Bulletin of Electrical Engineering and Informatics
Vol. 15, No. 6, December 2026, pp. 5240~5253
ISSN: 2302-9285, DOI: 10.11591/eei.v15i6.12147

ET-RCPSP: a novel RCPSP variant with earliness-tardiness penalties solved by enhanced evolutionary algorithm

Hao Nguyen Thi[1,2], Loc Nguyen The[1], Huu Dang Quoc[3]
[1]Faculty of Information Technology, School of Mathematics and Computer Science, Hanoi National University of Education, Hanoi, Vietnam
[2]Faculty of Engineering Technology, Hung Vuong University, Phu Tho, Vietnam
[3]Faculty of Applied Digital Technology, Thuongmai University, Hanoi, Vietnam

Article Info

Article history:

Received Dec 16, 2025
Revised Sep 9, 2026
Accepted Oct 4, 2026

Keywords:

Earliness-tardiness
Evolutionary algorithm
Multi-skill resource-constrained project scheduling problem
NP-hard problem
Project scheduling

ABSTRACT

The resource-constrained project scheduling problem (RCPSP) and its multi-skill variant (MS-RCPSP) are widely applied in manufacturing, logistics, construction, and software project management. Existing research predominantly focuses on minimizing total project completion time, with limited consideration of practical costs incurred when tasks finish earlier or later than their specified due dates. This study introduces the earliness-tardiness RCPSP (ET-RCPSP), which incorporates task-specific earliness-tardiness penalties to achieve more realistic schedule efficiency. To address this NP-hard problem, we propose an enhanced earliness-tardiness genetic algorithm (ET-GA) with two novel mechanisms: i) priority resource allocation for independent tasks to reduce early-phase congestion and ii) elitism preservation to enhance convergence and stability. Experimental results on the iMOPSE benchmark demonstrate that ET-GA achieves optimal or near-optimal solutions on 90% of test instances, with average earliness-tardiness penalties reduced by 98.71%, 99.24%, and 99.19% compared to differential evolution (DE), standard genetic algorithm (GA), and particle swarm optimization (PSO), respectively. ET-GA exhibits exceptional consistency, with 40% perfect reproducibility (where the best result equals the average), while baseline algorithms achieve 0%. Statistical tests confirm substantial improvements (Wilcoxon signed-rank, $p<0.000001$), demonstrating the method's effectiveness for practical multi-skill project scheduling with earliness-tardiness considerations.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Huu Dang Quoc
Faculty of Applied Digital Technology, Thuong Mai University
79 Ho Tung Mau. ST, Tu Liem, Hanoi, Vietnam
Email: huudq@tmu.edu.vn

1. INTRODUCTION

Project scheduling is a critical task in modern organizational management, encompassing diverse domains such as manufacturing, logistics, construction, and software development. The multi-skill resource-constrained project scheduling problem (MS-RCPSP) [1], [2] has emerged as an essential optimization framework for coordinating complex projects in which tasks must be scheduled under resource constraints and precedence constraints. These problems aim to determine the optimal allocation of resources and the timing of task execution to achieve project objectives while adhering to specified constraints.

Traditional research on scheduling problems has predominantly focused on minimizing the total completion time of all tasks (makespan). While makespan minimization is crucial, it often overlooks the practical cost implications of tasks finishing earlier or later than their specified deadlines. In many practical

project environments, completing tasks significantly earlier than their due dates may result in inventory holding costs or idle resource utilization. At the same time, late completion can lead to contractual penalties, supply chain disruptions, or customer dissatisfaction. The lack of explicit consideration of these earliness-tardiness costs in scheduling models limits their applicability in real-world scheduling environments, as completing in the shortest time is insufficient—adherence to deadlines demonstrates credibility and professionalism. Therefore, explicitly incorporating earliness–tardiness penalties into scheduling models is essential for accurately evaluating schedule performance in real-world project management. Despite the growing body of research on RCPSP and MS-RCPSP, the explicit integration of earliness–tardiness penalties within multi-skill project scheduling frameworks remains limited in the literature.

To address this limitation, this study introduces an extended variant of MS-RCPSP that incorporates task-specific earliness-tardiness penalty costs. The proposed problem formulation provides a more realistic representation of schedule performance by ensuring that execution times meet deadlines as precisely as possible. Given the NP-hard nature of RCPSP and its variants [3]–[5], exact optimization methods, such as modern constraint-programming solvers like CP-SAT [6], become computationally infeasible for practical-scale problems. Consequently, metaheuristic approaches [7], [8] have emerged as viable alternatives, capable of identifying near-optimal solutions within reasonable computational time. Among these, genetic algorithms (GA), a well-established class of evolutionary computation techniques [9], have demonstrated particular effectiveness due to their population-based search mechanisms and ability to explore diverse solution spaces. This study proposes an enhanced GA that integrates two novel mechanisms to improve solution quality and convergence behavior. First, a priority-based resource allocation strategy is introduced to prioritize resource assignment to independent tasks in early project phases, thereby reducing resource contention and improving schedule flexibility. Second, an elitism-preservation mechanism with roulette-wheel selection is implemented to retain high-quality individuals from previous generations, thereby enhancing convergence stability and preventing the loss of promising solutions during the evolutionary process.

The proposed algorithm is evaluated on the iMOPSE benchmark dataset [10], which provides standardized test instances for algorithm comparison. This evaluation strategy enables both controlled performance assessment and validation of practical applicability. Experimental results demonstrate that the proposed method generates schedules with reduced makespan and lower total earliness-tardiness penalties compared to baseline algorithms, including standard GA, differential evolution (DE), and particle swarm optimization (PSO). These findings affirm the effectiveness and practical potential of the proposed approach in solving the multi-skill project scheduling problems with realistic cost considerations. The contributions of this research extend both the theoretical understanding of MS-RCPSP variants and provide valuable insights for practitioners facing complex resource allocation decisions in time-sensitive project environments.

The main contributions of this research are: i) a comprehensive mathematical formulation for ET-RCPSP in multi-skill resource environments; ii) the proposed earliness-tardiness genetic algorithm (ET-GA), which combines GA with resource assignment techniques for tasks without predecessors and elitism preservation from previous generations; and iii) analysis and evaluation of experimental results on the iMOPSE benchmark dataset to demonstrate the algorithm's effectiveness for ET-RCPSP.

The remainder of this paper is organized as follows: section 2 presents a comprehensive literature review on ET-RCPSP and related scheduling problems. Section 3 presents the mathematical formulation of ET-RCPSP. Section 4 details the proposed ET-GA. Section 5 presents experimental results, performance evaluation, and statistical analysis of ET-GA. Section 6 concludes the research and outlines future directions for research.

2. RELATED WORKS

The RCPSP [1], [2] has been proven to be NP-hard. The traditional objective is to minimize project completion time subject to precedence constraints among tasks and resource availability. However, in modern manufacturing practice, particularly with the development of just-in-time (JIT) production systems, optimization objectives have expanded beyond mere time reduction to include cost minimization and ensuring on-time completion.

Shahabi-Shahmiri *et al.* [11] proposes a multi-objective MILP model for a multi-skill, multi-mode RCPSC problem under uncertainty, incorporating sustainability considerations such as job creation, energy consumption, and resource costs. The model is solved using an AUGMECON2-VIKOR-based approach. Kramer and Subramanian [12] developed a general heuristic algorithm applicable to various scheduling problem variants with earliness-tardiness considerations, providing a detailed annotated bibliography and demonstrating effectiveness across hundreds of instances. Otten *et al.* [13] demonstrated the feasibility of early-late optimization in surgical scheduling using the SVF rule, balancing resource waste (early completion) and service quality (late completion). Hoffmann *et al.* [14] developed an earliness-tardiness optimization model for COSP, emphasizing the importance of controlling delivery time in food and

pharmaceutical industries due to product perishability. These studies have not yet considered the characteristics of multi-skilled resources in modern production environments.

MS-RCPSP [1], [4] extends traditional RCPSP by considering resources with multiple skills and tasks that require specific skill sets. This model more accurately reflects modern manufacturing environments and has attracted substantial research attention in recent years. Hosseinian and Baradaran [15] developed a multi-mode MS-RCPSP variant and employed an evolutionary algorithm combined with multi-attribute decision-making methods to reduce project duration. Tirkolaee *et al.* [16] proposes a nonlinear model and two metaheuristic solution approaches to efficiently solve the multi-objective multi-mode RCPSP with payment planning. Myszkowski *et al.* [10] developed the iMOPSE library, which supports bi-objective optimization (cost and time) for MS-RCPSP; this was subsequently extended by Myszkowski and Laszczyk [17] to many-objective optimization, incorporating five comprehensive evaluation criteria on the same dataset.

Recent developments over the past three years have introduced several novel approaches. Povéda *et al.* [18] investigated MS-RCPSP with partial preemption, integrating practical constraints such as generalized precedence relations and resource calendars, with successful application in aerospace manufacturing. Antkiewicz *et al.* [19] evaluated specialized genetic operators for NSGA2, demonstrating their effectiveness when designed to match problem-specific characteristics. Snauwaert and Vanhoucke [20] further proposed a solution framework for multi-skilled project scheduling problems with hierarchical skills, extending the classical multi-skill setting to capture skill-level dependencies that better reflect real-world workforce structures.

Additionally, several studies have explored hybrid approaches for complex project scheduling problems. Torba *et al.* [21] proposed a memetic algorithm combining GA and simulated annealing for multi-project scheduling with multi-skilled resources. Vermeire and Vanhoucke [22] introduced a priority-rule-based heuristic with a parallel schedule-generation scheme, achieving competitive results on the MSLIB dataset. Myszkowski *et al.* [23] proposed a DE algorithm combined with a greedy strategy to solve the MS-RCPSP and improve search efficiency. Chen *et al.* [24] developed a hybrid GA with heuristic rules for the multi-mode resource-constrained multi-project scheduling problem. Pellerin *et al.* [25] provided a survey of hybrid metaheuristics for RCPSP.

Recent research on MS-RCPSP mainly focuses on three directions: extending models to capture real-world constraints better, developing specialized metaheuristics for multi-skill scheduling problems, and applying these approaches to large-scale industrial applications. However, most existing studies still emphasize objectives such as makespan or cost optimization. In contrast, objective functions incorporating earliness–tardiness penalties remain largely underexplored, despite their high practical relevance in JIT production environments. In particular, the explicit integration of task-level earliness–tardiness penalties within a multi-skill RCPSP framework has received limited attention in the literature. To address this gap, this study introduces a novel ET-RCPSP formulation that incorporates earliness–tardiness considerations into multi-skill project scheduling. To efficiently solve the proposed problem, an ET-GA is developed that integrates priority-based resource allocation and elitism preservation to improve search efficiency and solution quality.

Deep reinforcement learning (DRL) offers advantages in dynamic environments or scenarios requiring continuous rescheduling. Park *et al.* [26] proposed a GNN-based reinforcement learning approach that learns graph representations of scheduling states and optimizes scheduling policies via proximal policy optimization (PPO). Lei *et al.* [27] extended DRL to flexible job-shop scheduling within a multi-action framework, demonstrating deep learning's ability to handle complex decision spaces. Recent studies have explored DRL for scheduling problems using Markov decision process formulations, with models that combine graph neural networks and PPO achieving competitive results on RCPSP [28]. However, DRL still exhibits significant limitations, including the need for large training datasets, high computational costs, difficulty generalizing to skill-based or nonlinear multi-objective variants, and a lack of interpretability.

Given the objective of developing the novel ET-RCPSP variant and establishing a benchmark for the research community, this study prioritizes metaheuristic methods for their simplicity, transparency, reproducibility, low computational cost, and suitability for practical deployment. This approach also generates a reliable baseline for comparing and evaluating more complex methods in future work. DRL will be considered in subsequent research once standardized datasets and stable baselines are established.

3. PROBLEM FORMULATION

In the ET-RCPSP problem, a project consists of a finite set of tasks, each requiring one or more specific skills with predefined proficiency levels. Tasks may be subject to precedence constraints, where

predecessor tasks must be completed before their corresponding successors can begin. A task may have multiple predecessors and successors depending on the project structure.

Each resource is defined by a set of skills characterized by skill type and skill level and may possess multiple skills. At any time, a resource can handle only one task. A resource is eligible to perform a task when it has the required skill type and a skill level that meets or exceeds the task's requirement. Resources are reusable; once a task is finished, they can be allocated to another suitable task, but never concurrently.

The ET-RCPSP problem aims to minimize total earliness and tardiness simultaneously. Its objective is to determine a feasible schedule that ensures the effective allocation of multi-skilled resources while maintaining optimal adherence to task deadlines. Earliness occurs when the actual completion time of a task or project precedes its designated due date, whereas tardiness occurs when the completion time exceeds the assigned deadline. Excessive earliness may lead to unnecessary inventory holding costs, resource underutilization, or imbalances across workflow segments. Conversely, tardiness can result in contractual penalties, disputes, cascading delays throughout dependent project components, and reputational risk for the organization. Consequently, minimizing overall earliness and tardiness in ET-RCPSP is not only a technical requirement but also holds substantial practical significance for modern complex scheduling systems. The ET-RCPSP formulation inherits all constraints from the classical MS-RCPSP but differs in its objective function.

The mathematical model of the ET-RCPSP problem is formulated as follows: input: a set of multi-skilled resources, a set of tasks, and associated constraints. Output: a feasible schedule T such that resources execute all tasks in accordance with the constraints and the total earliness and tardiness penalties of the entire project are minimized.

Notation: $MA=\{Ma_1, Ma_2, ..., Ma_{mres}\}$: set of *mres* resources in the project, where Ma_k; $TA=\{Ta_1, Ta_2, ..., Ta_{ntas}\}$: set of *ntas* tasks in the project, where Ta_m denotes the m^{th} task; Fat_m: set of tasks that must be completed before task Ta_m can start; Ta^k: set of tasks that can be executed by resource Ma_k, where $Ta_k \subseteq TA$; dur_m: duration required to execute task Ta_m; $star_m$, fis_m: start time and finish time of task Ta_m; d_m: due date of task Ta_m; Ask^m: set of skills required by task m; CAP: set of skills in the project; Cap_j: skill j; Cap^k: set of skills possessed by resource k, $Cap^k \subseteq CAP$; typ_j, deg_j: type and proficiency level of skill j, respectively; $A_{m,k}{}^t$: binary variable taking values in {0, 1}, where $A_{m,k}{}^t$=1 if resource k is executing task m at time t, and 0 otherwise; T: feasible schedule; L_m, S_m: tardiness and earliness of task Ta_m, respectively, defined as (1):

$$L_m = max(0, fis_m - d_m),\ S_m = max(0, d_m - fis_m) \quad (1)$$

w_m^S, w_m^L: weights (or penalty costs) for earliness and tardiness of task Ta_m, respectively;

$Fit(T)$: objective function of the problem, representing the total earliness and tardiness penalties of all tasks in schedule T, computed as (2):

$$Fit(T) = \sum_{m=1}^{ntas}(w_m^S \times S_m + w_m^L \times L_m) \quad (2)$$

In practice, the values of the two weights w_m^S and w_m^L in (2) depend on the project's operational or contractual requirements. Certain project types only prohibit late completion, treating early and on-time completion as equivalent; in such cases, w_m^L=1 and w_m^S=0. Conversely, other projects only penalize early completion, with no distinction between late and on-time completion; thus, w_m^L=0 and w_m^S=1. In this study, we set w_m^L=w_m^S=1, as this configuration reflects the prevailing project category in practice, where deviations from the planned completion time, whether early or late, are undesirable. Therefore, the objective function becomes:

$$Fit(T) = \sum_{m=1}^{ntas}(S_m + L_m) \quad (2^*)$$

The objective of the ET-RCPSP problem is to find a schedule T such that: $Fit(T) \rightarrow min$. Schedule T must satisfy the following constraints:

$$Cap^k \neq \emptyset\ \forall Ma_k \in MA \quad (3)$$

$$dur_m \geq 0\ \forall Ta_m \in TA \quad (4)$$

$$fis_m \geq 0\ \forall\ Ta_m \in TA \quad (5)$$

$$fis_m \leq fis_n - dur_n \geq 0, \forall Ta_m \in TA, m \neq 1, Ta_m\ \in Fat_m \quad (6)$$

$$\forall Ta_m \in Ta^k, \exists Ask^m \in Cap^k : typ_{Cap^k} = typ_{Ask^m} \;\&\&\; deg_{Cap^k} \geq deg_{Ask^m} \quad (7)$$

$$\forall Ma_k \in MA, \forall p \in dur_m : \sum_{m=1}^{ntas} A_{m,k}^p \leq 1 \quad (8)$$

$$\forall Ta_m \in TA, \exists! \, p \in [0, dur_m], ! \, Ma_k \in MA : A_{m,k}^p = 1, where \; A_{i,k}^p \in \{0,1\} \quad (9)$$

The meanings of the above constraints are as follows: constraint (3) guarantees that each resource participating in the project is assigned a valid, non-empty skill set. Constraints (4) and (5) enforce non-negative task durations and completion times. Zero duration is allowed to represent dummy tasks for project start and end milestones. Constraint (6) enforces precedence relationships between tasks. This constraint establishes $fis_m \leq fis_n - dur_n$, for all precedence pairs *(m, n)*. Constraint (7) enforces skill compatibility: a resource assigned to task Ta_m must possess at least one skill of the required type and with a proficiency level not lower than the task requirement. Constraint (8) ensures resource non-preemption, limiting each resource to at most one task assignment at any time point. Constraint (9) enforces that every task is allocated to exactly one resource for a minimum of one time unit within its processing interval. Thus, the ET-RCPSP solution consists of determining a feasible schedule *T* that satisfies constraints (3)-(9) and $Fit(T) \longrightarrow min$.

Example 1: a website development project consists of four tasks, *TA*={Ta_1–interface design, Ta_2–frontend implementation, Ta_3–data processing, and Ta_4–database integration}. The project is executed by three resources, *MA*={Ma_1, Ma_2, Ma_3}. The skill set is defined as *CAP*={Cap_1–Designer, Cap_2–Frontend, Cap_3–Backend, and Cap_4–Developer}. Information about the resources, tasks, and constraints is provided in Table 1.

Table 1. Project information

Information about tasks in the project				
Task	Processing time	Predecessor task(s)	Required skills	Due date (weeks)
Ta_1	2		$(Cap_1, 2)$	2
Ta_2	3	Ta_1	$(Cap_2, 2)$	5
Ta_3	2	Ta_1	$(Cap_3, 3)$	4
Ta_4	4	Ta_2, Ta_3	$(Cap_2, 3)$, $(Cap_3, 3)$	8

Information about resources in the project	
Resource	Resource skills
Ma_1	$(Cap_1,3)$
Ma_2	$(Cap_2, 3)$, $(Cap_4, 4)$
Ma_3	$(Cap_2, 3)$, $(Cap_3, 3)$, $(Cap_4, 3)$

Based on the information in Table 1, a feasible schedule *T* can be constructed as illustrated in Figure 1. The resulting schedule yields an objective value of 0+0+0+1=1 week, since Ta_1, Ta_2, and Ta_3 meet their respective due dates, whereas Ta_4 is completed later than its assigned deadline.

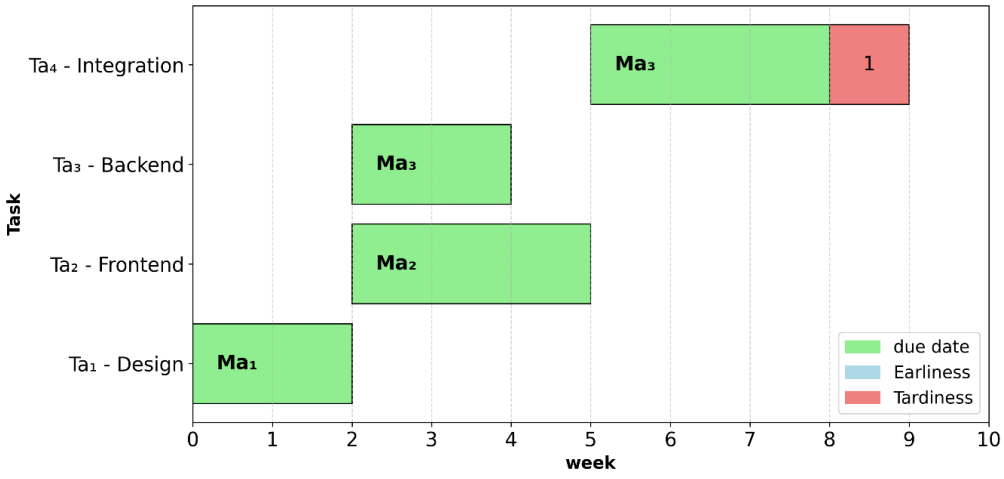


Figure 1. Feasible schedule *T*

4. PROPOSED ALGORITHM

In this study, we introduce the ET-GA, which embeds a priority-driven resource assignment mechanism for ready tasks throughout the evolutionary process, covering population initialization, crossover, and mutation. Furthermore, an elitism strategy is incorporated to ensure that elite individuals are retained across successive generations.

4.1. Priority technique for ready tasks

The ready task priority (RTP) technique is a priority-based resource assignment method for tasks without precedence constraints, which can therefore be scheduled immediately. In scheduling problems in general, and in the ET-RCPSP problem in particular, there are tasks with no precedence constraints or that are not constrained by other tasks during execution; such tasks are referred to as ready tasks. This offers an advantage in the scheduling process because resources can be assigned to these tasks immediately, an effective strategy that reduces both the computational search space and the number of tasks requiring resource allocation.

For the RTP technique, tasks ready for execution are determined before the optimization procedure begins. These tasks are provisionally matched with appropriate resources in advance, assuming that the available resources possess the necessary skill levels to perform them. By allocating resources to tasks that are not subject to precedence relationships during this preliminary stage, the algorithm reduces the complexity of the remaining scheduling decisions. As a result, the search process can concentrate on tasks with more complicated constraints. This separation simplifies the overall problem structure and accelerates convergence by easing schedule formation in the early phase of optimization.

Example 2: consider a project with tasks and precedence constraints as shown in Figure 2. Assume that both resources Ma_1 and Ma_2 satisfy the skill requirements for all tasks in the project. In Figure 2, tasks Ta_1, Ta_2, Ta_3, Ta_{10}, Ta_{11}, and Ta_{14} are identified as ready, since no other tasks constrain them. Therefore, resources with the necessary skill sets can be allocated to these tasks immediately without altering the overall logical structure of the project schedule. This process establishes the start of the schedule, while subsequent phases consider only tasks that have not yet reached the ready state. The RTP technique is described in detail in Algorithm 1.

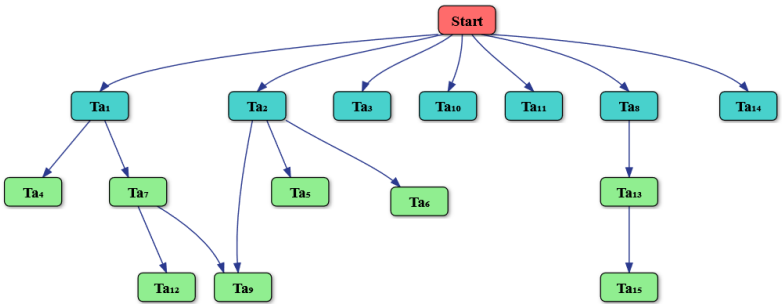


Figure 2. Task information in sample project

Algorithm 1. RTP

```
Function RTP(S)
        Input: S (individual). Output: S' (individual after resource priority assignment)
1:      freeList ← { i | readyTask(i) }; evals ← ∅
2:      for each i ∈ freeList do
3:              (bestRes, obj) ← FindBestResource(S, i) // Identify the optimal resource
4:              evals.add(i, bestRes, obj)
5:      end for
6:      Sort evals by obj ascending
7:      keep ← first k × |freeList| elements // k is the retention ratio
8:      for each (i, bestRes) ∈ keep do
9:              S[i] ← bestRes
10:     end for
11:     return S
```

Where: |freeList| denotes the size of the individual freeList*; readyTask(i)* denotes a function that determines whether task *i* is a ready task*; FindBestResource():* identify the best resource for the task; if one is unused and feasible, select it; otherwise, select the feasible resource with the lowest current usage.

Applying RTP techniques in the initialization phase not only reduces the computational burden of the subsequent optimization process but also improves the quality of the initial solution. Consequently, the convergence rate is accelerated, and the overall quality of the final schedule is enhanced. During the evolution process, RTP techniques are employed to maintain solution quality and prevent resource-load imbalance. Accordingly, the resource–task assignment for ready tasks should emphasize: i) distributing free tasks evenly across eligible resources; ii) prioritizing resources with lower instantaneous workloads; and iii) adopting cost functions that minimize either the average load or the workload standard deviation to guide task allocation.

The objective of the problem is to minimize the total delay of tasks relative to their due dates. Thus, during execution, resources should be assigned to tasks with the nearest due dates by computing the deviation between the actual completion time and the due date of each task, sorting tasks in ascending order of this deviation, and retaining 50% (subject to change) of tasks with the smallest deviations for the evolutionary phase. Example 3: Table 2 illustrates the RTP technique application process.

Table 2. Ready tasks and their due date deviations (left), and the refinement step retaining 50% of tasks with the smallest deviations (right)

Task	Due date	Actual completion time	Resource
Ta_1	20	30	Ma_1
Ta_2	24	28	Ma_2
Ta_3	45	54	Ma_1
Ta_8	26	33	Ma_2
Ta_{10}	60	72	Ma_1
Ta_{11}	80	85	Ma_2
Ta_{14}	72	78	Ma_1

Calculate the deviation →

Task	Deviation
Ta_1	10
Ta_2	4
Ta_3	9
Ta_8	7
Ta_{10}	8
Ta_{11}	5
Ta_{14}	6

Sort the deviation →

Task	Mark
Ta_2	1
Ta_{11}	1
Ta_{14}	1
Ta_8	1
Ta_{10}	0
Ta_3	0
Ta_1	0

In Table 2, tasks Ta_2, Ta_{11}, Ta_{14}, and Ta_8 are retained (marked 1), while the remaining tasks Ta_1, Tas_3, and Ta_{10} are left for later processing (marked 0).

4.2. Elite preservation mechanism

In each evolutionary cycle, after crossover and mutation are performed, the best individuals from the previous generation are preserved and passed directly to the next generation. Their quality is evaluated using the total earliness–tardiness cost, with lower values indicating better solutions. These elite individuals are excluded from genetic operations, which prevents well-performing allocation patterns from being altered or lost due to randomness during evolution. For problems with large solution spaces and complex constraints, such as ET-RCPSP, unintentionally discarding a high-quality solution can be computationally expensive to recover. Therefore, preserving elite individuals plays a crucial role in accelerating convergence and reducing the risk of losing good solutions throughout the evolutionary process. The elitism mechanism is illustrated in Algorithm 2.

Algorithm 2. Elite algorithm

```
Function EliteSelection(population, eliteCount)
       Input: population, eliteCount: number of elite individuals; Output: elite (elite
individual array)
1:     sortedPopulation ← Sort(population) // sort by fitness level (increasing)
2:      elite ← Array[eliteCount]
3:       for i = 0 to eliteCount - 1 do
4:            elite[i] ← Copy(sortedPopulation[i])
5:       end for
6:     return elite
```

By integrating RTP and elitism, the ET-GA preserves high-quality solutions while maintaining the level of diversity required to explore the solution space effectively. This combination enables the algorithm to deliver more consistent performance and enhances its search efficiency compared with traditional evolutionary methods.

4.3. Earliness-tardiness genetic algorithm

The RTP technique and the elitism preservation mechanism are integrated into the proposed evolutionary framework to form the enhanced ET-GA for solving the ET-RCPSP problem. Each candidate solution is encoded as a vector that represents both task ordering and resource assignment. A hybrid encoding

scheme is adopted to represent task precedence constraints and resource allocation decisions simultaneously. During decoding, a feasible schedule is generated that satisfies all constraints, including precedence relationships, resource capacity, skill requirements, and valid task start times. Figure 3 illustrates the overall workflow of the proposed ET-GA, including the stages of population initialization, RTP priority computation, schedule construction, and the evolutionary loop consisting of selection, crossover, mutation, and elitism preservation. Based on this workflow, the complete procedure of the proposed ET-GA is presented in Algorithm 3.

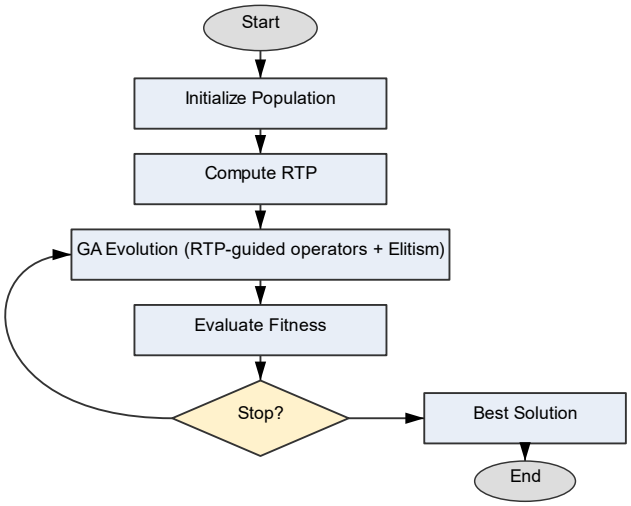


Figure 3. Workflow of the proposed ET-GA

Algorithm 3. ET-GA

```
Function ET-GA(popSize, generations, dataset)
Input: popSize, generations, dataset (ntasks, mres). Output: bestSolution, bestFitness
1: Initialize population
2: Compute RTP priorities for individuals
3: population ← BuildSchedules(population)
4: for gen = 1 to generations do
5:      Evaluate population and update bestSolution, bestFitness
6:      elite ← EliteSelection(population, eliteCount)
7:      while |newPopulation| < popSize do
8:             parent1, parent2 ← RouletteWheelSelection(population)
9:             child ← Crossover(parent1, parent2, RTP, Pc)
10:            child ← Mutation(child, RTP, Pm)
11:            child ← BuildScheduleForIndividual(child)
12:            Add child to newPopulation
13:     end while
14:     population ← elite U newPopulation
15: end for
16: return bestSolution, bestFitness
```

BuildSchedules(): Construct feasible schedules; *EliteSelection()*:Select elite individuals; *Crossover():* RTP-guided single-point crossover (Pc); *Mutation():* RTP-guided random mutation (Pm). *BuildScheduleForIndividual():* Generate a schedule for the offspring individual; *RouletteWheelSelection():*Selects an individual based on fitness probability.

5. PERFORMANCE ANALYSIS

The proposed algorithm for the ET-RCPSP problem was implemented in C# using visual studio 2022. All experiments were conducted on a Windows 10 system with an Intel Core i5 processor (2.4 GHz) and 8 GB of RAM. Four generation settings were tested: 1,000, 3,000, 5,000, and 10,000 generations. The population size was set to 100, with a crossover rate of 0.7 and a mutation rate of 0.1. The elitism retention ratio was fixed at 20% of the population size, and the RTP retention ratio was set to $k = 0.5$. Random seeds were not fixed; each run used an independent, system-generated seed. For each setting, each experiment was repeated 10 times per dataset to ensure the reliability of the results. To assess the performance and stability of the proposed algorithm, we used the iMOPSE dataset [10], a widely recognized benchmark for solving the

MS-RCPSP problem and its variants. The dataset provides standardized instances with different project sizes and resource configurations, enabling fair comparisons between scheduling algorithms. In this study, 30 benchmark instances from the iMOPSE dataset were utilized, comprising projects with 100–200 tasks, 5–40 resources, 22–145 precedence relations, and 9–15 skill types, thereby representing diverse scheduling scenarios and varying levels of problem complexity.

5.1. Results of the earliness-tardiness genetic algorithm

Table 3 summarizes the overall performance of the ET-GA on 30 instances from the iMOPSE dataset. The results show that the number of instances achieving the optimal value (best=0) increases from 21/30 (70%) under the 1000-generation configuration to 27/30 (90%) under the 10000-generation configuration. The average best value decreases substantially from 57.8 to 4.4, indicating a significant improvement in solution quality as the number of generations increases.

Table 3. Summary statistics for ET-GA

Generation	Optimal (best=0)	Avg best	Avg mean	Avg std	Max best	Min best
1000	21/30	57.83	252.06	142.59	713	0
3000	23/30	14.43	169.37	153.88	210	0
5000	24/30	19.43	165.22	125.24	246	0
10000	27/30	4.37	128.1	135.38	55	0

Figure 4 illustrates this improvement trend from two perspectives: the number of instances reaching the optimal solution and the average best value. The experimental results indicate that the GA exhibits markedly different performance across the datasets in the iMOPSE collection. A subset of instances (iM1, iM2, iM4, iM11, iM12, iM26, and iM28) achieves perfect outcomes, with the best, average, and standard deviation metrics all equal to 0 across all generation settings. These instances are relatively simple and can be optimally solved by the algorithm from the early stages of the search. In contrast, several datasets, such as iM16, iM18, iM19, iM20, and iM21, demonstrate substantially greater difficulty, with best values ranging from 0 to 713 and average values exceeding 1400, accompanied by very large standard deviations (300–1200). These results indicate not only poor solution quality but also instability in the search process. Notably, iM5 consistently attains an optimal solution (best=0), yet its average and standard deviation remain extremely high (up to 555.53), reflecting significant population divergence, with most individuals exhibiting very poor fitness.

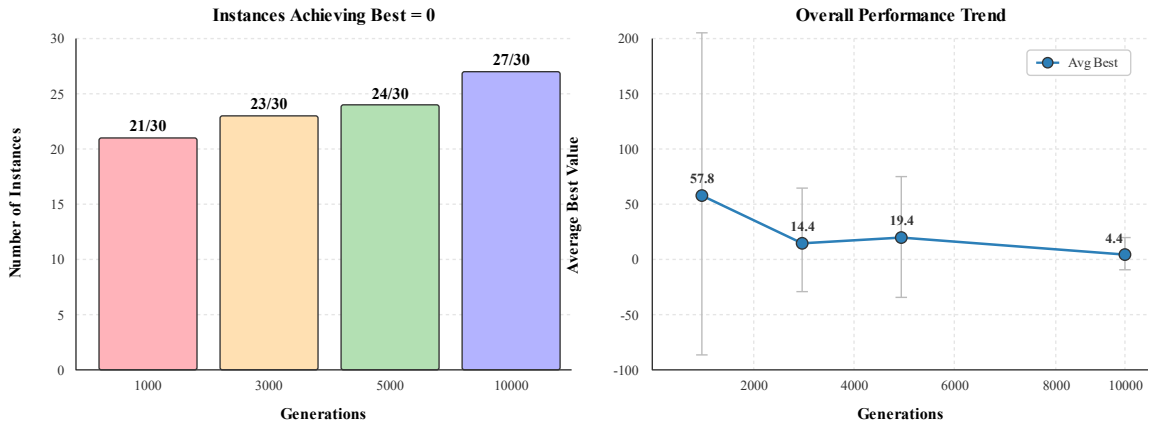


Figure 4. Performance improvement charts

Increasing the number of generations from 1000 to 10000 resulted in a significant improvement for approximately 40% of the datasets, particularly the iM9, iM10, iM15, iM16, and iM25 groups. iM9 and iM16, in particular, reduced their best values from 107 and 256 to 0, respectively, after running 10000 generations. However, the improvement trend was not uniform: while iM9, iM16, and iM25 showed clear benefits with increased generations, some datasets, such as iM5, iM17, and iM18, did not show a clear improvement trend. In terms of stability, approximately 50% of the dataset (including iM1-iM4, iM6, iM7,

iM11-iM14, iM22, iM23, and iM26-iM28) maintained a standard deviation below 50 across all generation counts, demonstrating the algorithm's high stability on these problems.

The ET-GA across four different generation configurations (1000, 3000, 5000, and 10000) provided important insights into its convergence and stability. The results showed a clear improvement trend as the number of generations increased from 1000 to 5000, with the number of instances achieving optimal solutions (best=0) increasing from 21/30 to 24/30 and the average best value decreasing from 57.8 to 19.4. However, increasing to 10,000 generations did not yield a commensurate improvement and even caused instability in some instances (iM5, iM17, and iM19). Therefore, the 5,000-generation configuration was determined to be the optimal choice, striking a good balance between performance and reliability.

5.2. Comparative evaluation of the earliness-tardiness genetic algorithm and baseline algorithms

To evaluate the superiority of the proposed ET-GA, we compared it with three popular metaheuristic algorithms: DE, GA, and PSO, across all 30 instances of the iMOPSE dataset. For a fair comparison, all algorithms used the same population size, maximum number of iterations, and number of independent runs. The algorithm-specific parameters were set as follows: for DE, a self-adaptive scheme was used in which the control parameters F and CR were adapted during the search, with F self-adapted within [0.1, 0.9) and CR within [0, 1); for GA, the crossover rate was 0.7, and the mutation rate was 0.1; and for PSO, the inertia weight w decreased linearly within [0.35, 0.95], with acceleration coefficients $c_1 = 2.0$ and $c_2 = 1.2$. Experimental results showed that ET-GA was absolutely superior in all aspects compared to the other algorithms. Specifically, ET-GA achieved an average mean objective value of only 128.1, significantly lower than DE (9924.9), GA (16923.5), and PSO (15845.8), corresponding to improvements of 98.71%, 99.24%, and 99.19%, respectively. In terms of the number of instances achieving optimal solutions (Best=0), ET-GA solves 27 out of 30 instances (90%), compared with 9/30 (30%) for DE, 11/30 (37%) for GA, and 12/30 (40%) for PSO. Win/Tie/Loss analysis shows ET-GA completely dominating, with a 30/0/0 win rate against all three opponents, demonstrating superior, consistently optimal solution-finding capabilities across all instances. The detailed comparison results of the four algorithms are summarized in Figure 5 and Table 4.

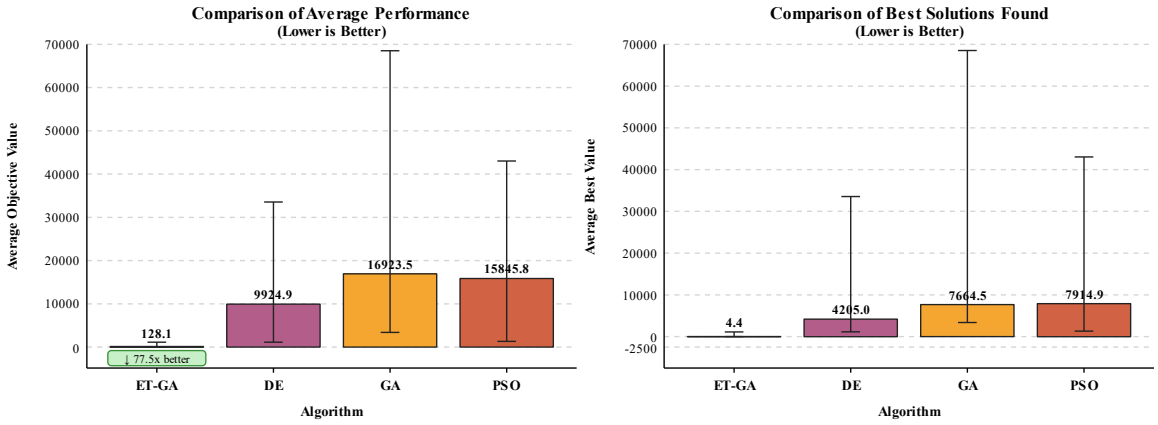


Figure 5. Bar chart comparison

Table 4. Summary comparison of four algorithms

Algorithm	Avg best	Avg mean	Std mean	Optimal (best=0)	Min	Max
ET-GA	4.4	128.1	251.9	27/30	0	1125.1
DE	4205	9924.9	7836.4	9/30	1140	33545.2
GA	7664.5	16923.5	14688.7	11/30	3379.5	68510
PSO	7914.9	15845.8	13304.2	12/30	1311.6	43027.1

To verify the statistical significance of the performance differences, the Friedman test was conducted and returned a p-value<0.000001, strongly rejecting the null hypothesis of no performance difference among the algorithms. Next, Wilcoxon signed-rank pairwise tests were applied to compare ET-GA with each algorithm, with p-values<0.000001 in all three cases, confirming that ET-GA has statistically significantly better performance (significance level α=0.05). The performance profile shows that ET-GA achieved 97% of instances with objective values≤1000, while the remaining algorithms still achieved 0% at the same threshold. To reach full coverage (100%), ET-GA requires a threshold of only 1125, compared with

33,545 for DE, 68,510 for GA, and 43,027 for PSO. Ranking analysis shows that ET-GA achieved an average ranking of 1.00, followed by DE (2.40), PSO (3.20), and GA (3.40). The substantial gaps in both the mean and the variance, clearly illustrated in the boxplots, demonstrate that ET-GA not only finds higher-quality solutions but also exhibits markedly greater stability than traditional algorithms. These results collectively confirm the effectiveness and reliability of the proposed method for solving the MS-RCPSP with earliness–tardiness optimization objectives. The statistical significance test results are reported in Table 5.

Table 5. Statistical significance tests

Test	Comparison	Statistic	P-value	Result
Friedman	All four algorithms	64.08	<0.000001	Significant difference exists
Wilcoxon	ET-GA vs. DE	0.00	<0.000001	ET-GA better (↑98.71%)
Wilcoxon	ET-GA vs. GA	0.00	<0.000001	ET-GA better (↑99.24%)
Wilcoxon	ET-GA vs. PSO	0.00	<0.000001	ET-GA better (↑99.19%)

5.3. Discussion

Experimental results highlight several key trade-offs among solution quality, convergence speed, computational complexity, and scalability. Increasing the generation count from 1,000 to 5,000 leads to steady improvements in solution quality: the number of instances reaching optimality increases from 21/30 to 24/30, while the average best value drops from 57.8 to 19.4. Scaling further to 10,000 generations yields the highest solution quality, with 27/30 instances achieving the optimum, and the average best value falling to 4.4. However, these gains come at the cost of substantially higher computational overhead and localized instability across a few instances (e.g., iM5, iM17, and iM19), where the standard deviation increases due to population divergence. This trade-off suggests that the 10,000-generation configuration is best suited when maximizing solution quality is the primary objective. In contrast, the 5,000-generation configuration offers a well-balanced compromise between quality, convergence rate, and computational cost under resource-constrained environments. In terms of computational complexity, the priority-based resource allocation and elitism mechanisms introduce only a modest per-generation overhead while significantly stabilizing convergence, thereby enabling ET-GA to achieve high computational efficiency relative to the performance gains.

Regarding scalability, the benchmark instances span a wide spectrum of problem scales (100–200 tasks, 5–40 resources, 22–145 precedence constraints, and 9–15 skill types). ET-GA maintains high solution quality and robustness on small- to medium-scale instances; however, its performance exhibits greater variance on the most complex instances (e.g., iM19, iM20, and iM21). In these challenging cases, the vast search space and dense task–resource constraints substantially increase search difficulty, resulting in a high standard deviation even at extended generation counts. This suggests that while the proposed mechanisms generally scale well, very large-scale scheduling problems with tightly coupled constraints remain challenging and could further benefit from parallel or hybrid search strategies.

From a practical perspective, the ET-RCPSP is readily applicable to various real-world domains where both resource utilization and deadline adherence are paramount. In engineering project management and construction planning, explicitly defined earliness–tardiness penalties help balance the costs of committing resources prematurely against the risks and costs associated with contractual delays. In manufacturing scheduling, particularly under the JIT paradigm, the framework enforces strict deadline compliance to minimize inventory holding costs and idle resource overhead. Similar benefits extend to logistics, where deadline-sensitive delivery and distribution planning must balance resource utilization against on-time performance, and to software development, where sprint and release scheduling operate under tight deadline pressure with multi-skilled personnel. Furthermore, the formulation naturally extends to cloud resource scheduling and other complex resource-constrained optimization problems, where multi-skilled resources and time-sensitive service-level agreements must be jointly optimized. These characteristics establish ET-RCPSP as a versatile foundation for intelligent decision-making across diverse project environments.

6. CONCLUSION

This study introduces ET-RCPSP, an extended variant of the MS-RCPSP that incorporates task-specific earliness–tardiness penalties to better reflect practical project management requirements. To tackle this NP-hard problem, we propose ET-GA, an enhanced evolutionary algorithm equipped with two novel mechanisms: a priority-based resource allocation strategy for independent tasks and an elitist preservation mechanism. Our methodological contributions are threefold: a rigorous ET-RCPSP formulation that

integrates earliness–tardiness costs into multi-skill scheduling; an enhanced evolutionary algorithm that boosts both search efficiency and convergence stability; and a comprehensive empirical evaluation across 30 benchmark instances from the iMOPSE dataset. ET-GA achieves optimal solutions on 90% of the benchmark instances and reduces the average earliness–tardiness penalty by over 98% compared to DE, GA, and PSO, while demonstrating markedly superior consistency and statistically significant advantages. Beyond these concrete results, the proposed framework contributes to the broader fields of intelligent optimization, engineering informatics, and complex resource scheduling by providing a reusable baseline for future research and demonstrating strong potential for industrial deployment in real-world engineering applications.

Despite these promising outcomes, several limitations should be acknowledged. First, regarding data scope, the experimental evaluation relies exclusively on the iMOPSE benchmark, which, while widely recognized, may not fully capture the complexity, stochasticity, and dynamic nature of real-world project environments. Second, regarding model assumptions and penalty structures, the current formulation assumes deterministic task durations and resource availabilities, as well as linear, task-specific earliness–tardiness penalties. In contrast, practical scenarios often involve stochastic parameters and more complex, non-linear penalty functions. Third, in terms of practical deployment, ET-GA's hyperparameters (population size, crossover and mutation rates, and elitism ratio) were empirically tuned rather than dynamically adapted, and performance variance remains noticeable on very large-scale instances with tightly coupled constraints.

Building upon these limitations, future research will pursue several promising directions: i) incorporating stochastic factors to enhance model fidelity under uncertainty; ii) developing adaptive parameter control mechanisms to replace manual tuning; iii) investigating hybrid evolutionary optimization, including hybridization with DRL and other learning-based methods, as well as multi-objective ET-RCPSP formulations to optimize competing objectives simultaneously; iv) extending the framework to dynamic resource-constrained scheduling with support for real-time/online rescheduling; and v) designing parallel optimization frameworks to enhance scalability and computational efficiency on large-scale problems. Finally, extensive validation on real-world industrial datasets will be conducted to substantiate practical applicability further.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Hao Nguyen Thi	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	✓
Loc Nguyen The	✓	✓			✓			✓	✓	✓				
Huu Dang Quoc	✓	✓	✓	✓	✓	✓			✓	✓			✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that there are no conflicts of interest regarding the publication of this paper.

DATA AVAILABILITY

The data that support the findings of this study are openly available in the iMOPSE benchmark dataset [10] at https://github.com/imopse/iMOPSE/tree/main/configurations/problems/MSRCPSP/d36.

REFERENCES

[1] B. Afshar-Nadjafi, "Multi-skilling in scheduling problems: A review on models, methods and applications," *Computers and Industrial Engineering*, vol. 151, p. 107004, Jan. 2021, doi: 10.1016/j.cie.2020.107004.

[2] R. Chaouch, H. Ghorbel, and S. Khalfallah, "Model for the classification of scheduling problems based on ontology," *Procedia Computer Science*, vol. 181, pp. 890–896, 2021, doi: 10.1016/j.procs.2021.01.244.
[3] Z. Bahroun, R. As'Ad, M. Tanash, and R. Athamneh, "The Multi-Skilled Resource-Constrained Project Scheduling Problem: A Systematic Review and an Exploration of Future Landscapes," *Management Systems in Production Engineering*, vol. 32, no. 1, pp. 108–132, Mar. 2024, doi: 10.2478/mspe-2024-0012.
[4] J. Snauwaert and M. Vanhoucke, "A classification and new benchmark instances for the multi-skilled resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 307, no. 1, pp. 1–19, May 2023, doi: 10.1016/j.ejor.2022.05.049.
[5] S. Hartmann and D. Briskorn, "An updated survey of variants and extensions of the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 297, no. 1, pp. 1–14, Feb. 2022, doi: 10.1016/j.ejor.2021.05.004.
[6] L. Perron, F. Didier, and S. Gay, "The CP-SAT-LP Solver (Invited Talk)," in *29th International Conference on Principles and Practice of Constraint Programming (CP 2023)*, 2023, pp. 3:1-3:2, doi: 10.4230/LIPIcs.CP.2023.3.
[7] H. Ding, C. Zhuang, and J. Liu, "Extensions of the resource-constrained project scheduling problem," *Automation in Construction*, vol. 153, p. 104958, Sep. 2023, doi: 10.1016/j.autcon.2023.104958.
[8] M. Guerrero-Luis, F. Valdez, and O. Castillo, "A Review on the Cuckoo Search Algorithm," in *Fuzzy Logic Hybrid Extensions of Neural and Optimization Algorithms: Theory and Applications*, Cham, Switzerland: Springer, 2021, pp. 113–124, doi: 10.1007/978-3-030-68776-2_7.
[9] H. T. T. Binh and P. D. Thanh, *Evolutionary Computation*. Hanoi, Vietnam: Bach Khoa Hanoi Publishing House, 2024.
[10] P. B. Myszkowski, M. Laszczyk, I. Nikulin, and M. Skowroński, "iMOPSE: a library for bicriteria optimization in Multi-Skill Resource-Constrained Project Scheduling Problem," *Soft Computing*, vol. 23, no. 10, pp. 3397–3410, May 2019, doi: 10.1007/s00500-017-2997-5.
[11] R. Shahabi-Shahmiri, R. Tavakkoli-Moghaddam, A. Dolgui, S. A. Mirnezami, M. Ghasemi, and M. Ahmadi, "Preemptive and non-preemptive multi-skill multi-mode resource-constrained project scheduling problems considering sustainability and energy consumption: A comprehensive mathematical model," *Journal of Environmental Management*, vol. 367, pp. 1–20, Sep. 2024, doi: 10.1016/j.jenvman.2024.121986.
[12] A. Kramer and A. Subramanian, "A unified heuristic and an annotated bibliography for a large class of earliness–tardiness scheduling problems," *Journal of Scheduling*, vol. 22, no. 1, pp. 21–57, Feb. 2019, doi: 10.1007/s10951-017-0549-6.
[13] M. Otten, A. Braaksma, and R. J. Boucherie, "Minimizing Earliness/Tardiness costs on multiple machines with an application to surgery scheduling," *Operations Research for Health Care*, vol. 22, pp. 1–9, Sep. 2019, doi: 10.1016/j.orhc.2019.100194.
[14] J. Hoffmann, J. S. Neufeld, and U. Buscher, "Minimizing the earliness–tardiness for the customer order scheduling problem in a dedicated machine environment," *Journal of Scheduling*, vol. 27, no. 6, pp. 525–543, Dec. 2024, doi: 10.1007/s10951-024-00814-z.
[15] A. H. Hosseinian and V. Baradaran, "An evolutionary algorithm based on a hybrid multi-attribute decision making method for the multi-mode multi-skilled resource-constrained project scheduling problem," *Journal of Optimization in Industrial Engineering*, vol. 12, no. 2, pp. 155–178, 2019, doi: 10.22094/JOIE.2018.556347.1531.
[16] E. B. Tirkolaee, A. Goli, M. Hematian, A. K. Sangaiah, and T. Han, "Multi-objective multi-mode resource constrained project scheduling problem using Pareto-based algorithms," *Computing*, vol. 101, no. 6, pp. 547–570, Jun. 2019, doi: 10.1007/s00607-018-00693-1.
[17] P. B. Myszkowski and M. Laszczyk, "Investigation of benchmark dataset for many-objective Multi-Skill Resource Constrained Project Scheduling Problem," *Applied Soft Computing*, vol. 127, p. 109253, Sep. 2022, doi: 10.1016/j.asoc.2022.109253.
[18] G. Povéda, N. Alvarez, and C. Artigues, "Partially Preemptive Multi Skill/Mode Resource-Constrained Project Scheduling with Generalized Precedence Relations and Calendars," in *29th International Conference on Principles and Practice of Constraint Programming (CP 2023)*, 2023, pp. 31:1–31:21, doi: 10.4230/LIPIcs.CP.2023.31.
[19] M. Antkiewicz, P. B. Myszkowski, K. Gmyrek, A. Krzeminski, and J. L. Calvo-Rolle, "Efficiency of Specialized Genetic Operators in Non-dominated Tournament Genetic Algorithm (NTGA2) Applied to Multi-objective Multi-skill Resource Constrained Project Scheduling Problem," in *Communications in Computer and Information Science (CCIS)*, vol. 2166, 2024, pp. 97–110, doi: 10.1007/978-3-031-70259-4_8.
[20] J. Snauwaert and M. Vanhoucke, "A solution framework for multi-skilled project scheduling problems with hierarchical skills," *Journal of Scheduling*, vol. 28, no. 3, pp. 289–310, Jun. 2025, doi: 10.1007/s10951-025-00836-1.
[21] R. Torba, S. Dauzère-Pérès, C. Yugma, C. Gallais, and J. Pouzet, "Solving a real-life multi-skill resource-constrained multi-project scheduling problem," *Annals of Operations Research*, vol. 338, no. 1, pp. 69–114, Jul. 2024, doi: 10.1007/s10479-023-05784-7.
[22] G. Vermeire and M. Vanhoucke, "A priority rule heuristic for the multi-skilled resource-constrained project scheduling problem," *Applied Soft Computing*, vol. 171, p. 112776, Mar. 2025, doi: 10.1016/j.asoc.2025.112776.
[23] P. B. Myszkowski, Ł. P. Olech, M. Laszczyk, and M. E. Skowroński, "Hybrid Differential Evolution and Greedy Algorithm (DEGR) for solving Multi-Skill Resource-Constrained Project Scheduling Problem," *Applied Soft Computing Journal*, vol. 62, pp. 1–14, Jan. 2018, doi: 10.1016/j.asoc.2017.10.014.
[24] J. C. Chen, H. Y. Lee, W. H. Hsieh, and T. L. Chen, "Applying hybrid genetic algorithm to multi-mode resource constrained multi-project scheduling problems," *Journal of the Chinese Institute of Engineers, Transactions of the Chinese Institute of Engineers,Series A*, vol. 45, no. 1, pp. 42–53, Jan. 2022, doi: 10.1080/02533839.2021.1983461.
[25] R. Pellerin, N. Perrier, and F. Berthaut, "A survey of hybrid metaheuristics for the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 280, no. 2, pp. 395–416, Jan. 2020, doi: 10.1016/j.ejor.2019.01.063.
[26] J. Park, J. Chun, S. H. Kim, Y. Kim, and J. Park, "Learning to schedule job-shop problems: representation and policy learning using graph neural network and reinforcement learning," *International Journal of Production Research*, vol. 59, no. 11, pp. 3360–3377, Jun. 2021, doi: 10.1080/00207543.2020.1870013.
[27] K. Lei *et al.*, "A multi-action deep reinforcement learning framework for flexible Job-shop scheduling problem," *Expert Systems with Applications*, vol. 205, p. 117796, Nov. 2022, doi: 10.1016/j.eswa.2022.117796.
[28] H. Cai, Y. Bian, and L. Liu, "Deep reinforcement learning for solving resource constrained project scheduling problems with resource disruptions," *Robotics and Computer-Integrated Manufacturing*, vol. 85, p. 102628, Feb. 2024, doi: 10.1016/j.rcim.2023.102628.

BIOGRAPHIES OF AUTHORS



Ms. Hao Nguyen Thi bachelor's degree was awarded in 2009 by Hung Vuong University, Vietnam, where she has been a lecturer ever since. Her master's degree was awarded in 2013 by Hanoi University of Science and Technology, Vietnam. Her research focuses on software engineering and evolutionary algorithm. She can be contacted at email: haont@hvu.edu.vn.



Loc Nguyen The received his bachelor's degrees (1998) and master's degrees (2001) in the Hanoi University of Science and Technology, Vietnam. He received his Ph.D. from Japan Advanced Institute of Science and Technology, Japan, in 2007. He has been a lecturer at HNUE since 1997. His research interests include evolutionary and approximate algorithms. He can be contacted at email: locnt@hnue.edu.vn.



Dr. Huu Dang Quoc received his bachelor's degrees (2000) and master's degrees (2015) in the National University, Hanoi, Vietnam. His Ph.D. was awarded in 2022 by the Military Institute of Science and Technology, Vietnam. Since 2006, he has been a lecturer at Thuong Mai University, Vietnam. His research interests include software engineering, evolution algorithms, and optimization algorithms. His research focuses on software engineering and evolutionary algorithms. He can be contacted at email: huudq@tmu.edu.vn.