Contents lists available at GrowingScience

Journal of Project Management

homepage: www.GrowingScience.com

Multi-mode multi-skill resource-constrained project scheduling problem with differentiated professional capabilities

Chunhao Li[a], Feng Wang[a], and Tsuiping Chung[b*]

[a]*School of Business and Management, Jilin University, Changchun, China*
[b]*Business School of Northeast Normal University, Changchun, China*

C H R O N I C L E

Article history:
Received: February 9, 2023
Received in revised format: April 28, 2023
Accepted: September 14, 2023
Available online:
September 14, 2023

Keywords:
Project scheduling
Resource constraints
Multiple modes
Multiple skill types
Differentiated professional capabilities
Artificial immune system algorithm

A B S T R A C T

Motivated by a practical situation in a digital transformation project, this paper considers a resource-constrained project scheduling problem with multiple modes, multiple skill types, and differentiated professional capabilities. In the proposed problem, each project activity has one or more alternative execution modes associated with a trade-off between processing time and resource consumption. In an execution mode, an activity requires a certain number of employees with specific skill types and required professional capabilities. A mixed integer programming model is developed to minimize the total project duration. Since this problem is NP-hard, an efficient immunoglobulin-based artificial immune system (EIAIS) algorithm with a new encoding and decoding scheme and novel components is proposed. The effectiveness of the proposed EIAIS algorithm is tested on randomly generated instances. Computational results show that the proposed EIAIS algorithm has better performance than the existing algorithms.

1. Introduction

As a standard problem in project scheduling, the resource-constrained project scheduling problem (RCPSP) is to schedule activities subject to precedence relations and resource constraints. A zero-one linear programming formulation representing the RCPSP was first proposed by Pritsker, Waiters, and Wolfe(1969). Subsequently, Talbot (1982) introduces methods for formulating and solving a general class of non-preemptive resource-constrained project scheduling problems. Since then, the RCPSP has been found in many industrial applications, such as software project management (Alba & Francisco Chicano, 2007), IT product development (R. Chen et al., 2017), construction program management (García-Nieves et al., 2019; Kerkhove & Vanhoucke, 2017), and high-end equipment development (Cui et al., 2021).

Motivated by a practical situation in an automotive manufacturer's digital transformation project, this article considers the problem of allocating a limited number of multi-skilled employees with differentiated professional capabilities (i.e., skill levels) to project activities with multiple execution modes. In recent years, automakers have embraced digital transformation to rapidly develop new products and optimize business performance to become more competitive. However, automakers also face a dilemma in that the digital systems developed by software companies cannot meet the personalized needs of the automakers themselves, and these developed systems are often not reusable when the production environment changes. One

* Corresponding author.
E-mail address: cpzhong@nenu.edu.cn (T. Chung)

doi: 10.5267/j.jpm.2023.9.002

way out of this dilemma is to develop highly personalized digital systems in-house. As a result, automakers face the challenge of allocating limited human resources to digital systems development projects. For instance, the automaker FAW-Volkswagen has a production management department responsible for various digital system development projects. The four digital modules currently being developed simultaneously by the department are as follows.

Material Supermarket Layout Module: This module requires employees to be familiar with the design of logistics channels, material shelf layout, material specifications, etc.

Production Material Distribution Module: This module requires employees familiar with the takt time, material transport vehicles, transfer boxes, distribution routes, etc.

Algorithm Design Module: This module requires employees to have skills in demand analysis, mathematical modeling, algorithm programming, etc.

Simulation Platform Development Module: This module requires employees to have skills in functional design, software development, testing, etc.

There are currently around fifteen employees in the production management department responsible for developing the four modules mentioned above. Each module has multiple development modes, such as waterfall model, spiral model, parallel model, etc. Employees with less than five years of service experience have junior skill levels and are familiar with one or two modules, while those with more than five years of service have senior skill levels and are familiar with two or more modules. In general, the number of skills and skill levels mastered by employees varies. The duration of the digital system development project depends on how multi-skilled employees with different professional capabilities are allocated to the modules with multiple development modes. This problem can be abstracted as a resource-constrained project scheduling problem with multi-mode, multi-skill, and hierarchical skill levels.

Although the scientific community is paying unabated attention to the RCPSP and its derivatives, to the best of our knowledge, many researchers individually tackle the extensions such as multi-mode resource-constrained project scheduling problem (MRCPSP), multi-skill resource-constrained project scheduling problem (MS-RCPSP), and MS-RCPSP with hierarchical skill levels, etc. In some practical situations, resource-constrained project scheduling should not only consider the choice of project execution mode but also consider the skills and skill levels mastered by employees. Once the execution mode is selected, each activity is performed by a certain number of employees with specific skills and corresponding skill levels. Although the durations of some activities may be reduced by assigning employees with skill levels higher than the required skill levels, other activities that can only be performed by high-level skilled employees may not be able to be processed simultaneously, which may increase the overall project completion time. Therefore, it is necessary to study the allocation problem of employees with different skill levels in multi-mode and multi-skill project scheduling.

This article is structured as follows. Section 2 briefly reviews the relevant literature. Section 3 presents the problem definition, assumptions, and mathematical formulation. Section 4 describes an efficient immunoglobulin-based artificial immune system algorithm with improved components and a new encoding and decoding scheme. Computational results from various experiments performed to verify the effectiveness of the proposed algorithm are discussed in Section 5. Finally, Section 6 presents some suggestions for future research.

2. Relevant literature review

As the recent paper by Hartmann and Briskorn (2022) reviews variations and extensions of the resource-constrained project scheduling problem (RCPSP) and identifies some notable trends, we limit our literature review in this section to some of the studies most relevant to the issues explored in this article. As an extension of the RCPSP, the multi-skill resource-constrained project scheduling problem (MS-RCPSP) proposed by Néron and Baptista (2002) has attracted an enormous research effort (Bellenguez-Morineau & Néron, 2007; Montoya et al., 2014; Zhu et al., 2021).

To reflect the requirements of project activities on the professional capability of employees, variants of the MS-RCPSP have considered hierarchical levels of skills. Some researchers have studied the single objective project scheduling problem where employees have multiple skill levels. Toroslu (2003) provides a highest-level-first greedy heuristic for the MS-RCPSP with hierarchical ordering constraints. The ordering constraints are considered when matching the ranks of the employees and their positions. Bellenguez and Néron (2005) assume that each employee masters one or more skills and that an activity must be performed by employees with specific skills at the required level. They establish an integer linear programming model to minimize the duration of the project. Yannibelli and Amandi (2011) consider the effectiveness level of employees, which determines whether an employee can be assigned to perform an activity. The objective is to allocate the most effective employee to the project activities. Zheng et al. (2017) investigate an MS-RCPSP with a single execution mode to minimize project duration where the processing times of activities are specified and only an employee with the same or a higher level than the required level can process project activities. Myszkowski et al. (2018) extend MS-RCPSP

by considering the degree of familiarity of the employees with the skills. They design a hybrid greedy algorithm and differential evolution approach to minimize project duration. Lin et al. (2020) deal with a project scheduling problem in which each employee has one or more skills with specified levels of familiarity, and each project activity requires several skills of certain types at a fixed minimum level. Chen et al. (2022) propose a genetic algorithm to minimize the integrated cost of a project scheduling problem where the employees have different skill efficiencies determined by the learning effect. Li et al. (2023) study the assignment of multi-skilled employees with dynamic skill levels to a construction project with unfixed activity duration. They develop a mathematical model and a tabu search algorithm with priority strategies to solve the small and large size problems, respectively.

In the study of multi-objective project scheduling considering differentiated skill levels of employees, Gutjahr et al. (2010) integrate the MS-RCPSP with employee competencies and learning effects. The level of competence mastered by an employee depends on whether that competence was used in the previous project. Both the multi-skilled employees and stochastic service times are considered by Barz and Kolisch (2014) to minimize the waiting and penalty costs in a network operator assignment problem, where each activity has only one execution mode. Maghsoudlou et al. (2017) assume that selecting a workforce that masters a higher level of required skills can reduce the rework risk of an activity and increase the cost of execution. However, they overlook the relationship between job processing times and project execution modes. Najafzad et al. (2019) consider level switching in an MS-RCPSP, where the skill levels required by activities may differ and the processing times of activities are independent of execution modes. Li et al. (2020) incorporate skill development and collaboration effectiveness into an MS-RCPSP where maximizing overall project effectiveness and skill evolution are considered. Tian et al. (2022) develop a further extension of the MS-RCPSP in which skill switching of resources can result in significant additional cost and execution time. Their study proposes a metaheuristic with employee leveling and swap operators to adjust the activity assignment sequence and employee scheduling.

Although some studies have considered differences in employee skill levels in multi-mode multi-skill project scheduling problems, the assumptions in these studies are not very close to the actual environment. Cui et al. (2021) tackle a multi-project collaborative scheduling problem considering multiple execution modes and multi-skilled human resources with different skill levels. They assume that each task should be performed by a single person and that the actual duration of a task is determined by the human resource assigned to it. Polancos and Seva (2023) tackle a cost minimization model which includes different skill levels in a multi-mode multi-skilled project scheduling problem. They also assume that an activity requires only one employee to complete it. However, in many practical situations, the execution mode of an activity usually corresponds to a fixed processing time and a given amount of each resource (Maghsoudlou et al., 2016). On the other hand, for the multi-skill project scheduling problem with hierarchical skill levels, it is always required that activities must be performed by employees with specific skills at a required minimum level.

A review of the relevant literature shows that although some studies have covered the skill levels of employees and have considered multiple objectives and the impact of worker proficiency on skill levels, there is a paucity of work that incorporates employee skill levels into the actual environment of a multi-mode multi-skill project scheduling problem. Therefore, to bring the study closer to real-world environments, it is more practical to include the skill level of the employees in the multi-mode and multi-skill project scheduling problem with realistic assumptions. Since the traditional RCPSP is strongly NP-hard (Blazewicz et al., 1983), the scheduling problem proposed in this article is also NP-hard in the strong sense.

The main contributions of this paper are threefold: first, this paper extends the existing multi-mode multi-skill project scheduling by considering the hierarchical skill levels of employees. Second, a mixed integer programming model is developed for the proposed problem. Finally, an efficient immunoglobulin-based artificial immune system algorithm with a new encoding and decoding scheme and novel components is established to obtain high quality solutions.

3. Problem Formulation

In this section, a mixed integer programming (MIP) model is proposed to formulate the multi-mode multi-skill resource-constrained project scheduling problem with differentiated professional capabilities (MM-MSRCPSP-PC). As an extension of multi- mode and multi-skill RCPSP, the MM-MSRCPSP-PC is much more complex. The details of the studied problem can be described formally as follows.

There is a project consisting of a set $V = \{0,1,\ldots,N,N+1\}$ of activities linked by priority relationships. The resources are employees with different skill types and differentiated professional capabilities (i.e., skill levels). For each activity, an execution mode is chosen from a set of available modes. Each execution mode corresponds to a fixed processing time and a certain number of employees with the required minimum skill level. In general, each execution mode involves a trade-off between the processing time of an activity and the allocation of employees. Under a given execution mode, activity requires a certain number of skills with a fixed minimum skill level. Only the employee with the required or a higher skill level can perform a given activity. The objective is to minimize the project duration, i.e., the makespan.

A solution of MM-MSRCPSP-PC can be obtained by determining the mode of executing each activity, the optimal assignment of employees with specific skill types and required or higher skill levels, and the starting times for all activity. Some assumptions of the proposed MIP model are defined below.

Activities numbered 0 and $N+1$ are dummy activities representing the start and end of the project.
Interruption is not permitted when executing an activity.
Each activity is carried out in an assigned mode, which should remain unchanged during the execution of the activity.
Employees involved in the project are always available.
Each employee can only be assigned to one skill of an activity at a time. Note that the selected employee should have an identical or a higher level of the skill required by the activity.
Each employee performs an activity, the duration of which is determined by their skill level.
All employees with specific skill types and required or higher skill levels to perform an activity should start processing simultaneously.

Before introducing the MIP model, some notations used to formulate the MM-MSRCPSP-PC problem are presented below.
Indices

i, j	Index of project activities where $i, j = 0,1,...,N,N+1$.
m	Index of modes where $m = 1,2,...,M$.
s	Index of employees where $s = 1,2,...,S$.
k	Index of skills where $k = 1,2,...,K$.
t	Index of time where $t = 1,2,...,T$.

Parameters

N	Total number of activities.
M	Total number of modes.
S	Total number of employees.
K	Total number of skills.
L	Total number of levels per skill.
KS	The number of skills mastered by each employee.
T	Upper bound of the project duration.
U_i	Set of all preceding activities for activity i.
h_{sk}	The level of skill k mastered by employee s. Note that $h_{sk} \in \{0,1,...,L\}$.
d_{im}	The duration of activity i executed in mode m.
b_{imk}	Number of employees required to perform skill k of activity i in mode m.
l_{imk}	Minimum level required to perform skill k of activity i in mode m. Note that $l_{imk} \in \{0,1,...,L\}$.
W	A very large positive number.

Decision variables

$$e_{im} = \begin{cases} 1 & \text{if activity } i \text{ is executed in mode } m \\ 0 & \text{otherwise} \end{cases}$$

$$x_{imst} = \begin{cases} 1 & \text{if activity } i \text{ is executed in mode } m \text{ by staff } s \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$y_{imsk}=\begin{cases}1 & \text{if activity } i \text{ is executed in mode } m \text{ by staff } s \text{ for skill } k\\ 0 & \text{otherwise}\end{cases}$$

$$z_{imt}=\begin{cases}1 & \text{if activity } i \text{ is executed in mode } m \text{ at time } t\\ 0 & \text{otherwise}\end{cases}$$

The proposed MM-MSRCPSP-PC can be formulated as follows.

$$\min C_{\max}=\sum_{m=1}^{M}\sum_{t=0}^{T} t\times z_{(N+1)mt} \tag{1}$$

s.t.

$$\sum_{m=1}^{M} e_{im}=1 \qquad \forall i \tag{2}$$

$$\sum_{m=1}^{M}\sum_{t=0}^{T} z_{imt}=1 \qquad \forall i \tag{3}$$

$$\sum_{m=1}^{M}\sum_{t=0}^{T} t\times z_{jmt}+\sum_{m=1}^{M} d_{jm}\times e_{jm}\le\sum_{m=1}^{M}\sum_{t=0}^{T} t\times z_{imt} \qquad \forall i,j\in U_i \tag{4}$$

$$\sum_{m=1}^{M}\sum_{t=0}^{T} x_{imst}\le 1 \qquad \forall i,s \tag{5}$$

$$\sum_{m=1}^{M}\sum_{k=1}^{K} y_{imsk}\le 1 \qquad \forall i,s \tag{6}$$

$$z_{imt}\le e_{im} \qquad \forall i,m,t \tag{7}$$

$$x_{imst}\le e_{im} \qquad \forall i,m,s,t \tag{8}$$

$$y_{imsk}\le e_{im} \qquad \forall i,m,s,k \tag{9}$$

$$y_{imsk}\times h_{sk}+W(1-y_{imsk})\ge l_{imk}\times e_{im} \qquad \forall i,m,s,k \tag{10}$$

$$x_{imst}\le z_{imt} \qquad \forall i,m,s,t \tag{11}$$

$$x_{imst}+1\ge z_{imt}+\sum_{k=1}^{K} y_{imsk} \qquad \forall i,m,s,t \tag{12}$$

$$\sum_{s=1}^{S} y_{imsk}=b_{imk}\times e_{im} \qquad \forall i,m,k \tag{13}$$

$$\sum_{s=1}^{S}\sum_{t=0}^{T} x_{imst}=\sum_{k=1}^{K} b_{imk}\times e_{im} \qquad \forall i,m \tag{14}$$

$$\sum_{i=0}^{N+1}\sum_{m=1}^{M}\sum_{\tau=t}^{t+d_{im}-1} x_{imst}\le 1 \qquad \forall s,t \tag{15}$$

$$\sum_{m=1}^{M}\sum_{t=0}^{T} x_{imst}=\sum_{m=1}^{M}\sum_{k=1}^{K} y_{imsk} \qquad \forall i,s \tag{16}$$

Constraint (1) states that the objective is to minimize the project duration, i.e., the makespan, which is equal to the start time of the dummy end activity $N+1$. Constraint (2) ensures that each activity is executed in only one mode. Constraint (3) enforces that each activity has a fixed time to start and an execution mode to perform. Constraint (4) implies that the start time of each activity should not be earlier than the completion times of all preceding activities. Constraint (5) guarantees that there is no more than one mode and one starting time for each employee allocated to an activity. Constraint (6) guarantees that at most one execution mode and one skill type can be selected for each employee allocated to an activity. The logical relationships between decision variables are specified by Constraints (7-9). Constraint (10) implies that any employee member can be assigned to perform a skill of an activity if the employee member masters the skill at least to the required level. Constraints (11-12) ensure that all employees allocated to the different skills required for each activity must start their work simultaneously. Constraint (13) ensures that the number of employees with skill k assigned to activity i should equal the number of employees with the same skill type required to process the same activity. Constraint (14) ensures that the number of employees allocated to process each activity should equal the required number of employees. Constraint

(15) guarantees that employees assigned to perform an activity in one mode do not process another activity during the entire processing time of the current activity. Constraint (16) ensures that employee s should process activities in a point without beyond the project time horizon if the employee is assigned to skill k of activity i.

Example 1. In order to exemplify the developed MIP model, an example of five employees implementing a project with six activities, two execution modes, and four skill types is used. In the project, each employee masters three of the four skill types. Each skill corresponds to two levels of proficiency (i.e., $N=6$, $M=2$, $S=5$, $K=4$, $KS=3$, $L=2$). The priority relationship of the six activities is shown in Figure 1. Note that the activities 0 and 7 are the dummy first and end activities. The personnel, duration, and skills required for each activity are presented in Table 1.

Table 1
Employee, duration, skill types, and skill levels required for each activity in Example 1

i	m	Requirements for performing skill k of activity i in mode m				Duration (d_{im})	Number of skills needed for activity i
		Skill 1	Skill 2	Skill 3	Skill 4		
		(b_{im1},l_{im1})	(b_{im2},l_{im2})	(b_{im3},l_{im3})	(b_{im4},l_{im4})		
1	1	(0,0)	(2,2)	(0,0)	(0,0)	4	1
	2	(0,0)	(3,1)	(0,0)	(0,0)	3	1
2	1	(1,2)	(1,2)	(0,0)	(0,0)	3	2
	2	(2,1)	(1,2)	(0,0)	(0,0)	4	2
3	1	(2,2)	(0,0)	(0,0)	(0,0)	2	1
	2	(3,1)	(0,0)	(0,0)	(0,0)	2	1
4	1	(0,0)	(0,0)	(2,2)	(0,0)	2	1
	2	(0,0)	(0,0)	(3,1)	(0,0)	2	1
5	1	(0,0)	(1,2)	(0,0)	(1,2)	2	2
	2	(0,0)	(1,1)	(0,0)	(1,2)	3	2
6	1	(0,0)	(0,0)	(0,0)	(2,2)	2	1
	2	(0,0)	(0,0)	(0,0)	(3,1)	2	1

The types and levels of skills possessed by employees are shown in Table 2. After coding the MIP model in AMPL and solving it with the CPLEX solver, the solution is 9, as shown in Fig. 2. The computational time is 0.266 seconds.

Table 2
Skill types and skill levels mastered by employees in Example 1

Employee (S)	Skill 1	Skill 2	Skill 3	Skill 4	KS
	h_{s1}	h_{s2}	h_{s3}	h_{s4}	
1	2	2	0	1	3
2	2	0	2	2	3
3	1	1	2	0	3
4	1	2	1	0	3
5	0	2	2	2	3

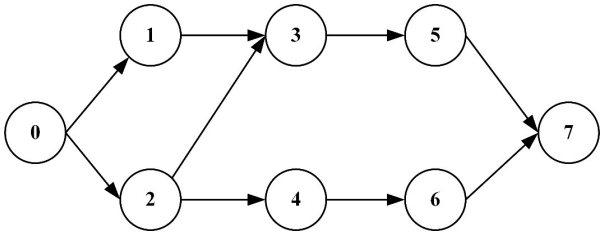

Fig. 1. The precedence graph of the project in Example 1

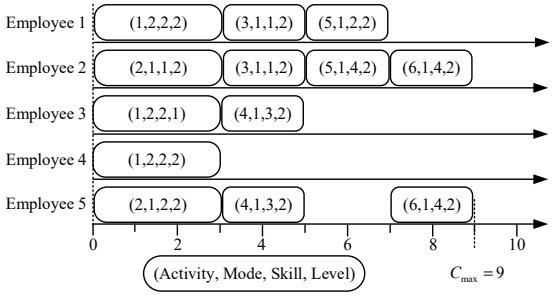

Fig. 2. The solution obtained by AMPL for Example 1

4. The efficient IAIS Algorithm

In this section, an efficient immunoglobulin-based artificial immune system (EIAIS) algorithm is designed to solve the resource-constrained project scheduling problem with multiple modes, multiple skill types, and differentiated professional capabilities. The original version of the immunoglobulin-based artificial immune system (IAIS) algorithm includes five components: encoding and decoding, somatic recombination, somatic hypermutation, isotype switching, and elimination (Chung & Liao, 2013). Based on the basic components, IAIS has been used to solve several NP-hard combinatorial optimization problems (Chung & Chen, 2019; C. Li et al., 2022).

In various versions of IAIS-based metaheuristics, each possible schedule is accepted as a receptor and represented by an integer-valued string with N activities. Once the receptor populations have been generated, the somatic recombination component helps the receptors to cluster around the currently found optimal individual. The somatic hypermutation component is then used to enhance the search breadth of the IAIS-based algorithms. Subsequently, isotype switching is performed to increase the search depth. After running these components, a new receptor population is created by component elimination. However, the solution representation used in the previous IAIS-based metaheuristics is insufficient to determine the execution modes, the assigned skill types, and the assigned personnel with different skill levels. Therefore, an EIAIS algorithm with a new encoding and decoding scheme is proposed. To describe the proposed EIAIS algorithm, we first briefly discuss its components.

4.1 Encoding and decoding

A feasible solution representation should take the processing sequence of activities under priority relationships and the choice of execution mode, skill types, and staffing into account. The proposed solution representation is a $(2+\Lambda+\max\sum_{k=1}^{K} b_{imk})\times N$ array, where Λ is the maximum number of skills required for all activities, $i=1,2,\cdots,N$, and $m=1,2,\cdots,M$. This demonstration array is accepted as a receptor and there are A receptors in the EIAIS algorithm. It is clear from Table 1 that $\Lambda=2$, $\max\sum_{k=1}^{K} b_{imk}=3$, and $N=6$. For example, a solution representation of the project in Example 1 is shown in Fig. 3, which has 7 rows and 6 columns. The positive integers between 1 and N in the first row determine the processing order of all activities. The real numbers between 0 and 1 in the other rows identify the assignment of execution modes, skill types, and employees to activities. Based on the encoding scheme of the receptor, a feasible solution is computed by the following decoding processes.

Activity	2	3	1	5	4	6
Mode	0.45	0.23	0.64	0.37	0.08	0.35
Λ Skill	0.32	0.13	0.81	0.86	0.26	0.72
Λ Skill	0.21	0.99	0.40	0.55	0.77	0.02
$\max\sum_{k=1}^{K} b_{imk}$ Employee	0.52	0.40	0.39	0.06	0.58	0.47
$\max\sum_{k=1}^{K} b_{imk}$ Employee	0.85	0.29	0.11	0.33	0.96	0.62
$\max\sum_{k=1}^{K} b_{imk}$ Employee	0.68	0.66	0.43	0.17	0.32	0.54

Fig. 3. Schematic representation of the receptor encoding scheme

The N activities in the first row of the demonstration array are first separated to determine the processing order of the project activities. An activity i is identified as an executable activity when all its predecessors have been completed, i.e., (pre-list $i=\varnothing$). All executable activities are placed in a temporary set called the optional list. Then, the executable activity with the corresponding smallest place value in the first row is selected. This process is performed iteratively until the processing order of all activities is confirmed. The steps for determining the processing order of all activities in the presented Example 1 are shown in Table 3.

Table 3
Steps of determining the processing order of activities from a receptor

Steps	Predecessor list	Optional list	Corresponding values of places in first row	The selected activity
Step 1	Pre-list 1 = ∅ Pre-list 2 = ∅ Pre-list 3 = [1,2] Pre-list 4 = [2] Pre-list 5 = [3] Pre-list 6 = [4]	[1,2]	3,1	2
Step 2	Pre-list 1 = ∅ Pre-list 3 = [1] Pre-list 4 = ∅ Pre-list 5 = [3] Pre-list 6 = [4]	[1,4]	3,5	1
Step 3	Pre-list 3 = ∅ Pre-list 4 = ∅ Pre-list 5 = [3] Pre-list 6 = [4]	[3,4]	2,5	3
Step 4	Pre-list 4 = ∅ Pre-list 5 = ∅ Pre-list 6 = [4]	[4,5]	5,4	5
Step 5	Pre-list 4 = ∅ Pre-list 6 = [4]	4	5	4
Step 6	Pre-list 6 = ∅	6	6	6

According to Table 3, in step 1, activities 1 and 2 are placed in the optional list because their predecessors have been completed. In the first row of the represented solution shown in Fig. 3, the corresponding values of places for executable activities 1 and 2 are 3 and 1, respectively. Activity 2 is then selected as the first activity in the processing order because it has the smallest place value. Subsequently, activity 2 is removed from the current predecessor list.

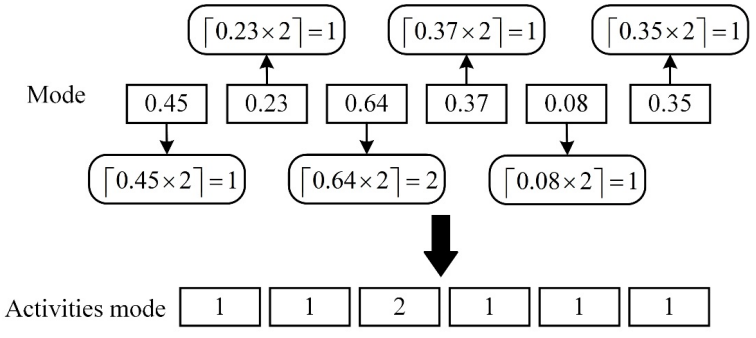

Fig. 4. Schematic representation of mode selection

These procedures are repeated until the processing order of the six activities is confirmed. Once the processing order of project activities has been determined, the mode selection is completed by the following procedures. N elements of the second row in the demonstration array are separated. Each element is multiplied by the number of modes and rounded up. Then, the execution mode of an activity is determined by the rounded-up value in the second row of the same column. The mode selection procedures are shown graphically in Fig. 4. Once the mode selection has been identified, the requirements of skill types, skill level, and duration for each project activity are determined. Once the processing order and execution modes of the activities have been determined, the skill allocation should be carried out. Obviously, the maximum number of skills required to perform each activity in Example 1 is 2. Therefore, the third and fourth rows of the demonstration array are used to determine the skills allocation. Specifically, the elements in the third row are multiplied by the number of remaining skills required in the selected execution mode and rounded up. If there are still skills to assign in the current activity, the same operation is performed on the elements in the fourth row. Otherwise, the decoding process moves to the next column to assign the skills for the next activity. In this way, all the skills required for each activity are selected. Note that the skill type and the skill level requirements of an activity are determined by how it is performed. Fig. 5 illustrates the allocation of skills.

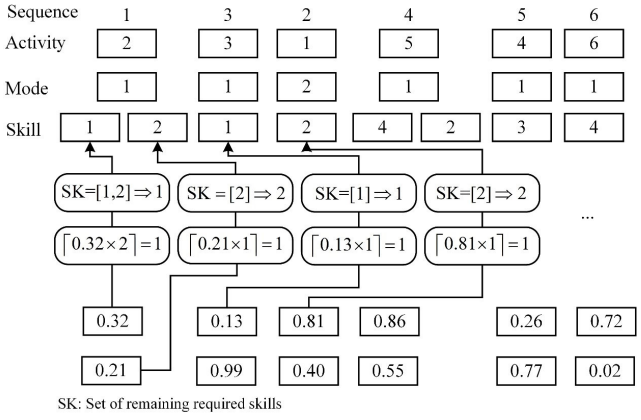

Fig. 5. Schematic representation of skill allocation

Once the skills have been assigned to the activities, the employee with the required skill level should be allocated to perform the skills. In Example 1, the maximum number of employees selected to perform an activity is 3. Therefore, the fifth, sixth, and seventh rows are used to assign employees to the activities.

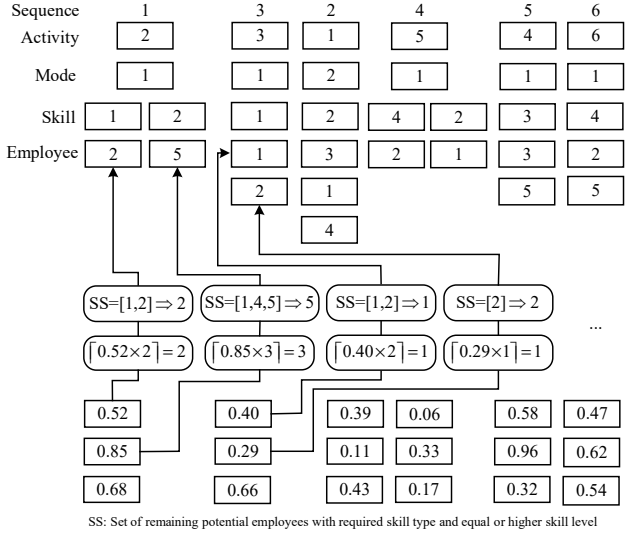

Fig. 6. Schematic representation of employee assignment

The assignment is achieved by rounding up the result of multiplying the corresponding elements of the fifth, sixth, and seventh rows by the number of remaining potential employees with the required skill type and equal or higher skill level to perform the activity. When the employee assignment for a particular activity is complete at a given time, the decoding

process moves to the next column to assign the required employees for the next activity. The employee assignment procedures are shown in Fig. 6. Using the makespan obtained by the above decoding procedures, the affinity value of each receptor can be calculated using the equation: $Affinity = 1/makespan$. As can be seen from the equation, a lower makespan has a higher affinity value.

4.2 Somatic recombination

The function of somatic recombination in adaptive immunity is to cut and recombine receptor gene fragments. In this way, somatic recombination can increase the likelihood that newly generated receptors will bind and eliminate invading antigens. For one generation in the proposed EIAIS algorithm, a receptor with the highest affinity value is defined as the standard receptor. For other receptors, R of N columns are randomly selected. First, these selected columns are inserted into the positions where they have the same elements in the first row of the standard receptor. Then, the elements in the selected column are replaced by those in the same position in the standard acceptor column. Fig. 7 shows a diagram of somatic recombination.

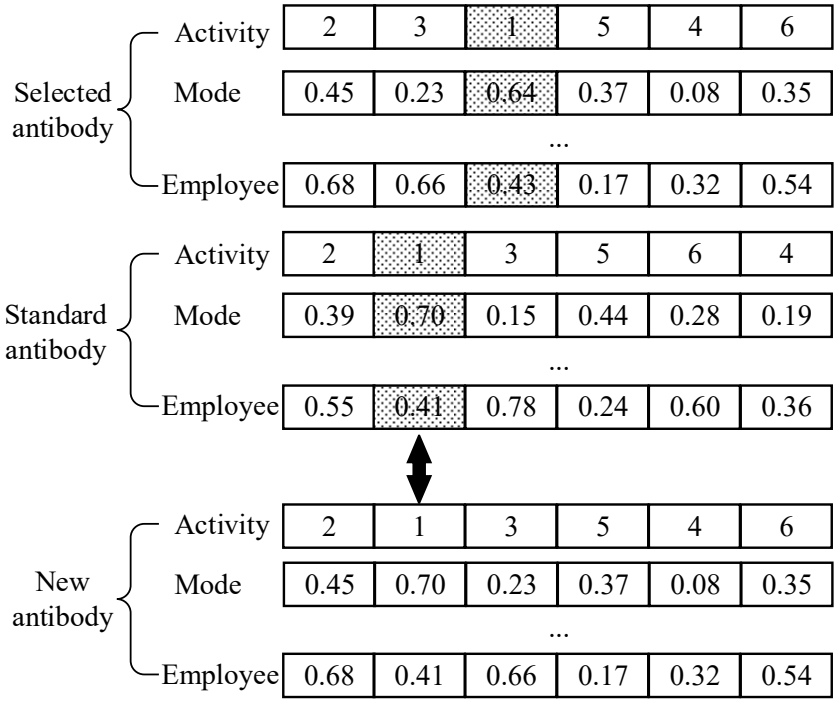

Fig. 7. Schematic representation of somatic recombination

4.3 Somatic hypermutation

As an early-stage receptor, somatic hypermutation has the task of finding the possible antigen positions. Hierarchical inversion is used in somatic hypermutation to explore a broader search space. Two binary vectors called BV and (1-BV) are used when a receptor undergoes hierarchical inversion. For instance, if BV=[0,1,1,1,0,1,0], then $(1-\mathrm{BV}) = [1,0,0,0,1,0,1]$. To perform the hierarchical inversion, two columns α and β of the chosen receptor are randomly selected. Note that the absolute difference between the positions of these two columns should not be less than 2. First, column α and column β are pointwise multiplied by BV and (1-BV), respectively. The new column α' is generated by adding the resulting two terms such that $\alpha' = \alpha \times BV + \beta \times (1-BV)$. The new column β' is generated in a similar way such that $\beta' = \alpha \times (1-BV) + \beta \times BV$. Second, the columns between α and β are subjected to the same processes. A new receptor is then obtained, as depicted graphically in Fig. 8.

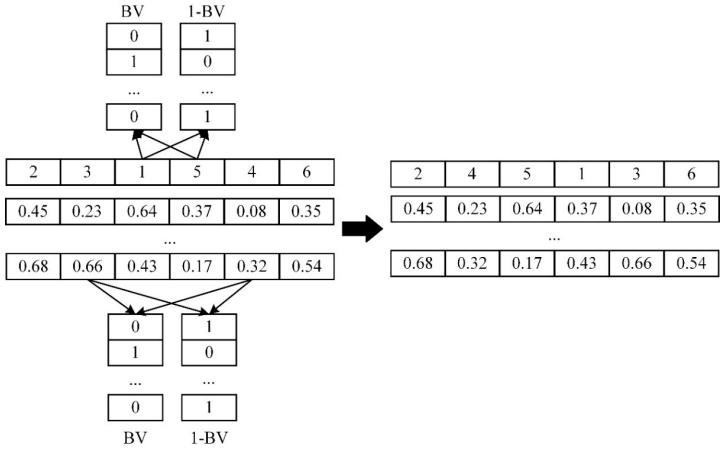

Fig. 8. Schematic representation of somatic hypermutation

4.4 Isotype switching

With the help of three more mature immunoglobulin receptors involved in isotype switching, the immune system can quickly bind and destroy antigens. For the EIAIS, there are three corresponding operators, i.e., IgA, IgE, and IgG. Depending on the unique structures, these three operators have different functions and can be used to generate much more effective receptors. After a receptor has undergone somatic hypermutation without increasing its affinity value, isotype switching is repeated I times, with IgG, IgA, and IgE randomly selected each time. During the processing of isotype switching, the original receptor is replaced by the newly generated receptor if the new receptor has a higher affinity value.

4.4.1 IgG

A remarkable property of IgG is its ability to access foreign antigens in the invaded tissues. The hierarchical pairwise swap is therefore used in IgG because it allows a fast and deep search of the neighborhood of the receptor. Let α and β be randomly selected columns for a receptor. Two binary vectors BV and (1-BV) are randomly generated. Then, α and β are updated respectively to $\alpha' = \alpha \times BV + \beta \times (1 - BV)$ and $\beta' = \alpha \times (1 - BV) + \beta \times BV$. Figure 9 shows a diagram of the IgG.

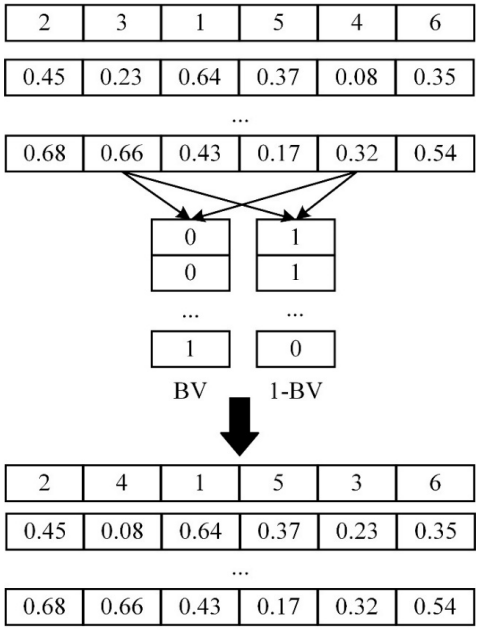

Fig. 9. Schematic representation of IgG

4.4.2 IgA

A prominent feature of IgA is its ability to penetrate the bloodstream and protect the regions that IgG cannot reach. The hierarchical mutation is therefore used in IgA because it can search for distant neighbors of the current receptor. For a receptor, let α be a randomly selected column in the receptor. The element of the first row in the column α remains unchanged. Instead, elements of other rows in column α are randomly changed from the interval (0,1). A schematic representation of IgA is graphically shown in Fig. 10.

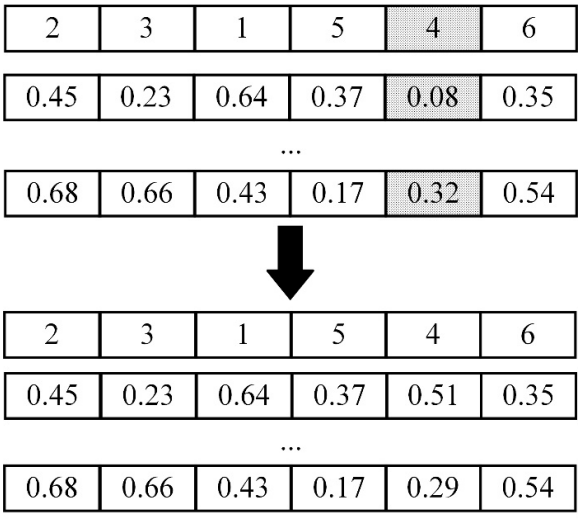

Fig. 10. Schematic representation of IgA

4.4.3 IgE

Unlike IgA and IgG, IgE can recognize and bind antigens. Therefore, both the hierarchical pairwise swap and hierarchical mutation are used in IgE to escape the local optimum. A new receptor is obtained by applying the hierarchical pairwise swap and hierarchical mutation presented in IgG and IgA.

4.5 Elimination

To explore a broader search space, the receptor with the highest affinity value in the current generation is retained. Other receptors in the current population are deleted and randomly regenerated. The proposed EIAIS algorithm will stop when the termination criterion is satisfied. Based on the various components presented above, the steps of the proposed EIAIS algorithm are described below. Fig 11 shows the framework of the EIAIS algorithm.

The Proposed EIAIS Algorithm

Input algorithm parameters:
A : Number of the receptor population size
R : Number of columns randomly selected in somatic recombination
I : Number of replicates for isotype switching
Generate an initial receptor population of A receptors:
While termination criterion≠True:
  Renew the receptor population through somatic recombination;
  For each receptor do:
    Somatic hypermutation results in a new receptor;
    If the new receptor has a higher affinity value
      Replace the original receptor with the new one;
    else
      Isotype switching results in a new receptor;
      If the new receptor has a higher affinity value
        Replace the original receptor with the new one;
  End For
  Perform the elimination;
End while

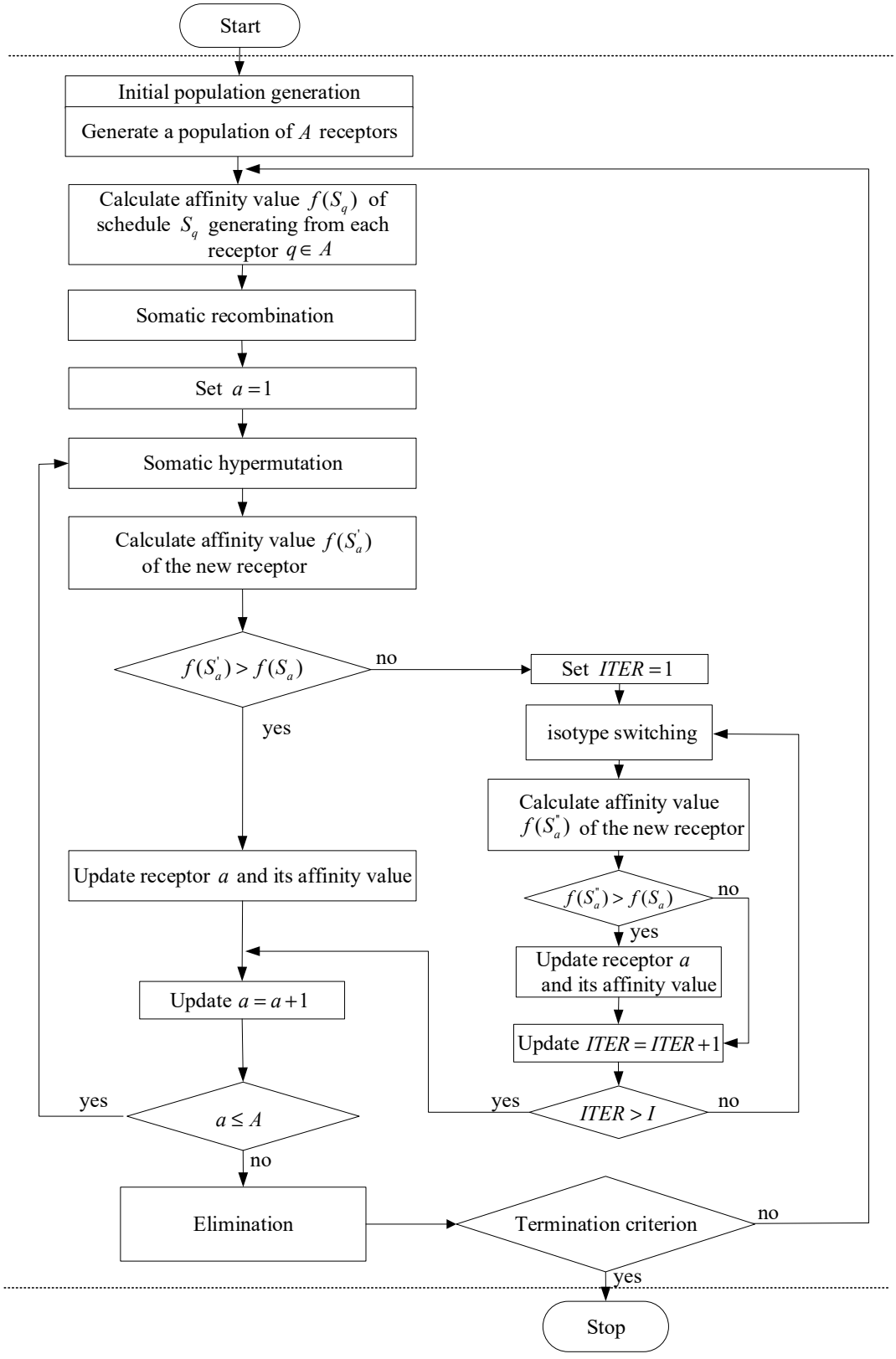

Fig. 11. Flowchart of the proposed EIAIS algorithm

The EIAIS algorithm starts with an initial population of A receptors. First, somatic recombination is applied to each receptor, resulting in a new population of receptors. Next, somatic hypermutation is performed for each receptor. The original receptor will be replaced by the new receptor obtained by somatic hypermutation if the new one has a higher affinity value. Otherwise, the current receptor undergoes I iterations of isotype switching. If a new receptor generated by IgG, IgA, or

IgE has a higher affinity value, the original receptor is replaced by the new one. When all receptors have undergone somatic hypermutation, elimination is performed to generate the next generation. These steps are repeated until the termination criterion is met.

5. Computational results

This section presents the computational experiments conducted in two parts: the MIP model and the EIAIS algorithm. All the experiments are performed on an Intel Core i7, 2.3GHz PC with 16 GB RAM. The MIP model is coded in AMPL software and solved by CPLEX 12.6.3.0 while the EIAIS algorithm is coded in C++. Since no benchmark instances exist for the proposed MM-MSPSP-PC, some problem instances are randomly generated to test the proposed model and algorithm. The priority constraints between activities are derived from the datasets in the Project Scheduling Problem Library (PSPLIB). Bellenguez and Néron (2005) assume that the number of skills for each instance is randomly generated between three and six. According to Drezet and Billaut (2008), each employee masters 70% of the total number of skills. Each skill is divided into junior, intermediate, and senior levels. In addition, the requirements for duration, number of employees, and skill types for each activity in each mode are generated from the uniform distributions shown in Table 4. Note that the level required to perform each skill is randomly generated.

Table 4
Distributions used to generate the requirements for performing activities in different modes

Mode	Duration	Employee quantity requirement	Skill requirement
Mode 1	Uniform (1,6)	Uniform ($\lceil 0.4S \rceil, \lceil 0.6S \rceil$)	Uniform ($\lceil 0.25K \rceil, K$)
Mode 2	Uniform (3,8)	Uniform ($\lceil 0.3S \rceil, \lceil 0.5S \rceil$)	
Mode 3	Uniform (5,10)	Uniform ($\lceil 0.2S \rceil, \lceil 0.4S \rceil$)	
Mode 4	Uniform (7,12)	Uniform ($\lceil 0.1S \rceil, \lceil 0.3S \rceil$)	

5.1 Experiments with the proposed MIP model

In this subsection, the MIP model presented in Section 3 is implemented in AMPL and solved using the CPLEX 12.6.3 solver. The number of activities is set to $N = 8,10,12$. Since the problem size starts from 10 activities in the PSPLIB, precedence constraints of 8 activity problems are generated by modifying the precedence relations of 10 activity problems from PSPLIB. The number of employees is set to $S = 6,8$. In addition, the number of skills and execution modes are set to $K = 3,4,6$ and $M = 2,3,4$, respectively. Table 5 shows the solutions and the CPU times obtained by running the MIP model. Table 5 shows that as the number of modes, employees, or activities increases, the average computation time of the MIP model increases. For the problem instances with 12 activities, the average computational time for solving the MIP model by AMPL increases significantly. AMPL cannot obtain an optimal solution within an acceptable CPU time when the number of activities exceeds 12.

Table 5
The result of the proposed MIP model

N	S	K	$M = 2$		$M = 3$		$M = 4$	
			Solution	Time(s)	Solution	Time(s)	Solution	Time(s)
8	6	3	25	3.890	25	16.422	25	26.766
		4	26	4.219	21	11.453	22	10.469
		6	27	2.813	28	2.437	25	22.812
		Ave.	26	3.641	24.7	10.104	24	20.016
	8	3	29	8.812	22	85.719	22	96.641
		4	28	6.953	26	19.813	29	47.937
		6	27	1.672	24	25.594	23	19.890
		Ave.	28	5.812	24	43.709	24.7	54.823
10	6	3	25	49.547	30	55.625	24	95.062
		4	27	24.750	27	39.875	28	81.391
		6	30	21.297	28	31.750	24	67.094
		Ave.	27.3	31.865	28.3	42.417	25.3	81.182
	8	3	28	401.016	33	841.281	34	935.625
		4	39	41.485	32	247.469	32	756.579
		6	32	8.688	27	96.625	32	117.765
		Ave.	33	150.396	30.7	395.125	32.7	603.323
12	6	3	27	389.469	28	1884.41	27	2283.69
		4	37	19.422	31	837.422	32	849.828
		6	35	45.156	29	283.938	32	1467.81
		Ave.	33	151.349	29.3	1001.923	30.3	1533.77
	8	3	32	1553.58	27	6165.19	30	8790.42
		4	40	318.016	38	3253.36	36	6394.14
		6	38	261.671	35	1586.55	33	1632.62
		Ave.	36.7	711.089	33.3	3668.367	33	5605.72

5.2 Experiments with the components in EIAIS

Based on the new encoding and decoding schemes, the EIAIS algorithm is developed using novel components, such as hierarchical inversion, hierarchical pairwise swap, and hierarchical mutation. In order to verify the advantage of the novel components used in EIAIS, the IAIS with the original components, i.e., inverse mutation, pairwise swap, and insertion mutation, is selected as a comparison. The small and large size problem instances with $N=\{10,14,20\}$ and $N=\{30,60,90\}$ respectively are used to evaluate the performance of the EIAIS and IAIS algorithms. According to the setting of activity requirements in some literatures (Maghsoudlou et al., 2016; Nemati-Lafmejani et al., 2019), as the number of activities increases from small to large size problems, the required number of execution modes, skill types and employees increases slightly. For the small size problem instances, the number of employees, execution modes, and skill types are set to $S=8,10$, $M=2,3$, and $K=3,4$, respectively. For the large size problem instances, the settings $S=12,14$, $M=3,4$, and $K=4,6$ are used. All test problem instances are randomly generated based on the uniform distributions shown in Table 4. Ten instances are randomly generated for each combination of *N*, *S*, *M* and *K*. Based on preliminary test results, the parameters of EIAIS are: $A=10$, $R=3n/4$, and $I=10$. The EIAIS and IAIS algorithms are run ten times for each problem instance. A time limit $T=0.1\times N\times M$ is used as the termination criterion. To measure the performance of the EIAIS and IAIS algorithms, the relative deviation index (*RDI*) is used and calculated as following.

$$RDI=\frac{C_{\max}^{H}-C_{\max}^{best}}{C_{\max}^{best}}\times 100 \tag{17}$$

where $C_{\max}^{H}$ can be $C_{\max}^{EIAIS}$ and $C_{\max}^{IAIS}$. $C_{\max}^{EIAIS}$ represents the makespan computed using the EIAIS algorithm, while $C_{\max}^{IAIS}$ represents the makespan computed using the IAIS algorithm. $C_{\max}^{best}$ is the optimal value of these two algorithms. Fig. 12 shows the *RDI* values of the EIAIS and IAIS algorithms for different numbers of employees. For the small size problems, the average *RDI* values of the EIAIS and IAIS algorithms are 6.26 and 16.06 when *S=8*, 6.93 and 20.71 when $S=10$. Specifically, the average *RDI* values of the EIAIS for *N=10*, 14, and 20 are 6.96, 6.30, and 5.51 when $S=8$, while the corresponding values of the IAIS are 15.27, 16.48, and 16.42. When *S =10*, the average *RDI* values of the EIAIS for *N=10*, 14, and 20 are 6.93, 6.45, and 7.42, while the corresponding values of the IAIS are 19.42, 20.62, and 22.10.

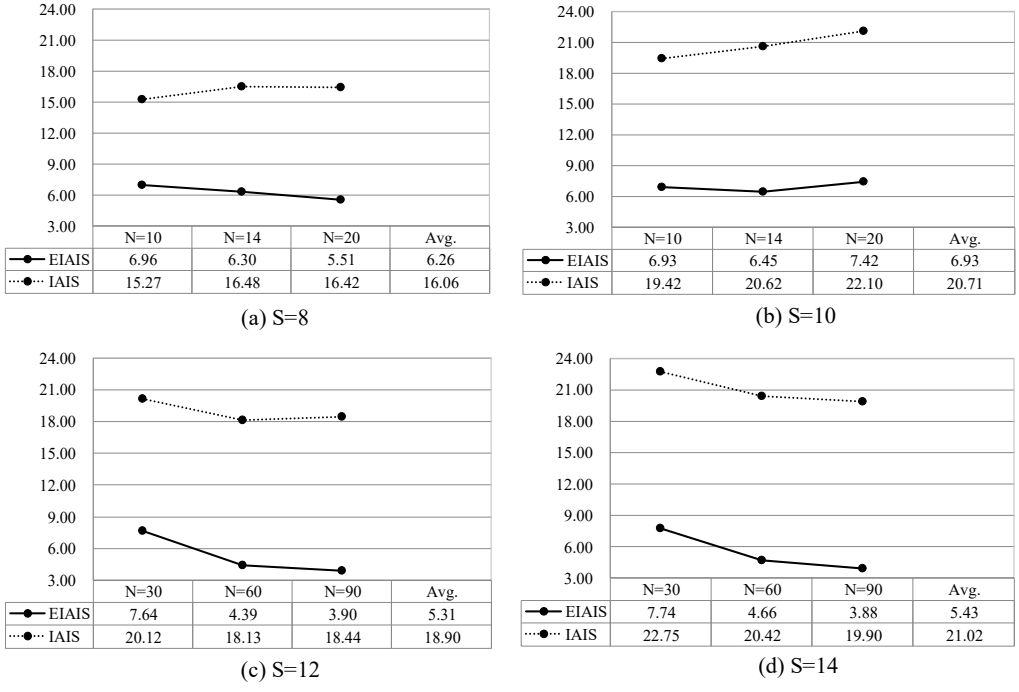

Fig. 12. *RDI* values of EIAIS and IAIS algorithms with different number of employees

For the large size problem, the average *RDI* values for EIAIS and IAIS algorithms are 5.31 and 18.90 when $S=12$, 5.43 and 21.02 when *S=14*. Specifically, the average *RDI* values of the EIAIS for *N=30*, 60, and 90 are 7.64, 4.39, and 3.90 when *S=12*, while the corresponding values of the IAIS are 20.12, 18.13, and 18.44. When *S=14*, the average *RDI* values

of the EIAIS for *N=30*, 60, and 90 are 7.74, 4.66, and 3.88, while the corresponding values of the IAIS are 22.75, 20.42, and 19.90. Thus, the results in Fig. 12 show that regardless of the number of employees and the size of the problem instance, the EIAIS with novel components has a lower average *RDI* value than the IAIS with original. Furthermore, the average *RDI* values of EIAIS and IAIS increase as the number of employees increases, regardless of the size of the problem.

5.3 Experiments with the metaheuristic algorithms

In order to evaluate the proposed EIAIS algorithm, four existing algorithms are selected as comparisons, which are particle swarm optimization (PSO) proposed by Kumar and Vidyarthi (2016), genetic algorithm (GA) by Nemati-Lafmejani, Davari-Ardakani, and Najafzad (2019), simulated annealing (SA) by Tirkolaee et al. (2019), and variable neighborhood search (VNS) used by Cui et al. (2021). Based on the problem instances generated in the previous subsection, the EIAIS, PSO, VNS, GA, and SA algorithms are run ten times for each problem instance. The RDI value of these algorithms is calculated as follows.

$$RDI = \frac{C_{\max}^{H'} - C_{\max}^{best}}{C_{\max}^{best}} \times 100 \tag{18}$$

where $C_{\max}^{H'}$ represents the makespan computed using EIAIS, PSO, VNS, GA, or SA and $C_{\max}^{best}$ represents the optimal solution obtained by these algorithms. The computational results presented in Tables 6 and 7 show that the EIAIS algorithm outperforms other metaheuristic algorithms as the average *RDI* values of EIAIS, PSO, VNS, GA, and SA are 5.98, 14.06, 14.31, 18.10, and 19.68, respectively.

Table 6
RDI values of the metaheuristic algorithms for the small size problems

N	M	K	$S=8$					$S=10$				
			EIAIS	PSO	VNS	GA	SA	EIAIS	PSO	VNS	GA	SA
10	2	3	5.36	9.85	9.96	15.69	19.51	7.12	12.25	12.77	19.03	24.96
		4	8.22	8.28	11.39	22.70	22.89	8.01	12.15	13.62	23.49	26.62
	3	3	5.72	9.44	12.62	19.84	22.85	6.05	9.55	13.42	22.12	25.89
		4	8.54	8.80	14.07	20.77	23.16	6.52	11.41	14..36	20.37	25.63
		Ave.	6.96	9.09	12.01	19.75	22.10	6.93	11.34	13.27	21.25	25.78
14	2	3	7.64	13.56	13.04	19.74	21.39	8.77	13.01	12.89	20.58	23.33
		4	6.80	8.05	11.28	23.45	17.90	4.48	12.18	12.49	23.07	23.77
	3	3	5.22	7.86	10.52	19.53	19.34	6.19	12.38	13.83	21.45	25.49
		4	5.52	6.78	10.80	17.62	17.87	6.36	10.70	12.92	20.84	21.78
		Ave.	6.30	9.06	11.41	20.09	19.13	6.45	12.07	13.03	21.49	23.59
20	2	3	5.60	9.45	8.90	15.34	15.52	8.57	13.40	10.88	22.19	22.00
		4	6.53	8.93	13.61	23.85	20.33	7.26	12.81	9.84	26.22	23.60
	3	3	5.49	9.12	12.07	19.43	18.98	6.85	12.13	12.82	23.62	26.06
		4	4.41	6.97	12.73	17.33	16.58	7.01	11.11	11.81	18.53	21.60
		Ave.	5.51	8.62	11.83	18.99	17.85	7.42	12.36	11.34	22.64	23.32
Agg.			6.25	8.92	11.75	19.61	19.69	6.93	11.92	12.55	21.79	24.23

Table 7
RDI values of the metaheuristic algorithms for the large size problems

N	M	K	$S=12$					$S=14$				
			EIAIS	PSO	VNS	GA	SA	EIAIS	PSO	VNS	GA	SA
30	3	4	7.47	15.40	13.95	13.23	14.54	6.56	14.37	13.62	13.34	16.53
		6	8.60	15.57	11.48	15.69	14.94	9.46	18.91	14.33	22.16	19.61
	4	4	7.26	9.82	16.52	26.22	20.13	7.11	15.07	15.81	22.25	24.30
		6	7.24	13.94	16.33	26.93	22.44	7.82	14.35	16.72	28.67	23.65
		Ave.	7.64	13.68	14.57	20.52	18.01	7.74	15.68	15.12	21.61	21.02
60	3	4	4.74	18.11	14.85	5.85	11.06	4.01	18.87	12.34	12.30	14.79
		6	3.55	16.69	12.06	6.79	9.00	5.46	20.32	15.05	16.45	16.85
	4	4	5.02	18.93	17.96	8.82	14.70	4.03	16.38	18.05	19.99	22.13
		6	4.26	17.38	26.54	22.75	20.30	5.12	16.95	18.91	24.68	23.07
		Ave.	4.39	17.78	17.85	11.05	13.77	4.66	18.13	16.09	18.36	19.21
90	3	4	3.32	18.98	14.98	4.58	10.44	4.39	22.20	15.67	8.32	14.49
		6	3.42	18.21	14.11	6.08	10.53	4.43	21.59	14.89	11.34	14.00
	4	4	4.74	21.69	20.13	11.43	20.43	2.31	19.88	19.18	11.31	18.95
		6	4.13	20.44	20.89	14.68	19.43	4.37	20.70	21.02	17.92	21.25
		Ave.	3.90	19.83	17.53	9.19	15.21	3.88	21.09	17.69	12.22	17.17
Agg.			5.31	17.10	16.65	13.86	15.66	5.42	18.30	16.30	17.39	19.14

From Table 6 for the problem instances with $N=\{10,14,20\}$, the average *RDI* values of EIAIS, PSO, VNS, GA, and SA are 6.25, 8.92, 11.75, 19.61, and 19.69 when $S=8$, 6.93, 11.92, 12.55, 21.79, and 24.23 when *S=10*. From Table 7 for the problem instances with $N=\{30,60,90\}$, the average *RDI* values of EIAIS, PSO, VNS, GA, and SA are 5.31, 17.10, 16.65, 13.86, and 15.66 when *S=12*, 5.42, 18.30, 16.30, 17.39, and 19.14 when *S=14*. Thus, EIAIS has a smaller average *RDI*

value than PSO, VNS, GA, and SA, regardless of the problem size and the number of employees. Meanwhile, for the same problem size, the difference between the average *RDI* values of EIAIS and the other metaheuristic algorithms increases slightly as the number of employees increases. In other words, EIAIS performs much better than PSO, VNS, GA, and SA when many more resources are devoted to performing project activities.

In conclusion, the EIAIS algorithm with new encoding and decoding schemes and novel components is relatively more effective in solving the MM-MSPSP-PC in the same computational time than existing metaheuristic algorithms such as PSO, VNS, GA, and SA.

6. Conclusion

In this paper, a new project scheduling problem involving multi-mode, multi-skill, and differentiated professional capabilities has been studied to minimize the project duration under resource constraints. Project activities must be performed by employees with specific skill types at a required minimum level. A mixed integer programming model is developed to formulate the considered problem. Since the established model is NP-hard, an efficient immunoglobulin-based artificial immune system algorithm with a new encoding and decoding scheme and novel components is presented to solve the model. In the absence of standard benchmark instances, all test problems are randomly generated to evaluate the performance of the proposed algorithm. The IAIS algorithm with the original components, such as inverse mutation, pairwise swap, and insertion mutation, is used to verify the effectiveness of the novel components developed in the proposed algorithm. Meanwhile, existing particle swarm optimization, variable neighborhood search, genetic algorithm, and simulated annealing are used for comparison. The computational results show that the proposed algorithm outperforms the comparison algorithms on the test instances.

This paper has considered a multi-mode multi-skill project scheduling problem with differentiated professional capabilities, which is common in practice. During the execution of project activities, human resources may have different availability constraints, such as planned holidays, training, labor laws in different countries, etc. Further research will focus on implementing the proposed efficient immunoglobulin-based artificial immune system algorithm for other project scheduling problems with resource availability constraints. Furthermore, multi-mode multi-skill project scheduling problems with differentiated professional capabilities in a multi-objective situation are also worth pursuing.

Acknowledgments

This work was supported by the Humanities and Social Science Foundation of Ministry of Education of China [Grant number 20YJA630028].

Data availability statement

The numerical data for this article will be available from the corresponding author, [Tsuiping Chung], upon reasonable request.

References

Alba, E., & Francisco Chicano, J. (2007). Software project management with GAs. *Information Sciences, 177*(11), 2380-2401.

Barz, C., & Kolisch, R. (2014). Hierarchical multi-skill resource assignment in the telecommunications industry. *Production and Operations Management, 23*(3), 489-503.

Bellenguez-Morineau, O., & Néron, E. (2007). A Branch-and-Bound method for solving Multi-Skill Project Scheduling Problem. *RAIRO - Operations Research, 41*(2), 155-170.

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

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

Chen, J. C., Chen, Y.-Y., Chen, T.-L., & Lin, Y.-H. (2022). Multi-project scheduling with multi-skilled workforce assignment considering uncertainty and learning effect for large-scale equipment manufacturer. *Computers & Industrial Engineering, 169*, 108240.

Chen, R., Liang, C., Gu, D., & Leung, J. Y. T. (2017). A multi-objective model for multi-project scheduling and multi-skilled staff assignment for IT product development considering competency evolution. *International Journal of Production Research, 55*(21), 6207-6234.

Chung, T.-P., & Chen, F. (2019). A complete immunoglobulin-based artificial immune system algorithm for two-stage assembly flowshop scheduling problem with part splitting and distinct due windows. *International Journal of Production Research, 57*(10), 3219-3237.

Chung, T.-P., & Liao, C.-J. (2013). An immunoglobulin-based artificial immune system for solving the hybrid flow shop problem. *Applied Soft Computing, 13*(8), 3729-3736.

Cui, L., Liu, X., Lu, S., & Jia, Z. (2021). A variable neighborhood search approach for the resource-constrained multi-project collaborative scheduling problem. *Applied Soft Computing, 107*, 107480.

Drezet, L. E., & Billaut, J. C. (2008). A project scheduling problem with labour constraints and time-dependent activities requirements. *International Journal of Production Economics, 112*(1), 217-225.

García-Nieves, J. D., Ponz-Tienda, J. L., Ospina-Alvarado, A., & Bonilla-Palacios, M. (2019). Multipurpose linear programming optimization model for repetitive activities scheduling in construction projects. *Automation in construction, 105*, 102799.

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

Hartmann, S., & Briskorn, D. (2022). An updated survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operational Research, 297*(1), 1-14.

Kerkhove, L. P., & Vanhoucke, M. (2017). Optimised scheduling for weather sensitive offshore construction projects. *Omega, 66*, 58-78.

Kumar, N., & Vidyarthi, D. P. (2016). A model for resource-constrained project scheduling using adaptive PSO. *Soft Computing, 20*(4), 1565-1580.

Li, C., Wang, F., Gupta, J. N., & Chung, T. (2022). Scheduling Identical Parallel Batch Processing Machines Involving Incompatible Families with Different Job Sizes and Capacity Constraints. *Computers & Industrial Engineering*, 108115.

Li, Q., Jiang, M., Tao, S., Hao, J., & Chong, H.-Y. (2023). The Integrated Problem of Construction Project Scheduling and Multiskilled Staff Assignment with Learning Effect. *Journal of Construction Engineering and Management, 149*(8), 04023064.

Li, Q., Sun, Q., Tao, S., & Gao, X. (2020). Multi-skill project scheduling with skill evolution and cooperation effectiveness. *Engineering, Construction and Architectural Management, 27*(8), 2023-2045.

Lin, J., Zhu, L., & Gao, K. (2020). A genetic programming hyper-heuristic approach for the multi-skill resource constrained project scheduling problem. *Expert Systems with Applications, 140*, 112915.

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

Maghsoudlou, H., Afshar-Nadjafi, B., & Niaki, S. T. A. (2016). A multi-objective invasive weeds optimization algorithm for solving multi-skill multi-mode resource constrained project scheduling problem. *Computers & Chemical Engineering, 88*, 157-169.

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

Myszkowski, P. B., Olech, Ł. P., Laszczyk, M., & Skowroński, M. E. (2018). Hybrid differential evolution and greedy algorithm (DEGR) for solving multi-skill resource-constrained project scheduling problem. *Applied Soft Computing, 62*, 1-14.

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

Nemati-Lafmejani, R., Davari-Ardakani, H., & Najafzad, H. (2019). Multi-mode resource constrained project scheduling and contractor selection: Mathematical formulation and metaheuristic algorithms. *Applied Soft Computing, 81*, 105533.

Néron, E., & Baptista, D. (2002). *Heuristics for the multi-skill project scheduling problem.* Paper presented at the International Symposium on Combinatorial Optimization (CO'2002).

Polancos, R. V., & Seva, R. R. (2023). *A Cost Minimization Model for a Multi-mode, Multi-skilled, Resource-constrained Project Scheduling Problem with Discrete Time-cost-quality-risk Trade-off.* Paper presented at the Proceedings of the 2023 10th International Conference on Industrial Engineering and Applications.

Pritsker, A. A. B., Waiters, L. J., & Wolfe, P. M. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management science, 16*(1), 93-108.

Talbot, F. B. (1982). Resource-constrained project scheduling with time-resource tradeoffs: The nonpreemptive case. *Management science, 28*(10), 1197-1210.

Tian, Y., Xiong, T., Liu, Z., Mei, Y., & Wan, L. (2022). Multi-Objective multi-skill resource-constrained project scheduling problem with skill switches: Model and evolutionary approaches. *Computers & Industrial Engineering, 167*, 107897.

Tirkolaee, E. B., Goli, A., Hematian, M., Sangaiah, A. K., & Han, T. (2019). Multi-objective multi-mode resource constrained project scheduling problem using Pareto-based algorithms. *Computing, 101*(6), 547-570.

Toroslu, I. H. (2003). Personnel assignment problem with hierarchical ordering constraints. *Computers & Industrial Engineering, 45*(3), 493-510.

Yannibelli, V., & Amandi, A. (2011). A knowledge-based evolutionary assistant to software development project scheduling. *Expert Systems with Applications, 38*(7), 8403-8413.

Zheng, H.-y., Wang, L., & Zheng, X.-l. (2017). Teaching–learning-based optimization algorithm for multi-skill resource constrained project scheduling problem. *Soft Computing, 21*, 1537-1548.

Zhu, L., Lin, J., Li, Y.-Y., & Wang, Z.-J. (2021). A decomposition-based multi-objective genetic programming hyper-heuristic approach for the multi-skill resource constrained project scheduling problem. *Knowledge-Based Systems, 225*, 107099.