	29.46	32.29	16.06
13	6.88	0.00	16.14	0.00	3.73	0.00	11.35	0.00	0.36	0.00	1.72	0.00	9.56	0.00	28.00	0.00
14	0.00	14.98	0.00	10.81	0.00	9.92	0.00	5.53	0.00	1.69	0.00	5.80	0.00	26.43	0.00	18.40
15	2.36	14.76	0.00	0.00	1.57	5.16	0.00	0.00	0.67	7.68	0.00	0.00	4.70	19.55	0.00	0.00
16	6.21	0.00	0.00	8.16	3.16	0.00	0.00	5.35	1.43	0.00	0.00	3.51	9.50	0.00	0.00	15.57
17	4.19	0.00	15.99	11.64	2.41	0.00	10.30	3.93	0.43	0.00	0.47	6.50	6.24	0.00	26.07	15.69
18	0.00	20.07	13.90	0.00	0.00	9.17	9.34	0.00	0.00	5.52	7.58	0.00	0.00	27.27	29.74	0.00
19	0.00	0.00	14.53	11.84	0.00	0.00	11.82	4.38	0.00	0.00	3.76	5.62	0.00	0.00	32.84	15.61
20	0.00	23.27	0.00	0.00	0.00	11.66	0.00	0.00	0.00	2.97	0.00	0.00	0.00	32.97	0.00	0.00

5.3. Performance measures

We have considered several performance measures to compare algorithms. These performance measures are: (1) the average of RPD (ARPD), (2) the standard deviation of RPD (SDRPD), (3) the best RPD (RPD*), (4) the worst RPD (WRPD), and (5) computation time (CPU time). The smaller the values of these metrics, the better the performance of the algorithms.

5.4. Comparative results

As stated in previous sections, four meta-heuristic algorithms are compared based on the ARPD, SDRPD, RPD* and WRPD. Tables 8 and 9 report the outcomes in terms of four performance measures for small size and large size problems, respectively. As reported in Tables 8 and 9, the PGA has prevailed other methods significantly. In terms of the ARPD, the PGA outperformed other methods in most of scenarios. The SDRPD results from the PGA are far better than the outputs from the HS, GA and Neurogenetic. As for RPD*, the PGA has reached the best values in most of test problems. Based on WRPD values, the superiority of PGA is remarkable in both small and large size problems.

Table 9
Comparisons between algorithms for large size problems in terms of ARPD, SDRPD, RPD* and WRPD

Prob. No	Performance Measures															
	ARPD				SDRPD				RPD*				WRPD			
	PGA	HS	GA	NeuroGA*	PGA	HS	GA	NeuroGA*	PGA	HS	GA	NeuroGA*	PGA	HS	GA	NeuroGA*
21	0.00	0.00	13.68	12.38	0.00	0.00	11.47	3.55	0.00	0.00	1.28	8.69	0.00	0.00	31.96	16.19
22	0.00	25.49	17.35	0.00	0.00	9.77	11.26	0.00	0.00	12.28	5.32	0.00	0.00	36.75	30.10	0.00
23	0.00	0.00	21.50	0.00	0.00	0.00	14.26	0.00	0.00	0.00	0.51	0.00	0.00	0.00	36.12	0.00
24	6.16	19.34	0.00	21.15	1.49	8.86	0.00	1.91	4.20	9.26	0.00	19.29	7.46	27.85	0.00	23.84
25	0.00	12.72	0.00	0.00	0.00	14.12	0.00	0.00	0.00	0.22	0.00	0.00	0.00	35.00	0.00	0.00
26	4.70	0.00	0.00	0.00	2.99	0.00	0.00	0.00	0.91	0.00	0.00	0.00	8.97	0.00	0.00	0.00
27	3.72	0.00	0.00	0.00	3.13	0.00	0.00	0.00	0.85	0.00	0.00	0.00	9.08	0.00	0.00	0.00
28	7.12	0.00	21.02	0.00	3.09	0.00	12.47	0.00	2.69	0.00	4.10	0.00	9.86	0.00	33.75	0.00
29	0.00	17.38	0.00	17.64	0.00	11.63	0.00	4.99	0.00	8.95	0.00	10.95	0.00	36.12	0.00	22.81
30	0.00	17.80	19.47	0.00	0.00	11.89	10.89	0.00	0.00	3.50	10.24	0.00	0.00	33.87	32.22	0.00
31	0.00	23.68	16.69	18.66	0.00	10.17	7.24	6.06	0.00	11.26	7.55	10.37	0.00	34.65	26.74	23.96
32	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
33	5.44	19.72	0.00	20.75	3.61	11.54	0.00	3.41	1.44	4.88	0.00	16.69	9.38	33.15	0.00	24.01
34	0.00	16.49	20.55	0.00	0.00	8.41	12.86	0.00	0.00	7.03	0.54	0.00	0.00	29.43	35.15	0.00
35	4.53	0.00	24.66	18.98	3.01	0.00	11.13	4.53	1.20	0.00	5.85	12.67	8.84	0.00	35.51	23.42
36	6.42	0.00	18.17	13.08	3.68	0.00	11.67	3.32	2.05	0.00	4.44	10.77	9.97	0.00	35.03	17.89
37	5.11	12.07	22.29	12.91	1.31	11.82	6.06	2.37	3.78	2.00	14.10	10.85	7.26	32.10	29.91	15.94
38	0.00	19.05	17.23	18.47	0.00	12.52	14.98	4.59	0.00	4.37	2.40	11.68	0.00	36.62	36.42	21.84
39	0.00	18.46	0.00	16.42	0.00	15.63	0.00	4.04	0.00	1.44	0.00	11.98	0.00	36.73	0.00	21.76
40	0.00	0.00	19.18	0.00	0.00	0.00	10.58	0.00	0.00	0.00	7.91	0.00	0.00	0.00	31.42	0.00

NeuroGA* denotes the Neurogenetic algorithm

To test the performance of the PGA, the results obtained by PGA is compared to the outputs of the GAMS software version 24.1.2 in small scale problems. Table 10 indicates the comparison of results obtained by meta-heuristics and the exact solutions. The objective function values and the gap between the meta-heuristic and the optimal solutions are shown in Table 10. It is noteworthy that the GAMS software has achieved the optimal solutions in around 1500 seconds. The relative gap between the optimal solutions acquired by GAMS and three meta-heuristics are computed using Eq. 42 (Baniamerian et al., 2017):

$$GAP = \frac{OFV^{alg} - OFV^{*}}{OFV^{*}} \quad (42)$$

Where, OFV^{alg} is the objective function value obtained by the algorithm, while OFV^{*} denotes the optimal objective function value obtained by GAMS. As indicated in Table 10, the PGA has found the optimal or near optimal solution for almost all test instances.

Table 10
Comparison of meta-heuristics and GAMS in small size instances

Prob. No	Algorithms					GAP			
	PGA	HS	GA	NeuroGA	GAMS	GAP_{PGA}	GAP_{HS}	GAP_{GA}	$GAP_{NeuroGA}$
1	77	79	77	77	77	0.000	0.026	0.000	0.000
2	97	102	99	98	97	0.000	0.052	0.021	0.010
3	102	111	105	102	102	0.000	0.088	0.029	0.000
4	92	96	100	94	92	0.000	0.043	0.087	0.021
5	102	102	108	105	102	0.000	0.000	0.059	0.029
6	106	110	106	106	106	0.000	0.038	0.000	0.000
7	107	99	113	110	99	0.081	0.000	0.141	0.111
8	101	106	101	102	101	0.000	0.050	0.000	0.009
9	92	95	89	94	87	0.057	0.092	0.023	0.080
10	109	109	109	109	109	0.000	0.000	0.000	0.000
11	93	93	93	93	93	0.000	0.000	0.000	0.000
12	101	103	106	102	101	0.000	0.020	0.050	0.009
13	111	109	113	109	109	0.018	0.000	0.037	0.000
14	87	90	87	89	87	0.000	0.034	0.000	0.022
15	94	99	92	92	92	0.022	0.076	0.000	0.000
16	101	98	98	104	98	0.031	0.000	0.000	0.061
17	95	94	97	96	94	0.011	0.000	0.032	0.021
18	80	85	87	80	80	0.000	0.063	0.088	0.000
19	106	106	110	109	106	0.000	0.000	0.038	0.028
20	78	83	78	78	78	0.000	0.064	0.000	0.000

Figure 15 illustrates the mean of ARPD, SDRPD, RPD* and WRPD obtained by three algorithms for all problems. As shown in Figure 15, the PGA prevailed other methods in terms of all performance measures.

To validate the proposed genetic algorithm statistically, multiple statistical experiments are implemented by means of a multifactor analysis of variance (ANOVA). These experiments are conducted to detect that whether the observed differences are statistically significant or not. The null hypothesis assumes that there is no significant difference between the outputs of algorithms in terms of ARPD. We have checked the hypotheses in terms of homoscedasticity, independence and normality using a residual analysis. The results showed no bias. The ANOVA tests are conducted at a 95% confidence interval. The null hypothesis is rejected if $P-Value < 0.05$. Tables 11 and 12 report the ANOVA test results in terms of ARPDs for small and large size problems, respectively. The outputs demonstrate that the performances of algorithms are statistically different.

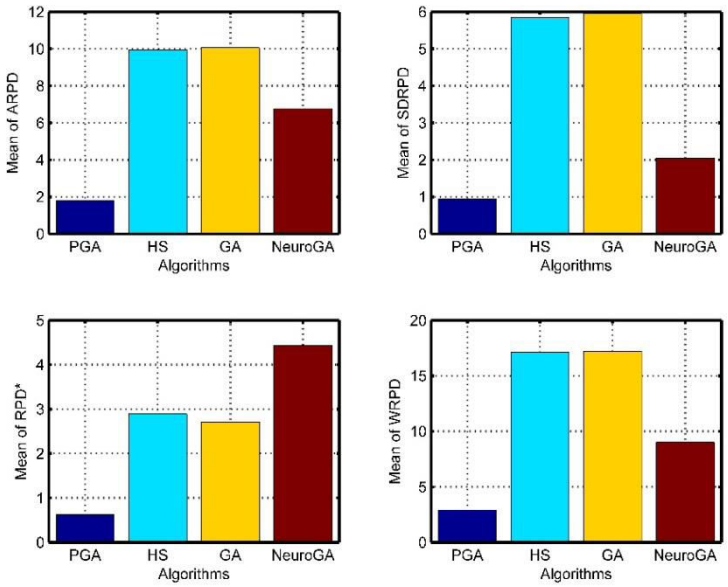

Fig. 15. Evaluating four performance measures of meta-heuristics

Table 11
The ANOVA test results for small size problems

Source	SS	df	MS	F	Prob>F
Columns	838.94	3	279.64	6.36	0.0007
Error	3341.74	76	43.97		
Total	4180.68	79			

Table 12
The ANOVA test results for large size problems

Source	SS	df	MS	F	Prob>F
Columns	1033.57	3	344.522	4.84	0.0039
Error	5412.33	76	71.215		
Total	6445.90	79			

Figure 16 illustrates the plots for mean values and Tukey intervals at a 95% confidence interval for both small and large scale problems. Based on Figure 16, the interval bars for the proposed genetic algorithm show better central tendency and variability in comparison to other methods.

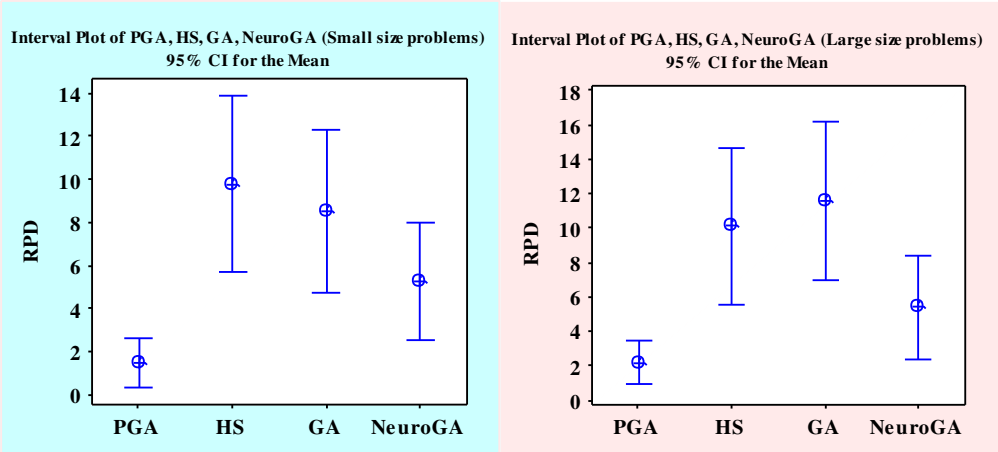

Fig. 16. Means and interval plots for small and large scale test instances in terms of RPD

Figures 17 and 18 compare the computation times (CPU times) of algorithms for small and large size problems, respectively. These figures show that the HS is the fastest method among all algorithms.

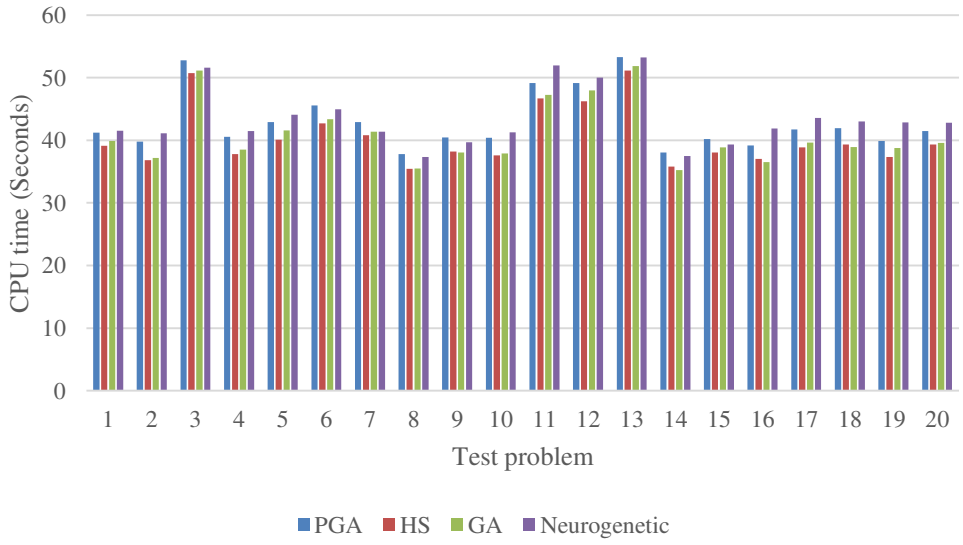

Fig. 17. CPU times of algorithms for small size problems

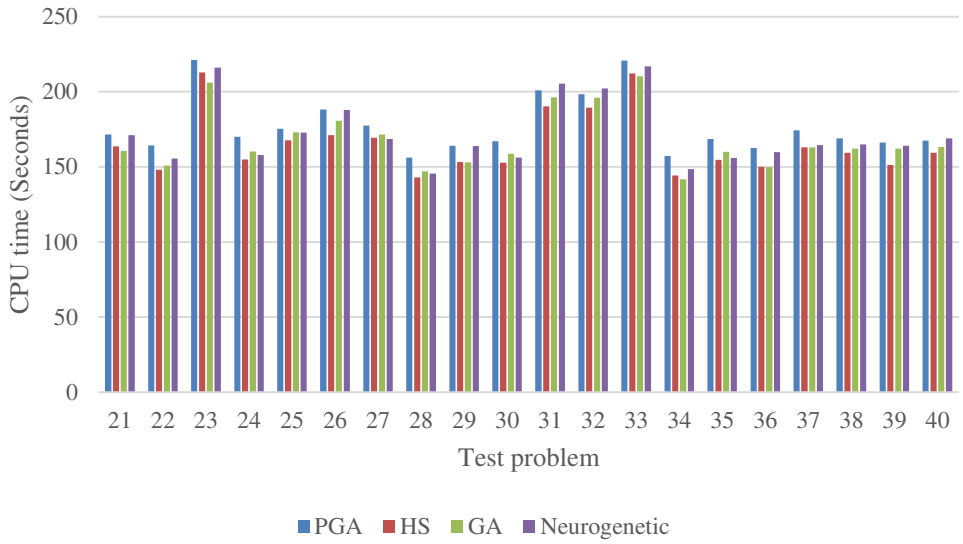

Fig. 18. CPU times of algorithms for large size problems

6. Conclusions and Future Studies

A multi-mode multi-skilled resource-constrained project scheduling problem (MMSRCPSP) was introduced in this paper in which the objective was to minimize the project completion time. We proposed a mathematical formulation for MMSRCPSP. Due to complexity of MMSRCPSP, a new genetic algorithm known as PGA was proposed to schedule project activities, efficiently. The proposed method produces promising outputs as it finds optimal solutions obtained by GAMS software. The most remarkable feature of PGA is its capability to explore solution space which is due to two strategies proposed to find high-quality individuals. Moreover, the PGA utilizes a hybrid MADM method consisting of Shannon's entropy approach and the VIKOR method to choose the best individuals for reproduction. This important feature remains the same as the size of problem increases. The proposed genetic algorithm is compared to harmony search algorithm and classical genetic algorithm in terms of various performance measures. The experimental results show that the PGA is superior to other methods in terms of most of performance measures. Using this hybrid MADM method in other meta-heuristics can be an interesting subject for further studies. Other constraints can be added to the proposed model to make it as close as possible to real-world cases such as considering generalized precedence relations between tasks or considering preemptive activities.

References

Afruzi, E., Najafi, A.A., Roghanian, E., & Mazinani, M., (2014). 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*, 50, 80-96.

Afshar-Nadjafi, B., Rahimi, A., & Karimi, H., (2012). An Exact Algorithm for the Mode Identity Project Scheduling Problem, *Journal of Optimization in Industrial Engineering*, 10, 55-63.

Agarwal, A., Colak, S., & Erenguc, S., (2011). A Neurogenetic approach for the resource-constrained project scheduling problem, *Computers & Operations Research*, 38, 44-50.

Agarwal, A., Colak, S., & Erenguc, S., (2015). *Metaheuristic Methods*, in: Schwindt, C., Zimmermann, J., (Eds.), Handbook on Project Management and Scheduling Vol.1, Springer International Publishing. New York, 57-74.

Al-Anzi, F.S., Al-Zame, K., & Allahverdi, A., (2010). Weighted Multi-Skill Resources Project Scheduling, *Journal of Software Engineering & Applications*, 3(12), 1125-1130.

Almeida, B.F., Correia, I., & Saldanha-da-Gama, F., (2016). Priority-based heuristics for the multi-skill resource constrained project scheduling problem, *Expert Systems with Applications*, 57(15), 91–103.

Azaron, A., Katagiri, H., Sakawa, M., Kato, K., & Memariani, A., (2006). A multi-objective resource allocation problem in PERT networks. *European Journal of Operational Research*, 172(3), 838-854.

Bellenguez, O., & Néron, E., (2005). Lower Bounds for the Multi-skill Project Scheduling Problem with Hierarchical Levels of Skills, *Practice and Theory of Automated Timetabling V*, 3616, 229-243.

Bellenguez, O., & Néron, E., (2007). A branch-and-bound method for solving multi-skill project scheduling problem, *Rairo Operations Research*, 41, 155-170.

Blazewicz, J., Lenstra, J.K. & Kan, A., (1983). Scheduling subject to resource constraints: Classification and complexity, *Discrete Applied Mathematics*, 5(1), 11-24.

Bouleimen, K., & Lecocq, H., (2003). A new efficient simulated annealing algorithm for the resource-

constrained project scheduling problem and its multiple mode version, *European Journal of Operational Research*, 149(2), 268-281.

Chen, R., Liang, C., Gu, D., and Leung, J., (2017). A multi-objective model for multi-project scheduling and multi-skilled staff assignment for IT product development considering competency evolution, *International Journal of Production Research*, 55(21), 1-29.

Cordeau, J., Laporte, G., Pasin, F., & Ropke, S., (2010). Scheduling Technicians and Tasks in a Telecommunications Company, *Journal of Scheduling*, 13(4), 393-409.

Corominas, A., Ojeda, J., & Pastor, R., (2005). Multi-objective allocation of multi-function workers with lower bounded capacity, *Journal of the Operational Research Society*, 56(6), 738-743.

Correia, I., & Saldanha-da-Gama, F., (2014). The impact of fixed and variable costs in a multi-skill project scheduling problem: An empirical study, *Computers & Industrial Engineering*, 72, 230-238.

Deb K., Pratap, A., Agrawal S., & Meyarivan, T., (2002). A Fast and Elitist Multi-objective Genetic Algorithm: NSGA-II, *IEEE transactions on evolutionary computation*, 6(2), 182-197.

Firat, M., & Hurkens, C.A.J., (2012). An improved MIP-based approach for a multi-skill workforce scheduling problem, *Journal of Scheduling*, 15(3), 363-380.

Gao, J., Chen, R., & Deng, W., (2013). An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem, *International Journal of Production Research*, 51(3), 641-651.

Gomar, J., Haas, C., & Morton, D., (2002). Assignment and Allocation Optimization of Partially Multiskilled Workforce, *Journal of construction engineering and management*, 128(2), 103-109.

Gutjahr, W.J., Katzensteiner, S., Reiter, P., Stummer, C., & Denk, M., (2008). Competence-driven project portfolio selection, scheduling and staff assignment, *Central European Journal of Operations Research*, 16(3), 281-306.

Gutjahr, W. J., Katzensteiner, S., Reiter, P., Stummer, C., & Denk, M., (2010). Multi-objective decision analysis for competence-oriented project portfolio selection, *European Journal of Operational Research*, 205(3), 670-679.

Hassanpour, J., Ghodoosi, M., & Hosseini, Z.S., (2017). Optimizing a Bi-objective Preemptive Multi-mode Resource-Constrained Project Scheduling Problem: NSGA-II and MOICA Algorithms, *Journal of Optimization in Industrial Engineering*, 21, 79-91.

Hartmann, S., & Kolisch, R., (2000). Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem, *European Journal of Operational Research*, 127, 394-407.

Hegazy, T., Shabeeb, K., Elbeltagi, E., & Cheema, T., (2000). Algorithm for scheduling with multiskilled constrained resources, *Journal of construction engineering and management*, 126(6), 414-421.

Hermerl, C., & Kolisch, R., (2010). Scheduling and staffing multiple projects with a multi-skilled workforce, *OR Spectrum*, 32, 343-368.

Hosseini, Z.S., Hassanpour, J., & Roghanian, E., (2014). A Bi-objective Pre-emption Multi-mode Resource Constrained Project Scheduling Problem with due Dates in the Activities, *Journal of Optimization in Industrial Engineering*, 15, 15-25.

Javanmard, S., Nadjafi, B., & Niaki, S.T.A., (2016). Preemptive multi-skilled resource investment project scheduling problem; mathematical modelling and solution approaches, *Computers and Chemical Engineering*, 96(1), 55-68.

Kadrou, Y., & Najid, N.M., (2006). Tabu Search Algorithm for the MRCPSP with Multi-Skilled Labor, *in: Proc. Comput. Eng. Syst. Appl., IMACS Multi-conference*, 1302-1309.

Kazemipoor, H., Tavakkoli-Moghaddam, R., Shahnazari-Shahrezaei, P., & Azaron, A., (2013). A differential evolution algorithm to solve multi-skilled project portfolio scheduling problems, *The International Journal of Advanced Manufacturing Technology*, 64(5-8), 1099-1111.

Kazemipoor, H., Tavvakoli-Moghaddam, E., & Sharezaei, P., (2013). Solving a novel multi-skilled project scheduling model by scatter search, *South African Journal of Industrial Engineering*, 24(1), 121-135.

Li, H., & Womer, K., (2009). A Decomposition Approach for Shipboard Manpower Scheduling, *Military Operations Research*, 14, 1-25.

Li, W., & Womer, K., (2009). Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm, *Journal of Scheduling*, 12, 281-298.

Lotfi, F.H., & Fallahnejad, R., (2010). Imprecise Shannon's Entropy and Multi Attribute Decision Making, *Entropy*, 12, 53-62.

Maghsoudlou, H.M., Nadjafi, B., & Niaki, S.T.A., (2016). A multi-objective invasive weeds optimization algorithm for solving multi-skill multi-mode resource constrained project scheduling problem, *Computers and Chemical Engineering*, 8(1), 157-169.

Maghsoudlou, H.M., Nadjafi, B., & Niaki, S.T.A., 2017. Multi-skilled project scheduling with level-dependent rework risk; three multi-objective mechanisms based on cuckoo search, *Applied Soft Computing*, 54, 46-61.

Mehmanchi, E., & Shadrokh, S., (2013). Solving a New Mixed Integer Non-Linear Programming Model of the Multi-Skilled Project Scheduling Problem Considering Learning and Forgetting Effect, *Proceedings of the 2013 IEEE IEEM*, Bangkok, Thailand.

Mehdizadeh, E., & Kivi, A.F., (2014). Three Metaheuristic Algorithms for Solving the Multi-item Capacitated Lot-sizing Problem with Product Returns and Remanufacturing, *Journal of Optimization in Industrial Engineering*, 16, 41-53.

Miller, B., & Goldberg, D.E., (1996). Genetic algorithms, selection schemes, and the varying effects of noise, *Evolutionary Computation*, 4(2), 113-131.

Montoya, C., Bellenguez-Morineau, O., Pinson, E., & Rivera. D., (2014). Branch-and-price approach for the multi-skill project scheduling problem, *Optimization Letters*, 8(5), 1721-1734.

Myszkowski, P.B, & Skowronski, M., (2013). Specialized genetic operators for multi skill resource-constrained project scheduling problem, *19th International Conference on Soft Computing MENDEL*, 57-62.

Myszkowski, P.B., Skowronski, M., & Podlodowski, L., (2013). Novel heuristic solutions for Multi–Skill Resource–Constrained Project Scheduling Problem, *Proceedings of the 2013 Federated Conference on Computer Science and Information Systems*, 159-166.

Myszkowski, P.B., Skowronski, M., Olech, L.P., & Oslizlo, K., (2015). Hybrid ant colony optimization in solving multi-skill resource-constrained project scheduling problem, *Soft Computing*, 19(12), 3599-3619.

Nader, B., (2013). The Project Portfolio Selection and Scheduling Problem: Mathematical Model and Algorithms, *Journal of Optimization in Industrial Engineering*, 13, 65-72.

Najafi, A.A., & Salimi, M., (2018). Modeling and Solution Procedure for a Preemptive Multi-Objective Multi-Mode Project Scheduling Model in Resource Investment Problems, *Journal of Optimization in Industrial Engineering*, 11(1), 181-190.

Opricovic, S., & Tzeng, G. H., (2004). Compromise Solution by MCDM Methods: A comparative analysis of VIKOR and TOPSIS, *European Journal of Operational Research*, 156, 445-455.

Opricovic, S., & Tzeng, G. H., (2007). Extended VIKOR method in comparison with outranking methods, *European Journal of Operational Research*, 178, 514–529.

Ranjbar, M., Kianfar, F., & Shadrokh, S., (2008). Solving the resource availability cost problem in project scheduling by path relinking and genetic algorithm, *Applied Mathematics and Computation*, 196, 879-888.

Tabrizi, B.H., Tavvakoli-Moghaddam, R., & Ghaderi, S.F., (2014). A two-phase method for a multi-skilled project scheduling problem with discounted cash flows, *Scientia Iranica*, 21(3), 1083-1095.

Valls, V., Perez, A., & Quintanilla, S., (2009). Skilled workforce scheduling in Service Centers, *European Journal of Operational Research*, 193(3), 791-804.

Walter, M., & Zimmermann, J., (2016). Minimizing average project team size given multi-skilled workers with heterogeneous skill levels. *Computers & Operations Research*, 70, 163–179.

Wu, M., & Sun, S., (2006). A project scheduling and staff assignment model considering learning effect, *The International Journal of Advanced Manufacturing Technology*, 28(11), 1190-1195.

Yaghoubi, S., Noori, S., Azaron, A., & Tavakkoli-Moghaddam, R., (2011). Resource allocation in dynamic PERT networks with finite capacity, *European Journal of Operational Research*, 215(3), 670-678.

Yaghoubi, S., Noori, S., Azaron, A., & Fynes, B., (2015). Resource allocation in multi-class dynamic PERT networks with finite capacity, *European Journal of Operational Research*, 247(3), 879-894.

Zheng, H., Wang, L., & Zheng, X., (2015). Teaching–learning-based optimization algorithm for multiskill resource constrained project scheduling problem, *Soft Computing*, 21(6), 1537-1548.

This article can be cited: Hosseinian, A.H. & Baradaran, V. (2019) An Evolutionary Algorithm Based on a Hybrid Multi-Attribute Decision Making Method for the Multi-Mode Multi-Skilled Resource-Constrained Project Scheduling Problem. *Journal of Optimization in Industrial Engineering*. 12 (2), 155-178.

http://www.qjie.ir/article_545828.html
DOI: 10.22094/JOIE.2018.556347.1531