The R-PSO algorithm solving multi-skill resource-constrained project scheduling problem

Dang Quoc Huu[1*], Nguyen The Loc[2], Nguyen Doan Cuong[3]

[1]Vietnam Comercial University, Ha Noi, Viet Nam;
[2]Ha Noi National University of Education, Ha Noi, Viet Nam;
[3]Military Institute of Science and Technology, Ha Noi, Viet Nam.
*Corresponding author: huudq@tmu.edu.vn.
Received 11 September 2021; Revised 13 December 2021; Accepted 15 December 2021.
DOI: https://doi.org/10.54939/1859-1043.j.mst.csce5.2021.71-82

ABSTRACT

The Multi-Skill Resource-Constrained Project Scheduling Problem (MS-RCPSP) is a combinational optimization problem with many applications in science and practical areas. This problem aims to find out the feasible schedule for the completion of projects and workflows that is minimal duration or cost (or both of them - multi objectives). The researches show that MS-RCPSP is classified into NP-Hard classification, which could not get the optimal solution in polynomial time. Therefore, we usually use approximate methods to carry out the feasible schedule. There are many publication results for that problem based on evolutionary methods such as GA, Greedy, Ant, etc. However, the evolutionary algorithms usually have a limitation that is fallen into local extremes after a number of generations. This paper will study a new method to solve the MS-RCPSP problem based on the Particle Swarm Optimization (PSO) algorithm that is called R-PSO. The new improvement of R-PSO is re-assigning the resource to execute solution tasks. To evaluate the new algorithm's effectiveness, the paper conducts experiments on iMOPSE datasets. Experimental results on simulated data show that the proposed algorithm finds a better schedule than related works.

Keywords: Swarm Intelligence; Evolutionary Computing; Optimization Methods; Resource-Constrained Project Scheduling.

1. INTRODUCTION

Scheduling problems have been studied since 1950, he scheduling objective is to find a matching method assigned task-resource satisfying the pre-constraints. The scheduling problem is applied in many fields such as operating systems [9], production lines, finance [4] and military [5]. Many published results have shown that this problem is classified into NP-Hard [14].

RCPSP (Resource-Constrained Project Scheduling Problem) [14] is a project scheduling problem with limited resources, which has been proven to be an NP-Hard problem. The goal is to find a good schedule with minimal time in terms of resource constraints. Currently, this problem is studied and applied in many fields.

An important problem extended from RCPSP is MS-RCPSP [1, 3, 11, 18] (Multi-kill RCPSP) that has a new constraint on the skill factor of implementation resources.

The rest of the paper is deivided into 6 parts. In part 2, we present some related works, such as previous approaches to solving the MS-RCPSP problem. Specifically, the summary of the PSO method is included in this section. PSO is an efficient evolutionary algorithm for solving NP-Hard problems. This paper proposes a new algorithm based on PSO to solve the MS-RCPSP problem.

In part 3, the MS-RCPSP problem is specifically defined by mathematical

formulations that show the problem constraints. A new method called Reallocate will be combined with PSO to find a satisfactory solution. The proposed algorithms will be introduced in Part 4. In part 5, the paper will run the new algorithm with the iMOPSE dataset. After that, the experiment results will be collected and evaluated to show the new algorithm's power. Finally, part 6 concludes the paper and outlines some directions for future work.

2. RELATED WORKS

The MS-RCPSP, widely studied by many scientists, has many applications in science and practice. By many quality publications of scientists, the research groups show that it is an NP-Hard problem [14], so the authors usually use evolutionary approaches to solve and get out the approximate solutions.

Myszkowski et al. [10, 13] have researched this problem for many years and they have many quality works presented regarding that. The authors proposed algorithms based on the evaluation methods such as Tabu Search [8], Ant colony [12], GA [11], etc. Most all of them are traditional methods so they usually have got the limited effective results. The best contribution of them are to have published a standard dataset called iMOPSE benchmark dataset [10]. It is used to experiment with new algorithms that solve the MS-RCPSP.

Hosseinian et al. [1-3] have many results from ms-rcpsp too. In 2018, Hosseinian and Baradaran published a paper [1] on the Multi-mode MS-RCSPSP (MMSRCPSP), a sub-class of the MS-RCPSP problem. MMSRCPSP has added constraints that each task can only be executed in a few predefined modes. When the mode is selected, it can not be changed. The authors proposed a method to choose individuals in the next generation, built from the GA algorithm combined with the decision-making method based on Shanon-entropy data measure. Experiments were carried out on randomly generated data sets by ProGen software. After that, in 2019, the authors published a paper [2] in which the Dandelion Algorithm. In 2020, this group of authors continued to publish an article [3] that solves the multi-objective MS-RCPSP problem with two objectives: total time and cost to complete a project. The authors use the Pareto-based Gray Wolf Optimizer algorithm and continue to test the results on the iMOPSE standard dataset [10].

In 2017, Javanmard et al. used two traditional evolutionary algorithms, GA and PSO, to schedule multi-skill resources (actually workers and engineers) to minimize the total cost in the chemical industry [16]. The proposed algorithms are tried on the PSPLIB dataset[15] not fully suitable for scheduling problems because of no cost parameter objectives. Moreover, at the experimental stage, the authors cannot compare their traditional algorithms with newer ones, but only compare them with each other.

Also studying multi-objective problems such as Hosseinian's group, H. Davari-Ardakani [6] proposed a multi-objective variant of the MS-RCPSP problem to minimize the time and cost of the current project. The proposed problem is called MSPSP [6], which is limited to projects with the following two characteristics:

- The timing of implementation is arbitrary, for example, the project can be done in the evenings or on weekends;

- Energy costs are very high, comparable to wages paid to employees.

However, the authors only gave the problem model without any new solution method to solve it. They only conducted experiments with the Max-Min method, a very traditional method.

3. PROBLEM DEFINITIONS

The MS-RCPSP [1, 3, 8] is a new RCPSP (Resource-Constrained Project Scheduling Problem) variant that added the skill domain to the renewable resources. The objective of this problem is to find the minimal makespan, i.e. the minimum time to complete the full project. In the running time, each task requires a particular skill that is equal to or greater than the task's requirement.

To define the formulation of the MS-RCPSP, we need a notation system presented in table 1.

Table 1. *The notation.*

C_i	The set of tasks needs be completed beforre the task *i* can be executed.
S	The set of all resource's skills S^i: subset of skills owned by the resource *i*, $S^i \subseteq S$;
S_i	The skill *i;*
t_j	The duration of task *j*
L	The resources used to execute tasks of the project
L^k	The subset of the resources which can be performed task *k*; $L^k \subseteq L$
L_i	The resource i
W	The tasks of the project need to do
W^k	The subset of task which can be executed by the resource *k* , $W^k \subseteq W$
W_i	The task *i*
r^i	The subset of the skill required by task i. A resource has the same skill and skill level equal or greater than requirement of task, that can be performed task.
B_k, E_k	The begin time and end time of the task k
$A_{u,v}{}^t$	The variable to identify the resource v is running task u at time t; 1: yes, 0: no;
h_i	The skill level i;
g_i	Type of skill i;
m	Makespan of the schedule
P	The feasible solution
P_{all}	The set of solutions
$f(P)$:	The function to calculate the makespan of P solution
n	Task number
z	Resource number

Using the notations presented in table 1, we can construct the mathematical model for the MS-RCPSP as follow:

$$f(P) \rightarrow \min$$

Subject to:

- $S^k \neq \varnothing$ $\forall L_k \in L$ (1)
- $t_j \geq 0$ $\forall W_j \in W$ (2)
- $E_j \geq 0$ $\forall W_j \in W$ (3)
- $E_i \leq E_j - t_j$ $\forall W_j \in W, j \neq 1, W_i \in C_j$ (4)
- $\forall W_i \in W^k \exists S_q \in S^k \; g_{S_q} = g_{r^i}$ và $h_{S_q} \geq h_{r^i}$ (5)
- $\forall L_k \in L, \forall q \in m : \sum_{i=1}^{n} A_{i,k}^q \leq 1$ (6)
- $\forall W_j \in W \; \exists! \, q \in [0, m], ! \; L_k \in L: A_{j,k}^q = 1;$ với $A_{j,k}^q \in \{0; 1\}$ (7)

As a new requirement of the MS-RCPSP problem, each task requires a resource that has the same skill type and skill level to execute. The assigning of resources and tasks can be described as a matrix illustrated in figure 1 below:

✓ Can be assigned
× Can not be assigned

Tasks

$S_{2.2}$	$S_{3.1}$	$S_{2.2}$	$S_{1.1}$
W_1	W_2	W_3	W_4

Resources

$S_{1.3}$, $S_{2.2}$	L_1
$S_{2.1}$, $S_{3.2}$	L_2
$S_{1.2}$, $S_{2.1}$	L_3
$S_{2.2}$, $S_{3.3}$	L_4

✓	×	✓	✓
×	✓	×	×
×	×	×	✓
✓	✓	✓	×

Figure 1. *The relation matrix between resources-tasks assignment.*

In figure 1, the $S_{i.j}$ denotes S_i skill type and it has j skill level. Tasks that require execution resources must meet a specific skill type and skill level. The feasible resource should have the same skill type and skill level greater than or equal to the required skill level.

Example 1:

In figure 1, to do task W_1, the resource must meet the requirement of the task that means the resource has to have skill type S_2 and skill level equal to or greater than *2*. Looking up into resources, we identify that resource L_1 has skill type S_2 and skill level is *2*, so L_1 can be assigned to execute W_1.

4. PROPOSED ALGORITHM

4.1. Schedule representation

In the scheduling algorithm, a schedule is represented as a vector that the vector's item number is equal to the task number of the project. The value of each item shows the resource index to perform it. The corresponding resource has to meet the requirement constraint of the task.

Example 2: Consider a project that has 10 tasks and 2 resources.

- The set of tasks: W = {W_1, W_2, W_3, W_4, W_5, W_6, W_7, W_8, W_9, W_{10}}

- The set of resources: L = {L_1, L_2}.

The priority graph of the tasks is shown in figure 2 below.

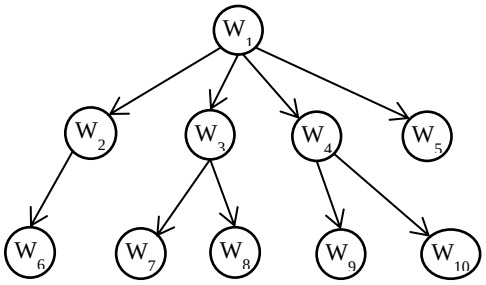

Figure 2. *The priority graph of project.*

The duration time of tasks (in hours) is shown in table 2.

Table 2. *The duration of task.*

Task	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9	W_{10}
Duration	2	4	3	5	2	2	5	3	4	2

Figure 3 shows the assigning of resources to execute tasks of the project. The resource to perform the task needs to meet the priority constraints of the problem containing the skill requirements and skill level.

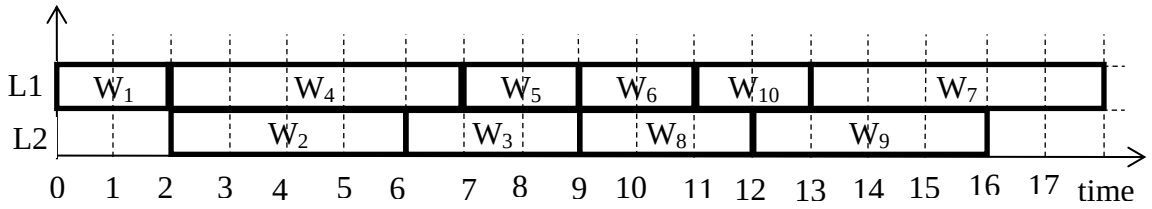

Figure 3. *The gantt diagram of the resource-task.*

The schedule shown in figure 3 can be presented by a vector as table 3 follow.

Table 3. *The duration of task.*

Task	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9	W_{10}
Resource	*1*	*2*	*2*	*1*	*1*	*1*	*1*	*2*	*2*	*1*

The table 3 is presented task-resource assignment of the project, where the resource L_1 performed tasks: W_1, W_4, W_5, W_6, W_7, W_{10}; The resource L_2 is assigned to execute tasks: W_2, W_3, W_8, W_9.

4.2. R-PSO Algorithm

To find a feasible solution with the minimum makespan for the MS-RCPSP problem, we research the Reallocation method combined with the PSO [7, 17] named R-PSO. Its pseudo-code is shown in Algorithm 1.

Algorithm 1. R-PSO function

```
   Input: tmax: number of generation
   Output: gbest: the best solution
1      Begin
2         Pall ← Load iMOPSE Dataset and init the population
3         t = 0
4         nf = 0
5         while ( t < tmax )
6            t = t + 1
7            for i=1 to size(Pall) do
8               Caculation the objective function: f(Pi)
```

```
9           end for
10          for i = 1 to size(Pall) do
11            if f(Pi)<f(fitnessi) then
12                pbesti = Pi
13                f(pbesti) = f(Pi)
14             end if
15          end for
16          for i=1 to size(Pall) do
17             if(f(Pi) < f(gbest)) then
18                gbest = Pi
19                f(gbest) = f(Pi)
20             end if
21          end for
22          for i=1 to size(Pall) do
23            Update the velocity vector
24            Update position vector
25          end for
26          for i=1 to size(Pall) do
27             Pi ← Reallocate(Pi)
28          end for
29        end while
30      return gbest
31    end
```

f: objective function

Lines 26 to 28 show that the Reallocate function is called to re-assignment the resource to execute the tasks of particles.

4.3. Function Reallocate

A schedule is a vector indicating the resource assigned to do the tasks. The Reallocate technical will re-find the new method to assign another resource to execute the task of schedule. This technical gives two effects:

- Expanding the search space, thereby improving the ability to find better solutions;
- Supporting the population to escape the local extremes.

The pseudo-code of this function is presented as algorithm 2.

Algorithm 2. Reallocate function

Input: currentBest // the best schedule among the current population
Output: // the improved schedule

```
1.  Begin
2.      makespan = f(best)
3.      newbest = currentBest;
4.      Lb ← maxResource (newbest) // the last resource to finish its job
5.      Wb ← set of tasks is performed by resource Lb
6.      For i=1 to size(Wb)        //Consider each task in Tb, the set of tasks performed
        by resource Rb
7.          Wi = Wb[i];
8.          Li ← set of resource that are skilled enough to execute the
            task i except Lb
9.          For j = 1 to size(Li)        // Consider each resource in turn
10.             Li[j] = Li[j] + {i}       // Reassign task i to resource Li[j]
11.             Lb = Lb – {i}             // Remove task i from the task list of Lb
12.             newmakespan = f(newbest) // The makespan of the new schedule
13.             If newmakespan  < makespan
```

```
14.                 makespan = newmakespan
15.                 Return newbest;      // Successful Reallocate, return the new and
                better schedule
16.             End if
17.             newbest = currentBest; // Unsuccessful Reallocate, reuse the original
                schedule for the next iteration
18.         End for
19.     End for
20.     Return best
21. End Function
```

Where:

- `currentBest`: The best schedule among the population of the current generation.
- L_b: According to the arrangement of schedule *newbest* (line 3), L_b is the last resource that finishes its job. In Figure.3, before the Reallocate function was called, L_b is resource 1.
- `maxResource()`: The function which returns the last resource to finish (line 3).
- `size()`: the function which returns the number of elements of the given set or array (line 5).
- `newmakespan`: Makespan of the new schedule (line 11), which obtained from the old schedule by reassigning task *i*. Task *i* is moved from resource L_b (line 9) to resource $L_i[j]$ (line 10).

Line 12 and 13 demonstrate that the new schedule `(newbest)` is better than the old schedule `(currentBest)` in terms of makespan, which means that the function *Reallocate* certainly returns a better schedule. Moreover, the *Reallocate* function guides algorithm R-PSO in the correct direction by founding a new schedule `(newbest)` based on the best solution `(currentBest)`. Therefore, it inherits the advanced genomes of the population as well as the old schedule.

Example 3:

Given set of task W ={1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, set of resource L^k ={1, 2, 3}. Resource 1 can execute tasks 1, 2, 3, 4, 6, 8, 9, and 10; Resource 2 can perform tasks 1,3,7, and 9; Resource 3 can handle tasks 1,4,5,8,9, and 10. The relationship between tasks is depicted in figure 1, the duration time of the tasks is given in table 4.

Consider a schedule (P) as shown in table 4. From observation, figure 4 shows that makespan f(P) = 17.

Table 4. *Resource – task assignment of P.*

Task	1	2	3	4	5	6	7	8	9	10
Resource	1	1	2	3	3	1	2	1	3	3

New schedule, the result of the function *Reallocate*, is described in table 5.

Table 5. *Resource – task assignment of new schedule.*

Task	1	2	3	4	5	6	7	8	9	10
Resource	1	1	2	3	3	1	2	1	1	3

Figure 4 illustrates that function *Reallocate* converts the schedule P into the new schedule by assigning task 9 to resource 2 instead of resource 1 as before. Thanks to this

allocation, the new schedule achieves a better makespan (15) compared to the old schedule (17).

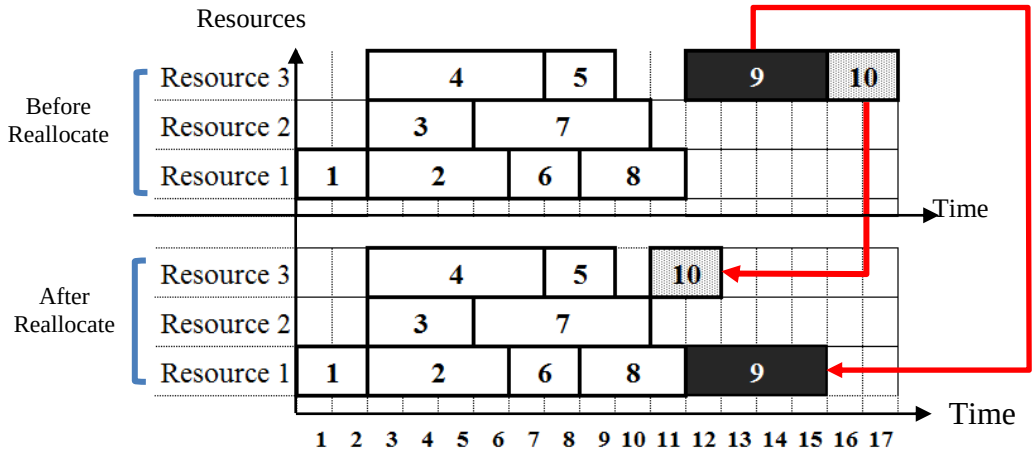

Figure 4. *Gantt diagram of schedule befor and after function Reallocate.*

5. EXPERIMENT

To evaluate the efficiency of our proposed algorithm, we developed the simulator on Matlab environment conducted on the iMOPSE benchmark datasets. The experiment results are compared with the GA-M [10] and GRASP [13] to evaluate the efficiency of our proposed algorithm.

5.1. The benchmark dataset

In the simulated program, we use the iMOPSE [10] installers which are investigated many existing algorithms as GRASP, GA-M [10, 13], etc. The iMOPSE dataset has the characteristics follow:

- The number of tasks and resources;
- The number of precedence relations between tasks;
- The number of skills.

Table 6 shows 12 iMOPSE's instances to experiment with our algorithms.

The simulation is run on a PC with Intel Core i5-CPU 2.2 GHz, 6 GB RAM, and using the Windows 10 OS. The dataset used in our simulator are listed in table 6.

Table 6. *iMOPSE benchmark dataset.*

Dataset instance	Tasks	Resources	Precedence Relations	Skills
100-5-22-15	100	5	22	15
100-5-46-15	100	5	46	15
100-10-26-15	100	10	26	15
100-10-47-9	100	10	47	9
100-20-22-15	100	20	22	15
100-20-47-9	100	20	47	9
200-10-50-9	200	10	50	9
200-10-50-15	200	10	50	15

200-20-55-9	200	20	55	9
200-20-54-15	200	20	54	15
200-40-45-9	200	40	45	9
200-40-45-15	200	40	45	15

The input parameters and initialize values as:

- The experiment environment: Matlab;
- Data: 12 iMOPSE installers that are shown in table 6;
- Size of the population Np: 100;
- Number of generations Ng: 50,000;
- Number of testing rounds: 35.

All result data is collected containing average values, the standard deviation values, and the best values.

5.2. Experiment results

The results of the R-PSO are presented in table 7. This table also shows the values of GA-M and GRASP algorithms published together with the iMOPSE dataset.

Table 7. *The experiment results on the iMOPSE dataset.*

Dataset	GA-M			GRASP			R-PSO		
	Best	**Avg**	**Std**	**Best**	**Avg**	**Std**	**Best**	**Avg**	**Std**
100-5-22-15	517	524	5.3	503	504	***0.5***	***488***	***490***	1.9
100-5-46-15	584	587	4.7	552	556	2.6	***528***	***529***	***0.7***
100-10-26-15	292	292	0.0	251	258	3.2	***231***	***232***	***0.3***
100-10-47-9	296	296	0.0	263	264	***0.7***	***260***	264	4
100-20-22-15	163	169	5.8	134	137	***1.6***	***133***	***135***	1.9
100-20-47-9	185	186	0.5	133	140	2.2	***125***	***127***	***1.5***
200-10-50-9	585	589	5.0	506	508	***1.4***	***493***	***497***	3.9
200-10-50-15	553	577	17.5	524	528	3.8	***487***	***488***	***0.8***
200-20-55-9	312	318	4.2	257	258	***0.6***	***253***	***254***	1.1
200-20-54-15	363	385	20.8	304	308	3.7	***254***	***255***	***0.8***
200-40-45-9	209	213	2.9	***144***	147	***1.6***	150	156	5.7
200-40-45-15	201	210	6.3	***164***	164	***0.0***	165	169	3.2

In table 5, GA-M[18] and GRASP[18] are proposed algorithms of Myszkowski, which are executed on iMOPSE dataset.

Comparing the R-PSO with GA-M:

- Regarding BEST and AVG values: R-PSO is better than GA-M in all instances, in which BEST value is better than GA-M from 5.6% to 32.4%, AVG value is better than GA-M from 6.6 % to 35.8%.

- Regarding STD (standard deviation), the total standard deviation of GA-M is 73, of R-PSO is 25.8, which denotes that the R-PSO algorithm is more efficient and more stable than GA-M algorithm.

Comparing the R-PSO with GRASP:

- The results of R-PSO have better results than GRASP in most cases, specifically: better than GRASP 10/12 datasets (from 0% to 16.4%) with BEST value, 9/12 instances data with AVG value.

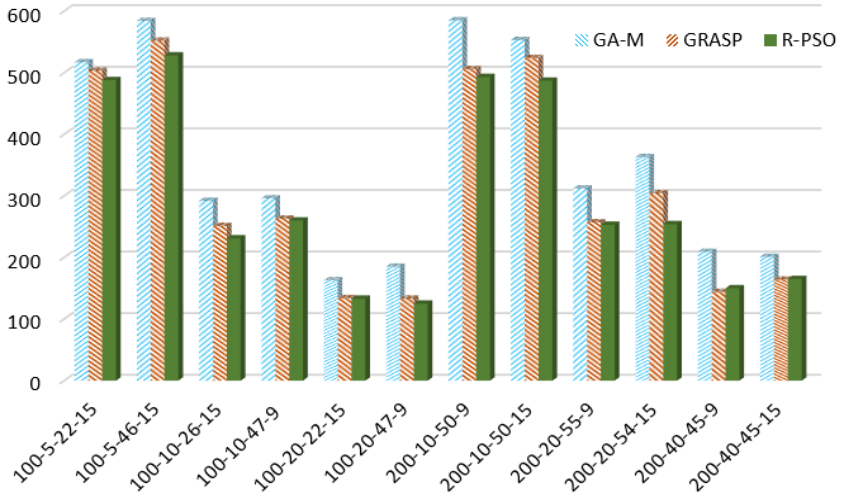

Figure 5. *The comparison of BEST values between GA-M, GRASP and R-PSO.*

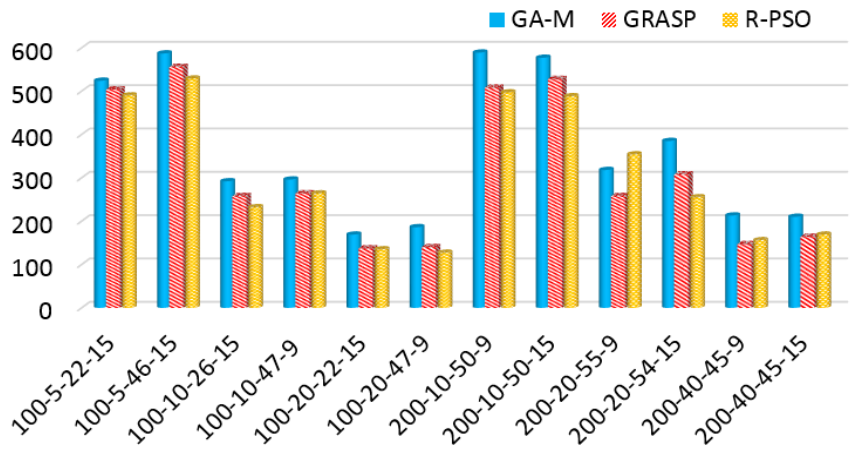

Figure 6. *The comparison of AVG values between GA-M, GRASP and R-PSO.*

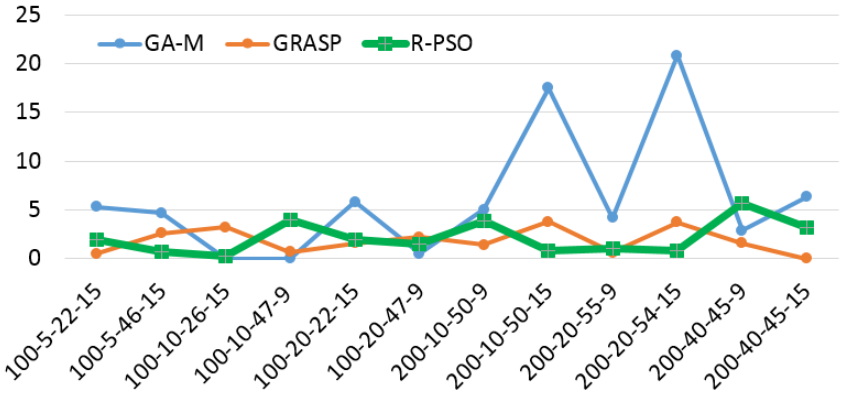

Figure 7. *The comparison of STD values between GA-M, GRASP and R-PSO.*

6. CONCLUSION AND FUTURE WORK

In this paper, we have presented the MS-RCPSP problem, one of the combinatorial optimization problems that have a wide range of practical applications in science, technology, and life. The article describes the mathematical model of the problem and proposes a new algorithm to find a feasible solution for the MS-RCPSP problem. The proposed algorithm is R-PSO, which is improved from the PSO algorithm by combination with Reallocate method to expand the searching space and escape the local extremes. The article has conducted experiments on iMOPSE datasets and compared it with algorithms such as GA-M and GRASP. Experimental results show that R-PSO's results give better than previous algorithms, specifically: better than GA-M from 5.6% to 32.4% and GRASP up to 16.4%.

In the future, the author will continue to research and improve the algorithm based on other approximation methods, using random moves based on Gauss, Cauchy, etc. to improve the effective of the suggested algorithm.

REFERENCES

[1]. A.H. Hosseinian, 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, 12.2, pp. 155-178,2019.

[2]. A.H. Hosseinian, V. Baradaran, *"Detecting communities of workforces for the multi-skill resource-constrained project scheduling problem: A dandelion solution approach."*, Journal of Industrial and Systems Engineering, pp. 72-99, 12.2019.

[3]. A.H. Hosseinian, V. Baradaran, *"P-GWO and MOFA: two new algorithms for the MSRCPSP with the deterioration effect and financial constraints (case study of a gas treating company)."*, Applied Intelligence, 50 , pp. 2151-2176 , 2020.

[4]. F. Black, and M. Scholes, *"The pricing of options and corporate liabilities",* Journal of Political Economy, 81, pp. 637-654, 1973

[5]. H. Li, K. Womer, *"Stochastic Resource-Constrained Project Scheduling and Its Military Applications",* IEEE Trans Computer, 65(12), pp. 3702–3712, 2016.

[6]. H. Najafzad, H. Davari-Ardakani, R. Nemati-Lafmejani, *"Multi-skill project scheduling problem under time-of-use electricity tariffs and shift differential payments."*, Energy Journal, vol. 168, pp. 619-636, Elsevier,2019.

[7]. J. Kennedy, R. Eberhart, *"Particle Swarm Optimization"*, IEEE International Conference on Neural Networks, 1995.

[8]. M. Skowroński, P.B. Myszkowski, P. Kwiatek, M. Adamski, *"Tabu Search approach for Multi–Skill Resource–Constrained Project Scheduling Problem"*, Annals of Computer Science and Information Systems, Volume 1, Proceedings of the 2013 Federated Conference on Computer Science and Information Systems, pp. 153-158, 2013.

[9]. O. Sinnen, *"Task scheduling for parallel systems",* Published by JohnWiley & Sons, Inc., Hoboken, New Jersey, Vol 60, 2007.

[10].P.B. Myszkowski, M. Laszczyk, I. Nikulin, M. Skowro, *"iMOPSE: a library for bicriteria optimization in Multi-Skill Resource-Constrained Project Scheduling Problem",* Soft Computing Journal, 23: 32397, 2019.

[11].P.B. Myszkowski, M. Skowroński, *"Specialized genetic operators for Multi–Skill Resource–Constrained Project Scheduling Problem"*, 19th International Conference on Soft Computing – Mendel 2013, pp. 57-62, 2013.

[12].P.B. Myszkowski, M. Skowroński, L. Olech, K. Oślizło, *"Hybrid Ant Colony Optimization in solving Multi–Skill Resource–Constrained Project Scheduling Problem",* Soft Computing Journal, Volume 19, Issue 12, pp.3599–3619, 2015.

[13].P.B. Myszkowski, M.E. Skowronski, K.Sikora, *"A new benchmark dataset for Multi-Skill Resource-Constrained Project Scheduling Problem",* Computer Science and Information Systems, ACSIS, Vol. 5, pp. 129–138, 2015. DOI: 10.15439/2015F273.

[14].R. Klein: *"Scheduling of Resource-Constrained Projects",* Springer Science & Business Media, Vol. 10, 2012.

[15].R. Kolisch, A. Sprecher, *"PSPLIB-a project scheduling problem library: OR software-ORSEP operations research software exchange program.",* European journal of operational research, 96(1), pp.205-216, 1997.

[16].S. Javanmard, B. Afshar-Nadjafi, S.T. Niaki, *"Preemptive multi-skilled resource investment project scheduling problem: Mathematical modeling and solution approaches.",* Computers & Chemical Engineering, 96, pp. 55-68, 2017.

[17].W. Guo, J.H. Park, L.T. Yang, A.V. Vasilakos, N. Xiong, G. Chen, *"Design and Analysis of a MST-Based Topology Control Scheme with PSO for Wireless Sensor Networks,",* 2011 IEEE Asia-Pacific Services Computing Conference, Jeju Island, pp. 360-367, 2011. doi: 10.1109/APSCC.2011

[18].Huu Dang Quoc, Loc Nguyen The, Cuong Nguyen Doan, Toan Phan Thanh, Naixue Xiong, *"Intelligent Differential Evolution Scheme for Network Resources in IoT",* Scientific Programming (IF:1.28, Q3), Volume 2020, Article ID 8860384 | 12, 2020. DOI: 10.1155/2020/8860384

[19].H.N.S. Krishnamoorthy, Z. Jacob, E. Narimanov, I. Kretzschmar, V.M. Menon, Science 336 (2012) 205.

TÓM TẮT

THUẬT TOÁN R-PSO GIẢI BÀI TOÁN LẬP LỊCH THỰC HIỆN DỰ ÁN VỚI TÀI NGUYÊN GIỚI HẠN

Bài toán lập lịch thực hiện dự án với tài nguyên giới hạn và đa kỹ có nhiều ứng dụng trong khoa học và thực tiễn. Mục tiêu của bài toán này là tìm phương án lịch biểu tốt nhất trong việc thực hiện các dự án, các luồng công việc. Việc đánh giá kết quả bài toán dựa trên yếu tố thời gian thực hiện hoặc chi phí thực hiện hoặc cả hai yếu tố thời gian và chi phí (đa mục tiêu). Do vậy, việc nghiên cứu giải bài toán MS-RCPSP là quan trọng, giúp nâng cao hiệu quả vận hành của các luồng công việc, các dây chuyền sản xuất,... MS-RCPSP đã được chứng minh là bài toán thuộc lớp NP-Khó nên không thể tìm được lời giải trong thời gian đa thức mà cần sử dụng các phương pháp cận tối ưu để tìm được nghiệm đủ tốt. Hiện đã có nhiều nhà khoa học nghiên cứu giải bài toán này bằng các phương pháp tiến hóa như GA, Greedy, Ant,... Tuy nhiên, khi áp dụng các thuật toán tiến hóa với không gian lời giải lớn, thuật toán thường rơi vào các cực trị địa phương, nên sẽ không tìm ra được các nghiệm tốt hơn. Bài báo này sẽ nghiên cứu một phương pháp mới để giải bài toán MS-RCPSP dựa trên thuật toán tối ưu bầy đàn bằng cách sử dụng kỹ thuật tái cấp phát tài nguyên thực hiện tác vụ (Reallocate) để mở rộng không gian tìm kiếm, giúp tăng khả năng tìm được các lời giải tốt hơn và tránh được các vùng cực trị địa phương. Để kiểm chứng thuật toán, bài báo tiến hành thực nghiệm trên bộ dữ liệu chuẩn iMOPSE, các kết quả thực nghiệm cho thấy thuật toán đề xuất mới mang lại hiệu quả tốt hơn so với một số thuật toán trước đây.

Từ khoá: Tính toán tiến hóa; Tối ưu bày đàn; Phương pháp tối ưu; Bài toán MS-RCPSP.