SOLVING A NOVEL MULTI-SKILLED PROJECT SCHEDULING MODEL BY SCATTER SEARCH

H. Kazemipoor[1*], R. Tavakkoli-Moghaddam[2] & P. Shahnazari-Shahrezaei[3]

[1] Department of Industrial Engineering,
Parand Branch, Islamic Azad University, Tehran, Iran
h.kazemipoor@piau.ac.ir

[2] Department of Industrial Engineering,
University of Tehran, Tehran, Iran
tavakoli@ut.ac.ir

[3] Department of Industrial Engineering,
Firoozkooh Branch, Islamic Azad University,Firoozkooh, Iran
parisa_shahnazari@iaufb.ac.ir

ABSTRACT

A multi-skilled project scheduling problem (MSPSP) has generally been presented to schedule information technology projects in deterministic conditions. The contribution of this model is to consider the resources, called 'staff members'. These members are regarded as valuable, renewable, and discrete resources with different multiple skills. The different skills of staff members, as well as the project network's activity requirement of different skills, cause this problem to become a special type of multi-mode resource-constrained project scheduling problem (MM-RCPSP), with a huge number of modes. Taking into account the importance of this issue and the few studies performed on this problem, a novel mathematical model for the MSPSP is presented. Since the complexity of this problem is NP-hard, an efficient scatter search (SS) algorithm is developed to solve such a difficult problem. This proposed SS is capable of generating optimised solutions in small sizes, and the excellent solutions in large sizes are compared with the solutions reported by a proposed Tabu search (TS) algorithm.

OPSOMMING

'n Veeldoel projekskeduleringsvraagstuk (MSPSP) word voorgehou vir skedulering onder deterministiese toestande van inligtingstegnologieprojekte. Die model word aangewend vir hulpbronne genaamd "personeellede". Die bronne is waardevol, hernubaar, diskreet en beskik oor uiteenlopende vaardighede. Die eienskappe van die vraagstuk berus by uiteenlopendheid. Vir die voorafgaande omstandighede word 'n nuwe model geskep vir die hantering van eienaardighede. Aangesien die problem NP-moeilik is, word 'n doeltreffende spreisoek-algoritme gebruik met groot sukses.

* Corresponding author

1. INTRODUCTION

Blazewicz et al. [1] first considered a resource-constrained project scheduling problem (RCPSP) that belongs to the class of NP-hard optimisation problems. The RCPSP is one of the most complicated operations research problems. The way to solve it has been improved considerably through exact and heuristic methods in recent decades, and new optimisation techniques have been employed to solve the problem [2-4]. Recently, a multi-skilled project scheduling problem (MSPSP), which is a fairly new version of a project scheduling problem, has been presented by Neron and Boptista [5]. As a general rule, this problem is an extended model of the multi-mode RCPSP [6]. The main differences between the MSPSP and the MM-RCPSP are in the resources being considered and the type of requirement for each resource. In the MSPSP, the resources are staff members with different skills assigned to the activities of the project. A staff member can be assigned at most to one requirement for a specific time to perform an activity using one required skill. Different combinations of use of these staff members are determined during the project assignment on the basis of the skills required for each activity and the skills of the staff members. To accomplish each activity in any skill, the proper resources are selected from among a set of qualified resources. In other words, if all staff members in the whole project had a single skill, the MSPSP would be the same as a classic RCPSP. According to the expressions mentioned above, and to clarify the MSPSP, a problem with a feasible solution is illustrated in Figures 1(a), 1(b) and 1(c).

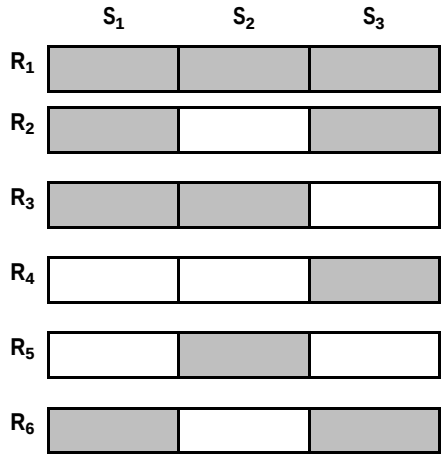

Figure 1a: Staff members' skills for the MSPSP instance[1]

[1] There are 6 staff members in the project according to Figure 1.a. It should be noted that the coloured [copyeditor comments: no colours can be discerned; I suggest that different shades of grey be used] rectangles for each resource (R_M) denote the expertise of the M^{th} resource at the k^{th} skill level, while the white rectangles denote a lack of expertise of the M^{th} resource at the k^{th} skill level.

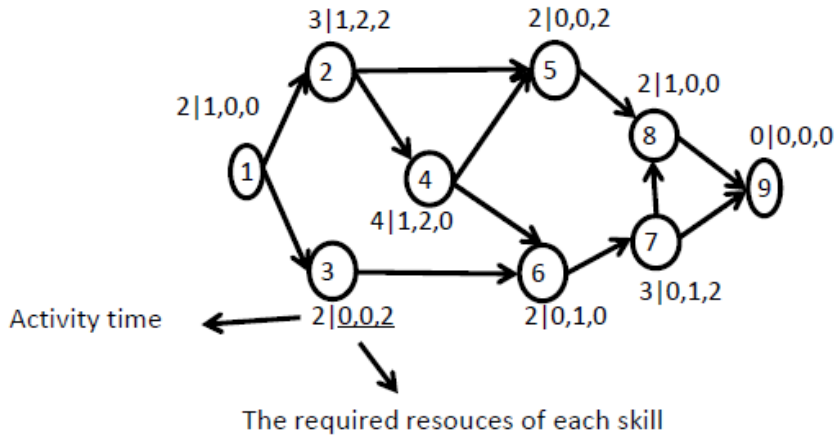

Figure 1b: Precedence network and activities requirement for the MSPSP instance

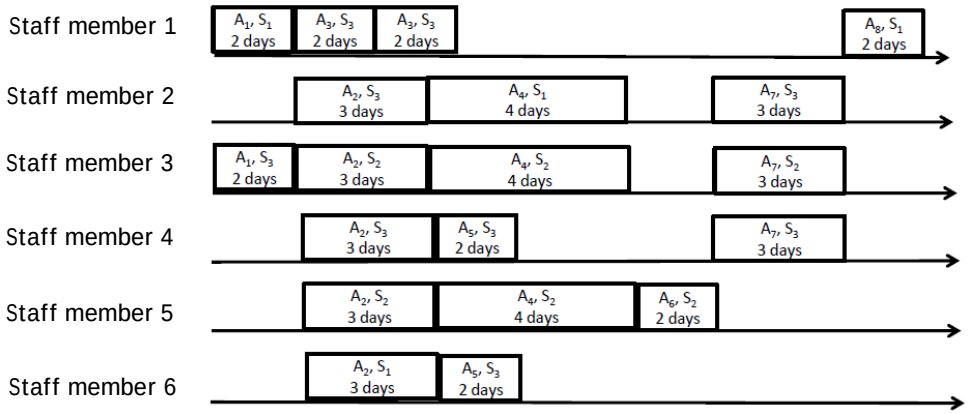

Figure 1c: Staff members' assignment and scheduling for the MSPSP instance

The fact that resources can be used to satisfy different requirements has been presented in the definition of job-shop scheduling. As a result, the MSPSP can be regarded as a combination of a classic RCPSP model and multi-process machines (MPM) [7,8]. Taking the related definitions with the given problem into account, and referring to Figure 1, the MSPSP can be classified in the category of the multi-mode RCPSP (MM-RCPSP) [5] with very large modes. In this case, the execution modes of an activity consist of a subset of qualified resources that can be assigned to that activity. Infinite execution modes of this problem originate from the different skill of each resource. As shown in the example in Figure 1, activity 2 can be executed by resources (i.e., R_1, R_2, R_3, R_6) in skill 1, by resources (i.e., R_1, R_3, R_5) in skill 2, and by resources (i.e., R_1, R_2, R_4, R_6) in skill 3. So activity 2 can be executed in 48 different modes. If the entire network is investigated, the number of execution modes for activities will be huge. Hence, the MSPSP is a type of the MM-RCPSP with huge modes to execute each activity. This means that the MSPSP cannot be solved even in medium sizes by the presented solution methods for the MM-RCPSP, which are often based on the enumeration [5]. Taking these facts into consideration, new techniques have been proposed to solve this type of problem [9]. In general, three classes of solution methods have been developed to solve these problems: lower bound, branch-and-bound, and heuristic and meta-heuristic methods.

Bellenguez-Morineau [10] summarised some proposed methods. Another example in this direction is the work carried out by Kadro and Najid [11]. They extended a solution method

by Tabu search, accompanied by strong local search, for a manpower constrained MM-RCPSP class. In order to exhibit the strength of the algorithm, they also found a lower bound for the problem and compared the generated solutions with this bound. Another example of research on solving a manpower scheduling problem in projects is that done by Valls et al. [12]. They recommended a hybrid genetic algorithm (GA) to solve a skilled manpower scheduling problem in computer centres. Heimerl and Kolisch [13] proposed a mixed-integer linear programming (MILP) model with hard constraints in order to minimise the manpower cost, considering the in-source and out-source manpower, for an information technology projects portfolio. Gutjahr et al. [14] took a project portfolio into account and introduced a non-linear MIP model on the basis of maximising the average economic income for selecting the projects, optimising the time, and assigning the persons to selected projects. They presented a heuristic process that included two components of a greedy heuristic algorithm to schedule and assign the persons, and a meta-heuristic algorithm to choose the project. Then they compared the obtained solutions with the bounds acquired by the exact solution of the simplified mathematical model, and proved the efficiency of their method. The MSPSP can have various applications, such as manpower scheduling in preventive maintenance. Since preventive maintenance is performed with human skills in the form of a project, Pessan [15] formulated it as an MSPSP. A significant point in these problems is the objective function. In most studies done in this area, the objective is to provide a schedule in order to minimise the project execution time [5,11]. Some researchers have also taken account of personnel costs in their model. Li and Womer [30] defined the minimum cost for staff members as an objective function. Heimerl and Kolisch [13] and Gutjahr et al. [14] entered these costs in the MSPSP in special cases as well.

This paper has been organised as follows. Section 2 deals with a novel mathematical model, the simplified model, the parameters used, and the decision variables. An efficient scatter search algorithm to solve the problem, and an elite Tabu search algorithm to compare the results, are presented in Section 3. Section 4 is allotted to computational results. Section 5 presents concluding remarks.

2. PRESENTING A NOVEL MATHEMATICAL MODEL FOR MSPSP

In the mathematical model for the MM-RCPSPs, two types of formulations are applied: sequence-based models, and time index formulation [9]. In this paper, the formulation is done with sequence-based models for the MSPSP. In this section, indices, parameters, and decision variables are presented first, and then the objective function and constraints of the mathematical model for developing the MSPSP are considered. Since this model is inspired by a real situation, the initial model is a mixed-nonlinear one that is converted to a linear model at the end of this section. It should be noted the network in question is defined as 'activity on node' (AON).

2.1 Indices

i, i' : Activities counters (1,…,n)
k, k' : Skills counters (1,…,K)
m, m' : Resources counters (1,…,M)

2.2 Parameters and sets

$A = \{A_1, ..., A_n\}$: Non-preemptive executive activities set
$S = \{S_1, ..., S_K\}$: Set of the existing skills for running the project
$R = \{R_1, ..., R_M\}$: Resources set (staff members)
G : Precedence relations graph for the project
p_i : Time of executing the activity A_i
b_{ik} : Number of required resources for doing skill S_k during the execution of activity A_i

$$r_{mk} = \begin{cases} 1 & \text{If person } mth \text{ has skill } kth \\ 0 & \text{Otherwise} \end{cases}$$

2.3 Decision variables

Two decision variables in this model are defined as follows:

$$x_{ik}^{m} = \text{Start time of activity } ith \text{ in skill } kth \text{ by person } mth$$

$$y_{ik}^{m} = \begin{cases} 1 & \text{If person } mth \text{ done activity } ith \text{ at Skill } kth \\ 0 & \text{Otherwise} \end{cases}$$

$$f_{ii'kk'}^{m} = \begin{cases} 0 & x_{ik}^{m} \geq x_{i'k'}^{m} \\ 1 & x_{ik}^{m} \leq x_{i'k'}^{m} \end{cases}$$

2.4 Novel linear mathematical model

Regarding the above-mentioned cases, a mathematical model for the MSPSP that is inspired by a real situation is presented below.

Model (I)

$$\min \{ \max_{\forall k,m}(x_{nk}^{m}) \} \quad (1)$$

s.t.

$$x_{ik}^{m} - x_{i'k'}^{m} \geq y_{ik}^{m} * y_{i'k'}^{m} * p_{i'} - \mathrm{M} f_{ii'kk'}^{m}; \forall (i,i',k,m), i \neq i' \quad (2)$$

$$x_{i'k'}^{m} - x_{ik}^{m} \geq y_{ik}^{m} * y_{i'k'}^{m} * p_{i} - \mathrm{M}(1 - f_{ii'kk'}^{m}); \forall (i,i',k,m), i \neq i' \quad (3)$$

$$\sum_{\mathrm{m}=1}^{M} r_{mk} y_{ik}^{m} = b_{ik}; \forall (i,k) \quad (4)$$

$$\min_{\forall k',m'}(x_{i'k'}^{m'} | y_{i'k'}^{m'} = 1) - \max_{\forall k,m}(x_{ik}^{\mathrm{m}} \mid y_{ik}^{\mathrm{m}} = 1) \geq p_i; \forall (i,i') \in \text{ Precedence relations} \quad (5)$$

$$x_{ik}^{m} \leq \mathrm{M} * y_{ik}^{m}; \forall (i,k,m) \quad (6)$$

$$x_{ik}^{m} \geq 0, \text{Integer}; \forall (i,k,m) \quad (7)$$

$$y_{ik}^{m} = \{0,1\}; \forall (i,k,m) \quad (8)$$

In the MSPSP, the objective function can be considered as the minimisation of the greatest start time of the last activity for different skills. Therefore, the finish time of the last activity will be minimised. Equation (1) describes this. One of the fundamental constraints in the MSPSP is the non-preemptive allocation of resources to the activities' skills. This constraint is expressed so that if a resource allocates to an activity at a special skill, this resource cannot be employed for another activity until it has finished the current activity. Eqs. (2)-(3) guarantee this constraint. It should be noted that M in Eqs. (2) and (3) shows big-M in big-M method [16]. The required amount of each activity at any skill is equal to b_{ik} in terms of a resource per day that should be guaranteed for each activity at any skill until finishing the activity time at different skills. Eq. (4) accompanied by Eqs. (2) and (3) provides this constraint. Execution of each project activity involves taking account of the precedence network. In a precedence network, the succeeding activity can be started when all preceding activities of this activity have been finished at all skills. Eq. (5) takes this matter into account. It should be mentioned that the first and last activity in the precedence network are generally the starting and ending activities, which are sometimes defined as virtual activities. An activity in each skill can be started when enough resources are allocated to it. Eq. (6) expresses this constraint. In this equation, M is also big-M. Finally, Eqs. (7) and (8) explain the type of decision variables in the mathematical model. As seen, Model (I) is nonlinear. Using the property below from the operations research science [16], Model (I) can be converted to a linear model.

Property 1:

The objective function of Model (I) can take the linear form as follows:

$$\min\ ST_n \tag{1-1}$$

s.t.

$$ST_n \geq x_{nk}^m; \forall m,k \tag{10}$$

Proof:

In this part, the standard linearisation methods of mathematical models [17] are applied. In fact, if $\max(x_{nk}^m; \forall m,k) = ST_n$ is defined, we have $ST_n \geq x_{nk}^m; \forall m,k$. In this manner, the objective function will become linear.

Property 2:

Eq. (5) is rewritten as follows:

$$\underset{x_{jk'}^{m'}>0}{x_{i'k'}^{m'}} \geq x_{ik}^{\mathrm{m}} + p_i; \forall (i,i') \in \text{Precedence Relations}, \forall (k,m',k,m) \tag{1-5}$$

Proof:

It is obvious that: $\max(x_{ik}^{\mathrm{m}}) = \max(x_{ik}^{\mathrm{m}} \mid y_{ik}^{\mathrm{m}} = 1, y_{ik}^{\mathrm{m}} = 0) = \max(x_{ik}^{\mathrm{m}} \mid y_{ik}^{\mathrm{m}} = 1, x_{ik}^{\mathrm{m}} \mid y_{ik}^{\mathrm{m}} = 0)$

By considering Eq. (6), if $y_{ik}^{\mathrm{m}} = 0$, we have $x_{ik}^{\mathrm{m}} = 0$. As a result, $\max(x_{ik}^{\mathrm{m}} \mid y_{ik}^{\mathrm{m}} = 0) = 0$. Thus:

$$\max(x_{ik}^{\mathrm{m}}) = \max(x_{ik}^{\mathrm{m}} \mid y_{ik}^{\mathrm{m}} = 1)$$

Consequently, we have:

$$\min_{\forall k',m'}(x_{i'k'}^{m'} | y_{i'k'}^{m'} = 1) - \max_{\forall k,m}(x_{ik}^{\mathrm{m}} \mid y_{ik}^{\mathrm{m}} = 1) \geq p_i \equiv \min_{\forall k',m'}(x_{i'k'}^{m'} | x_{jk'}^{m'} > 0) - \max_{\forall k,m}(x_{ik}^{\mathrm{m}}) \geq p_i$$

$$\Rightarrow \min_{\forall k',m'}(x_{i'k'}^{m'} | x_{i'k'}^{m'} > 0) \geq \max_{\forall k,m}(x_{ik}^{\mathrm{m}}) + p_i$$

Now, the abovementioned equation can be easily rewritten as:

$$\min_{\forall j,k',m'}(x_{i'k'}^{m'} | x_{i'k'}^{m'} > 0) \geq x_{ik}^{\mathrm{m}} + p_i; \forall (i,j) \in \text{ Precedence relations}, \forall i,k,m$$

Regarding the above equation, the minimum of $(x_{i'k'}^{m'} \mid x_{i'k'}^{m'} > 0)$ should be greater than $x_{ik}^{\mathrm{m}} + p_i$. Hence, the entire amounts of $(x_{i'k'}^{m'} \mid x_{i'k'}^{m'} > 0)$ will be greater than $x_{ik}^{\mathrm{m}} + p_i$, and:

$$\underset{x_{i'k'}^{m'}>0}{x_{i'k'}^{m'}} \geq x_{ik}^{\mathrm{m}} + p_i; \forall (i,i') \in \text{ Precedence relations}, \forall (k,m',k,m)$$

Property 3:

Let $z_{iki'k'}^m = y_{ik}^m y_{i'k'}^m$. Obviously, $z_{iki'k'}^m$ is a new 0-1 variable which is dependent on y_{ik}^m and $y_{i'k'}^m$. It can be proved that Eqs. (2) and (3) are converted to linear equations by changing and adding Eqs. (10)-(12):

$$x_{ik}^m - x_{i'k'}^m \geq z_{ii'kk'}^m p_{i'} - \mathrm{M} f_{ii'kk'}^m; \forall (i,i',k,m), i \neq i' \tag{2-1}$$

$$x_{i'k'}^m - x_{ik}^m \geq z_{iki'k'}^m p_i - \mathrm{M}(1 - f_{iki'k'}^m); \forall (i,k,m), i \neq i' \tag{3-1}$$

$$y_{ik}^m + y_{i'k'}^m - 2z_{iki'k'}^m \geq 0 \tag{10}$$

$$z^{m}_{iki'k'} \geq y^{m}_{ik} + y^{m}_{i'k'} - 1; \forall i,k,i',k',m \quad (11)$$

$$z^{m}_{iki'k'} = \{0,1\}; \forall i,k,i',k',m \quad (12)$$

Proof:

Assuming that Eqs. (2) and (3) are equal to Eqs. (2-1) and (3-1), $z^{m}_{iki'k'}$ should be equal to 0, when y^{m}_{ik} and $y^{m}_{i'k'}$ are 0 or alternatively are 0 and 1; else 1. It is clear that, when y^{m}_{ik} and $y^{m}_{i'k'}$ are 0 or alternatively are 0 and 1, $z^{m}_{iki'k'}$ will get 0 in respect with Eqs. (10) and (12), and Eq. (11) will be redundant. In other words, if y^{m}_{ik} and $y^{m}_{i'k'}$ are equal to 1, $z^{m}_{iki'k'}$ will get 1 with regard to Eqs. (9) to (11). Something like this proof, with minor changes, has been propounded in the quadratic assignment problem (QAP) [18].

Thus the NLMIP (i.e., Model (I)) is converted to the LMIP (i.e., Model (II)) including Eqs. (1-1), (2-1), (3-1), (4), (5-1), and (6) to (13). Although this simplification makes the model more efficient, Model (II) is also placed on the class of NP-Hard problems and cannot be solved in the real sizes at a limited time. As a result, the scatter search and Tabu search algorithms are used to solve the model in large sizes. In the next part of this paper, the details of the proposed methods are presented.

3. TWO META-HEURISTICS

3.1 Proposed scatter search

Scatter search (SS), which is one of the evolution methods to solve NP-hard problems, was first presented by Glover [19] to solve integer programming problems. Afterwards, Glover [20-22] examined different aspects of the scatter search algorithm and compared it with the genetic algorithm. In the original proposal, Glover explained scatter search as a method that uses a succession of coordinated initialisations to generate solutions. He introduced the reference set (RefSet) of solutions and several guidelines – including that the search occurs in a systematic way, as opposed to the random designs of other methods (such as GAs). The approach was conceived to begin by identifying a convex combination of the reference points. This central point, together with subsets of the initial reference points, was then used to define new sub-regions. Consequently, analogous central points of the sub-regions were examined in a logical sequence. At the end, these latter points were rounded (in a broad sense, depending on the solution representation) to acquire the desired solutions.

The general pattern of this method is used for the majority of the SS executions. There are five general stages in the main pattern of SS [23,24]: generation of various solutions, improving the generated solutions, generation of a reference set, the method of choosing subsets from the reference set, and the subsets' solutions combination method. In this paper, the proposed SS is designed on the basis of the five above-mentioned stages to solve the MSPSP in accordance with Figure 2. Then the details of Figure2 are described as follows.

Initialise *N* feasible solutions (population size) by a serial scheduling method.

Out loop:

Apply the diversification method for the current solution.

In loop:

- Apply improvement method for each current solution.
- Apply the reference set method for construction of reference set1 & reference set2.
- Apply the subset generation method for construction of binary subset of reference set1 & reference set2.
- Apply combination method on created binary subsets for generation of a new solution.
- Choose *N* solutions for the next iteration.

End in loop.

End out loop.

Figure 2: Structure of the SS algorithm

3.1.1 Solution representation method

In this paper, two structures are used to represent the solutions obtained by the proposed algorithms. The first structure, which is used to represent the order in which the activities are completed, is a one-dimensional matrix. The second structure is a three-dimensional matrix that indicates the allocation and start time of the required skills of all project activities in the MSPSP. In this matrix, the number of rows is equal to the number of all the required skills for the project; each column expresses the assigned resource's number to each skill and its start time; and the third dimension of the matrix represents each activity's number. The first and second structures are shown in Figures3 and 4, based on Figures 1a, 1b, and 1c.

Sequence of activities	1	2	3	5	4	6	7	8	9

Figure 3: A problem instance to represent the solutions in terms of the first structure according to Figures 1a, 1b, and 1c.

	Staff number	**Time**	**Staff number**	**Time**
S_1	2	5	0	0
S_2	3	5	5	5
S_3	0	0	0	0

Figure 4: A problem instance to represent the solutions in terms of the second structure for the fourth activity (for example) of Figures. 1a, 1b, and 1c.

In fact, Figure 3 is a representation method for a feasible sequence of the completion of the project activities in Figure 1.b, which has been previously shown in Figure 1.c. In addition, Figure4 illustrates the representation method of allocation and the start time of the required skills of the fourth activity in Figure 1.b, which can also be seen in Figure 1.c. In other words, in accordance with Figure 4, the second resource (staff member with number 2) has been assigned to the first skill of the fourth activity at time 5, and the third and fifth resources (staff members with numbers 3 and 5) have been assigned to the second skill of the fourth activity at time 5. As Figure 1.b shows, this activity does not require the third skill to be completed.

3.1.2 Generation of initial solutions

In this research, the initial solutions were firstly generated by stochastic procedures. After evaluating the quality of the generated initial solutions, it was specified that they are not acceptable for use. Hence, a bi-directional scheduling generation scheme [25] is used to generate initial solutions. In this method, a forward and backward partial scheduling is regarded simultaneously; and at any iteration, an activity can be scheduled when it is just schedulable in one direction. In this process, three sets of activities are defined: 1) the set

of activities whose precedent activities have been scheduled (ready for forward scheduling); 2) the set of activities whose succeeding activities have been scheduled (ready for backward scheduling); and 3) the set of activities whose precedent *and* succeeding activities have been scheduled (ready for forward and backward scheduling). At any iteration, the activities that are forward schedulable are transferred to set1, the activities that are backward schedulable are transferred to set 2, and the activities that are both forward and backward schedulable are transferred to set3 and removed from sets 1 and 2. After defining the sets, an activity is selected randomly from set1, and in one of the random possible modes is scheduled at the earliest time and removed from the related set; and then all sets are updated. In set 2 too, an activity is selected randomly and a random possible mode is assigned to it; and it is scheduled at the latest time on the condition that it does not violate the precedence activities and resources. This activity is removed from set 2, and then all sets are updated. This process is continued until no activity is schedulable in one direction. In this case, just the activities of set 3 have not been scheduled. The activities of set 3 are also selected randomly, and are scheduled in random directions byassigning random possible modes to them. Afterwards set 3 is updated. This process is continued until all activities of set 3 are assigned. At present, two lists of forward and backward scheduling are provided. Thereafter, to determine the final scheduling list, the activities of set 2 are added to set 1,in the reverse order of selecting them.

3.1.3 Diversification method for the current solution

In this paper, each existing solution is entered into the diversification method, and is scattered by the multi-start Tabu search method as much as is possible, according to Figure 5.

0. Counter p is set to 1 (p^{th} solution is A).
1. Counter q is set to 1.
2. Counter r is set to 1.
3. Move procedure is applied to p^{th} solution and the solution B is obtained.

3.1. If the solution B is not better and the move is not found in the Tabu list too, the solution is replaced with the probability of 5%.

4. Add one unit to counter r. If the counter is less than max-iteration, Go to 3; else Go to 5.
5. Add one unit to counter. If the counter is less than max-iteration, Go to 2.
6. Add one unit to counter p. If the counter is less than the population size (N), Go to 1.

Figure 5: Diversification method

The move procedure and Tabu list are described as follows:

- **Move procedure:** In this procedure, two indices of activities that can be started simultaneously are selected randomly and displaced according to the precedence relations. The resources assignment to the relevant skills of these activities are also changed randomly by checking the feasibility conditions.
- **Tabu list:** The abovementioned move procedure employs an intelligent TS strategy, based on avoiding a return to the solutions just visited, using an adaptive memory that is called a Tabu list. This Tabu list depends on the search direction and avoids repeating a move in the Tabu tenure. For example, if an obtained solution is accepted in a move, the move of the two mentioned points is added to the Tabu list. The forbidden moves remain inthe list in the Tabu tenure, and then exit the list.

3.1.4 Solutions improvement procedure

According to Figure 6, two parallel move procedures are used in order to improve the quality of the solutions.

For each existing solution in the population:

1. Counter is set to 0, and the following steps are repeated to reach the max-iteration:
2. The existing solution is improved as follows:
    2.1. Apply the move (neighbour) structure 1 on the solution.
    2.2. Apply the move structure 2 on the solution.
    2.3. Select the best solution among three solutions and consider it as the current solution for the next iteration.
3. Add one unit to counter. If the counter is not greater than max-iteration, Go to 4; else, Go to 2.
4. End.

Figure 6: Solutions improvement procedure

Structures 1 and 2 are explained as follows:

- **Move structure 1:** This structure is the same as the move approach in the solutions diversification procedure.
- **Move structure 2:** Index *a* is generated randomly in the interval [1,…, *n*-1] (*n* is the number of activities). *B* is the index of the first preceding activity existing in the *a*th cell, and *c* is the index of the last precedence activity existing in the *a*th cell index *d* is also generated randomly in the interval [*b,*…., *c*].
    - If *d*<*a*, the activity existing in the *a*th cell is inserted in the *d*th cell.
    - If *d*>*a*, the activity existing in the *a*th cell is inserted in the (*d*-1)*th* cell.

3.1.5 Updating the reference set

The reference set is formed of two subsets,*Refset*1 (the solutions of quality) and *Refset*2 (diversified solutions). The maximum capacity of these subsets is b_1 and b_2 respectively. In other words, $|refset| = b \le b_1 + b_2$. To make the reference set, the subset *Refset*1 is first formed. To do this, at most b_1 non-repetitive solutions are selected from among the best solutions and added to *Refset*1. Among the remaining solutions, b_2 non-repetitive solutions with the maximum Euclidean distance[1] from *Refset*1 are selected and added to *Refset2*.

3.1.6 Formation of the subsets

In this section, the subsets are formed from the solutions of the reference set in order to combine the solutions and generate the new ones. Hence, three types of pair subsets (S_1, S_2, S_3) are formed from the solutions of the reference set:

- **S_1**: All pair subsets of *Refset*1 that have at most $b_1 - 1$ members.
- **S_2**: All pair subsets of *Refset*2 that have at most $b_2 - 1$ members.
- **S_3**: A pair subset including *Refset*1 and *Refset*2 solutions whose members are made as succeeding. The first element of S_3 is each member of *Refset*1 that is selected in order. The second element is also one of the members of *Refset*2 that has the greatest euclidean distance from the first selected element. This subset contains b_1 members.

3.1.7 Subsets combination method

In this paper, a single-point crossover operator is used to combine each generated subset with the SS algorithm. The first and second members of each generated subset in the previous section are considered as parent1 and parent2. To generate offspring, one index *i* is generated randomly in the interval [1…*n*] (*n* is the number of activities) at first, and from the first to the *i*th cell of parent1 is assigned to the first to the *i*th cell of offspring1. The rest of the sequence of offspring1 is obtained from parent2 in the order of their appearance in this parent, for the activities that have not yet been selected in offspring1. Like offspring1, offspring2 is also generated by displacing the parents.

[1]The Euclidean distance between two points p and q is the length of the line segment connecting them.

3.2 Tabu search (TS)

The Tabu search was first developed by Glover [26]. In his ongoing research, Glover [27,28] explained the details of the TS algorithm. TS and Simulated Annealing (SA) algorithms have in common the fact that they guide the applied local search algorithm to avoid bad and low quality local optimum solutions by creating a Tabu list and supervising the examined solutions. A Tabu list stores the characteristics of examined solutions and prevents their being re-examined. Although the TS algorithm permits moves that make the objective function worse, in spite of the SA algorithm, it uses an exact acceptance criterion rather than a probable one. The TS chooses these moves around a corrected neighborhood structure. While running the algorithm, and based on the Tabu list, the neighborhoods are corrected and improved and lead to new neighborhoodsthat are in fact the original ones that have been improved. The improved neighborhoods can give rise to better solutions. The Tabu list has a size, and can store a determined number of characteristics. These characteristics are stored in the Tabu list according to the 'earliest in / earliest out' system. Therefore, characteristics are added to the beginning of the Tabu list and exit from the end of it (finishing the Tabu tenure) [28]. In order to investigate the quality of the solutions generated by the proposed SS, the MSPSP is also solved by using the TS algorithm [29], as in Figure 7:

0. Generate an initial solution for the MSPSP and consider it as current solution (x).
Create an empty Tabu list.
Counter=0
While Counter <= Number of iterations in the neighborhood structure of the TS do
1. Use move procedure to move from current solution to new solution (x').
2. Check acceptance criterion for x' and in case of acceptance, store it in Tabu list during Tabu tenure in order to avoid returning to it. Consider the accepted solution as current solution (x).
3. Update Tabu list.

End while
Consider x as the best solution of TS algorithm.

Figure 7: The pseudo code of the TS algorithm

The details of Figure 7 can be described as follows:

- **Initial solution:** In order to generate an initial solution, the same procedure used in the SS algorithm is employed.
- **Move procedure:** To compare the SS and TS algorithms, the move structures1 and 2 in the improvement procedure of SS are applied for the neighborhood operator of the TS algorithm. In this procedure, an intelligent TS strategy based on avoiding a return to generated solutions in the previous iterations is used by an adaptive memory called the Tabu list.
- **Tabu list:** The Tabu list depends on the **search direction,** and avoids the repetition of a move in the Tabu tenure. For instance, if index*j* is generated in a move and the obtained solution is accepted, its change move is added to the list. The forbidden moves remain in the list in the Tabu tenure, and after finishing Tabu tenure they exit the list.
  - **Search direction:** In order to determine the acceptance criterion of a new solution, the variable η is expressed by:

$$\eta = obj_A - obj_B \qquad (14)$$

where*A* is the current solution and *B* is a solution that has been generated by the TS method and by applying the move procedure. Therefore, the acceptance criterion is as given in one of the cases illustrated in Figure 8.

1. If $\eta \leq 0$ and the move does not exist in the Tabu list, the solution *B* will be replaced to *A*.
2. If $\eta \leq 0$ and the move exists in the Tabu list, the aspiration strategy will be applied and the solution *B* will be replaced to *A*.
    - The aspiration strategy is designed so that if the number of iterations for which the solution remains unchanged is greater than the defined allowable limit, the solution will be accepted even though it is forbidden.
3. If $\eta \geq 0$ and the move does not exist in the Tabu list, the solution *B* will be replaced to A if the solution *A* is not dominated by *B*.
4. If $\eta \geq 0$ and the move exist in the Tabu list, the solution *A* will remain unchanged.

Figure 8: Acceptance criterion

4. COMPUTATIONAL RESULTS

Since there is no special problem instance in the field of the MSPSP [6,9,12], some reasonable instances are generated randomly with a different number of activities and features,as listed in Table 1.

Table 1: Mathematical parameters of instances for the MSPSP

Parameter	Value
r_{mk}	$D \sim Uniform\ (0,1)^{*}$
b_{ik}	$D \sim Uniform\ (0,5)$
p_i	$D \sim Uniform\ (1,20)$
No. of skills in each project	$D \sim Uniform\ (5,7)$
No. of activities in project	10-20-30-50-100-500
No. of staff members for each instance	6 to 31 (depending on the no. of activities)

In order to analyse the developed algorithms further, the problem is solved in small stepsby the Lingo 9 software to find the exact solution. It should be noted that, in order to determine the parameters of the proposed algorithms, a trial and error method has been used in this paper. These parameters have been chosen so that the solutions acquired by the proposed algorithms are veryclose to the optimum solutions of the mathematical model for small-sized problems. These parameters are shown in Table 2. In this paper, the MSPSP is coded by the Matlab R2009a software, and all computational experiments are run on a Pentium 4 PC with 4 GB of memory and 2.20 GHz of CPU. The results of the small-sized problem after ten executions (in second) are shown in Table 3.

Table 2: Parameters of the SS and TS algorithms

Algorithms	Parameter	Value
SS	Population size	50
	$\|Refset1\|$	25
	$\|Refset2\|$	15
	Number of iterations	60
	Number of iterations in the neighbourhood structure of SS	5
TS	Number of iterations in the neighbourhood structure of the TS	5
	Maximum allowable iterations for keepingthe solution unchanged in the TS	10

Table 3: Computational results for small sizes

No. of (A,S,M)[1]	Results for ten executions						Lingo	
	SS			TS				
	(Min, Ave., Max) of OFV[2]	Ave. Time	NBS[3]	(Min, Ave., Max) of OFV	Ave. Time	NBS	OFV	Time
(10-6-6)	(296-296-296)	12	10	(296-298-325)	4	3	296	9750
(10-5-8)	(280-280-280)	15	10	(283-292-319)	6	4	280	10867
(20-6-6)	(517-517-517)	24	10	(543-601-631)	17	3	517	28940
(20-6-8)	(499-499-499)	42	10	(500,510,512)	21	2	499	42923
(20-5-10)	(406-406-406)	59	10	(426-471-497)	29	4	---	---[4]
(20-7-13)	(373-374-375)	76	6	(387-412-429)	38	4	---	---

As shown in Table 2, the exact solution of the mathematical model takes a long time to produce the optimised solutions in small sizes. The proposed SS produces the same results, but taking less time than Lingo 9. The proposed TS achieves feasible solutions in very short times, which can be considered as an upper bound to solve the given problem. The results of Table 2 show that the designed SS algorithm produces excellent solutions.

It should be mentioned that the model presented in this paper defines the variables and parameters from a different point of view, despite the approach of other papers [6, 13, 15, and 30]; and that the defined problem has many parameters. Note that, there is a final answer in all existing papers. These matters make it impossible to draw any exact comparisons among the existing papers.

In connection with these comments, the problem in larger and more real sizes is presented in Table 4. In fact, excellent procedures have been created to solve the MSPSP in the real sizes by mathematical programming, and by exact and meta-heuristic solutions in this paper.

5. CONCLUSION

In many projects (such as information technology projects, research projects, engineering activities, preventive maintenance activities, and the like) staff members should be assigned to project activities in accordance with their skills and taking into consideration the precedence network of projects. As mentioned before, this problem has become well-known as the multi-skilled project scheduling problem (MSPSP). Since the MSPSP is a special type of the multi-mode resource-constrained project scheduling problem (MM-RCPSP) with infinite modes, which has received less attention from researchers, a novel linear mixed-integer programming (LMIP) model has been presented for the MSPSP in this paper, and solved by an efficient scatter search (SS) algorithm. To verify the efficiency of the proposed SS, the generated random problems have been compared with the optimised solutions obtained by the Lingo 9 software in small sizes, and with the solutions obtained by the proposed elite Tabu search (TS) algorithm in medium and large sizes.

[1]Number of (Activities, Skills, Resources)
[2]Objective function value (OFV)
[3]Number of times of achievement to the best known solution by each proposed algorithm (NBS)
[4]After 15 hours

Table 4: Computational results for medium and large sizes

No. of (A,S,M)	Results for ten executions					
	TS			SS		
	(Min, Ave., Max) of OFV	Ave. Time(s)	NBS	(Min, Ave., Max) of OFV	Time(s)	NBS
(30-6-6)	(1429-1429-1429)	47	10	(1339-1339-1339)	42	10
(30-5-8)	(601-601-601)	51	10	(575-575-575)	47	10
(30-5-10)	(753-757-759)	63	8	(734-734-734)	66	10
(30-4-13)	(497-498-499)	71	8	(409-416-422)	118	7
(30-6-16)	(534-536-537)	79	8	(461-469-483)	138	8
(50-5-6)	(1458-1461-1466)	83	8	(1359-1359-1359)	159	10
(50-5-8)	(1442-1446-1449)	99	8	(1290-1290-1290)	230	10
(50-6-10)	(1051-1056-1062)	104	7	(941-953-959)	259	7
(50-6-13)	(800-807-811)	119	7	(750-763-771)	301	6
(50-4-16)	(592-595-601)	134	7	(500-500-500)	318	10
(50-6-19)	(671-673-675)	150	8	(583-592-597)	380	8
(50-6-31)	(450-457-461)	192	6	(382-388-391)	450	7
(50-6-39)	(398-399-400)	217	8	(339-342-346)	554	6
(50-7-47)	(417-423-432)	287	7	(366-371-374)	590	6
(100-6-13)	(1831-1843-1850)	393	6	(1597-1620-1661)	710	6
(100-7-19)	(1384-1388-1392)	499	7	(1214-1287-1312)	787	5
(100-7-31)	(961-965-971)	539	7	(835-869-901)	815	5
(100-6-39)	(707-717-724)	667	6	(601-653-671)	960	5
(100-6-47)	(621-641-645)	701	5	(513-549-582)	980	5
(500-6-13)	(8000-8026-8071)	799	7	(7208-7404-7508)	1009	6
(500-7-16)	(5983-6103-6121)	810	6	(5018-5101-5204)	1101	5
(500-4-19)	(4005-4089-4140)	846	6	(3573-3625-3688)	1232	5
(500-5-31)	(3401-3483-4013)	959	5	(2832-2846-2915)	1304	5

REFERENCES

[1] **Blazewicz, J., Lenstra, J.K. &RinnooyKan, A.H.G.** 1983. Scheduling subject to resource constraints: Classification and complexity, *Discrete Applied Mathematics*, 5, pp 11–24.

[2] **Herroelen, W. &Leus, R.** 2005. Project scheduling under uncertainty: Survey and research potentials, *European Journal of Operational Research*, 165, pp 289-306.

[3] **Hartmann, S. &Briskorn, D.** 2010. A survey of variants and extensions of the resource-constrained project scheduling problem, *European Journal of Operational Research*, 207, pp 1-14.

[4] **Demeulemeester, E.L. &Herroelen, W.S.** 2002. *Project scheduling: A research handbook*, Kluwer Academic Publisher.

[5] **Neron, E. &Boptista, D.** 2002. Heuristics for the multi-skill project scheduling problem, *International Symposium on Combinatorial Optimization*, Paris.

[6] **Bellenguez-Morineau, O. &Neron, E.** 2005. Lower bounds for the multi-skill project scheduling problem with hierarchical levels of skills, in: E.K. Burke and M. Trick, eds, Practice and theory of automated timetabling, *Lecture Notes in Computer Science*, Vol. 3616, Springer-Verlag, pp 229-243.

[7] **Dauzere-Peres, S., Roux, W. &Lassere, J.B.** 1996. Multi-resource shop scheduling with resource flexibility, *European Journal of Operation Research*, 107, pp 289-305.

[8] **Jurish, B.** 1992. Scheduling jobs in shops with multi-purpose machines, PhD Thesis, University of Osnabruck, Germany.

[9] **Artigues, C., Demassey, S. &Neron, E.** 2008. *Resource constraint project scheduling problem*, John Wiley and Sons.

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

[11] **Kadrou, Y. &Najid, N.M.** 2006. Tabu search algorithm for the MRCPSP with multi-skilled personnel, *Computational Engineering in System Application, IMACS Multi Conference*, Vol. 2(4-6), pp 1302-1309.

[12] **Valls, V., Perez, A. & Quintanilla, S.** 2009. Skilled workforce scheduling in service centres, *European Journal of Operational Research*, 193, pp 791-804.

[13] **Heimerl, C. &Kolisch, R.** 2009. Scheduling and staffing multiple projects with a multi-skilled workforce, *OR Spectrum*, 32 (2), pp 343-368.

[14] **Gutjahr, W.J., Katzensteiner, S., Raiter, P., Stummer, C. &Denk, M.** 2008. Competence-driven project portfolio selection scheduling and staff assignment, *Central European Journal of Operation Research*, 16, pp 281-306.
[15] **Pessan, C., Bellenguez-Morineau, O. &Neron, E.** 2007. Multi-skill project scheduling problem and total preventive maintenance, *MultidisciplinaryInt. Conf. onScheduling: Theory and Applications (MISTA)*, pp 608-610.
[16] **Taha, H.A.** 2011. *Operations research: An introduction*, Prentice Hall, 9th edition.
[17] **Bazaraa, M.S. &Sheraly, H.D.** 1993. *Nonlinear programming*, 2nd edition, John Wiley and Sons.
[18] **Lawler, E.L.** 1963. The quadratic assignment problem, *Management Science*, 9, pp 586–599.
[19] **Glover, F.** 1977. Heuristics for integer programming usingsurrogate constraints, *Decision Sciences*, 8, pp 156–166.
[20] **Glover, F.** 1994. Genetic algorithms and Scatter Search: Unsuspected potentials, *Statistics and Computing*, 4, pp 131–140.
[21] **Glover, F.** 1994. Tabu Search for nonlinear and parametric optimization (with links to genetic algorithms), *Discrete Applied Mathematics*, 49, pp 231–255.
[22] **Glover, F.** 1995. Scatter Search and star paths: Beyond the genetic metaphor, *OR Spektrum*, 17, pp 125–137.
[23] **Glover, F.** 1998. A template for scatter search and path relinking, in: J.K. Hao, E. Lutton, E. Ronald, D. Snyers (eds), *Lecture Notes in Computer Science*, 1363, Springer, pp 13–54.
[24] **Laguna, M.** 2002. Scatter Search,in: P.M. Pardalos& M.G.C. Resende (eds), *Handbook of applied optimization*, Oxford University Press.
[25] **Seifi, M. &Tavakkoli-Moghaddam, R.** 2008. A new bi-objective model for a multi-mode resource-constrained project scheduling problem with discounted cash flows and four payment models, *Int. J. of Engineering, Transaction A: Basic*, 21(4), pp 347-360.
[26] **Glover, F.** 1986. Future paths for integer programming and links to artificial intelligence, *Computer and Operations Research*, 13(5), pp 533-549.
[27] **Glover, F.** 1989. Tabu search-Part I, *INFORMS Journal on Computing*, 1 (3), pp 190-206.
[28] **Glover, F.** 1989. Tabu search-Part II, *INFORMS Journal on Computing*, 2 (1), pp 4-32.
[29] **Tavakkoli-Moghaddam, R., Azarkish, M. &Sadeghnejad, A.** 2010. Solving a multi-objective job shop scheduling problem with sequence-dependent setup times by a Pareto archive PSO combined with genetic operators and VNS, *Int. J. of Advanced Manufacturing Technology*, Article in Press, DOI:10.1007/s00170-010-2847-4.
[30] **Li, H. &Womer, K.** 2009. Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm, *Journal of Scheduling*, 12 (3), pp 281-298.