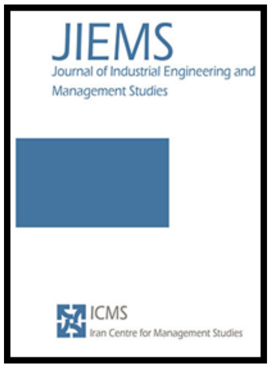
JIEMS

Journal of Industrial Engineering and Management Studies

Vol. 7, No. 1, 2020, pp. 59-86

DOI: 10.22116/JIEMS.2020.110007

www.jiems.icms.ac.ir



Modified Pareto archived evolution strategy for the multi-skill project scheduling problem with generalized precedence relations

Amir Hossein Hosseinian[1], Vahid Baradaran[1,*]

Abstract

In this research, we study the multi-skill resource-constrained project scheduling problem, where there are generalized precedence relations between project activities. Workforces are able to perform one or several skills, and their efficiency improves by repeating their skills. For this problem, a mathematical formulation has been proposed that aims to optimize project completion time, reworking risks of activities, and costs of processing the activities, simultaneously. A modified version of the Pareto Archived Evolution Strategy (MV-PAES) is developed to solve the problem. Contrary to the basic PAES, this algorithm operates based on a population of solutions. For the proposed method, we devised crossover and mutation operators, which strengthen this algorithm in exploring solution space. Comprehensive numerical tests have been conducted to evaluate the performance of the MV-PAES in comparison with two other meta-heuristics. The outputs show the excellence of the MV-PAES in comparison with other methods. A real-world software development project has been studied to demonstrate the practicality of the proposed model for real-world environment. The influence of competency evolution has been investigated in this case study. The results imply that the competency evolution has a considerable impact on the objective function values.

Keywords: project scheduling; multi-objective optimization; multi-skill resources; competency evolution.

Received: July 2019-07

Revised: August 2019-04

Accepted: November 2019-25

1. Introduction

The Resource-Constrained Project Scheduling Problem (RCPSP) is one of the most complex problems in the field of operations research that appears in a large spectrum of real-world situations (Brucker et al., 1999). The RCPSP is an optimization problem, in which a set of activities must be scheduled such that a predefined objective is optimized (Chakrabortty, et al., 2016). This problem is mainly concerned with determining the feasible start times of activities with respect to precedence and resource constraints such that the makespan is minimized (Hartmann and Briskorn, 2010).

* Corresponding author; v_baradaran@iau-tnb.ac.ir

[1] Department of Industrial Engineering, Faculty of Engineering, Islamic Azad University, Tehran North Branch, Tehran, Iran.

Due to the NP-hard nature of the RCPSP, finding optimal solutions by exact methods for large size problems becomes computationally intractable (Blazewicz et al., 1983; Alcaraz and Maroto, 2001). Hence, many meta-heuristic strategies have been developed to solve this hard optimization problem. The classical RCPSP assumes that the capacity of resources is constant in each period (Hartmann and Briskorn, 2010). However, in real-world projects, the capacity of resources might change over time due to holidays, weekends, machine breakdowns, scheduled maintenance operations, etc. (Cheng et al., 2015). Besides, in the classical form of the RCPSP, the dependencies between activities are defined as Finish-to-Start (FS) relations. Based on the Finish-to-Start relations, an activity cannot be started before completion of all its predecessors (Hartmann and Briskorn, 2010). In real-world situations, in addition to the Finish-to-Start relations, there might be other dependencies between activities such as Finish-to-Finish (FF), Start-to-Start (SS), and Start-to-Finish (SF) relations. These dependencies are known as Generalized Precedence Relations (GPR) which imply that the starting or finishing times of a pair of activities are separated by a minimum or a maximum time lag (Schwindt and Zimmermann, 2015).

A remarkable variant of the classical RCPSP is the Multi-Skill Resource-Constrained Project Scheduling Problem (MSRCPSP). The aim of the MSRCPSP is to identify the start times of activities subject to prerequisite relations and resource constraints. Moreover, this problem seeks to find the most appropriate combination of multi-skill workforces assigned to each activity (Schwindt and Zimmermann, 2015). In the MSRCPSP, each worker is able to perform at least one skill. The MSRCPSP assumes that each activity may need one or several skills to be executed. Since workers are capable to perform one or more skills, there are many possible combinations of workers which can be assigned to each skill of an activity. Each worker is allowed to use only one of his/her skills in each period to accomplish an activity. It is possible to assign workforces to several activities as long as their processing times do not overlap. The workers allocated to an activity should be available until all its required skills are completed (Maghsoudlou et al., 2017).

In real world scenarios, workers have different efficiencies in performing each of their skills. In this case, their efficiencies can improve by learning. Learning phenomenon occurs when workforces spend more time on performing a skill. Since the makespan of the project depends on the efficiency of workers, the learning effect plays a remarkable role in decreasing the required time to accomplish the project. Performing a skill by more efficient workers, increases the execution cost, while it reduces the risk of reworking (Maghsoudlou et al., 2017).

In this paper, we have concentrated on the multi-skill RCPSP when resource availabilities vary over time and the activities are subject to generalized precedence relations. Most real-world scheduling problems involve heterogeneous objectives that should be optimized, simultaneously. In this respect, we have focused on the multi-objective resource-constrained project scheduling problem. Thus, the proposed model in this research is called the MOMSRCPSP-GRP. The MOMSRCPSP-GPR assumes that the efficiency of workers is different in performing each skill. Each skill of an activity should be executed by competent workers. A worker is competent if his/her efficiency is greater than or equal to the required efficiency of a skill. Since all workers are not competent to perform all skills, the learning factor plays an important role in increasing the number of competent workers. The importance of learning is elevated when the availability of competent workers is not constant in each period. In the proposed model, as a worker spends time performing a skill, his/her efficiency will become better in performing that particular skill. The MOMSRCPSP-GPR aims to minimize the makespan of the project as the first objective function. For the second and the third objective functions, the proposed model targets to minimize the reworking risks of processed activities and to minimize costs of processing activities, respectively.

Due to the high complexity of the proposed model, we have employed the Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al. 2000) and the Pareto Envelope-based Selection Algorithm II (PESA-II) (Corne et al., 2001) to solve the MOMSRCPSP-GPR. In addition to these methods, we have proposed a Modified Version of the Pareto Archived Evolution Strategy (PAES) which is called the MV-PAES. The classical PAES algorithm uses a single mutation operator to produce new individuals. For the proposed method, we have designed a new crossover operator which makes it possible to explore the solution space more accurately. In addition, the PAES method maintains a single parent in each iteration, while the MV-PAES algorithm preserves a population of solutions, from which the parents are selected to generate new individuals. Furthermore, the mutation operator embedded in the MV-PAES has been modified to secure the feasibility of mutated solutions.

Contributions of this study are as follows: First, a multi-objective multi-skill RCPSP model with time-dependent resource capacities is presented. The proposed model considers generalized precedence relations between activities and it aims to minimize makespan, reworking risks of the processed activities and costs of processing activities. Second, a modified version of the Pareto Archived Evolution Strategy is proposed to solve the model. A population of solutions has been considered for this developed method. Third, the performance of the MV-PAES is evaluated in comparison with two evolutionary methods, i.e. classical NSGA-II and PESA-II based on some test problems. Fourth, a software development real-world project has been described to indicate practical aspects of the model. The MV-PAES, NSGA-II, and PESA-II have been used to schedule activities of this project and their outputs have been compared as well. Ultimately, the impact of competency evolution has been investigated on the values of the objective functions.

The structure of the paper is as follows: Section 2 presents a literature review for the RCPSP with time-varying resource availabilities and the multi-skill RCPSP. Section 3 is dedicated to the problem description and mathematical formulation of the MOMSRCPSP-GPR. The proposed algorithm is discussed in detail in Section 4. Section 5 is devoted to computational experimentations, and ultimately, Section 6 provides concluding remarks and discusses further research opportunities.

2. Literature review

In this section, we review the most related studies to the current research. Bartusch et al. (1988) investigated on using generalized precedence constraints to capture notions of time-dependent resource capacities and requests. Buddhakulsomsiri and Kim (2006) studied a multi-mode RCPSP, in which the activities can be split during their scheduling procedure due to variations in availability of resources. Hartmann (2013) presented a mathematical model in which resource availabilities and requests are time-dependent. Gomar et al. (2002) proposed a linear programming model to allocate proper multi-skill workers to activities of a construction project. Bellenguez and Neron (2005) presented a mathematical formulation for the MSRCPSP, where the multi-skill resources have different levels of efficiency in performing each skill. Corominas et al. (2005) proposed a multi-objective non-linear mixed integer program to assign multi-function workers to activities. Wu and Sun (2006) proposed a mixed non-linear formulation for the project scheduling and staff allocation problems with the learning effect. Gutjahr et al. (2008) developed a non-linear mixed integer formulation for the project portfolio selection, scheduling and staff assignment problems. Li and Womer (2009) proposed a mathematical formulation for staff scheduling as a multi-purpose RCPSP. Valls et al. (2009) proposed a multi-skill workforce project scheduling model considering generalized precedence relations between activities, percentage time lags and variable activity durations. Ho and Leung (2010) studied staff scheduling problem with job time windows and job-skills compatibility constraints. A construction heuristic and an adaptive large scale neighborhood search method were introduced by Cordeau et al. (2010) to schedule both activities and multi-skill workforces in Telecommunications Company. Liu and Wang (2012) introduced a Constraint Programming (CP) technique and several heuristic rules to solve the multi-skill RCPSP with generalized precedence relations. Kazemipoor et al. (2013a) proposed a goal programming model for the multi-skill project portfolio scheduling problem with infinite number of modes for each activity. Myszkowski et al. (2013) proposed some novel heuristics for the MSRCPSP. Mehmanchi and Shadrokh (2013) investigated the effects of learning and forgetting on efficiency of human resources in the MSRCPSP. Kazemipoor et al. (2013b) proposed a mixed integer model for the multi-mode MSRCPSP to schedule Information Technology (IT) projects. Tabrizi et al. (2014) applied a two-phase method based on genetic operators and path relinking algorithms to maximize the Net Present Value (NPV) of the project. Correia and Saldanha-da-Gama (2014) considered resource usage, fixed and variable costs for the MSRCPSP. Myszkowski et al. (2015) proposed a hybrid algorithm consisting of classical heuristic priority rules with Ant Colony Optimization (ACO) method. Zheng et al. (2015) introduced a Teaching-Learning-Based Optimization (TLBO) algorithm to minimize the makespan. Almeida et al. (2016) introduced a Parallel Scheduling Scheme (PSS) for the MSRCPSP. Javanmard et al. (2016) integrated the multi-skill project scheduling problem with the Resource Investment Problem (RIP). Maghsoudlou et al. (2016) proposed a multi-objective mixed integer formulation for the multi-mode MSRCPSP. They presented a new Multi-Objective Invasive Weeds Optimization (MOIWO) algorithm to approximate the Pareto frontier. Maghsoudlou et al. (2017) presented a bi-objective mathematical model for the MSRCPSP in order to minimize total costs and the reworking risks of the activities. Chen et al. (2017) addressed a multi-project multi-skill project scheduling problem considering learning and forgetting effects on efficiency of workers. Dai et al. (2018) investigated the multi-skill RCPSP with the step-deteriorating effect. Myszkowski et al. (2018) developed a hybrid Differential Evolution and Greedy algorithm (DEGR) for the multi-skill RCPSP. A knowledge-guided multi-objective Fruit-fly Optimization Algorithm (FOA) was developed by Wang and Zheng (2018) for the MSRCPSP to minimize the makespan and total cost, concurrently. Zabihi et al. (2019) proposed a multi-objective TLBO method to the multi-skill RCPSP, where the efficiency of workers is influenced by the learning phenomenon. Hosseinian et al. (2019) developed the linear threshold model in social networks to model the diffusion of dexterity among workforces for the MSRCPSP. Hosseinian and Baradaran (2019a) proposed a community detection based approach to find the most compatible working groups in the MSRCPSP. Hosseinian and Baradaran (2019b) developed a Genetic Algorithm (GA) based on a hybrid Multi-Criteria Decision Making (MCDM) method for the multi-mode multi-skill RCPSP. Tirkolaee et al. (2019) studied the multi-mode RCPSP considering payment planning in order to optimize makespan and net present value of the project, simultaneously. In another study, Hosseinian and Baradaran (2019c) proposed an energy-efficient mathematical model for the RCPSP. Najafzad et al. (2019) proposed an energy-efficient bi-objective model for the MSRCPSP in which electricity tariffs are determined based on time of use. Objectives of their model are minimization of project completion time and cost, concurrently. Laszczyk and Myszkowski (2019) modified the selection operator of the NSGA-II and proposed three new evolutionary algorithms for the multi-skill RCPSP. They showed that their modifications in the selection operator improves convergence and diversity of the obtained Pareto fronts.

Based on the studies reviewed in this section, one of the research gaps is the multi-objective multi-skill RCPSP with time-dependent resource capacities and generalized precedence relations between activities (MOMSRCPSP-GPR). This gap motivated us to focus on this research opportunity and develop a mathematical formulation that suits the multi-skill RCPSP with the aforementioned characteristics.

Since genetic algorithms have been successful in many of optimization problems and to the best of the authors' knowledge, none of the previous studies has used the PAES algorithm for the multi-skill RCPSP. Therefore, we decided to develop a modified version of the PAES algorithm to solve the problem, and compare its efficacy with other meta-heuristics.

3. Problem description and mathematical formulation

3.1. Impact of learning phenomenon on efficiency of workforces

Despite a great number of studies investigated the learning effects in machine scheduling settings, these effects are rarely used in the field of project scheduling. In this paper, we study the impact of learning on project completion time, reworking risks of processed activities, and total costs of processing activities. To model the learning process, the learning curve proposed by Wu and Sun (2006) is used. They presented an average efficiency curve that represents the knowledge of a worker in performing a specific activity after a period of time. They showed that the efficiency of workforces in executing an activity increases by spending more time on that specific activity. According to this learning curve, the efficiency of a worker in executing an activity is computed as follows (Wu and Sun, 2006):

$$\overline{Eff}_t = \overline{Eff}_1 \times t^b \tag{1}$$

Where, $\overline{Eff_t}$ denotes the cumulative average efficiency when an individual has spent *t* periods on activity *j*. In the Equation (1), *t* is the total number of periods spent by worker *s* on activity *j*. $\overline{Eff_1}$ is the initial efficiency of a worker in performing activity *j* and *b* is the learning factor calculated by (Wu and Sun, 2006):

$$b = -\frac{Ln(l)}{Ln(2)}, (0 < l \le 1) \tag{2}$$

In the above equation, *l* is the learning percentage. Smaller values of *l* leads to larger values of *b*. However, the learning curve proposed by Wu and Sun (2006) needs some modifications to be applicable in multi-skill project scheduling environments. Hence, we have extended this learning curve for the MSRCPSP. In this respect, for each skill of a worker, a learning curve is considered. Based on this learning curve, the longer an individual performs a skill, the more efficient he/she will become on that particular skill. For instance, a worker is capable of performing skills "1", "2", "4", and "5". This worker has different initial efficiencies in performing each skill. Considering the learning percentage of $l = 0.95$, Fig. 1 shows the cumulative average efficiency of this worker for each of his/her skills. According to Fig. 1, the cumulative average efficiency continues to increase over time until it reaches 1.0. For the learning curve used in this research, the cumulative average efficiency of worker *s* in performing skill *k* in period *t*+1 can be computed using the following formula:

$$\overline{Eff}_{sk(t+1)} = \overline{Eff}_{sk1} \times TT_{skt}^b \tag{3}$$

Where, $\overline{Eff_{sk1}}$ denotes the initial cumulative average efficiency of worker *s* in performing skill *k*. TT_{skt} is the total time spent by worker *s* on performing skill *k* up to period *t*. The modified learning curve has been used in our mathematical formulation in Section 3.2.

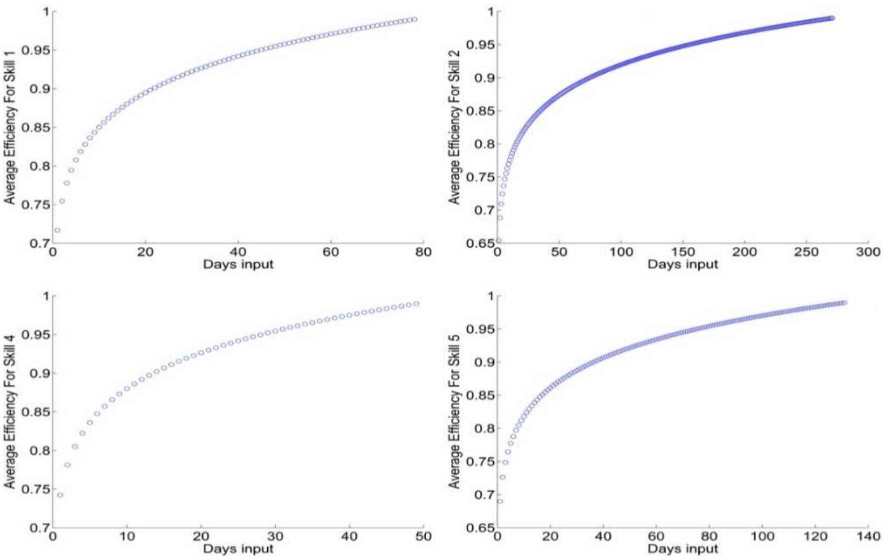


Figure 1. Learning curves

3.2. Mathematical formulation of the MOMSRCPSP-GPR

In this study, some of the most important features of the real-world projects have been considered in our mathematical formulation. The MOMSRCPSP-GPR seeks to minimize the makespan, reworking risks of processed activities, and total costs of processing activities, concurrently. Makespan is defined as the required time to complete all project activities. Using more efficient workers reduces the required durations for activities, while it imposes more costs on the project. The budget in each period and for the whole project is limited. The greater the efficiency of workers assigned to an activity, the lower the risk of reworking for this activity. Generalized precedence relations between activities and time-varying resource availabilities are another features that make the proposed model close to real-world situations. The multi-skill RCPSP lies within the scope of NP-hard problems (Correia et al., 2012). Therefore, we can conclude that the proposed problem with generalized precedence relations and time-dependent resource availabilities is an NP-hard problem as well. The most important features of the proposed problem is as follows: The project is defined as an acyclic directed network $G(V, A)$, where V is a set of nodes corresponding to project activities and A denotes a set of arcs representing precedence relations among various activities in the Activity-On-Node (AON) representation (Hartmann and Briskorn 2010). The activities are numbered in topological order labeled as $j = 0,1,2, \dots, N+1$. The activities 0 and $N+1$ are dummy activities that represent the start and the finish of the project, respectively. Dummy activities have zero standard duration and they require no resources. Generalized precedence relations are considered among project activities. A single mode is available for activities to be performed. All resources are multi-skill resources. The availability of resources may change from period to period. Each activity may require one or more skills. A given number of workers are needed to execute each required skill. Resource requirements are deterministic and known in advance. The standard durations of activities are non-negative and integer valued. It is assumed that if the efficiency of workers assigned to an activity is 1.0, they are able to complete this activity in its standard duration. Each worker is able to execute one or several specific skills. However, the efficiency of workers in performing each skill is different. Each required skill of an activity has to be carried out by individuals who are efficient enough with respect to minimum required efficiency. Each worker has to be assigned to only one skill of an activity at each period. All workers assigned to an activity should start performing required skills, concurrently. The efficiency of workers in performing their skills changes over time. The longer an individual performs a particular skill, the more efficient will be this worker on that skill. The amount of budget in each period is limited. The amount of budget for the whole project is limited.

3.2.1. Sets

V: Set of project activities, $(j, j' = 0,1,2,...,N+1)$,
K: Set of project's required skills, $(k, k' = 1,2,...,K)$,
Λ: Set of workforces, $(s, s' = 1,2,...,S)$,
Π: Set of time periods, $(t, t' = 0,1,2,...,T)$,
$pred_j$: Set of immediate predecessors of activity j,
A_{FS}: Set of arcs representing finish-to-start relations,
A_{FF}: Set of arcs representing finish-to-finish relations,
A_{SS}: Set of arcs representing start-to-start relations,
A_{SF}: Set of arcs representing start-to-finish relations,
I_k: Set of activities that require skill k,
RS_j: Set of skills required by activity j,
φ_k: Set of workers who are capable to perform skill k,
L_{jkt}: Set of competent workers to perform skill k of activity j in period t,

3.2.2. Parameters

p_j: Standard duration of activity j without learning,
r_{jk}: The number of required workers to perform skill k of activity j,
ME_{jk}: The minimum required efficiency to perform skill k of activity j,
$lag_{jj'}$: Time lag between activity j and activity j',
b: Learning factor,
PDS: The per-day salary of a worker with efficiency of 1.0,
MF: The maximum amount of fund allowed to be spent in each period,
UBB: Upper bound of budget considered for the whole project,
M: A positive large number,
δ_{sk}: Equals 1 if worker s has skill k, otherwise it equals 0

3.2.3. Decision variables

X_{jt}: Equals 1 if activity j is started in period t, otherwise it equals 0
ω_{jskt}: Equals 1 if worker s begins to work on activity j in period t, otherwise it equals 0
O_{jskt}: Equals 1 if worker s is performing skill k of activity j in period t, otherwise it equals 0
H_{jkt}: Equals 1 if the execution of skill k for activity j is in progress in period t, otherwise it equals 0
ξ_{jskt}: Equals 1 if worker s is efficient to perform skill k of activity j in period t (if $\overline{Eff}_{skt} \geq ME_{jk}$), otherwise it equals 0
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

γ_{jk} : Completion time of execution of skill *k* for activity *j*
Eff_{skt} : The efficiency of worker *s* in performing skill *k* in period *t*
$\overline{Eff}_{skt}$: Cumulative average efficiency of worker *s* in performing skill *k* up to period *t*
Pr_{skt} : The reworking risk if worker *s* performs skill *k* in period *t*
D_{jskt} : Processing time required by worker *s* to perform skill *k* of activity *j* in period *t*
d_j : The overall processing time required by workers assigned to activity *j*
R_{jkt} : The number of competent workers that can perform skill *k* in period *t*
TR_{jkt} : The required time to accomplish skill *k* of activity *j* in period *t*
TS_{skt} : The time spent by worker *s* on performing skill *k* in period *t*
C_{skt} : Cost of performing skill *k* by worker *s* in period *t*
Z_1 : The first objective function variable
Z_2 : The second objective function variable
Z_3 : The third objective function variable

3.2.4. Problem modelling

$$MinZ_1 = \sum_{t=0}^{T} t \times X_{(N+1)t} \tag{4}$$

$$MinZ_2 = \sum_{t=0}^{T}\sum_{s=1}^{S}\sum_{k=1}^{K}\left(Pr_{skt} \times \sum_{j=0}^{N+1} O_{jskt}\right) \tag{5}$$

$$MinZ_3 = \sum_{t=0}^{T}\sum_{s=1}^{S}\sum_{k=1}^{K}\left(C_{skt} \times \sum_{j=0}^{N+1} O_{jskt}\right) \tag{6}$$

s.t.

$$\sum_{t=0}^{T} X_{jt} = 1 \qquad \forall j \tag{7}$$

$$D_{jskt} = (2 - \overline{Eff}_{skt}) \times O_{jskt} \qquad \forall j, \forall s \in L_{jkt}, \forall k \in RS_j, \forall t \tag{8}$$

$$TR_{jkt} = \max_{s}(D_{jskt}) \qquad \forall j, \forall k \in RS_j, \forall t = 1, \dots, p_j \tag{9}$$

$$d_j = \sum_{t=1}^{p_j} \max_{k}(TR_{jkt}) \qquad \forall j, \forall k \in RS_j \tag{10}$$

$$d_j \ge p_j \qquad \forall j \tag{11}$$

$$R_{jkt} = \sum_{t'=0}^{t}\sum_{s=1}^{S} \xi_{jskt'} \qquad \forall j, \forall k \in RS_j \tag{12}$$

$$r_{jk} \times H_{jkt} \le R_{jkt} \qquad \forall j, \forall k \in RS_j, \forall t \tag{13}$$

$$\overline{Eff}_{skt} \ge 0 \qquad \forall s, \forall k, \forall t \tag{14}$$

$$\overline{Eff}_{skt} \le 1 \qquad \forall s, \forall k, \forall t \tag{15}$$

$$\overline{Eff}_{skt} \ge ME_{jk} \times O_{jskt} \qquad \forall j, \forall s, \forall k \in RS_j, \forall t \tag{16}$$

$$O_{jskt} \le \delta_{sk} \qquad \forall j, \forall s, \forall k, \forall t \tag{17}$$

$$\sum_{j=0}^{N+1}\sum_{k=1}^{K} O_{jskt} \le 1 \qquad \forall s, \forall t \tag{18}$$

$$H_{jkt} \le \sum_{s=1}^{S} O_{jskt} \qquad \forall j, \forall k, \forall t \qquad (19)$$

$$U_{jk} = \sum_{t=0}^{T} t \times X_{jt} \qquad \forall j, \forall k \in I_j \qquad (20)$$

$$ST_j = U_{jk} \qquad \forall j, \forall k \in RS_j \qquad (21)$$

$$U_{jk} = U_{jk'} \qquad \forall j, \forall (k \ne k') \in RS_j \qquad (22)$$

$$\gamma_{jk} = U_{jk} + \sum_{t=U_{jk}}^{U_{jk}+p_j} TR_{jkt} \times H_{jkt} \qquad \forall j, \forall k \in RS_j \qquad (23)$$

$$FT_j = ST_j + d_j \qquad \forall j \qquad (24)$$

$$ST_{(N+1)} \le T \qquad (25)$$

$$FT_j \ge \gamma_{jk} \qquad \forall j, \forall k \in RS_j \qquad (26)$$

$$FT_j + lag_{jj'} \le ST_{j'} \qquad \forall (j, j') \in V; \forall (j, j') \in A_{FS} \qquad (27)$$

$$ST_j + lag_{jj'} \le FT_{j'} \qquad \forall (j, j') \in V; \forall (j, j') \in A_{SF} \qquad (28)$$

$$ST_j + lag_{jj'} \le ST_{j'} \qquad \forall (j, j') \in V; \forall (j, j') \in A_{SS} \qquad (29)$$

$$FT_j + lag_{jj'} \le FT_{j'} \qquad \forall (j, j') \in V; \forall (j, j') \in A_{FF} \qquad (30)$$

$$\sum_{s=1}^{S} \sum_{t=0}^{T} \omega_{jst} = \sum_{k=1}^{K} r_{jk} \qquad \forall j \qquad (31)$$

$$\sum_{t=0}^{T} \omega_{jst} \le 1 \qquad \forall j, \forall s \qquad (32)$$

$$\sum_{j=0}^{N+1} \sum_{t'=t-d_j+1}^{t} \omega_{jst'} \le 1 \qquad \forall s, \forall t \qquad (33)$$

$$\omega_{jst} \le X_{jt} \qquad \forall j, \forall s, \forall t \qquad (34)$$

$$C_{skt} = \overline{Eff}_{skt} \times PDS \qquad \forall s \in S, \forall k, \forall t \qquad (35)$$

$$Pr_{skt} = 1 - \overline{Eff}_{skt} \qquad \forall s, \forall k, \forall t \qquad (36)$$

$$\sum_{j=0}^{N+1} \sum_{k \in RS_j} \sum_{s \in L_{jkt}} C_{skt} \times O_{jskt} \le MF \qquad \forall t \qquad (37)$$

$$\sum_{t=0}^{T} \sum_{j=0}^{N+1} \sum_{k \in RS_j} \sum_{s \in L_{jk}} C_{skt} \times O_{jskt} \le UBB \qquad (38)$$

$$TS_{skt} = \sum_{j \in I_k} \left(2 - \overline{Eff}_{skt}\right) \times O_{jskt} \qquad \forall s, \forall k, \forall t \qquad (39)$$

$$\overline{Eff}_{sk1} = Eff_{sk1} \qquad \forall s, \forall k \qquad (40)$$

$$\overline{Eff}_{sk(t+1)} \le \overline{Eff}_{sk1} \left(\sum_{t'=1}^{t} TS_{skt'} \times O_{jskt'} \right)^{b} + M\left(1 - O_{jskt}\right) \qquad \forall s, \forall k, \forall t \qquad (41)$$

$$\overline{Eff}_{sk(t+1)} \ge \overline{Eff}_{sk1} \left(\sum_{t'=1}^{t} TS_{skt'} \times O_{jskt'} \right)^{b} - M\left(1 - O_{jskt}\right) \qquad \forall s, \forall k, \forall t \qquad (42)$$

$$\overline{Eff}_{sk(t+1)} \le \overline{Eff}_{skt} + M \times O_{jskt} \qquad \forall j, \forall s, \forall k, \forall t \qquad (43)$$

$$\overline{Eff}_{sk(t+1)} \ge \overline{Eff}_{skt} - M \times O_{jskt} \qquad \forall j, \forall s, \forall k, \forall t \qquad (44)$$

$$Eff_{sk(t+1)} = \max\left(\frac{\overline{Eff}_{sk(t+1)} \times \sum_{t'=1}^{t} TS_{skt'} \times O_{jskt'} - \overline{Eff}_{skt} \times \sum_{t'=1}^{t-1} TS_{skt'} \times O_{jskt'}}{\max\left(\left(\sum_{t'=1}^{t} TS_{skt'} \times O_{jskt'} - \sum_{t'=1}^{t-1} TS_{skt'} \times O_{jskt'} \right), 1 \right)}, Eff_{skt} \right) \qquad \forall j, \forall s, \forall k, \forall t \qquad (45)$$

$$L_{jkt} = \begin{cases} L_{jk(t-1)} \cup \{s\}; & if \quad \overline{Eff}_{skt} \geq ME_{jk} \\ L_{jk(t-1)}; & if \quad \overline{Eff}_{skt} < ME_{jk} \end{cases} \qquad \forall j, \forall s \in \varphi_k, \forall k \in RS_j, \forall t \qquad (46)$$

$$X_{jt}, \omega_{jst}, O_{jskt}, H_{jkt}, \xi_{jskt} \in \{0,1\} \qquad \forall j, \forall s, \forall k, \forall t \qquad (47)$$

$$ST_j, FT_j, U_{jk}, \gamma_{jk}, Eff_{skt}, \overline{Eff}_{skt}, D_{jskt}, C_{skt}, R_{jkt}, TR_{jkt}, TS_{skt}, d_j, Pr_{skt} \geq 0 \qquad \forall j, \forall s \in S, \forall k, \forall t \qquad (48)$$

3.2.5. Model description

The first objective function in Equation (4) is concerned with minimizing the makespan of the project. The second objective function (5) aims to minimize the reworking risks of processed activities. The third objective function in Equation (6) shows the costs associated with processing activities, which should be minimized. Constraint (7) secures that each activity is started exactly once. Equation (8) calculates the time needed by worker *s* to perform skill *k* of activity *j* in each time unit. Equation (8) has been obtained by considering the efficiency of workers assigned to each skill of an activity. Equation (9) determines the required time to execute skill k of activity j in period t. Equation (10) computes the overall processing time needed by workers allocated to activity *j* to perform all required skills. Constraint (11) implies that the overall processing time required by the workers assigned to activity *j* must be equal or greater than the standard duration of this activity. Equation (12) calculates the number of competent workers that can perform skill *k* in period *t*. Constraint (13) reflects the limitations on time-varying resource availabilities. Constraints (14) and (15) specify the lower and upper bounds on efficiency of workforces, respectively. Constraint (16) secures that worker *s* is efficient enough in period *t* to execute skill *k* of activity *j*. Constraint (17) guarantees that the worker assigned to skill *k* of activity *j* is able to perform this skill. Constraint (18) assures that each worker is allowed to perform just one activity at a particular period. The logical relation between H_{jkt} and O_{jskt} is stated in constraint (19). Equation (20) computes the start time of execution of each skill. Equation (21) implies that the start time of an activity is equal to start time of its required skills. Equation (22) forces that all skills required by activity j start simultaneously. Equation (23) calculates the completion time of required skills. Equation (24) is dedicated to calculation of finish times of activities. Constraint (25) defines the upper bound of project completion time. Constraint (26) reflects the logical relation between FT_j and γ_{jk}. Constraint (27) takes care of Finish-to-Start precedence relations between activities, while constraint (28) preserves Start-to-Finish precedence relationships among activities. Constraint (29) shows precedence relations between project activities with the Start-to-Start relations. In constraint (30), the Finish-to-Finish precedence relations among activities are preserved. Constraint (31) ensures that the number of workers that has been allocated to activity j to perform skill k, is equal to the number of workers needed to accomplish the activity. Constraint (32) indicates that at most one start time is determined for workers allocated to an activity. Constraint (33) assures that workers assigned to different skills of activities must carry out these skills without any interruption. This constraint secures uninterrupted assignment of staff members to different skills of activities. Constraint (34) states the logical relation between ω_{jst}

and X_{jt}. Equation (35) computes the cost of performing skill k by worker s in period t. Equation (36) computes the reworking risk if worker s performs skill k in period t. Constraint (37) secures that the amount of costs incurred in each period cannot be greater than the maximum amount of fund considered for each period. Constraint (38) guarantees that total costs of processing activities cannot be greater than the upper bound of budget considered for the whole project. Equation (39) computes the time spent by worker s on performing skill k in period t. Equation (40) sets the initial efficiency of workers for all skills. Constraints (41) to (44) imply that the cumulative average efficiency of workers in performing a skill improves when they spend more time on executing that specific skill. Constraints (41) and (42) are for modelling the case when $O_{jskt} = 1$, while constraints (43) and (44) are considered when $O_{jskt} = 0$. The relations between cumulative average efficiency for a time interval and the efficiency at a specific period are modeled in Equation (45). Equations (46) show how to update the set L_{jkt}. Finally, constraints (47) and (48) denote the type of decision variables in this mathematical formulation.

4. Solution approaches

4.1. Encoding and decoding process

In this paper, each solution is represented by two lists. Both lists have N positions, where N is the number of non-dummy activities in the project. The first list is the standard activity list proposed by Kolisch and Hartmann (1999). This representation is a precedence-feasible list which means each activity appears in any position after all its predecessors. The second list has been presented in this paper to show the resources assigned to each activity. Fig.2 illustrates a feasible solution. The first row of this solution shows a feasible activity list with seven non-dummy activities. The second row of the solution shown in Fig.2 displays the resources assigned to project activities. For instance, the first and the fourth workers denoted as “W1” and “W4” have been assigned to activity “1”. There is a difference between the representations used in the multi-mode RCPSP and the representation proposed in this paper. In the multi-mode RCPSP, each mode represents different durations and resource requirements for an activity. However, the proposed model in this paper assumes that there is only one execution mode for project activities and therefore processing times and resource requirements are known and predefined.

Activity List	**1**	**4**	**2**	**6**	**3**	**5**	**7**
List of Assigned Resources	**W1 , W4**	**W2 , W5**	**W3**	**W2**	**W4 , W5**	**W1 , W3**	**W1 , W2**

Figure 2. A two-vector solution representation for the MOMSRCPSP-GPR

Decoding schedules to real solutions of the RCPSP is necessary to calculate fitness functions. We have used Serial Schedule Generation Scheme (S-SGS) in Hartmann (2013) as decoding procedure to determine the makespan of schedules. The S-SGS consists of N stages. In each stage, one activity is scheduled as soon as possible with respect to its precedence and resource constraints.

4.2. Multi-objective evolutionary algorithms

Since the proposed model (MOMSRCPSP-GPR) belongs to the class of NP-hard problems, a multi-objective meta-heuristic is needed to approximate the Pareto optimal frontier in a reasonable computation time. Hence, we propose a modified version of PAES algorithm called the MV-PAES to solve the model. The procedures of the proposed algorithm are discussed as follows.

4.2.3. Modified Version of Pareto Archived Evolution Strategy (MV-PAES)

The Pareto Archived Evolution Strategy (PAES) is a renowned sequential multi-objective algorithm which is known for being a (1+1) evolution strategy combined with an archive set that stores non-dominated solutions previously found (Knowles and Corne, 1999). Despite most of the evolutionary algorithms that preserve a population of solutions, from which the individuals are selected to generate offspring, the PAES maintains a single parent in each iteration to generate a new single candidate solution. Each mutated solution is compared against non-dominated solutions of the archive set. If the mutated solution succeeds to dominate any individual in the archive, the non-dominated solution will be removed from the archive set. To maintain the diversity of solutions, the PAES uses a crowding procedure that divides phenotype space in a recursive manner. Based on the values of objective functions, each solution is located in a certain grid location. The number of individuals existing in each grid location is applied to make decisions about selecting and archiving candidate solutions when a location is crowded. Except for the number of divisions in the phenotype space, no other parameters are needed. In this respect, the PAES has a lower computational complexity comparing to other niching procedures (Knowles and Corne, 2000).

As mentioned above, the classical PAES preserves a single solution in each iteration to produce a new single candidate solution. To obtain high quality solutions in terms of convergence and diversity, our modified version of the PAES preserves a population of solutions. In addition, the classical PAES does not use any crossover operator to generate new solutions. Hence, we propose a crossover operator to find better candidate solutions in solution space. The proposed algorithm also includes a new mutation operator which generates feasible chromosomes for the MOMSRCPSP-GPR. These operators are explained in the following sections. For the proposed method, we have used the Pareto dominance sorting used in the NSGA-II (Deb et al. 2000; Bensmaine et al., 2013) to rank solutions of the population. A brief description of the MV-PAES is shown in Fig.3.

Algorithm 1: Modified Version of Pareto Archived Evolution Strategy (MV-PAES)

Notations:

CurrentSol	Current solution
BestSol	Best candidate solution found in each iteration
pop	Current population of solutions
ARCH	Archive of non-dominated solutions
p_m	Mutation rate
Npop	Population size
MaxIt	Maximum number of iterations
ARS	Archive size
HGS	Hyper-grid size

Require: p_m, *Npop*, *MaxIt*, *ARS*, and *HGS*

Generate a random solution (*CurrentSol*) and add to *pop*;
Evaluate *CurrentSol* and add to the archive;
Do for $it = 1: MaxIt$
    Do for $\rho = 1: 2$
        Mutate *CurrentSol* to generate a candidate solution;
        Evaluate the candidate solution;
        Add the candidate solution to *pop*;
    End For
    While $|pop| \leq Npop$ do
        Select two parents from *pop* randomly;
        Employ crossover operator to produce two offspring;
        Evaluate both offspring;
        Add both offspring to *pop*;
    End While

```
    Rank solutions based on Pareto dominance sorting;
    Select the best solution of the population (BestSol) based on its rank;
    If CurrentSol dominates BestSol
        Discard BestSol;
    else
        If any archive member dominates BestSol
            Discard BestSol;
        else
            If  BestSol dominates any member of archive
                Remove dominated solutions from archive;
                Add BestSol to the archive;
                Replace CurrentSol with BestSol;
            else
                If  |ARCH| ≤ ARS
                    Add BestSol to the archive;
                        If BestSol resides in a less crowded region than CurrentSol
                            Replace CurrentSol with BestSol;
                        else
                            Discard BestSol;
                        End if
                else
                    If BestSol increases diversity in the archive;
                        Add BestSol to the archive;
                        Remove the solution existing in the most crowded grid location;
                        If BestSol resides in a less crowded region than CurrentSol
                            Replace CurrentSol with BestSol;
                        else
                            Discard BestSol;
                        End if
                    else
                        If BestSol resides in a less crowded region than CurrentSol
                            Replace CurrentSol with BestSol;
                        else
                            Discard BestSol;
                        End if
                    End if
                End if
            End if
        End if
    End if
End For
Output: Non-dominated solutions of archive
```

Figure 3. Description of the MV-PAES

Proposed crossover operator

To employ crossover operator, two parents denoted as P_1 and P_2 are selected, randomly. The crossover operator initiates with generating two integer random numbers $RINT_1$ and $RINT_2$ from the interval $[1, N]$ to select two activities from P_1 and P_2, respectively. The crossover operator consists of two stages. Having selected two solutions as parents, a two-stage operation is applied to produce two chromosomes denoted as CH_1 and CH_2. For the first stage, we used the method proposed by Hartmann (2002), in which, the positions of the nearest predecessor and the nearest successor of the chosen activities are identified on activity lists of the P_1 and P_2. To obtain activity list of the CH_1, the first chosen activity moves to a random place within its nearest predecessor and successor on the P_2. Another feasible activity list is generated by moving the second chosen activity to a random position between its nearest predecessor and successor on the P_1. Fig.4 shows two solutions as parents. Suppose that activities “4” and “5” have been selected randomly from the P_1 and P_2, respectively. The activity “4” can move anywhere within the highlighted genes on the P_2 to obtain a precedence-feasible activity list for the CH_1. Similarly, the activity "5" is allowed to take a random position from the highlighted genes on the P_1. In the example illustrated in Fig.4, the activity "4" moves to the second gene on the P_2, while the activity "5" takes the fifth gene on the P_1. In the second stage, a simple procedure is proposed to generate resource lists of both offspring. A random integer number denoted as RNW, is generated on the interval $[1, N]$ to determine the number of activities for which the allocated workers should change. Suppose that the workers of two activities should change ($RNW = 2$). In the next step, two random integer numbers are randomly chosen from $[1, N]$ again. These two random numbers are denoted as RNP_1 and RNP_2 which represent the activities for which the resources should change. Suppose that $RNP_1 = 4$ and $RNP_2 = 7$. The workers assigned to activities "4" and "7" on the P_1, will be the workers assigned to these activities on the CH_2. Similarly, the workers allocated to activities "4" and "7" on the P_2, will be the workers assigned to these activities on the CH_1. The workers of remaining activities are taken from resource lists of the P_1 and P_2. Fig.4 illustrates the exchange of workers for the activities "4" and "7".

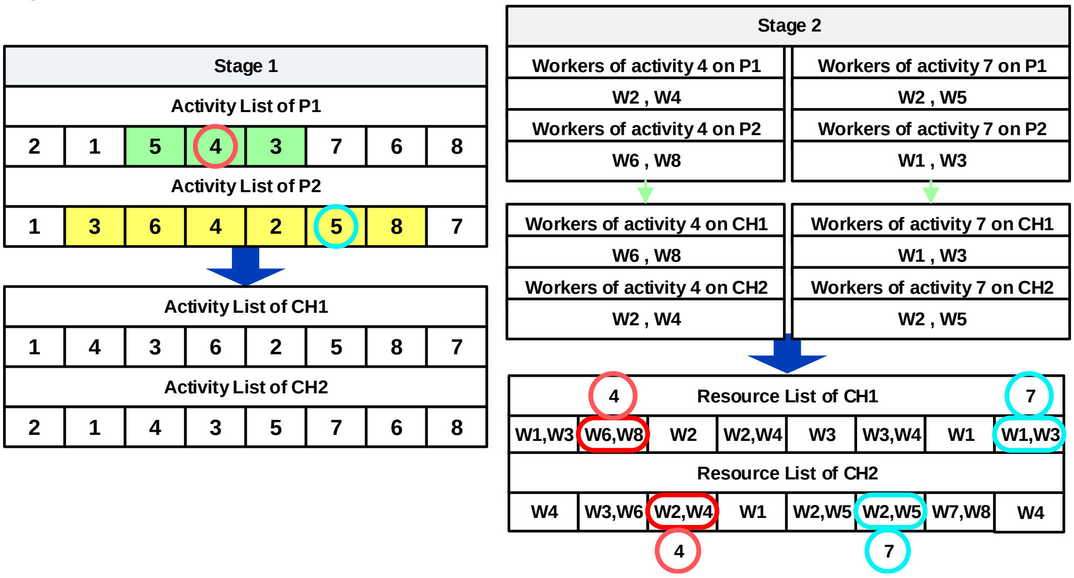


Figure 4. Two offspring generated by the proposed crossover operator

Proposed mutation operator

In this section, a mutation operator is designed for the PAES algorithm to generate feasible candidate solutions for the MSRCPSP. Similar to crossover operator, the proposed mutation operator consists of two stages. In the first stage, a feasible precedence activity list is created. The second stage is dedicated to determining the combination of resources assigned to activities. In the first stage, an integer random number is generated on the interval $[1, N]$ to choose an activity to change its position on the activity list. Afterwards, the positions of the nearest predecessor and successor of the selected activity are determined. The chosen activity takes a random position within its nearest predecessor and successor to produce a new activity list (Afshar-Nadjafi et al., 2013). Fig.5 illustrates first row of a chromosome. Suppose that activity "3" is chosen randomly to change its position. The highlighted genes represent the positions between the nearest predecessor and successor of activity 3. To respect the precedence relations, the activity 3 can be moved anywhere within the highlighted genes. Assume that the second gene of the activity list has been selected for activity 3 to move. Fig.5 also shows the activity list generated by the mutation operator.

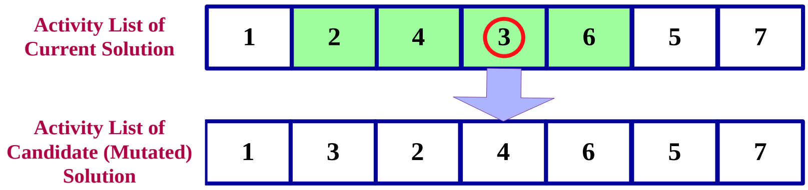


Figure 5. The activity list generated by the proposed mutation operator

For the second stage, we have proposed a procedure for the solution representation described in Section 4.1 to generate activity resource list of the candidate solution. This procedure has two versions called mutation operator version 1 (MOV1) and mutation operator version 2 (MOV2). In each iteration, a random number (RND) on the interval [1, 2] is generated to select either of the versions. $RND = 1$ corresponds to the first version of mutation operator, while $RND = 2$ corresponds to the second version. For both operators, a random binary string is generated to determine the activities for which the workforces need to change. The MOV1 changes the workers of activities whose corresponding elements on binary string are "0" and it copies the workforces of the other activities to the second row of the mutated solution. On the opposite side, MOV2 alters the workers allocated to activities whose corresponding elements on binary string are "1" and the remaining workers are taken from the current solution. The description of the MOV1 and MOV2 are shown in Fig.6. It is necessary to introduce some notations before the description of the MOV1 and MOV2.

Algorithm 2: Proposed mutation operator

Notations:

ACT	Set of activities for which the workers need to change
SK_j	Set of required skills of activity *j*
TAS	A tabu set
RNW_{jk}	Required number of workforces to perform skill *k* of activity *j*

Mutation operator version 1 (MOV1)	**Mutation operator version 2 (MOV2)**
Require: A random (1×*N*) binary string	**Require:** A random (1×*N*) binary string
Copy all workers of activities whose corresponding elements on binary string are "1" to the candidate solution;	Copy all workers of activities whose corresponding elements on binary string are "0" to the candidate solution;
Do for $j \in ACT$	Do for $j \in ACT$
Empty set *TAS*	Empty set *TAS*
Do for $k \in SK_j$	Do for $k \in SK_j$
Do for $\rho = 1{:}RNW_{jk}$	Do for $\rho = 1{:}RNW_{jk}$
Find a worker (s) which is not included in *TAS*;	Find a worker (s) which is not included in *TAS*;
Update *TAS* with worker (s) ($TAS \cup \{s\}$);	Update *TAS* with worker (s) ($TAS \cup \{s\}$);
End For	End For
End For	End For
End For	End For
Output: Second row of mutated solution	**Output:** Second row of mutated solution

Figure 6. Description of the MOV1 and MOV2

5. Experimentation and computational results

The parameters of the proposed model are first tuned in Section 5.1. Performance measures are described in Section 5.2. Then, the algorithms are calibrated via the Taguchi design method in Section 5.3. The paper continues with a comprehensive analysis to evaluate the effectiveness of the MV-PAES. The proposed method is compared with two state-of-the-art evolutionary algorithms; NSGA-II (Deb et al., 2000) and PESA-II (Corne et al., 2001). The results of computational experiments that indicate the performance of algorithms come in Section 5.4. All algorithms studied in this paper have been 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 the MOMSRCPSP-GPR

To evaluate the performance of the algorithms, we have selected 30 random test problems from the project scheduling problems library (PSPLIB) (Kolisch and Sprecher, 1996). These problems have different precedence networks and they provide standard durations of activities and the precedence relationships between the activities. However, since there is no specific benchmark instances available in the literature for the proposed model, some new data are required to be added to these test problems. The required data are generated as follows: The processing times of activities are extracted from the chosen test instances available in the PSPLIB. The number of required skills (*K*) to complete the whole project follows a uniform distribution on [3, 12]. The number of available workers for the whole project follows a uniform distribution on [8, 20]. The required number of skills for each activity is randomly selected on the interval $[1, K]$. The required number of workers to perform skill k of activity j (r_{jk}) is randomly generated by a uniform distribution $U[1,10]$. Generalized precedence relations between activities and time lags are generated using the heuristic introduced in (Tavana et al. 2014). Test problems are classified into two sizes of small and large instances. The general features of these test problems are the number of activities (*N*), the number of available workers (*S*) and the number of required skills (*K*). Test problems 1 to 15, which have 30 non-dummy activities are considered as small-size instances, while test instances 16 to 30, which have 120 non-dummy activities are large size problems.

5.2. Performance measures:

To compare the performance of optimizers, we have employed various metrics to measure the convergence and the diversity of the solutions obtained by each algorithm. The performance measures used to compare algorithms are: (1) Error Ratio (*ER*) (Zitzler, 1999), (2) Generational Distance (*GD*) (Deb 2001), (3) Mean Ideal Distance (*MID*) (Zitzler and Thiele, 1998), (4) Spacing Metric (*SM*) (Schott, 1995), (5) Diversification Metric (*DM*) (Zitzle,r 1999), (6) Set Coverage (*SC*) or C-Metric (Zitzler, 1999), and (7) Computation time (CPU time) (Zitzler, 1999).

5.3. Tuning parameters of multi-objective algorithms

The performance of optimizers significantly relies on the adjustment of their parameters. In this research, the Taguchi method (Afruzi et al. 2014) has been employed to adjust parameters of algorithms. We used the $j122$ dataset to carry out the Taguchi experiments. To obtain better convergence and diversity by Pareto-based algorithms, we have used the *MOCV* response variable proposed in Rahmati et al. (2013) for the Taguchi method. To save spaces, the details of Taguchi experiments have not been reported.

5.4. Experimental results

In this section, we compare the MV-PAES to the NSGA-II and PESA-II based on various performance metrics and objective function values. We have run the algorithms five times for each problem. Table 1 shows the performance of the three algorithms in terms of the *GD*, *MID*, *ER* and computation time for small and large size problems. It can be inferred from Table 1 that the MV-PAES has outperformed the other methods in terms of the *GD*, *MID* and *ER* metrics. The PESA-II has been superior to other two methods in terms of computation time. Table 1 reveals that the MV-PAES has won the second place in terms of CPU time.

Table 1. Comparison of algorithms in terms of convergence metrics and computation time

Problem size		Convergence Metrics									Computation time		
		GD			*MID*			*ER*			Run time (Seconds)		
		MV-PAES	NSGA-II	PESA-II	MV-PAES	NSGA-II	PESA-II	MV-PAES	NSGA-II	PESA-II	MV-PAES	NSGA-II	PESA-II
Small	Average	9.82	15.25	16.38	21.46	33.31	35.77	0.38	0.58	0.62	62.93	73.59	60.25
	Standard deviation	8.08	9.68	10.36	17.64	21.14	22.61	0.29	0.34	0.37	12.64	10.10	3.04
Large	Average	32.31	52.08	48.49	46.61	112.79	116.71	0.36	0.51	0.50	123.82	154.98	105.32
	Standard deviation	26.56	45.27	39.40	38.28	98.02	94.83	0.25	0.31	0.30	15.23	6.54	3.63

Table 2 shows the comparative results in terms of the *SM* and *DM* metrics. In terms of the *SM* metric, the MV-PAES has succeeded to provide the most uniformly distributed solutions. Furthermore, the proposed algorithm has been successful to provide better solutions in terms of the *DM* metric.

Table 2. Comparison of algorithms in terms of diversity metrics

Problem size		Diversity Metrics					
		SM			*DM*		
		MV-PAES	NSGA-II	PESA-II	MV-PAES	NSGA-II	PESA-II
Small	Average	0.00756	0.01876	0.01667	1684.33	1406.57	1323.68
	Standard deviation	0.00363	0.01915	0.01661	1226.36	1156.34	1130.59
Large	Average	0.03594	0.05429	0.05460	7268.69	6518.15	6513.36
	Standard deviation	0.01841	0.00532	0.00683	2631.92	2448.72	2539.69

Fig. 7 shows the pairwise comparisons between the MV-PAES and the other two algorithms in terms of the set coverage metric (C-Metric). According to Fig. 7, the Pareto solutions produced by the MV-PAES dominate the solutions obtained by the NSGA-II in 80% of j32 cases. Moreover, the solutions of the MV-PAES are dominant in comparison to the NSGA-II in 73.3% of j122 cases. The NSGA-II has outperformed the proposed method in just 3 out of 15 j32 cases and 4 out of 15 j122 cases. It is clear from Fig. 7 that in 14 out of 15 j32 cases, the Pareto fronts formed by the MV-PAES have dominated the solutions obtained by the PESA-II. For %80 of j122 test problems, the MV-PAES outweighed the PESA-II. The PESA-II has won the competition against the MV-PAES for one instance of j32 cases and 3 instances of *j*122 cases, respectively.

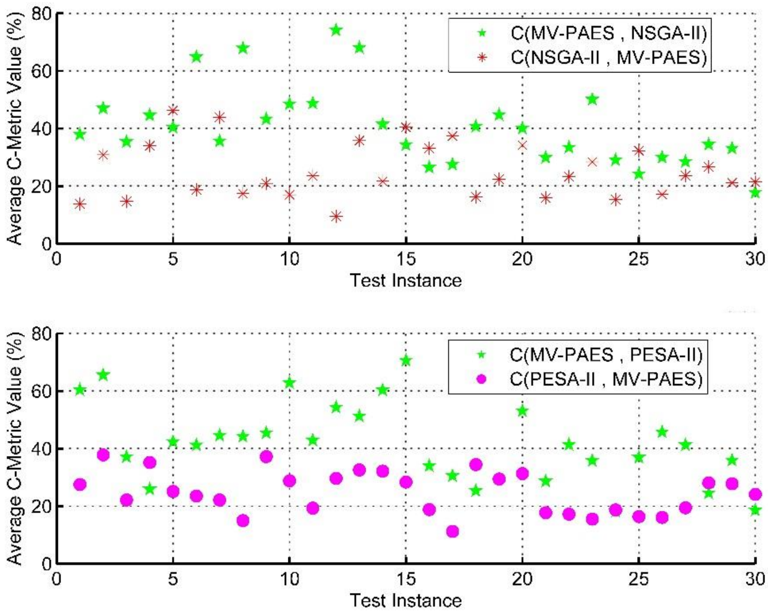


Figure 7. Pairwise comparisons between algorithms based on average *C*-Metric values

This section continues with conducting various hypothesis tests, which evaluate the performance of optimizers, statistically. The hypothesis tests are based on the difference between the means of two populations with regard to each metric. The standard deviations of these populations are unknown. Null hypotheses (H_0) of these tests assume that there is no significant difference between the means of populations. To have a statistical test of these hypotheses, we used the 2-sample t-test at a significance level of 5%. The P-values of these tests on each metric are summarized in Table 3. If P-value of a hypothesis test is smaller than 5%, the null hypothesis is rejected which means the difference between the means of two populations is significant. Table 3 implies that for the MID, ER and SM metrics, there is a significant difference between the means of populations.

Table 3. Statistical comparisons between algorithms

Metric	Algorithms	Lower Bound	Upper Bound	T-Value	P-Value	Result
GD	MV-PAES vs NSGA-II	-28.479	3.284	-1.587	0.117	H_0 is not rejected.
	MV-PAES vs PESA-II	-25.852	3.119	-1.570	0.121	H_0 is not rejected.
MID	MV-PAES vs NSGA-II	-70.689	-7.346	-2.466	0.016	H_0 is rejected.
	MV-PAES vs PESA-II	-73.441	-10.971	-2.704	0.009	H_0 is rejected.
ER	MV-PAES vs NSGA-II	-0.328	-0.018	-2.239	0.029	H_0 is rejected.
	MV-PAES vs PESA-II	-0.350	-0.030	-2.380	0.020	H_0 is rejected.
SM	MV-PAES vs NSGA-II	-0.025	-0.003	-2.703	0.009	H_0 is rejected.
	MV-PAES vs PESA-II	-0.024	-0.002	-2.525	0.014	H_0 is rejected.
DM	MV-PAES vs NSGA-II	-1216.796	2245.099	0.594	0.554	H_0 is not rejected.
	MV-PAES vs PESA-II	-1188.272	2304.254	0.639	0.524	H_0 is not rejected.
CPU time	MV-PAES vs NSGA-II	-21.346	22.362	0.046	0.963	H_0 is not rejected.
	MV-PAES vs PESA-II	13.929	49.166	3.584	0.000	H_0 is rejected.

Fig. 8 illustrates boxplots on the average of objective function values obtained by algorithms in five runs. This figure shows that MV-PAES has prevailed the other optimizers in terms of central tendency and variability.

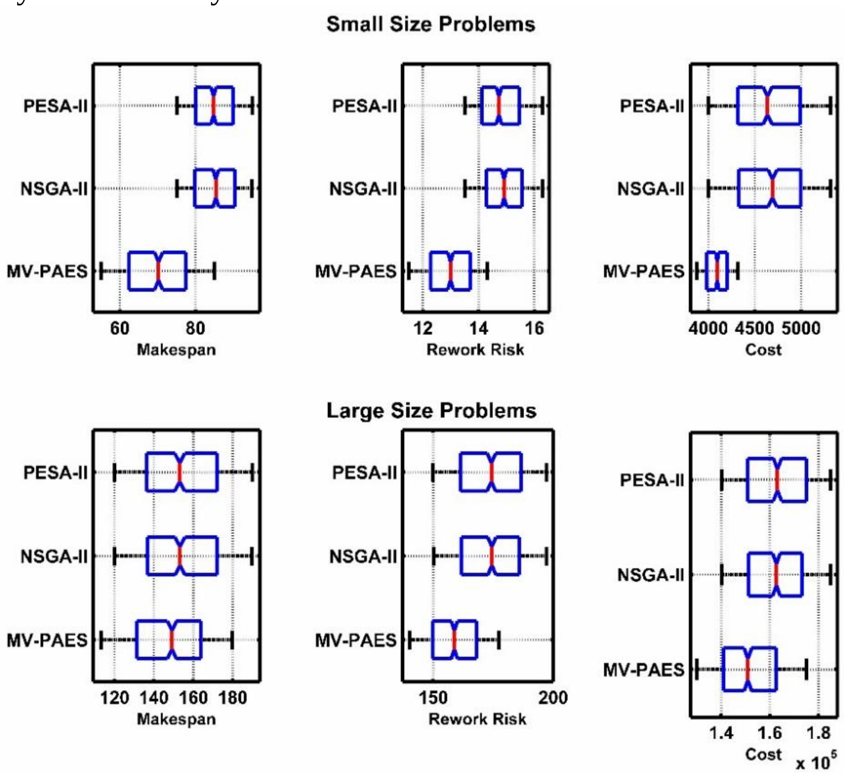


Figure 8. Boxplots of the average objective function values obtained by algorithms

In the following, we have conducted a computational analysis to investigate the effect of learning on objective function values. Hence, the MV-PAES algorithm has been hired to solve all 30 benchmark problems with and without learning effect. Fig. 9 illustrates the average of the outputs obtained by the proposed method in five runs. As shown in Fig. 9, the effect of learning on efficiency of workers reduces the first and the second objective function values, while it increases the cost of processing activities.

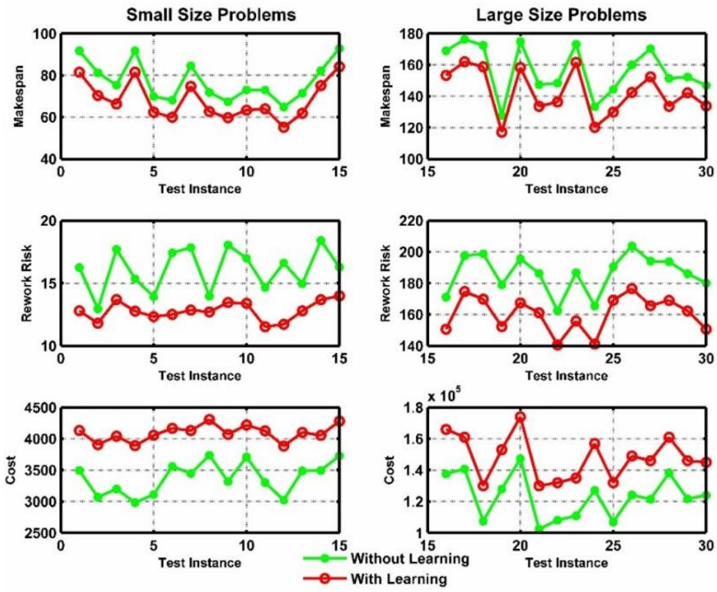


Figure 9. Comparing objective function values with consideration and without consideration of learning effect

To validate the proposed algorithm, we compared the results of the meta-heuristics with the outputs of the ε-constraint method which is an exact algorithm. The ε-constraint method is used to solve small-size Multi-Objective Optimization Problems (MOOP). Hence, the comparisons have been made for small-size test problems. Due to lack of space, we have not detailed the ε-constraint method. Full description of this method can be found in Tirkolaee et al. (2019). In this respect, the gaps between the outputs of meta-heuristics and the ε-constraint method have been computed and depicted in Figs. 10 to 12. As shown in these figures, the results of the MV-PAES are closer to the outputs of the ε-constraint method than other meta-heuristics.

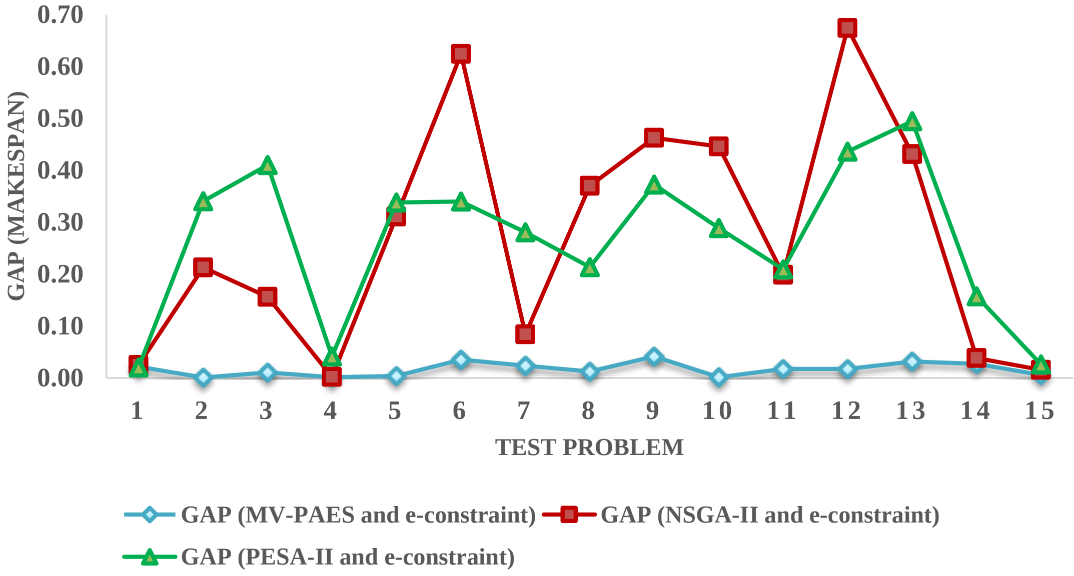


Figure 10. Comparison of algorithms in terms of makespan

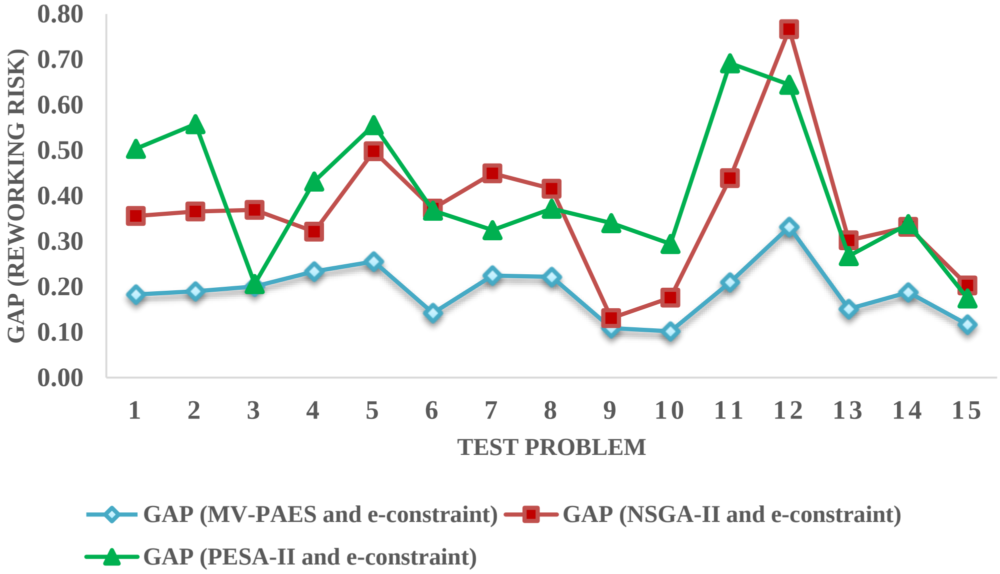


Figure 11. Comparison of algorithms in terms of reworking risk

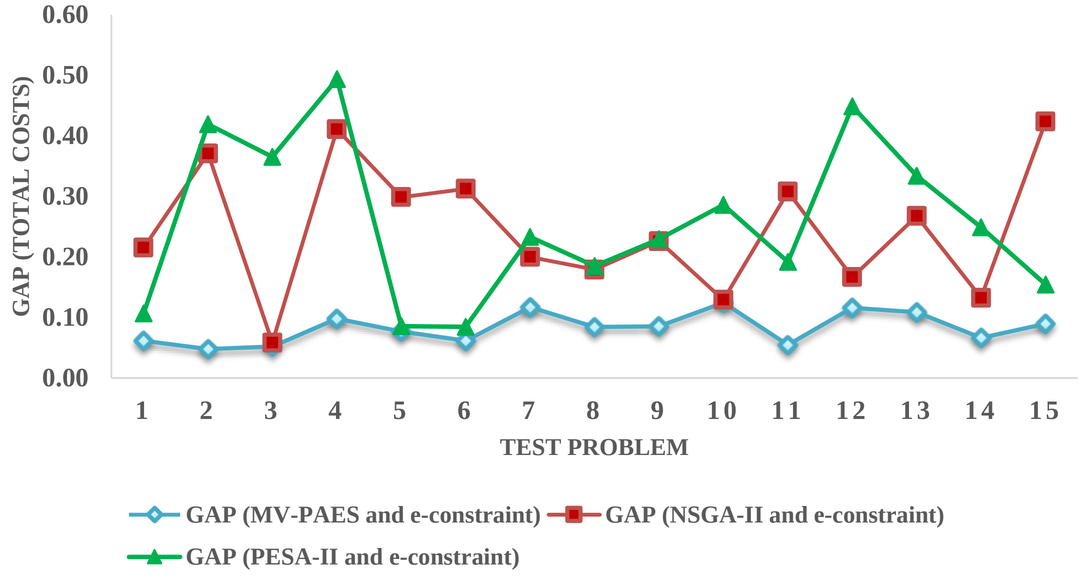


Figure 12. Comparison of algorithms in terms of total costs

We conducted multiple sensitivity analyses to examine the validity of the proposed model. One of the small-size test problems have been chosen randomly. The values of parameters have increased and decreased for 25% and we investigated these changes on the objective function values. The effect of learning factor has been investigated earlier. Each experiment has been conducted for five times, and the average of outputs have been shown in Table 4. The conclusions about changing values of parameters have been reported in Table 4 as well. Based on the remarks of Table 4, it can be concluded that the proposed model shows rational trends and obtains valid results.

Table 4. Impact of changing parameters on objective function values

Parameter	Change	Makespan	Reworking risk	Total cost	Remarks
p_j	+25% -25%	105.72 61.53	12.07 11.46	4652.29 2734.41	Significant impact on makespan and total cost.
r_{jk}	+25% -25%	98.80 65.03	18.22 7.17	5265.57 3853.90	Significant impact on all objectives.
ME_{jk}	+25% -25%	94.56 52.95	5.09 21.17	4912.66 2805.03	Significant impact on all objectives.
$lag_{jj'}$	+25% -25%	113.33 57.46	11.35 11.79	5012.82 3150.65	Significant impact on makespan and total cost.
PDS	+25% -25%	82.39 80.81	12.48 12.91	5976.12 3015.87	Significant impact on total cost.

5.5. Case study

In this section, we have applied the proposed model for a software development project to show its practicality for real-world projects. A software development company located in Iran wants to schedule activities of a software development project. This company has twenty multi-skill workforces, each of them is able to perform at least one skill. The workforces normally include: (1) industrial engineers to discover the needs for a software development project based on market studies. They study the demography of existing and new customers and use forecasting models to predict sales prospects, (2) programmers who are capable of coding with at least one programming language such as Python, C++, SQL and Power BI, (3) software analysts that provide requirements of a software, prepare specification documents and convey the needs of users to developers. The software development project studied in this research has more than 300 activities which is considered as a large-sized project in the literature of project scheduling problem. However, there are some main activities in this project which have been mentioned in Table 5. Table 5 also shows the required skills of main activities. The efficiencies of workforces in performing each of their skills are different. The longer an employee uses his/her skills, the more efficient he/she will become in performing that specific skill.

Table 5. Main activities and their required skills

Main activities	Skills						
	Business knowledge	Problem solving	Interpersonal relations with people	Market research	Forecasting knowledge	Computer programming	Documentation skill
Planning	✓	✓	✓	✓	✓		✓
Domain analysis	✓			✓			
Market studies and investigation	✓	✓	✓	✓	✓		
Providing requirements for the business plan	✓	✓	✓		✓		
Designing system	✓	✓				✓	
Software implementation						✓	
Testing						✓	
Debugging and maintenance						✓	
Documentation							✓

We have used the MV-PAES to solve the model with characteristics of the case study in two modes: (1) with consideration of learning effect, and (2) without consideration of learning effect. The MV-PAES has been run for twenty times and the results have been reported in Table 6. The results reported in Table 6 have been obtained for all activities of this project. As shown in Table 6, the MV-PAES achieved better results with consideration of learning effect regarding project completion time and reworking risks of processed activities. On the other hand, since more efficient workforces demand higher salaries, consideration of learning effect has led to more total costs of project.

Table 6. Results of the case study

Objective function	With learning		Without learning	
	Best	**Average**	**Best**	**Average**
Makespan	280	334.61	356	395.29
Reworking risk	39	47.31	63.47	75.23
Total costs	361724	473018.85	286683	314531.26

6. Conclusions and future research directions

In this paper, we have proposed a multi-objective mixed-integer formulation for the MSRCPSP with generalized precedence relations that considers: (1) Workforces have different efficiencies in performing their skills, (2) Workforces can improve their efficiency by performing their skills repeatedly, and (3) The resource availabilities are time-dependent. The proposed model targeted to minimize project completion time, risk of reworking and cost of processing activities. Due to the high complexity of the proposed model, a new Pareto-based multi-objective evolutionary algorithm called the MV-PAES was proposed to approximate the Pareto frontier. The MV-PAES uses novel crossover and mutation operators to generate feasible solutions. The performance of the MV-PAES has been compared to the NSGA-II and the PESA-II in solving several test problems available on the PSPLIB. The optimizers were compared in terms of several well-known multi-objective metrics. The outputs showed that the MV-PAES prevailed other two methods based on most of the metrics. To demonstrate the impact of learning on objective function values, we used the MV-PAES algorithm to solve all test problems with and without consideration of learning effect. The outputs imply that considering learning effect will decrease both makespan and reworking risks, while it increases costs of project. The validity of the proposed model have been demonstrated by carrying out several sensitivity analyses. To demonstrate the practicality of the proposed model, a real-world software development project was scheduled by the MV-PAES with and without consideration of learning effect. For future studies, it possible to extend the model by considering forgetting effect on efficiency of workers. Time-dependent resource requirements of activities can be added to the MOMSRCPSP-GPR to provide more reasonable schedules for real world projects. As another research opportunity, the proposed model can consider multiple execution mode for project activities.

References

Afruzi, E., Najafi, A.A., Roghanian, E., and 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*, Vol. 50, pp. 80-96.

Afshar-Nadjafi, B., Rahimi, A., and Karimi, H., (2013). "A genetic algorithm for mode identity and the resource constrained project scheduling problem", *Scientia Iranica*, Vol. 20, No. 3, pp. 824-831.

Alcaraz, J., and Maroto, C., (2001). “A Robust Genetic Algorithm for Resource Allocation in Project Scheduling”, *Annals of Operations Research*, Vol. 102, No. 1-4, pp. 83-109.

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

Bartusch, M., Mohring, R.H., and Radermacher, F.J., (1988). "Scheduling project networks with resource constraints and time windows", *Annals of Operations Research*, Vol. 16, pp. 201–240.

Bellenguez, O., and Néron, E., (2005). "Lower Bounds for the Multi-skill Project Scheduling Problem with Hierarchical Levels of Skills", In: Burke E., Trick M. (eds) Practice and Theory of Automated Timetabling V. PATAT 2004. Lecture Notes in Computer Science, vol 3616. Springer, Berlin, Heidelberg, pp. 229-243.

Bensmaine, A., Dahane, M., and Benyoucef, L., (2013). "A non-dominated sorting genetic algorithm based approach for optimal machines selection in reconfigurable manufacturing environment", *Computers & Industrial Engineering*, Vol. 66, No. 3, pp. 519-524.

Blazewicz, J., Lenstra, J.K., and Kan, A., (1983). "Scheduling subject to resource constraints: Classification and complexity", *Discrete Applied Mathematics*, Vol. 5, No.1, pp. 11-24.

Brucker, P., Drexl, A., Mohring, R., Neumann, K., and Pesch, E., (1999). "Resource-constrained project scheduling: notation, classification, models, and methods", *European Journal of Operational Research*, Vol. 112, No. 1, pp. 3-41.

Buddhakulsomsiri, J., and Kim, D.S., (2006). "Properties of multi-mode resource-constrained project scheduling problems with resource vacations and activity splitting", *European Journal of Operational Research*, Vol. 175, pp. 279-295.

Chakrabortty, R.K., Sarker, R.A., and Essam, D.L., (2016). "Multi-mode resource-constrained project scheduling under resource disruptions", *Computers & Chemical Engineering*, Vol. 88, pp. 13-29.

Cheng, J., Fowler, J., Kempf, K., and Mason, S., (2015). "Multi-mode resource-constrained project scheduling Problems with non-preemptive activity splitting", *Computers & Operations Research*, Vol. 53, pp. 275-287.

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*, Vol. 55, No. 21, pp. 6207-6234.

Cordeau, J., Laporte, G., Pasin, F., and Ropke, S., (2010). "Scheduling Technicians and Tasks in a Telecommunications Company", *Journal of Scheduling*, Vol. 13, No. 4, pp. 393-409.

Corne, D.W., Jerram, N.R., Knowles, J.D., and Oates, M.J., (2001). "PESA-II: region-based selection in evolutionary multi-objective optimization", In: Proceedings of the genetic and evolutionary computation conference (GECCO), pp. 283-290.

Corominas, A., Ojeda, J., Pastor, R., (2005). "Multi-objective allocation of multi-function workers with lower bounded capacity", *Journal of the Operational Research Society*, Vol. 56, No. 6, pp. 738-743.

Correia, I., Lourenco, L.L. and Saldanha-da-Gama, F., (2012), "Project scheduling with flexible resources: formulation and inequalities", *OR Spectrum*, Vol. 34 No. 3, pp. 635-663.

Correia, I., and 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*, Vol. 72, pp. 230-238.

Dai, H., Cheng, W., and Guo, P., (2018). "An Improved Tabu Search for Multi-skill Resource-Constrained Project Scheduling Problems Under Step-Deterioration", *Arabian Journal for Science and Engineering*, Vol. 43, No. 6, pp. 3279-3290.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T., (2000). "A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II", In: Schoenauer M. et al. (eds) Parallel Problem Solving from Nature PPSN VI. PPSN 2000. Lecture Notes in Computer Science, vol. 1917. Springer, Berlin, Heidelberg.

Deb, K., (2001). "Multi-objective optimization using evolutionary algorithms", Wiley, Hoboken, New York, USA, ISBN: 047187339X.

Gao, J., Chen, R., and Deng, W., (2013). "An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem", *International Journal of Production Research*, Vol. 51, No. 3, pp. 641-651.

Gomar, J., Haas, C., and Morton, D., (2002). "Assignment and Allocation Optimization of Partially Multi-skilled Workforce", *Journal of Construction Engineering and Management*, Vol. 128, No. 2, pp. 103-109.

Gutjahr, W.J., Katzensteiner, S., Reiter, P., Stummer, C., and Denk, M., (2008). "Competence-driven project portfolio selection, scheduling and staff assignment", *Central European Journal of Operations Research*, Vol. 16, No.3, pp. 281-306.

Hartmann, S., (2002). "A self-adapting genetic algorithm for project scheduling under resource constraints", *Naval Research Logistics*, Vol. 49, No.5, pp. 433-448.

Hartmann, S., (2013). "Project scheduling with resource capacities and requests varying with time: a case study", *Flexible Services and Manufacturing Journal*, Vol. 25, No. 1-2, pp. 74-93.

Hartmann, S. and Briskorn, D., (2010). "A survey of variants and extensions of the resource-constrained project scheduling problem", *European Journal of Operational Research*, Vol. 207, No. 3, pp. 1-14.

Ho, S., and Leung, J., (2010). "Solving a manpower scheduling problem for airline catering using metaheuristics", *European Journal of Operational Research*, Vol. 202, pp. 903-921.

Hosseinian, A.H., and Baradaran, V., (2019a). "Detecting communities of workforces for the multi-skill resource-constrained project scheduling problem: A dandelion solution approach", *Journal of Industrial and Systems Engineering*, Vol. 12, pp. 72-99.

Hosseinian, A.H., and Baradaran, V., (2019b). "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*, Vol. 12, No. 2, pp. 155-178.

Hosseinian, A.H., and Baradaran, V., (2019c). "An Energy-efficient Mathematical Model for the Resource-constrained Project Scheduling Problem: An Evolutionary Algorithm", *Iranian Journal of Management Studies*, Vol. 12, No. 1, pp. 91-119.

Hosseinian, A.H., Baradaran, V., and Bashiri, M., (2019). "Modeling of the time-dependent multi-skilled RCPSP considering learning effect: An evolutionary solution approach", *Journal of Modelling in Management*, Vol. 14, No. 2, pp. 521-558.

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

Kazemipoor, H., Tavakkoli-Moghaddam, R., Shahrezaei, P., and Azaron, A., (2013a). "A differential evolution algorithm to solve multi-skilled project portfolio scheduling problems", T*he International Journal of Advanced Manufacturing Technology*, Vol. 64, No. 5-8, pp. 1099-1111.

Kazemipoor, H., Tavvakoli-Moghaddam, R., and Sharezaei, P., (2013b). "Solving a novel multi-skilled project scheduling model by scatter search", *The South African Journal of Industrial Engineering*, Vol. 24, No.1, pp. 121-135.

Knowles, J., and Corne, D., (1999). "The Pareto archived evolution strategy: a new baseline algorithm for Pareto multi-objective optimization", In: Proceedings of the 1999 Congress on Evolutionary Computation, IEEE, pp. 98-105, Washington, DC, USA.

Kolisch, R., and Hartmann, S., (1999). "Heuristic algorithms for solving the resource-constrained project scheduling problem: classification and computational analysis", In: Weglarz, J., ed. Project scheduling: recent models, algorithms and applications. New York: Kluwer Academic, pp. 147–178.

Kolisch, R., and Sprecher A., (1996). "PSPLIB - A project scheduling problem library", *European Journal of Operational Research*, Vol. 96, No. 1, pp. 205-216.

Laszczyk, M., and Myszkowski, P., (2019). "Improved selection in evolutionary multi–objective optimization of multi–skill resource–constrained project scheduling problem", *Information Sciences*, Vol. 481, pp. 412-431.

Li, H., and Womer, K., (2009). "Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm", Journal of Scheduling, Vol. 12, pp. 281-298.

Liu, S., and Wang C., (2012). "Optimizing linear project scheduling with multi-skilled crews", *Automation in Construction*, Vol. 24, pp. 16-23.

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

Maghsoudlou, H.R., Afshar-Nadjafi, B., and 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*, Vol. 54, pp. 46-61.

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

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

Myszkowski, P., Skowronski, M., Olech, L.P., and Oslizlo, K., (2015). "Hybrid ant colony optimization in solving multi-skill resource-constrained project scheduling problem", *Soft Computing*, Vol. 19, No. 12, pp. 3599-3619.

Myszkowski, P., Olech, L.P., Laszczyk, M., and Skowronski, M., (2018). "Hybrid Differential Evolution and Greedy Algorithm (DEGR) for solving Multi-Skill Resource-Constrained Project Scheduling Problem", *Applied Soft Computing*, Vol. 63, pp. 1-14.

Najafzad, H., Davari-Ardakani, H., and Nemati-Lafmejani, R., (2019). "Multi-skill project scheduling problem under time-of-use electricity tariffs and shift differential payments", *Energy*, Vol. 168, pp. 619-636.

Rahmati, S.H.A, Hajipour, V., and Niaki, S.T.A., (2013). "A soft-computing Pareto-based meta-heuristic algorithm for a multi-objective multi-server facility location problem", *Applied Soft Computing*, Vol. 13, pp. 1728-1740.

Schott, J.R., (1995). "Fault tolerant design using single and multi-criteria genetic algorithms optimization", Dissertation, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA.

Schwindt, C., and Zimmermann, J., (2015). "Handbook on Project Management and Scheduling Vol.1", Springer International Publishing Switzerland. https://doi.org/10.1007/978-3-319-05443-8.

Tabrizi, B.H., Tavvakoli-Moghaddam, R., and Ghaderi, S.F., (2014). "A two-phase method for a multi-skilled project scheduling problem with discounted cash flows", *Scientia Iranica*, Vol. 21, No. 3, pp. 1083-1095.

Tavana, M., Abtahi, A.R., and Khalili-Damghani, K., (2014). "A new multi-objective multi-mode model for solving preemptive time–cost–quality trade-off project scheduling problems", *Expert Systems with Applications*, Vol. 41, pp. 1830-1846.

Tirkolaee, E.B., Goli, A., Hematian, M., Kumar Sangaiah, A., and Han, T., (2019). "Multi-objective multi-mode resource constrained project scheduling problem using Pareto-based algorithms", *Computing*, Vol. 101, No. 6, pp. 547-570.

Valls, V., Perez, A., and Quintanilla, S., (2009). "Skilled workforce scheduling in Service Centers", *European Journal of Operational Research*, Vo. 193, No. 3, pp. 791-804.

Wang, L., and Zheng, X.L., (2018). "A knowledge-guided multi-objective fruit fly optimization algorithm for the multi-skill resource constrained project scheduling problem", *Swarm and Evolutionary Computation*, Vol. 38, pp. 54-63.

Wu, M., and Sun, S., (2006). "A project scheduling and staff assignment model considering learning effect", *The International Journal of Advanced Manufacturing Technology*, Vol. 28, No. 11, pp. 1190-1195.

Zabihi, S., Rashidi Kahag, M., Maghsoudlou, H.R., and Afshar-Nadjafi, B., (2019). "Multi-objective teaching-learning-based meta-heuristic algorithms to solve multi-skilled project scheduling problem", *Computers & Industrial Engineering*, Vol. 136, pp. 195-211.

Zheng, H., Wang, L., and Zheng, X., (2015). "Teaching–learning-based optimization algorithm for multi-skill resource constrained project scheduling problem", *Soft Computing*, Vol. 21, No. 6, pp. 1537-1548.

Zitzler, E., and Thiele, L., (1998). "Multi-objective optimization using evolutionary algorithms — A comparative case study", In: Eiben A.E., Bäck T., Schoenauer M., Schwefel HP. (eds) Parallel Problem Solving from Nature — PPSN V. PPSN 1998. Lecture Notes in Computer Science, Vol. 1498. Springer, Berlin, Heidelberg.

Zitzler, E., (1999). "Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications", Dissertation, Swiss Federal Institute of Technology (ETH), Zürich, Switzerland.

This article can be cited: Hosseinian A.H., Baradaran V., (2020). " Modified Pareto archived evolution strategy for the multi-skill project scheduling problem with generalized precedence relations", *Journal of Industrial Engineering and Management Studies*, Vol. 7, No. 1, pp. 59-86.