RAIRO-Oper. Res. 53 (2019) 1877–1898
https://doi.org/10.1051/ro/2018103

RAIRO Operations Research
www.rairo-ro.org

PREEMPTIVE MULTI-SKILLED RESOURCE CONSTRAINED PROJECT SCHEDULING PROBLEM WITH HARD/SOFT INTERVAL DUE DATES

Hamidreza Maghsoudlou[1], Behrouz Afshar-Nadjafi[2,*] and Seyed Taghi Akhavan Niaki[3]

Abstract. This paper considers a preemptive multi-skilled resource constrained project scheduling problem in a just-in-time environment where each activity has an interval due date to be completed. In this problem setting, resuming a preempted activity requires an extra setup cost, while each time unit violation from the given due date incurs earliness or tardiness penalty. Also, processing cost of each skill to execute any activity depends on the assigned staff member to accomplish the skill. The objective function of the model aims to minimize the total cost of allocating staff to skills, earliness–tardiness penalties and preemption costs. Two integer formulations are proposed for the model which are compared in terms of number of variables, constraints and elapsed run-time to optimality. Furthermore, an ant colony based metaheuristic is developed to tackle real life scales of the proposed model. This algorithm relies on two intelligent local search heuristics. Parameters of the algorithm are calibrated using Taguchi method. The results of the experiments for the proposed algorithm confirm that the proposed algorithm has satisfying performance.

Mathematics Subject Classification. 90B35, 90C11, 90C59.

Received August 13, 2017. Accepted November 13, 2018.

1. Introduction and literature review

Multi-skilling plays an important role in many industries like chemical and construction industries where concept of work flexibility is crucial. Case studies show that with multi-skilling as an alternative, successful managers exploit it to improve their competiveness and productivity. In addition to avoiding from convey of workforces and new employments, multi-skilling gives rise to more motivated and more creative workers [39].

There are many extensions of resource constrained project scheduling problem (RCPSP) in the literature, which is one of the most challenging problems in the combinatorial optimization. Ballestin *et al.* developed an evolutionary algorithm for RCPSP with minimal and maximal time-lags [5]. Artigues *et al.* considered uncertainty of durations in RCPSP using robust optimization approach [4]. They developed two different algorithms

Keywords. Multi-skilled, project scheduling, due date, preemption, ant colony optimization.

[1] Department of Industrial Engineering, Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran.

[2] Department of Industrial Engineering, Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran.

[3] Department of Industrial Engineering, Sharif University of Technology, Tehran, Iran.

*Corresponding author: afsharnb@alum.sharif.edu

called scenario-relaxation algorithm and scenario-relaxation based heuristic algorithm for solving the model. Koné *et al.* extended the classical RCPSP assuming that resources can be produced and consumed by the activities [28]. Okudo *et al.* proposed a new heuristic called mask calculation algorithm to solve RCPSP considering power restriction during peak hours, energy consumption during setup operations and contract demand [37]. Kellenbrink and Helber formulated RCPSP as a binary linear programming model assuming a flexible structure for project [26].

Preemption has been a crucial aspect of many research works in the literature of project scheduling. Ballestin *et al.* formulated the RCPSP for preemptive case assuming that an activity can be interrupted at most once. Their results show that allowed preemption can decrease project's makespan [6]. Afshar-Nadjafi and Majlesi formulated preemptive RCPSP as a mixed integer linear programming model assuming a setup time to resume preempted activities [1] Moukrim *et al.* proposed a mathematical model and an efficient branch and bound algorithm for preemptive RCPSP [35].

Well-timed scheduling is a major characteristic of just-in-time (JIT) environment within machine scheduling. There are many works in the field of project scheduling which deal with well-timed scheduling. Ranjbar *et al.* proposed a mixed integer linear programming model for RCPSP to minimize total weighted tardiness penalty of resources [38]. They suggested a branch and bound algorithm for solving the proposed model. Afshar-Nadjafi and Shadrokh developed a branch and bound algorithm to solve project scheduling problem to minimize weighted earliness–tardiness costs [2]. Khoshjahan *et al.* proposed a binary linear programming model along with two metaheuristic algorithms for the RCPSP with activity due dates to minimize net present value of earliness–tardiness penalties [27].

One of the interesting extensions of RCPSP which is common in many real word situations is multi-skilled project scheduling problem (MSPSP). This problem was firstly introduced by Néron and Baptista which its assumptions are inspired from constraints of multi-purpose machines [36]. Therefore, MSPSP models are integration of two different models including the classic RCPSP and multi-purpose machine scheduling problems. In the last two decades, several works are devoted to different formulations and solution algorithms for MSPSP [7, 16] and [18].

Fırat and Hurkens formulated a multi-level multi-skilled project scheduling problem supposing that all technicians have special skills [21]. It was assumed in their work that a special group of technicians should work together during a workday to execute tasks which are assigned to them. The objective was to assign the groups' workload in order to maximize daily executed tasks. Their solution algorithm is an improved version of the solution procedure presented by Hurkens where incorporating the strength of MIP modeling in schedule construction were studied [23]. Correia and Saldanha-da-Gama revisited MSPSP to minimize total costs including fixed and variable costs of using resources [17]. In order to have a practical model, they considered several real-life assumptions in their proposed methodology. A summary of the recent works dealing with different aspects of the MSPSP is represented in Table 1.

As mentioned above, there are many works regarding different versions of the multi-skilled project scheduling problem. However, to the best of the authors' knowledge, there is no work devoted to the preemptive multi-skilled project scheduling problem in a JIT environment, which is the main motivation of this paper. More specifically, contributions of the current work are as follows. (1) We propose two integer mathematical formulations for the preemptive multi-skilled resource constrained project scheduling problem with given interval due dates for the activities. (2) The proposed formulations are compared for 30 small scale instances in terms of number of variables, number of constraints and elapsed run-time to obtain the optimal solution by a commercial solver, GAMS. (3) We develop an ant colony based metaheuristic algorithm with calibrated parameters to tackle the proposed problem in large instances. Two local search heuristics are embedded in the proposed method to improve the performance of the developed algorithm in exploration of the search space. (4) Performance of the solution approach and efficiency of the local search heuristics are evaluated by applying the algorithm on problem instances with varying number of activities, skills and staff members.

The rest of this paper is organized as follows. Section 2 describes the preemptive multi-skilled project scheduling problem (PMSPSP) and the proposed integer mathematical formulations. Section 3 represents the proposed

TABLE 1. A summary of recent works on the MSPSP.

Work No.	Problem features and solution approach
[3]	Human resources with different skills – Lower bound
[8]	Hierarchical levels of skills – Lower bound
[9]	Branch and bound
[13]	Bi-objective – Fuzzy environment
[17]	Fixed and variable costs of using resources
[21]	Multi-level- Technicians with special skills – MIP
[22]	MSPSP for IT projects – CPLEX
[24]	A real production scheduling – Hybrid heuristic
[25]	project portfolio scheduling- Differential evolution
[29]	Minimize staff costs – hybrid benders decomposition algorithm
[31]	Single and multi-skilled crews – Constraint programming
[32]	Multi-objective – Multi-mode – Weed optimization algorithm
[34]	Branch and price

ant colony based metaheuristic algorithm to solve the PMSPSP, including the developed local search heuristics and the parameters tuning. Section 4 provides the computational results of a comprehensive experimental analysis. Finally, Section 5 includes summary of the paper and several suggestions for future works.

2. PROBLEM DESCRIPTION

The preemptive multi-skilled project scheduling problem (PMSPSP) can be represented as an activity-on-node (AON) graph, $G = (N, A)$, where N denotes activities of the project, and A denotes the finish-to-start prerequisite relations with zero time-lags between activities of the project. All the activities are preempt-able and are numbered from a dummy start activity 0 to a dummy end activity $n + 1$. Indeed, interval due date is a realistic assumption for situations in which completion of an activity is preferred or have to be within a time interval. For instance, when meteorological reports forecast 6 sunny days in rainy spring, we *have to* complete concreting of a building's foundation within these days. In such a situation, completion of the activity before the lower bound of the interval and after the upper bound of the interval is impossible because it results in imperfect concreting. Such an interval is called "hard due-date", in which the activity *have to be* completed. In addition, we assume that there is a tighter interval called "soft due-date" for each activity, in which the activity is *preferred* to be completed. For instance, the misty day immediately before precipitation, and the sopped day immediately after precipitation may lead to an extra cost and effort in concreting. So, it would be better to avoid doing the activity within these days. Therefore, we will have a preferred soft interval including 4 days for doing the concreting process. Each time unit violation from the soft interval due date gives rise to earliness or tardiness penalty. The objective is to find the optimal assignment of staff members to the skills and the optimal schedule of the activities to minimize total cost of the project. Moreover, the following further assumptions are considered:

- All renewable resources are multi-skilled manpower.
- All parameters of the problem are deterministic.
- Set up time of all activities are zero.
- Processing time of all activities are non-negative integer.
- Each staff cannot be devoted to more than one activity at any time frame.
- Each staff cannot be devoted to more than one skill at any time frame.
- Execution of each skill with different staffs has different costs.
- Activity preemption is permitted with penalty and in discrete time frames.

TABLE 2. Notations used in formulation of PMSPSPs.

Indices and sets	
i, j	Index of activity $N = \{1, 2, \ldots, n\}$
m, h	Index of staff member $R = \{1, 2, \ldots, M\}$
k	Index of discrete time unit of activity's duration $K_i = \{1, 2, \ldots, P_i\}$
t	Index of time $t = 0, 1, \ldots, T$
s	Index of skill $s = 0, 1, \ldots, S$
$G = (N, A)$	Prerequisite relations graph
Parameters	
P_i	Processing time of activity i
PC_{ms}	Cost of performing skill s by staff m per time unit
b_{is}	Required number of staffs to perform skill s of activity i
$[D_i^l, D_i^u]$	Soft interval due date of activity i
$[TW_i^l, TW_i^u]$	Hard interval due date of activity i
e_i	Earliness penalty of activity i per time unit
t_i	Tardiness penalty of activity i per time unit
η_i	Preemption penalty of activity i
r_{ms}	1 If staff m has skill s; 0 otherwise
M	Large constant positive value
Variables	
Z_{ikt}	1 If part k of activity i is done at time t; 0 otherwise
X_{imkt}	1 If part k of activity i is done by staff m at time t; 0 otherwise
Y_{imks}	1 If skill s for part k of activity i is done by staff m; 0 otherwise
E_i	Earliness of activity i
T_i	Tardiness of activity i
C_i	Completion time of activity i

- All the required resources have to be available at the start of each activity.
- Each activity may need one or more skills to be performed.
- Each staff may be devoted to perform different skills of activities in different time frames.
- All the required skills of each activity have to be started concurrently.
- All the required skills need to be available during the execution of each activity.

There are several applications for the above mentioned problem in practice. Overhaul maintenance projects, event projects and construction projects are among the long list of projects dealing with multi-skilled workforce, which their preemptive activities have to be completed within predefined time interval due dates.

Herein, two mathematical integer programming formulations are developed for the problem PMSPSP. We call them PMSPSP-A and PMSPSP-B. In order to formulate the PMSPSP-A and PMSPSP-B, the notation is defined in Table 2.

2.1. The formulation of PMSPSP-A

In the first formulation for the PMSPSP, we use all binary and integer decision variables defined in Table 2. Inspiring from the variables defined in the model developed by Montoya *et al.*, the preemptive multi-skilled project scheduling problem can be formulated as follows [34]:

$$\begin{aligned}\text{Min } f = &\sum_{i=1}^{n} e_i \times E_i + \sum_{i=1}^{n} t_i \times T_i + \sum_{i=1}^{n} \eta_i \sum_{k=1}^{P_i - 1} \frac{1}{2} \left[\sum_{t=1}^{T-1} |Z_{ikt} - Z_{i,k+1,t+1}| \right] \\ &+ \sum_{i=1}^{n} \sum_{m=1}^{M} \sum_{k=1}^{P_i} \sum_{s=1}^{S} PC_{ms} \times Y_{imks} \end{aligned} \tag{2.1}$$

$$\sum_{t=TW_i^l}^{TW_i^u} Z_{ikt} = 1 \quad ; \quad \forall i \epsilon N, k \epsilon K_i \tag{2.2}$$

$$\sum_{t=TW_i^l}^{TW_i^u-1} t \times Z_{ikt} \leq \sum_{\acute{t}=TW_i^l}^{TW_i^u} \acute{t} \times Z_{i,k+1,\acute{t}} \quad ; \quad \forall i \epsilon N, k \epsilon K_i | P_i \tag{2.3}$$

$$\sum_{t=TW_i^l}^{TW_i^u} t \times Z_{ikt} \leq \sum_{\acute{t}=TW_j^l}^{TW_j^u} \acute{t} \times Z_{j1\acute{t}} \quad ; \quad \forall (i,j) \epsilon A, k \epsilon K_i \tag{2.4}$$

$$\sum_{i=1}^{N} \sum_{k=1}^{P_i} X_{imkt} \leq 1 \quad ; \quad \forall m \epsilon R, t = 0, 1, \ldots, T \tag{2.5}$$

$$Y_{imsk} \leq r_{ms} \quad ; \quad \forall i \epsilon N, \forall m \epsilon R, \forall k \epsilon K_i, s = 1, \ldots, S \tag{2.6}$$

$$X_{imkt} \leq Z_{ikt} \quad ; \quad \forall i \epsilon N, \forall m \epsilon R, \forall k \epsilon K_i, t = TW_i^l, \ldots, TW_i^u \tag{2.7}$$

$$X_{imkt} + 1 \leq Z_{ikt} + \sum_{s=1}^{S} Y_{imsk} \quad ; \quad \forall i \epsilon N, \forall m \epsilon R, \forall k \epsilon K_i, t = TW_i^l, \ldots, TW_i^u \tag{2.8}$$

$$\sum_{m=1}^{M} \sum_{t=TW_i^l}^{TW_i^u} X_{imkt} \leq \sum_{s=1}^{S} b_{is} \quad ; \quad \forall i \epsilon N, k \epsilon K_i \tag{2.9}$$

$$\sum_{m=1}^{M} Y_{imks} = b_{is} \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, s = 1, \ldots, S \tag{2.10}$$

$$\sum_{t=TW_i^l}^{TW_i^u} X_{imkt} = \sum_{s=1}^{S} Y_{imks} \quad ; \quad \forall i \epsilon N, \forall m \epsilon R, \forall k \epsilon K_i \tag{2.11}$$

$$C_i \geq t \times Z_{ikt} \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, t = TW_i^l, \ldots, TW_i^u \tag{2.12}$$

$$C_i \leq t + M \times \sum_{\acute{t}=t+1}^{TW_i^u} Z_{ik\acute{t}} \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, t = TW_i^l, \ldots, TW_i^u \tag{2.13}$$

$$E_i \geq D_i^l - C_i \quad ; \quad \forall i \epsilon N \tag{2.14}$$

$$T_i \geq C_i - D_i^u \quad ; \quad \forall i \epsilon N \tag{2.15}$$

$$E_i, T_i, C_i \geq 0 \text{ and integers} \quad ; \quad \forall i \epsilon N \tag{2.16}$$

$$Z_{ikt}, X_{imkt}, Y_{imks} \in \{0, 1\} \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, \forall m \epsilon R, t = 0, \ldots, T, s = 1, \ldots, S. \tag{2.17}$$

Objective function in equation (2.1) minimizes the total cost of the project including earliness, tardiness, preemption and processing costs. Constraints set (2.2) guarantee that each part of any activity should be done only once during its hard permitted interval. Constraints set (2.3) preserve precedence relations between successive parts of each activity. Constraints set (2.4) maintain finish to start prerequisite relations between activities. Constraints set (2.5) describe that each staff should be devoted to at most one part of one activity at any time frame. Constraints set (2.6) stipulates that the staff member who is devoted to implement a part of an activity should has the required skill. Constraints sets (2.7) and (2.8) enforce all the assigned staffs to start their work on the related part of the activity concurrently. Constraints set (2.9) maintain that the number of assigned staffs to a part of an activity should not be more than the required number of staffs for performing that activity. Constraints set (2.10) guarantee that all the assigned staff members to perform a skill for a part of an

activity should be equal to the required number of staff members for performing the related skill. Constraints set (2.11) states that each part of any activity is implemented by the assigned staff members to it. Constraints sets (2.12) and (2.13) compute completion time of activities, where M is a considerably large positive value. Constraints sets (2.14) and (2.15) compute earliness and tardiness of activities, respectively. Constraints sets (2.16) and (2.17) describe that decision variables E_i, T_i and C_i are integers, while Z_{ikt}, X_{imkt} and Y_{imks} are binary. It is clear that, except the third element of the objective function in equation (2.1), the model is linear. The third part of the objective function which is nonlinear, can be stated in linear form as follows:

$$\sum_{i=1}^{n} \eta_i \sum_{k=1}^{P_i} \frac{1}{2} \left[\sum_{t=1}^{T} (Z'_{ikt} + Z''_{ikt}) \right] \tag{2.18}$$

where Z'_{ikt} and Z''_{ikt} are complementary binary variables which satisfy the following condition:

$$Z_{ikt} - Z_{i,k+1,t+1} = Z'_{ikt} + Z''_{ikt} \quad ; \quad \forall i \epsilon N, k \epsilon K_i | P_i, \mathrm{t} = 0, \ldots, \mathrm{T}. \tag{2.19}$$

By replacing the third element of the objective function in equation (2.1) by equation (2.18), and adding the Constraints set (2.19), the model is completely linear. In doing so, the number of constraints is increased by at most $n.T.\sum_{i=1}^{n} P_i$, and the number of binary variables is increased by at most $2n.T.\sum_{i=1}^{n} P_i$. Thus, Constraints set (2.17) should be replaced by Constraints set (2.20) as follows:

$$Z_{ikt}, X_{imkt}, Y_{imks}, Z'_{ikt}, Z''_{ikt} \in \{0, 1\} \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, \forall m \epsilon R, t = 0, \ldots, T, s = 1, \ldots, S. \tag{2.20}$$

2.2. The formulation of PMSPSP-B

In the second formulation of the PMSPSP, we use all three integer decision variables and only two binary decision variables Y_{imks} and X_{imkt} defined in Table 2. Although we have less number of variables than PMSPSP-A, the intensity of the constraint coefficients and number of constraints are the other factors which contribute to complexity of the formulations. Inspiring from the model developed by Néron and Baptista in dealing with precedence constraints, we have [36]:

$$\begin{aligned} \mathrm{Min}\, f = & \sum_{i=1}^{n} e_i \times E_i + \sum_{i=1}^{n} t_i \times T_i + \frac{1}{2} \sum_{i=1}^{n} \eta_i \sum_{t=TW_i^l}^{TW_i^u - 1} \sum_{k=1}^{P_i - 1} \left[\left| \frac{\sum_{m=1}^{M} X_{imkt}}{\sum_{s=1}^{S} b_{is}} - \frac{\sum_{m=1}^{M} X_{i,m,k+1,t+1}}{\sum_{s=1}^{S} b_{is}} \right| \right] \\ & + \sum_{i=1}^{n} \sum_{m=1}^{M} \sum_{k=1}^{P_i} \sum_{s=1}^{S} PC_{ms} \times Y_{imks} \end{aligned} \tag{2.21}$$

$$\sum_{t=TW_i^l}^{TW_i^u} X_{imkt} = 1 \quad ; \quad \forall i \epsilon N, \forall m \epsilon R, \forall k \epsilon K_i \tag{2.22}$$

$$\frac{\sum_{t=TW_i^l}^{TW_i^u} \sum_{m=1}^{M} X_{imkt}}{\sum_{s=1}^{S} b_{is}} + 1 \leq \frac{\sum_{\acute{t}=TW_i^l}^{TW_i^u} \sum_{m=1}^{M} X_{im,k+1,\acute{t}}}{\sum_{s=1}^{S} b_{is}} \quad ; \quad \forall i \epsilon N, k \epsilon K_i | P_i \tag{2.23}$$

$$\frac{\sum_{t=TW_i^l}^{TW_i^u} \sum_{m=1}^{M} t \times X_{imkt}}{\sum_{s=1}^{S} b_{is}} + 1 \leq \frac{\sum_{\acute{t}=TW_j^l}^{TW_j^u} \sum_{m=1}^{M} \acute{t} \times X_{j1k\acute{t}}}{\sum_{s=1}^{S} b_{js}} \quad ; \quad \forall (i,j) \epsilon A, k \epsilon K_i. \tag{2.24}$$

Constraints sets (2.5) and (2.6)

$$\sum_{t=TW_i^l}^{TW_i^u} t \times X_{imkt} \leq \frac{\sum_{\acute{t}=TW_i^l}^{TW_i^u} \sum_{h=1}^{M} \acute{t} \times X_{ihk\acute{t}}}{\sum_{s=1}^{S} b_{is}} \quad ; \quad \forall i \epsilon N, \forall m \epsilon R, \forall k \epsilon K_i. \tag{2.25}$$

Constraints sets (2.9), (2.10) and (2.11)

$$t \times \frac{\sum_{m=1}^{M} X_{imkt}}{\sum_{s=1}^{S} b_{is}} \leq C_i \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, t = TW_i^l, \ldots, TW_i^u \tag{2.26}$$

$$t + M \times \sum_{\acute{t}=t+1}^{TW_i^u} \sum_{m=1}^{M} X_{imk\acute{t}} \geq C_i \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, t = TW_i^l, \ldots, TW_i^u. \tag{2.27}$$

Constraints sets (2.14), (2.15) and (2.16)

$$X_{imkt}, Y_{imks} \in \{0,1\} \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, \forall m \epsilon R, t = 0, \ldots, T, s = 1, \ldots, S. \tag{2.28}$$

Objective function in equation (2.21) minimizes the total cost of the project as mentioned before. Constraints set (2.22) describe that each part of any activity should be done only once during its hard permitted interval. Constraints set (2.23) preserve precedence relations between successive parts of each activity. Constraints set (2.24) maintain finish to start prerequisite relations between activities. Constraints set (2.25) enforce all the assigned staff memberss to start their work simultaneously. Constraints sets (2.26) and (2.27) compute completion time of activities, where inf is a considerably large positive value. Constraints set (2.28) describe that X_{imkt} and Y_{imks} are binary. Notice that Constraints sets (2.5), (2.6), (2.9), (2.10), (2.11), (2.14), (2.15) and (2.16) are duplicated from model PMSPSP-A to PMSPSP-B. Again, we just face with a non-linear part in the objective function in equation (2.21). In order to linearize the third part of the objective function we replace it with equation (2.18) where Z'_{ikt} and Z''_{ikt} are complementary binary variables such a way that satisfy:

$$\frac{\sum_{m=1}^{M} X_{imkt}}{\sum_{s=1}^{S} b_{is}} - \frac{\sum_{m=1}^{M} X_{i,m,k+1,t+1}}{\sum_{s=1}^{S} b_{is}} = Z'_{ikt} - Z''_{ikt} \quad ; \quad \forall i \epsilon N, k \epsilon K_i | P_i, t = 0, \ldots, \mathrm{T}. \tag{2.29}$$

Therefore, equation (2.28) should be replaced by equation (2.30) as follows:

$$X_{imkt}, Y_{imks}, Z'_{ikt}, Z''_{ikt} \in \{0,1\} \quad ; \quad \forall i \epsilon N, \forall k \epsilon K_i, \forall m \epsilon R, t = 0, \ldots, T, s = 1, \ldots, S. \tag{2.30}$$

2.3. Comparison of the formulations

In order to compare the proposed formulations, 50 small scale instances with up to 15 activities, are generated. Each problem instance is solved by both formulations using GAMS software (Solver CPLEX) version 24.1.3 on a PC with 4 GB RAM and Core i5 CPU. The number of variables, number of constraints, elapsed run-time to optimality and the optimal values of the objective function is reported in Table 3. As it would be expected, the optimal values of the objective function for the two models are identical. It can be seen in Table 3 that the average number of decision variables and constraints for the PMSPSP-A is more than those of the PMSPSP-B. On the other hand, the elapsed run-time to optimality for the PMSPSP-A is considerably less than the one in PMSPSP-B. In order to compare the formulations in solving the instances, a paired t-test that allows one to compare the means of the elapsed run-times of the two formulations is implemented here. Indeed, a single factor (formulation) experiment with two levels (PMSPSP-A and PMSPSP-B) is designed. From a statistical analysis, we calculate a 95% confidence interval for the mean difference of the elapsed run-time to optimality as $(-4152, -12)$. Since this confidence interval does not contain zero, it means that there is a significance difference between the elapsed run-time in using GAMS to solve the problems modeled by the two developed formulations. This surprising result can just be ascribed to the density of the constraint coefficients, *i.e.*, the constraint matrix for PMSPSP-A is sparser than PMSPSP-B. Finally, while the two formulations are correct in achieving the optimal solution, the PMSPSP-A outperforms the PMSPSP-B measured by average computational time.

TABLE 3. Computational results of PMSPSP-A and PMSPSP-B.

Problem no	# act.	# skill	# staff	PMSPSP-A				PMSPSP-B			
				# var.	# const.	Run-time (s)	f^*	# var.	# const.	Run-time (s)	f^*
1	3	2	2	666	862	0.025	156	470	617	0.025	156
2	3	3	3	1162	1397	0.035	244	702	1111	0.056	244
3	4	2	2	1087	1519	0.005	504	804	954	0.004	504
4	4	3	3	1258	1645	0.032	367	1240	804	0.032	367
5	5	2	2	2318	2212	0.545	285	1993	1269	0.438	285
6	5	3	3	2602	2493	0.841	451	2550	1291	2.484	451
7	5	4	4	2023	2264	0.941	455	3328	1189	2.238	455
8	6	2	2	1851	2035	0.436	438	1715	1107	0.542	438
9	6	3	3	2947	2925	0.622	433	2939	1493	1.621	433
10	6	4	4	3643	3739	0.864	793	3637	1789	16.903	793
11	7	2	2	5082	3448	0.775	608	4468	2172	0.657	608
12	7	3	3	4621	3734	0.662	675	4411	2031	1.267	675
13	7	4	4	4818	4440	0.927	745	4914	2162	1.613	745
14	8	2	2	3377	2975	0.573	710	3181	1705	0.558	710
15	8	3	3	4309	3814	0.858	613	4397	1982	2.685	613
16	8	4	4	5833	5621	1.273	1438	5619	2705	207.98	1438
17	8	5	5	5985	5764	0.663	1091	6288	2715	65.57	1091
18	9	2	2	6760	4514	1.1	674	6031	2817	1.038	674
19	9	3	3	8776	5816	1.285	816	8014	3306	2.84	816
20	9	4	4	9568	6634	1.482	786	9559	3501	4.128	786
21	9	5	5	8958	7133	2.121	1207	9593	3462	1558.398	1207
22	10	2	2	6971	4749	0.937	872	6221	2989	1.272	872
23	10	3	3	6811	5847	3.336	1004	6501	3053	81.719	1004
24	10	4	4	7735	6,139	1.003	749	8217	3113	3.688	749
25	10	5	5	10 426	8065	1.514	1401	10 940	3934	39.03	1401
26	11	2	2	7272	5254	1.05	987	6491	3207	4.963	987
27	11	3	3	7222	5,910	1.524	915	7148	3136	74.908	915
28	11	4	4	9186	7044	2.413	1448	9568	3622	433.52	1448
29	11	5	5	11 024	8579	1.859	1359	11 725	4218	40.691	1359
30	12	2	2	9965	6249	1.811	953	8810	3994	1.33	953
31	12	3	3	9826	7035	2.293	1097	9480	3894	558.492	1097
32	12	4	4	15 949	9605	2.184	1353	15 340	5280	45.466	1353
33	12	5	5	14 870	10 327	10.149	1585	15 146	5188	2876.415	1585
34	12	6	6	16 777	11 581	2.317	1303	18 024	5674	519.536	1303
35	13	2	2	8612	6010	2.419	1066	7876	3706	2.81	1066
36	13	3	3	11 077	7770	2.495	1249	10 789	4339	292.65	1249
37	13	4	4	14 932	9718	4.762	1288	14 744	5204	1,036.361	1288
38	13	5	5	24 056	13 316	2.854	1726	23 328	7080	11.98	1726
39	13	6	6	17 908	12 890	3.332	1728	18 696	6172	816.82	1728
40	14	2	2	16 225	9192	39.561	1229	16 097	5538	33.459	1229
41	14	3	3	17 522	10 024	2.064	1197	17 686	5796	103.585	1197
42	14	4	4	18 819	10 855	2.493	1482	19 235	6053	281.425	1482
43	14	5	5	21 761	13 494	6.464	2084	21 202	6957	5168.055	2084
44	14	6	6	28 069	15 741	4.508	2665	28 648	8,064	7371.535	2665
45	15	2	2	14 938	8592	2.329	1178	13 410	5636	2.92	1178
46	15	3	3	19 558	10 794	4.763	1548	18 154	6538	101.199	1548
47	15	4	4	15 802	10 996	6.989	1501	16 288	5780	2369.95	1501
48	15	5	5	26 376	14 619	3.853	1787	26 075	7828	7539.428	1787
49	15	6	6	26 872	15 690	13.288	2429	27 813	8049	39952.54	2429
50	15	7	7	34 023	19 284	6.225	2333	35 015	9625	32617.09	2333
Average	–	–	–	**10 764**	**7207**	**3.137**	1100	**10 690**	**3877**	**2085.078**	1100

3. Proposed algorithm for PMSPSP

In this section, an Ant Colony Optimization (ACO) based metaheuristic algorithm is presented to solve the real life scales of the PMSPSP. Dorigo firstly suggested the idea of optimization based on ants' behavior [19]. The basic structure of ACO relies on iterative creation of solutions by dispatching a population of artificial ants. During the ants move, they face with several options regarding direction of their next trajectory. The choice of a direction by any ant is a function of dynamic pheromone trails secreted by previously transited ants. Heuristic information is another crucial factor which influences on the chosen direction by an artificial ant. This factor is predominantly dependent on the special structure of the problem in hand. Notable successful applications of ACO dealing with a variety of resource-constrained project scheduling problems motivated us to tackle with PMSPSP by ACO [11, 14, 15] and [30]. In the proposed algorithm, not only the typical intelligence of ACO is exploited, but also two efficient local search heuristics are developed to cope with the special characteristics of PMSPSP.

3.1. Construction of solutions

While in many evolutionary algorithms the initial solutions are generated at random, solutions in ACO are created using a special guided procedure. In this mechanism, n^{ant} feasible solutions are created by n^{ant} artificially contrived ants considering pheromone update. In our proposed algorithm, the problem is partitioned into two sub-problems; determination of the activity list representing the sequence of activities, and assignment of workforces to the required skills of the activities. Each artificial ant in each cycle of the search process creates an activity list and a staff assignment scheme. Before beginning the construction phase, we replace each activity i with duration of p_i by p_i activities with duration of 1 by preserving all the prerequisite relations. Therefore, the reorganized project has $\sum_{i=1}^{n} p_i$ activities (sub-activities) with duration of 1. For the first sub-problem, ant constructs an activity list by selecting activity i to be located at position j with probability p_{ij}^k as follows [33]:

$$p_{ij}^k = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \epsilon S_j} [\tau_{lj}]^\alpha [\eta_{lj}]^\beta} \quad ; \quad \forall j, \forall i \in S_j \tag{3.1}$$

where τ_{ij} is pheromone, S_j is set of activities that are selectable at position j, α and β are control parameters related to the relative impact of pheromone and the heuristic information, respectively. Herein, the heuristic information η_{ij} is defined as follows:

$$\eta_{ij} = \frac{1}{(\text{et})_{ij}} \tag{3.2}$$

where $(\text{et})_{ij}$ is earliness–tardiness cost of locating activity i at position j. This gives higher priority to select the activity with small earliness–tardiness cost. For the second sub-problem, ant assigns a staff i to do skill j with the same probability defined as equation (3.1), where S_j is set of staffs that are assignable to skill j. In this case, the heuristic information η_{ij} is defined as follows:

$$\eta_{ij} = \frac{1}{PC_{ij}}. \tag{3.3}$$

This gives higher priority to assign the staff i with small cost PC_{ij} to process skill j.

With illustrative purpose, let us consider an example with 3 activities, 3 staffs and 3 skills. Assume that the first skill can be performed by staffs 2 and 3, the second skill can be done by all staffs and the third skill can be done by staffs 1 and 2. The other data are summarized in Table 4.

By reorganizing the example project we will have $\sum_{i=1}^{3} p_i = 6$ activities with duration of 1. Construction of a solution for the first sub-problem (activity list) related to this example is depicted in Figure 1, where ($1(^1)$, $2(^1)$, $2(^2)$, $1(^2)$, $3(^1)$, $3(^2)$) is created as a solution. Also, construction of a solution for the second sub-problem (staff assignment) related to this activity list is depicted in Figure 2, where 1st skill of activity $1(^1)$ is assigned to staff 2, 1st and 2nd skill of activity $2(^1)$ is assigned to staffs 2 and 3, respectively, and so on.

TABLE 4. The data for the illustrative example.

Activity	Predecessors	b_{is}	p_i	D_i^l	D_i^u	TW_i^l	TW_i^u
1	–	1st skill	2	2	4	0	6
		$b = 1$					
2	–	1st skill 2nd skill	2	4	5	1	7
		$b = 1$ $b = 1$					
3	1, 2	2nd skill 3rd skill	2	4	7	3	8
		$b = 1$ $b = 2$					

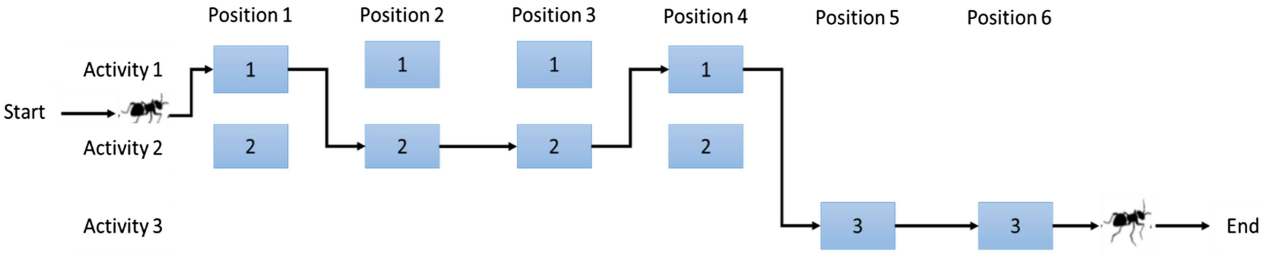

FIGURE 1. Schematic representation for construction of activity list.

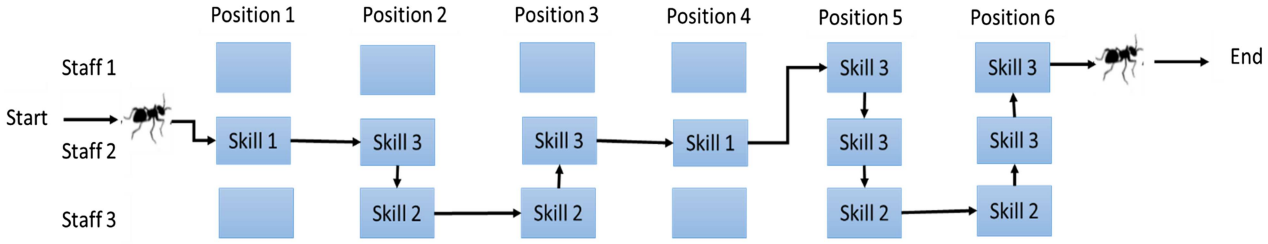

FIGURE 2. Schematic representation for construction of staff assignment.

The above mentioned activity list and staff assignment should be transformed to a precedence and resource feasible schedule. In order to determine the start or finish time for activities we use a scheduling generation scheme which is compatible with special structure of PMSPSP. Let C be set of scheduled activities in a partial schedule. To do this, we select the first unscheduled activity (with duration of 1) from the activity list. Then, we schedule the selected activity with its assigned staffs to be finished at the first possible time after $t = \max(\mathrm{FT}^{\mathrm{Pr}_i} + 1, D_i^l - P_i + 1)$. In doing so, staffs availability and the hard interval due-date are satisfied. $\mathrm{FT}^{\mathrm{Pr}_i}$ denotes the finish time of the all predecessors of activity i. This process is done for the activity provided it doesn't be the last part of an original activity. In this case, the activity will be set to finish at the nearest time to D_i^u which the required staffs are available. Implementation of this procedure for the solution associated with Figures 1 and 2 is demonstrated in Figure 3. It is obvious that the activities 1 and 2 are preempted, while the activity 3 is scheduled without preemption. Also, all the three activities are finished within their hard/soft interval due date (Tab. 5).

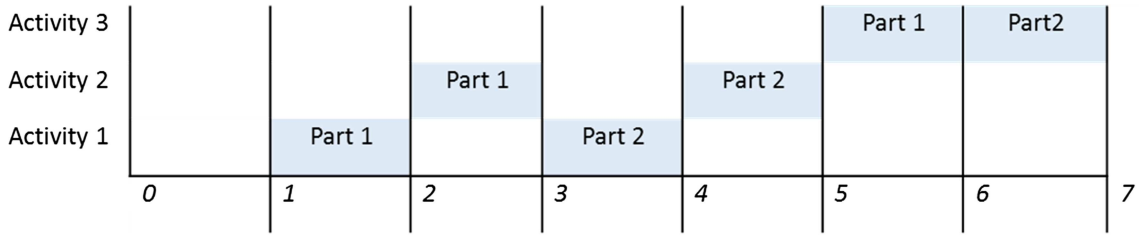

FIGURE 3. Gant chart related to scheduling activities for the illustrative example.

TABLE 5. The pseudo code for construction of solutions.

Initialize parameters: number of ants $[n^{ant}]$, pheromone exponential weight $[\alpha]$, heuristic exponential weight $[\beta]$;
Replace each activity i with duration of p_i by p_i activities with duration of 1, $nVar$;

Calculate heuristic matrix for the first sub-problem from equation (3.1);
Generate an activity list by selecting activity i to be located at position j with probability computed from equation (3.3);
Calculate heuristic matrix for the second sub-problem from equation (3.2);
Assign a staff i to do skill j with probability computed from equation (3.3);

for k =1 to n^{ant} **do**
  for j=1 to $nVar$
    Select the first unscheduled activity from the activity list;
    Schedule the selected activity with its assigned staffs to be finished at the first possible time after $t = \max(\text{FT}^{\text{Pr}_i} + 1, D_i^l - P_i + 1)$**;**
  End for
End for
Report Ants;

3.2. Pheromone updating

In order to update pheromone density, we use local and global pheromone updating mechanisms for both of the sub-problems. At the beginning of the algorithm the values of pheromone are set $\tau_0 = 1$. Local mechanism for the first sub-problem updates the value of pheromone when an activity is added to the current partial activity list, while for the second sub-problem value of pheromone is updated when a staff is assigned to the corresponding required skill as follows:

$$\tau_{ij} = (1 - \xi)\,\tau_{ij} + \xi\tau_0 \tag{3.4}$$

where $\xi \in [0, 1]$ is control parameter of pheromone such a way that $\tau_0 \leq (1 - \xi)\,\tau_{ij} + \xi\tau_0 \leq \tau_{ij}$. This guarantees $\tau_{ij} \geq \tau_0$ during throughout the algorithm.

Global mechanism emphasizes on forgetting worse solutions and updates the value of pheromone based on the best found solution at the end of each iteration. To do this, the value of pheromone is updated as follows when all the ants have built their solutions [12]:

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \rho \cdot \Delta\tau_{ij}^{bs,gs} \tag{3.5}$$

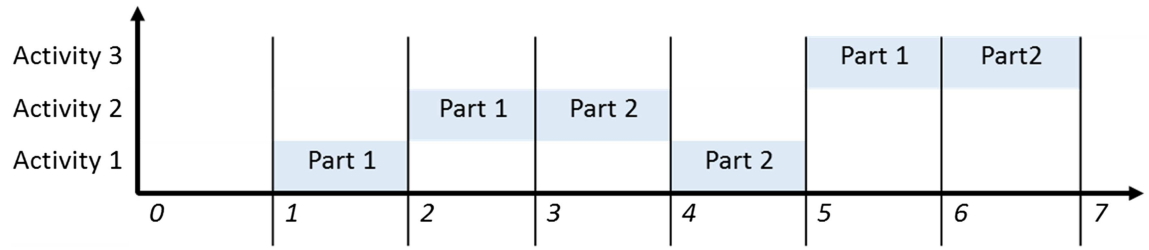

FIGURE 4. Schedule for the illustrative example after implementation of PBLS.

where $\rho \in [0,1]$ is the rate of evaporation and $\Delta\tau_{ij}^{bs,gs}$ denotes the pheromone increase on the arcs applied by best found solution at the end of each iteration. $\Delta\tau_{ij}^{bs,gs}$ for both of the sub-problems is computed as:

$$\Delta\tau_{ij}^{bs,gs} = \left(\frac{\text{first} - \text{Cost}^{bs,gs}}{\text{first} - \text{firstBest}} + 1\right) \cdot \psi \tag{3.6}$$

where first and firstBest represent the objective value of the first found solution by the first ant and the best objective value in the first iteration of the algorithm. Also, $\text{Cost}^{bs,gs}$ denotes the objective value of the best found solution at end of the iteration and throughout the algorithm. $\psi \geq 1$ is control parameter of the impact of pheromone values.

3.3. Local search heuristics

Upon the termination of each iteration, we implement two local search heuristics on the feasible solutions constructed by the ants. The first heuristic is a preemption based local search (PBLS) which tries to remove the time gaps between different parts of an activity to decrease the preemption penalty. On the other hand, the second heuristic is an early–tardy based local search (ETBLS) which tries to shift the activities with earliness or tardiness to right or left to decrease their earliness or tardiness penalties.

- PBLS

As mentioned before, each activity is replaced by sub-activities with duration of 1. Also, in construction of solutions the last part of each activity i is set to finish as much near as to D_i^u. Although this procedure leads to the solutions with minimal earliness–tardiness cost, but each activity may be preempted several times which incurs huge amount of preemption penalty. The preemption based local search (PBLS) iteratively considers the first unconsidered activity in the activity list which is preempted. Assume that in the current schedule qth part of activity i is preempted from its $(q+1)$th part. PBLS procedure shifts the part $(q+1)$ of the activity i to the left such a way that this part is connected to part q. In doing so, the precedence and resource constraints are preserved. Then, the cost-decreasing benefits of this shift is investigated. If it is a beneficial shift, finish time of the involved activities are updated and the current schedule is replaced by the resulting schedule with updated activity list. This procedure is repeated until no further improvement be possible. Implementation of PBLS on the solution depicted in Figure 3 is shown in Figure 4 where it has been beneficial to schedule the activity 2 without preemption.

- ETBLS

After implementation of PBLS, the resulting solution would be subject to an early–tardy based local search (ETBLS). This procedure is conducted on the activities with earliness and tardiness in a different way. For the activities with earliness, ETBLS iteratively considers the first unconsidered activity in the activity list which has earliness in the current schedule, *i.e.* its last part is finished before D_i^l. This procedure shifts the last part

TABLE 6. The pseudo code for the proposed ACO algorithm.

```
model=input-problem-data

initialize parameters: number of ants [n^ant], pheromone exponential weight [α],
heuristic exponential weight [β], evaporation rate [ρ], initial pheromone [τ0], and maximum number iterations [MaxIt];

Calculate heuristic matrix for the first sub-problem and save it to η^1;
Calculate heuristic matrix for the second sub-problem and save it to η^2;
Calculate initial pheromone for the first sub-problem and save it to τ_0^1;
Calculate initial pheromone for the second sub-problem and save it to τ_0^2;
Generate n^ant empty ant structure;

for Iter=1 to MaxIt do
  for k=1 to n^ant do
    for j=1 to nVar
        Select activity for position j of the ant(k);
        Update pheromone matrix for the first sub-problem from equation (3.4);
    End for

    for j=1 to nVar
        Select staff for the activity in the position j of ant (k);
        Update pheromone matrix for the second sub-problem from equation (3.4);
    End for

    Schedule solution obtained by ant (k);
    Evaluate ant (k);
    Apply PBLS for solution obtained by ant(k);
    Apply ETBLS for solution obtained by ant(k);

   if cost of ant(k) is lower than GlobalAnt
        GlobalAnt = ant(k);
    End if
  End for

  Find the best ant in this iteration and save it to BestAnt;
  Update pheromone matrix for the first sub-problem from equation (3.5);
  Update pheromone matrix for the second sub-problem from equation (3.5);
End for

Report globalAnt;

where
nVar is number of positions
```

of the activity one time unit to right if it is beneficial so that the precedence and staff constraints may not be violated. The same strategy acts on the activities with tardiness, which shifts the last part of the activity one time unit to left if it is beneficial. After each beneficial shift, finish time of the activities are updated. This procedure continues until no further improvement be possible.

3.4. General structure of the algorithm

The proposed ant colony algorithm will be continued until a maximum number of search cycles is met. The pseudo code of the algorithm is shown in Table 6.

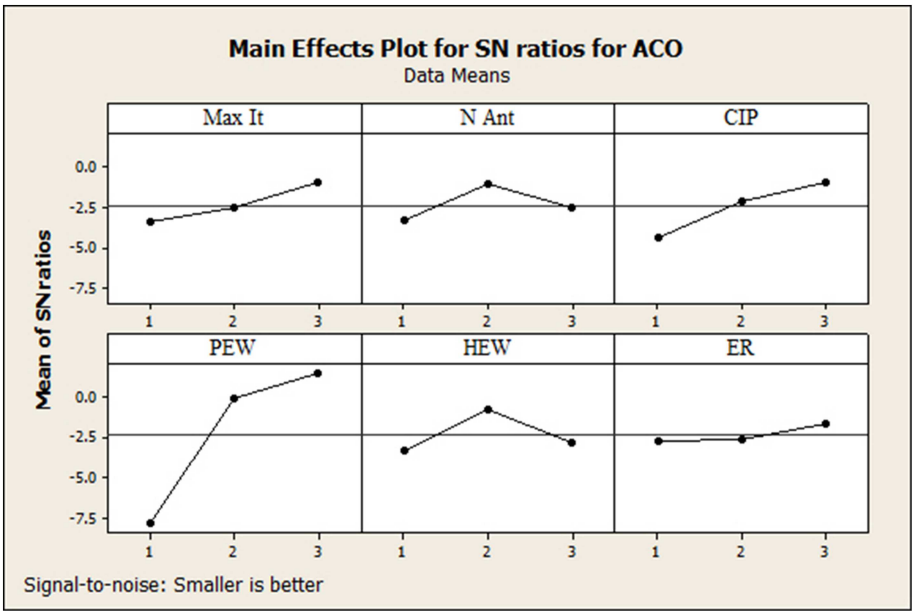

FIGURE 5. S/N ratio plots for each level of the factors for the proposed ACO.

TABLE 7. Factors of ACO and their levels.

Parameters	Parameter levels		
	Level 1	Level 2	Level 3
Maximum number of iterations (Max It)	10*n	20*n	**30*n**
Number of ants (N Ant)	5	**10**	15
Controller for impact of pheromone (CIP): ψ	1	2	**3**
Pheromone exponential weight (PEW): α	1	2	**3**
Heuristic exponential weight (HEW): β	2	**3**	4
Evaporation rate (ER): ρ	0.05	0.10	**0.15**

3.5. Calibration

In order to escalate performance of the developed algorithm, the Taguchi design is applied to calibrate its key parameters [40]. This method suggests the optimal level of factors to minimize the effect of noise. In the proposed algorithm, there are six controllable factors that should be tuned. These factors and their different levels are shown in Table 7.

As it is concluded in [10], there is no uniquely optimal set of ACS parameters yielding best quality solutions in all instances. So, we randomly selected an instance to tune the parameters. To do this, problem instance 8 from Table 3 is applied for calibration of the parameters and a L^{27} orthogonal array design is executed using MINITAB software which is shown in Table 8. To prepare more reliable data we do 3 runs for each experiment and we use the best result. In Taguchi method, signal to noise ratio measures the effect of noise which is defined as follows:

$$\frac{S}{N} = -10 \log \left(\frac{S\left(Y^2\right)}{n} \right). \tag{3.7}$$

TABLE 8. The orthogonal L^{27} design for calibration of the parameters.

Experiments	Max It	N Ant	CIP	PEW	HEW	ER	Response
1	1	1	1	1	1	1	1138
2	1	1	1	1	2	2	1142.3
3	1	1	1	1	3	3	1143.3
4	1	2	2	2	1	1	1113
5	1	2	2	2	2	2	1108
6	1	2	2	2	3	3	1107.3
7	1	3	3	3	1	1	1115.7
8	1	3	3	3	2	2	1103
9	1	3	3	3	3	3	1108
10	2	1	2	3	1	2	1108
11	2	1	2	3	2	3	1110.7
12	2	1	2	3	3	1	1108
13	2	2	3	1	1	2	1130.7
14	2	2	3	1	2	3	1110.7
15	2	2	3	1	3	1	1120.7
16	2	3	1	2	1	2	1113
17	2	3	1	2	2	3	1098
18	2	3	1	2	3	1	1111.3
19	3	1	3	2	1	3	1108
20	3	1	3	2	2	1	1103
21	3	1	3	2	3	2	1115.7
22	3	2	1	3	1	3	1103
23	3	2	1	3	2	1	1108
24	3	2	1	3	3	2	1098
25	3	3	2	1	1	3	1131.3
26	3	3	2	1	2	1	1126
27	3	3	2	1	3	2	1112.7

TABLE 9. Response table for S/N ratios of ACO parameters.

Level	Max It	N Ant	CIP	PEW	HEW	ER
1	−3.4564	−3.3613	−4.439	−7.8705	−3.338	−2.7179
2	−2.5011	−1.1161	−2.1415	−0.0775	−0.8061	−2.6655
3	−1.0205	−2.5126	−0.9927	1.52168	−2.8488	−16 8691
Delta	2.43591	2.2452	3.4463	9.3922	2.53195	1.03103
Rank	4	5	2	1	3	6

The average S/N ratio for each level is depicted in Figure 5. The level with maximum average S/N ratio determines the optimal level of parameters which are highlighted in Table 7.

The response table for the signal to noise ratios is reported in Table 9. It is obvious that pheromone exponential weight (PEW), Controller for impact of pheromone (CIP) and Heuristic exponential weight (HEW) are the three most important factors which influence on the signal to noise ratio S/N, respectively.

4. COMPUTATIONAL RESULTS

This section is devoted to evaluating performance of the proposed ACO algorithm. To do this, a set of 30 small scale project instances with up to 30 real activities are applied to measure the efficiency of the suggested

TABLE 10. The parameter settings for the test problems.

Control parameter	Value
Activity durations	Integer [2, 4]
Number of initial activities	Integer [1, 3]
Number of terminal activities	Integer [1, 2]
Processing cost of skills by staffs	[10, 30]
Earliness and tardiness penalty	[10, 50]
Preemption penalty	[5, 10]
Maximal number of successors and predecessors	3
Lower bound of soft interval due dates	$[\max(D^l_{\text{per}}), \max(D^l_{\text{per}}) + p_i]$
Upper bound of soft interval due dates	$D^l_i + 3$
Lower bound of hard interval due dates	$D^l_i - 5$
Upper bound of hard interval due dates	$D^l_i + 5$

local search heuristics beside the results obtained by GAMS software. In addition, 60 project instances with 40 and 50 activities are applied to measure the robustness of the developed method. The algorithm is coded in MATLAB environment and the experiments are implemented on a PC with 4 GB RAM and Core i5 CPU.

4.1. Test problems setting

Project networks of all the problem instances are generated using a program called ProGen developed by Drexl *et al.* which is a problem generator [20]. Other parameters of the test problems which are used throughout the paper are given in Table 10.

The notation $[a, b]$ denotes that the corresponding parameter is generated uniformly from $[a, b]$. Also, D^l_{per} denotes the maximum lower bound of soft interval due date for predecessors of the activity. The problem instances are generated with different purposes. As mentioned before in section 2.3, 50 problem instances are applied to compare the performance of the developed integer formulations. For that problem set, number of skills and staffs are considered equally from 2 to 7. For the 30 instances which are used in evaluation of the proposed local search heuristics, number of skills and staffs are set equally between 3 and 8. Finally, for the problem set with 40 and 50 activities applied in assessment of the algorithm's robustness, number of skills and staffs are considered equally between 5 and 10.

4.2. The LS heuristics efficiency

The experimental results obtained from implementation of the proposed ACO is reported in Table 11. Each problem is solved 3 times without and with the local search heuristics and the average result is recorded as Aver. Sol. In Table 11, the optimal solutions obtained by the GAMS software is reported as GAMS(f^*). Besides, %Gap denotes the deviation percentage of the average results from the optimal solution obtained by GAMS software. It is clear that the developed ACO without local search heuristics has the worst results measured by %Gap, while the algorithm with just ETBLS or PBLS leads to better solutions. However, the ACO algorithm with both the embedded heuristics culminates in solutions with 5.1 average %Gap from the optimal results, which is a satisfying result for such sophisticated optimization problem. It is also noticeable that the ACO algorithm with both the heuristics which has resulted in the best solutions, has the most computational time (666 s) as it would be expected. The last two columns of the table represent the number of variables (var.) and the number on constraints (const.), respectively, which are increasing functions of the number of project's activities.

TABLE 11. Computational results of ACO with and without the local search heuristics.

Problem	#	#	#	ACO without LS			ACO with ETBLS			ACO with PBLS			ACO with ETBLS and PBLS			GAMS	#	#
no	Act.	Skill	Staff	Aver. Sol.	Time (s)	% Gap	Aver. Sol.	Time (s)	% Gap	Aver. Sol.	Time (s)	%	Aver. Sol.	Time (s)	% Gap	(f^*)	var.	const.
1	8	3	3	1365	56	39.6	985	92	0.7	1013	117	3.6	978	416	0.0	978	5617	4532
2		3	3	673	78	34.6	527	135	5.4	519	109	3.8	500	448	0.0	500	4277	4204
3		4	3	1479	95	41.7	1052	157	0.8	1070	144	2.5	1044	467	0.0	1044	5113	4501
4		4	3	934	108	43.3	664	163	1.8	675	154	3.5	652	468	0.0	652	5209	4742
5	10	3	3	1248	119	33.5	957	180	2.4	967	178	3.4	935	507	0.0	935	8281	6074
6		3	3	1346	124	32.7	1037	206	2.3	1041	187	2.7	1014	517	0.0	1014	8911	6339
7		4	4	1914	132	31.3	1480	212	1.5	1502	221	3.0	1458	524	0.0	1458	12 851	8251
8		4	4	1326	132	21.9	1121	220	3.0	1129	254	3.8	1088	542	0.0	1088	9751	7567
9	12	4	4	1337	144	27.6	1118	224	6.7	1145	255	9.3	1048	555	0.0	1048	12 629	8633
10		4	4	1267	145	14.4	1198	257	8.1	1193	262	7.7	1108	572	0.0	1108	18 649	10 245
11		5	4	2241	148	19.1	1901	282	1.0	2015	275	7.1	1882	577	0.0	1882	14 213	9465
12		5	4	2531	151	57.1	2174	297	34.9	1904	279	18.2	1611	580	0.0	1611	13 554	10 431
13	15	4	4	1599	157	59.3	1195	325	19.0	1103	290	9.9	1024	592	2.0	1004	17 506	10 648
14		4	4	3310	175	79.9	2691	329	46.3	2248	332	22.2	1886	627	2.5	1840	20 186	12 416
15		5	5	1658	176	21.4	1503	333	10.0	1469	337	7.5	1392	630	1.9	1366	17 542	10 916
16		5	5	2204	188	27.9	1834	349	6.4	1792	364	4.0	1764	648	2.4	1723	18 850	11 484
17	20	5	5	4596	191	20.0	4085	365	6.7	4084	365	6.6	4010	720	4.7	3830	42 201	20 763
18		5	5	2509	197	29.7	2250	379	16.3	2328	388	20.4	2128	735	10.0	1934	32 171	17 740
19		6	6	4221	209	44.0	3300	386	12.6	3314	391	13.0	3099	750	5.7	2932	51 133	24 397
20		6	6	3102	213	44.8	2472	406	15.4	2309	417	7.8	2237	778	4.4	2142	39 469	20 331
21	25	6	6	4578	238	52.3	3513	419	16.9	3538	436	17.7	3251	737	8.2	3005	57 700	27 290
22		6	6	3719	250	29.0	3447	424	19.5	3295	440	14.3	3092	738	7.2	2884	58 096	25 646
23		6	6	5037	255	61.4	3845	424	23.2	3964	453	27.0	3434	742	10.0	3121	69 766	30 440
24		7	7	6002	285	26.7	5339	442	12.7	5566	467	17.5	5187	795	9.5	4736	92 495	36 490
25		7	7	5425	295	29.3	4915	465	17.1	4977	488	18.6	4719	803	12.4	4197	79 244	32 602
26	30	7	7	8207	302	59.1	6617	460	28.2	6710	500	30.0	5894	837	14.2	5160	112 309	43 690
27		7	7	6062	315	47.8	4914	483	19.8	5097	502	24.3	4761	893	16.1	4102	103 231	40 518
28		8	8	7063	347	16.6	6710	473	10.8	6819	523	12.6	6733	912	11.1	6058	163 571	53 487
29		8	8	6284	367	56.4	5969	478	48.6	5743	548	43.0	4894	935	21.8	4017	94 819	41 959
30		8	8	6938	372	29.8	6177	537	15.6	6423	561	20.2	5876	939	10.0	5344	136 515	47 879
Aver.	–	–	–	–	199	37.7	–	330	13.8	–	341	12.8	–	666	5.1	–	–	–

TABLE 12. Computational results of the proposed ACO for instances with 40 activities.

Problem no	# skill	# staff	Results of ACO					RPD					
			Run 1	Run 2	Run 3	Run 4	Run 5	Run 1	Run 2	Run 3	Run 4	Run 5	Aver. RPD
1	5	5	7694	7483	7781	7487	7536	2.820	0.000	3.982	0.053	0.708	1.513
2	5	5	6698	6600	6631	6581	6622	1.778	0.289	0.760	0.000	0.623	0.690
3	5	5	6618	6764	6875	6914	6824	0.000	2.206	3.883	4.473	3.113	2.735
4	5	5	7961	7710	7617	7678	7782	4.516	1.221	0.000	0.801	2.166	1.741
5	5	5	4766	4897	4884	4771	4873	0.000	2.749	2.476	0.105	2.245	1.515
6	6	6	6381	6322	6384	6514	6381	0.933	0.000	0.981	3.037	0.933	1.177
7	6	6	6342	6302	6586	6385	6503	0.635	0.000	4.507	1.317	3.189	1.930
8	6	6	9479	9216	9402	9264	9282	2.854	0.000	2.018	0.521	0.716	1.222
9	6	6	5663	5641	5589	5737	5677	1.324	0.930	0.000	2.648	1.575	1.295
10	6	6	6720	6827	6692	6636	6632	1.327	2.940	0.905	0.060	0.000	1.046
11	7	7	7042	6951	6899	6935	6922	2.073	0.754	0.000	0.522	0.333	0.736
12	7	7	7367	7349	7250	7346	7218	2.064	1.815	0.443	1.773	0.000	1.219
13	7	7	7232	7190	7171	7225	7248	0.851	0.265	0.000	0.753	1.074	0.588
14	7	7	8401	8472	8345	8217	8276	2.239	3.103	1.558	0.000	0.718	1.524
15	7	7	5977	6157	6104	6234	6217	0.000	3.012	2.125	4.300	4.015	2.690
16	8	8	8117	8206	8067	8354	8119	0.620	1.723	0.000	3.558	0.645	1.309
17	8	8	6354	6532	6616	6535	6616	0.000	2.801	4.123	2.849	4.123	2.779
18	8	8	6388	6173	6179	6163	6428	3.651	0.162	0.260	0.000	4.300	1.675
19	8	8	8750	8627	8674	8749	8785	1.426	0.000	0.545	1.414	1.831	1.043
20	8	8	7753	7676	7710	7677	7685	1.003	0.000	0.443	0.013	0.117	0.315
21	9	9	8555	8910	8765	8708	8502	0.623	4.799	3.093	2.423	0.000	2.188
22	9	9	9400	9211	9396	9397	9462	2.052	0.000	2.008	2.019	2.725	1.761
23	9	9	9997	9887	9947	9925	10 046	1.113	0.000	0.607	0.384	1.608	0.742
24	9	9	7377	7354	7359	7461	7386	0.313	0.000	0.068	1.455	0.435	0.454
25	9	9	10 698	10 860	10 466	10 809	10 986	2.217	3.765	0.000	3.277	4.968	2.845
26	10	10	8769	8776	8672	8841	8816	1.119	1.199	0.000	1.949	1.661	1.185
27	10	10	8720	8569	8591	8607	8870	1.762	0.000	0.257	0.443	3.513	1.195
28	10	10	10 512	10 552	10 614	10 788	10 547	0.000	0.381	0.970	2.626	0.333	0.862
29	10	10	8817	8838	8969	8670	8688	1.696	1.938	3.449	0.000	0.208	1.458
30	10	10	9212	9138	9327	9195	9268	0.810	0.000	2.068	0.624	1.423	0.985

TABLE 13. Computational results of the proposed ACO for instances with 50 activities.

Problem no	# skill	# staff	Results of ACO					RPD					
			Run 1	Run 2	Run 3	Run 4	Run 5	Run 1	Run 2	Run 3	Run 4	Run 5	Aver. RPD
1	5	5	7524	7604	7238	7251	7432	3.951	5.057	0.000	0.180	2.680	2.374
2	5	5	8530	8356	8334	8129	8325	4.933	2.792	2.522	0.000	2.411	2.532
3	5	5	9524	9531	9547	9000	9523	5.822	5.900	6.078	0.000	5.811	4.722
4	5	5	10 118	10 070	10 188	10 164	10 288	0.477	0.000	1.172	0.933	2.165	0.949
5	5	5	18 696	18 650	18 741	18 826	18 413	1.537	1.287	1.781	2.243	0.000	1.370
6	6	6	10 873	10 488	10 756	10 962	10 489	3.671	0.000	2.555	4.519	0.010	2.151
7	6	6	17 338	16 664	17 102	16 757	17 161	4.045	0.000	2.628	0.558	2.982	2.043
8	6	6	10 693	10 263	10 327	10 603	10 450	4.190	0.000	0.624	3.313	1.822	1.990
9	6	6	8915	9033	8853	8534	9033	4.464	5.847	3.738	0.000	5.847	3.979
10	6	6	8264	8398	8162	8182	7885	4.807	6.506	3.513	3.767	0.000	3.718
11	7	7	11 761	11 571	11 725	11 717	11 477	2.475	0.819	2.161	2.091	0.000	1.509
12	7	7	10 830	11 306	10 947	11 188	11 154	0.000	4.395	1.080	3.306	2.992	2.355
13	7	7	10 813	11 011	10 728	10 537	10 874	2.619	4.498	1.813	0.000	3.198	2.426
14	7	7	9595	9317	9342	9308	9303	3.139	0.150	0.419	0.054	0.000	0.752
15	7	7	7160	7107	7248	7465	7234	0.746	0.000	1.984	5.037	1.787	1.911
16	8	8	10 740	10 989	10 895	10 467	10 763	2.608	4.987	4.089	0.000	2.828	2.902
17	8	8	9711	10 161	10 240	10 196	10 079	0.000	4.634	5.447	4.994	3.790	3.773
18	8	8	10628	10452	10586	10576	10198	4.217	2.491	3.805	3.707	0.000	2.844
19	8	8	11 111	11 440	11 171	11 214	11 327	0.000	2.961	0.540	0.927	1.944	1.274
20	8	8	9787	10 185	10 358	10 355	10 435	0.000	4.067	5.834	5.804	6.621	4.465
21	9	9	13 150	13 191	13 283	13 158	13 354	0.000	0.312	1.011	0.061	1.551	0.587
22	9	9	11 914	11 359	11 513	11 433	11 578	4.886	0.000	1.356	0.651	1.928	1.764
23	9	9	10 537	10 529	10 451	10 456	10 547	0.823	0.746	0.000	0.048	0.919	0.507
24	9	9	11 438	11 668	11 960	11 387	11 976	0.448	2.468	5.032	0.000	5.173	2.624
25	9	9	11 108	11 165	10 963	11 136	11 158	1.323	1.843	0.000	1.578	1.779	1.304
26	10	10	13 566	13 228	13 341	13 511	13 117	3.423	0.846	1.708	3.004	0.000	1.796
27	10	10	13 573	13 791	13 624	13 641	13 692	0.000	1.606	0.376	0.501	0.877	0.672
28	10	10	12 435	12 397	12 566	12 148	12 376	2.363	2.050	3.441	0.000	1.877	1.946
29	10	10	11 126	11 169	11 147	11 211	11 353	0.000	0.386	0.189	0.764	2.040	0.676
30	10	10	11 785	12 056	12 024	11 815	11 859	0.000	2.300	2.028	0.255	0.628	1.042

4.3. The algorithms robustness

A metaheuristic algorithm would be reliable to apply in real-life problems if its results in different runs of on a given instance be as closer together as possible. To test this attribute of the developed ACO, a set of 60 problem instances are considered. Each instance is solved 5 times and their relative percent difference (RPD) in different runs are computed as follows:

$$\mathrm{RPD}(i) = \left[\frac{\mathrm{Solution}_i - \mathrm{BestSolution}}{\mathrm{BestSolution}}\right] \times 100. \tag{4.1}$$

The test set includes 30 instances with 40 activities and 30 instances with 50 activities. The number of skills and the number of staffs are considered between 5 and 10.

Herein, the ACO algorithm with both of the heuristics which has resulted in the best solutions is applied for robustness analysis. The details of the results obtained in different runs are reported in Tables 12 and 13.

It can be observed from Table 12 that the maximum values of average RPD is 4.968%. Also, Table 13 shows that the maximum values of average RPD is 6.621%. The last column in Tables 12 and 13 represents the average values of RPD which is an index of the algorithm's robustness. It can be noticed that the average value of RPD for the instances with 40 activities is 2.845%, while this index for the instances with 50 activities is 4.722. Another noticeable result is that the maximum and the average values of RPD increases with the size of the instances, measured by the number of activities. To sum up, the results reveal that all the values of RPD for all the experiments is less than 7% which is a satisfying value in support of the robustness of the proposed ACO algorithm.

5. Conclusion

This paper deals with a multi-skilled resource constrained project scheduling problem which in activity preemption is permitted with penalty. Execution of any skill of an activity has staff-dependent processing cost. Also, completion time of activities is subjected to hard interval due dates that should be strictly preserved, and soft interval due dates that may by violated with earliness or tardiness penalty. This newly defined problem is called PMSPSP. Two integer programming formulation is proposed for the problem with an objective function to minimize total processing costs, earliness–tardiness and preemption penalties. Based on 30 small scale randomly generated test problems, the developed formulations are compared on GAMS commercial solver. It is observed that while the first formulation has more number of variables and constraints on average, but its average run-time to optimality is considerably lower than the second formulation. To tackle the large scale instances of PMSPSP, a solution procedure based on ACO with calibrated parameters is proposed. To cope with the complicated structure of the solution space, a preemption based (PBLS) and an earliness–tardiness based (ETBLS) local search heuristic is contrived at end of each iteration of the algorithm. Experimental results on 30 test problems reveal that the designed heuristics are intelligent in the exploration of the search space in such a way that the average gap between the algorithm's results and the optimal results is 5.1%. In addition, the maximum value of 6.621% for the average relative percent deviation (RPD) for test problems with up to 50 activities confirms that the proposed algorithm gives robust solutions. From these analysis, it can be concluded that while exact methods and commercial solvers fail to obtain optimal solution for real-size instances of the problem, the proposed methodology can help decision makers to provide a satisfying schedule for PMSPSP within a reasonable computational time.

The work of this paper can be extended in several ways. One potential extension is to consider this model with different levels for the skills with different quality, duration and processing cost. Another extension would be considering inter-related sequence between different skills of an activity. Furthermore, it is suggested to interested reader to extend PMSPSP by considering other real life assumptions like setup time after preemptions, rework and multiple objectives.

References

[1] B. Afshar-Nadjafi and M. Majlesi, Resource constrained project scheduling problem with setup times after preemptive processes. *Comput. Chem. Eng.* **69** (2014) 16–25.

[2] B. Afshar-Nadjafi and S. Shadrokh, An algorithm for the weighted earliness-tardiness unconstrained project scheduling problem. *J. Appl. Sci.* **8** (2008) 1651–1659.

[3] F.S. Al-Anzi, K. Al-Zame and A. Allahverdi, Weighted multi-skill resources project scheduling. *J. Softw. Eng. Appl.* **3** (2010) 1125–1130.

[4] C. Artigues, R. Leus and F.T. Nobibon, Robust optimization for resource-constrained project scheduling with uncertain activity durations. *Flex. Serv. Manuf. J.* **25** (2013) 175–205.

[5] F. Ballestín, A. Barrios and V. Valls, An evolutionary algorithm for the resource-constrained project scheduling problem with minimum and maximum time lags. *J. Sched.* **14** (2011) 391–406.

[6] F. Ballestín and V. Valls and S. Quintanilla, Preemption in resource-constrained project scheduling. *Eur. J. Oper. Res.* **189** (2008) 1136–1152.

[7] O. Bellenguez-Morineau, Methods to solve the multi-skill project scheduling problem. *4OR* **6** (2008) 85–88.

[8] O. Bellenguez-Morineau and E. Neron, Lower Bounds for the multi-skill project scheduling problem with hierarchical levels of skills. In practice and theory of automated timetabling. *Lectures Notes Comput. Sci.* **3616** (2005) 229–243.

[9] O. Bellenguez-Morineau and E. Neron, A branch-and-bound method for solving multi-skill project scheduling problems. *RAIRO: OR* **41** (2007) 155–170.

[10] D.G. Cabrero and D.N. Ranasinghe, Fine-tuning the Ant Colony System Algorithm Through Particle Swarm Optimization. Technical Report-University of Valencia, Spain, (2005).

[11] W.N. Chen and J. Zhang, Scheduling multi-mode projects under uncertainty to optimize cash flows: a Monte Carlo ant colony system approach. *J. Comput. Sci. Technol.* **27** (2012) 950–965.

[12] W.N. Chen, J. Zhang, H.S.H. Chung, R.Z. Huang and O. Liu, Optimizing discounted cash flows in project scheduling – an ant colony optimization approach. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **40** (2010) 64–77.

[13] H. Cheng and X. Chu, Task assignment with multi-skilled employees and multiple modes for product development projects. *Int. J. Adv. Manuf. Technol.* **61** (2012) 391–403.

[14] C.W. Chiang, Y.Q. Huang and W.Y. Wang, Ant colony optimization with parameter adaptation for multi-mode resource-constrained project scheduling. *J. Intell. Fuzzy Syst.* **19** (2008) 345–358.

[15] S. Christodoulou, Scheduling resource-constrained projects with ant colony optimization artificial agents. *J. Comput. Civ. Eng.* **24** (2009) 45–55.

[16] I. Correia, L. Lourenco and F. Saldanha-da-Gama, Project scheduling with flexible resources: formulation and inequalities. *OR Spectr.* **34** (2012) 635–663.

[17] I. Correia and F. Saldanha-da-Gama, The impact of fixed and variable costs in a multi-skill project scheduling problem: An empirical study. *Comput. Ind. Eng.* **72** (2014) 230–238.

[18] C. Dhib, A. Soukhal and E. Neron, Mixed-integer linear programming formulation and priority-rule methods for a preemptive project staffing and scheduling problem, edited by C. Schwindt and J. Zimmermann. In: Handbook on Project Management and Scheduling. Springer (2015) 603–617.

[19] M. Dorigo, *Optimization, learning and natural algorithms*. Ph.D. thesis, Politecnico di Milano, Italy (1992).

[20] A. Drexl, R. Nissen, J.H. Patterson and F. Salewski, ProGen/πx – An instance generator for resource constrained project scheduling problems with partially renewable resources and further extensions. *Eur. J. Oper. Res.* **125** (2000) 59–72.

[21] M. Fırat and C.A.J. Hurkens, An improved MIP-based approach for a multi-skill workforce scheduling problem. *J. Sched.* **15** (2012) 363–380.

[22] C. Heimerl and R. Kolisch, Scheduling and staffing multiple projects with a multi-skilled workforce. *OR Spectr.* **32** (2010) 343–368.

[23] C.A. Hurkens, Incorporating the strength of MIP modeling in schedule construction. *RAIRO: OR* **43** (2009) 409–420.

[24] Y. Kadrou and N.M. Najid, A new heuristic to solve RCPSP with multiple execution modes and multi-skilled labor. *Comput. Eng. Syst. Appl. IMACS Multi Conf.* **2** (2006) 1302–1309.

[25] H. Kazemipoor, R. Tavakkoli-Moghaddam, P. Shahnazari-Shahrezaei and A. Azaron, A differential evolution algorithm to solve multi-skilled project portfolio scheduling problems. *Int. J. Adv. Manuf. Technol.* **64** (2013) 1099–1111.

[26] C. Kellenbrink and S. Helber, Scheduling resource-constrained projects with a flexible project structure. *Eur. J. Oper. Res.* **246** (2015) 379–391.

[27] Y. Khoshjahan, A.A. Najafi and B. Afshar-Nadjafi, Resource constrained project scheduling problem with discounted earliness–tardiness penalties: mathematical modeling and solving procedure. *Comput. Ind. Eng.* **66** (2013) 293–300.

[28] O. Koné, C. Artigues, P. Lopez and M. Mongeau, Comparison of mixed integer linear programming models for the resource-constrained project scheduling problem with consumption and production of resources. *Flex. Serv. Manuf. J.* **25** (2013) 25–47.

[29] H. Li and K. Womer, Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm. *J. Sched.* **12** (2009) 281–298.

[30] H. Li and H. Zhang, Ant colony optimization-based multi-mode scheduling under renewable and nonrenewable resource constraints. *Autom. Constr.* **35** (2013) 431–438.

[31] S.S. Liu and C.J. Wang, Optimizing linear project scheduling with multi-skilled crews. *Autom. Constr.* **24** (2012) 16–23.

[32] H. Maghsoudlou, B. Afshar-Nadjafi and S.T.A. Niaki, A multi-objective invasive weeds optimization algorithm for solving multi-skill multi-mode resource constrained project scheduling problem. *Comput. Chem. Eng.* **88** (2016) 157–169.

[33] D. Merkle, M. Middendorf and H. Schmeck, Ant colony optimization for resource-constrained project scheduling. *IEEE Trans. Evol. Comput.* **6** (2002) 333–346.

[34] C. Montoya, O. Bellenguez-Morineau, E. Pinson and D. Rivreau, Branch-and-price approach for the multi-skill project scheduling problem. *Optim. Lett.* **8** (2013) 1721–1734.

[35] A. Moukrim, A. Quilliot and H. Toussaint, An effective branch-and-price algorithm for the preemptive resource constrained project scheduling problem based on minimal Interval order enumeration. *Eur. J. Oper. Res.* **244** (2015) 360–368.

[36] E. Néron and D. Baptista, Heuristics for multi-skill project scheduling problem. *Int. Symp. Comb. Optim.* (CO'2002) (2002).

[37] H. Okubo, T. Miyamoto, S. Yoshida, K. Mori, S. Kitamura and Y. Izui, Project scheduling under partially renewable resources and resource consumption during setup operations. *Comput. Ind. Eng.* **83** (2015) 91–99.

[38] M. Ranjbar, M. Khalilzadeh, F. Kianfar and K. Etminani, An optimal procedure for minimizing total weighted resource tardiness penalty costs in the resource-constrained project scheduling problem. *Comput. Ind. Eng.* **62** (2012) 264–270.

[39] H. Rolfe, Qualifications and international mobility: a case study of the European chemicals industry. *Nat. Inst. Econ. Rev.* **175** (2001) 85–94.

[40] G. Taguchi, Introduction to Quality Engineering. Asian Productivity Organization, Tokyo (1986).