



Sharif University of Technology
Scientia Iranica
Transactions A: Civil Engineering
<http://scientiairanica.sharif.edu>



Chaotic vibrating particles system for resource-constrained project scheduling problem

A. Kaveh* and Y. Vazirinia

Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran, P.O. Box 16846-13114, Iran.

Received 14 July 2018; received in revised form 24 September 2018; accepted 10 December 2018

KEYWORDS

Resource-constrained project scheduling problem;
Metaheuristic algorithms;
Chaotic maps;
Chaotic vibrating particles system;
Chaos theory;
Global optimization.

Abstract. Project scheduling in the resource-constrained situation is one of the key issues of project-oriented organizations. The aim of Resource-Constrained Project Scheduling Problem (RCPSP) is to find a schedule with minimum makespan by considering precedence and resource constraints. RCPSP is a combinatorial optimization problem and belongs to the NP-hard class of problems. The exact methods search the entire search space and are unable to solve a large-sized project network problem. Thus, metaheuristics are used to solve this problem in a short computational time. Due to the probabilistic nature of metaheuristics, it is a challenging problem to make a balance between exploitation and exploration phases. The literature review shows that embedding of chaos improves both the convergence speed and the local optima avoidance of metaheuristics. This paper presents a Chaotic Vibrating Particles System (CVPS) optimization algorithm for solving the RCPSP. Vibrating Particles System (VPS) is a physics-inspired metaheuristic which mimics the free vibration of single-degree-of-freedom systems with viscous damping. The performance and applicability of the CVPS are compared with the standard VPS and five well-known algorithms on three benchmark instances of the RCPSPs. Experimental studies reveal that the proposed optimization method is a promising alternative to assist project managers in dealing with RCPSP.

© 2020 Sharif University of Technology. All rights reserved.

1. Introduction

Scheduling of projects, such as product development, maintenance, construction project, etc., depends on resource limitations because they expend different renewable and non-renewable resources during their execution and these resources are usually limited. Resource constraints include a limited number of employees, limited project budget, vehicle capacity,

etc. Given that ‘project scheduling’ in a resource-constrained situation is an important and challenging issue for project-oriented organizations, project management researchers are particularly interested in the matter. The purpose of Resource-Constrained Project Scheduling Problem (RCPSP) is to find a schedule with minimum makespan based on the precedence relationship between constraints and resource constraints [1]. RCPSP is a combinatorial optimization problem and belongs to the NP-hard class of problems [2,3]. A variety of methods including exact methods [4–6], heuristic [3,7], and metaheuristic [8–10] methods have been used to solve RCPSP with different assumptions. Exact methods search the entire search space and are

*. Corresponding author.

E-mail address: alikaveh@iust.ac.ir (A. Kaveh)

computationally infeasible or face the ‘combinatorial explosion’ problem in solving large-sized project networks. Thus, in recent decades, metaheuristics have been utilized to solve this problem with a manageable computational time.

In today’s extremely competitive world, solving real-life problems by metaheuristic algorithms has become an interesting topic. Many researchers have tried to provide the best method for solving the Construction Engineering Optimization Problems (CEOPs) [8–11]. Many metaheuristics with different philosophies and characteristics have been developed and applied to different fields. The objective of these methods is to explore the search space efficiently in order to find global or near-global solutions. These methods are not problem specific and do not require the derivatives of the objective function; therefore, they have drawn increasing attention from both academia and industry [12–14]. Metaheuristics are global optimization methods that mimic natural phenomena such as Genetic Algorithm (GA) [15], Particle Swarm Optimization (PSO) [16], Dolphin Echolocation (DE) [17], humans social behavior such as Imperialist Competitive Algorithm (ICA) [18], or physical phenomena such as Magnetic Charged System Search (MCSS) [19], Colliding Bodies Optimization (CBO) [20], Vibrating Particles System (VPS) [14], and Harmony Search (HS) [21]. Two important characteristics of the metaheuristic optimization techniques are exploitation and exploration. Exploitation serves to search around the current best solutions and to select the best possible points, and Exploration allows the optimizer to explore the search space more efficiently, often by randomization [22]. Due to the probabilistic nature of the metaheuristics, making a balance between exploitation and exploration phased is a challenging problem. Review of the literature [23–26] reveals that by embedding of chaos, both the convergence speed and the local optima avoidance of metaheuristics can be improved. A chaotic system is a complex system which is intensively sensitive to initial conditions like social, market, economic, astronomical, or earth’s weather system. In these systems, any change, even small changes, in the beginning will produce large changes in the future behavior of the system [27]. Typical features of chaotic systems include nonlinearity, determinism, irregularity, sensitivity to initial conditions, and long-term unpredictability [28].

This paper presents a Chaotic Vibrating Particles System optimization algorithm for solving the Resource-Constrained Project Scheduling Problem (CVPS-RCPSP). Vibrating Particles System (VPS) is a newly developed metaheuristic which mimics the free vibration of single-degree-of-freedom systems with viscous damping [14].

After this introduction, the rest of the paper is organized as follows: the RCPSP is described in

Section 2. The basic VPS algorithm and 10 chaotic maps are described in Section 3. The proposed CVPS technique is presented in Section 4. Numerical examples are studied in Section 5, and the results are discussed in Section 6. Finally, conclusions are deduced in Section 7.

2. Resource-Constrained Project Scheduling Problem (RCPSP)

Due to the usability of resource-constrained problems in construction project management, these problems have been investigated extensively by researchers of this field. This paper elaborates on a typical RCPSP which is defined as follows [9].

A project includes the scheduling of $j = 1, 2, \dots, J$ activities that are delineated in an activity-on-node network $G = (V, E)$, where the nodes and arcs represent the set of activities V and finish-to-start precedence relationship (with zero lag) E , respectively. The numerator of the activities in the project network is from 0 to $J + 1$, where activities 0 and $J + 1$ are dummy activities and specify the start and end of the project and duration of these activities is zero. Precedence relationships between some of the activities in the project necessitate that for starting an activity j , all its predecessors “ P_j ” must be finished because of technological requirements. Activity j requires r_j renewable resource k for each period of operation. The time that the activity j takes to be executed “ D_j ” depends on the amount of its allocated resource. When the activity j begins, any interruption like changing the duration or resource amount cannot occur and it must be continued in sequential periods of D_j . Moreover, the availability of the k th resource is given by R_k .

The purpose is to achieve a solution with minimum total time, considering precedence relationships among different activities and resource constraints at the same time. The objective functions of the RCPSP model are formulated to minimize the total project time with the allocation of resources in the entire project makespan, simultaneously.

When an activity is selected, its corresponding duration and resource requirement will be assigned. Afterward, a feasible schedule based on activity information and given constraints will be produced. The outcome of the resulting schedule is the determination of the project finish time. The objective of RCPSP model is to minimize the duration of the project, which is the finish time of the last activity f_{j+1} in a project. Therefore, the total project duration F_t is given below:

$$\min z = \max f_j, \quad j = 1, 2, \dots, J. \quad (1)$$

In the above formulation, the objective function minimizes the project time F_t . The constraints are expressed as follows:

$$f_j - D_j \geq f_i, \quad (i, j) \in E. \quad (2)$$

These constraints guarantee the consideration of the precedence relationships. In this formula, f_j is the finish time of the activity j , D_j is the duration of activity j , and f_i is the finish time of the predecessor of activity j called i .

$$\sum_{i \in A} r_{j,k} \leq R_k, \quad k = 1, 2, \dots, K,$$

$$A_t = \{j | f_j - D_j < t \leq f_j\}. \quad (3)$$

The constraint set in Eq. (3) indicates that at each time instant t and for each resource type k , the renewable resource amounts required by the activities, which are currently processed (i.e., A_t), cannot exceed the resource availability, where r_{jk} is the amount of resource k required by the activity j .

$$f_i \geq 0, i = 0, 1, \dots, j + 1. \quad (4)$$

Finally, the constraint set in Eq. (4) ensures that every output has been positive and the schedule logic will be true.

3. Vibrating Particles System (VPS)

The VPS is a population-based algorithm that simulates free vibrations of single-degree-of-freedom systems with viscous damping [14]. Similar to other multi-agent methods, VPS has a number of individuals (or particles) consisting of the variables of the problem. In the VPS, each solution candidate is defined as “ X ”, contains a number of variables (i.e., $X_i = \{X_i^j\}$), and is considered as a particle. Particles are damped based on three equilibrium positions with different weights and, during each generation, the position of particles is updated based on using (i) the historically best position of all particles population (HB), (ii) a good particle (GP), and (iii) a bad particle (BP). The solution candidates gradually approach their equilibrium positions that are achieved by the current population and historically best position in order to have a proper balance between diversification and intensification. The main procedure of this algorithm is given as follows:

Step 1: Initialization. Initial locations of particles are created randomly in an n -dimensional search space by:

$$x_i^j = x_{\min} + rand \times (x_{\max} - x_{\min}), \quad (5)$$

where x_i^j is the j th variable of the i th particle; x_{\max} and x_{\min} are the minimum and maximum allowable values of vector of variables, respectively; $rand$ is a random number at the interval $[0,1]$; and n is the number of particles;

Step 2: Evaluating candidate solutions. The objective function value of each particle is calculated;

Step 3: Updating the particle positions. In order to select the GP and BP for each candidate solution, the current population is sorted according to their objective function values in increasing order and then, GP and BP are chosen randomly from the first half and second half, respectively.

According to the above concepts, the particle's position is updated by:

$$x_i^j = \omega_1 \cdot [D.A.R1 + HB^j] + \omega_2 \cdot [D.A.R2 + GP^j] + \omega_3 \cdot [D.A.R3 + BP^j], \quad (6)$$

where x_i^j is the j th variable of the particle i ; ω_1 , ω_2 , and ω_3 are three parameters for measuring the relative importance of HB , GP , and BP , respectively ($\omega_1 + \omega_2 + \omega_3 = 1$); and $R1$, $R2$, and $R3$ are the uniformly distributed random numbers in the range of $[0, 1]$. The parameter A is defined as follows:

$$A = \left[\omega_1 \cdot (HB^j - x_i^j) \right] + \left[\omega_1 \cdot (GP^j - x_i^j) \right] + \left[\omega_1 \cdot (BP^j - x_i^j) \right]. \quad (7)$$

Parameter D is a descending function based on the number of iterations:

$$D = \left(\frac{iter}{iter_{\max}} \right)^{-\alpha}. \quad (8)$$

In order to have a fast convergence rate in the VPS, the role of BP is sometimes considered in updating the position formula. Therefore, for each particle, a parameter like p within $(0,1)$ is defined and it is compared with $rand$ (a random number uniformly distributed in the range of $[0,1]$) and if $p < rand$, then $\omega_3 = 0$ and $\omega_2 = 1 - \omega_1$.

Particles move towards HB and, therefore, self-adaptation is provided. Any particle has the chance to affect the new position of the other one; thus, the cooperation between the particles is supplied. Because of the p parameter, the effect of GP (good particle) is more than that of BP (bad particle) and, therefore, a completion is established;

Step 4: Handling the side constraints. There is a possibility of boundary violation when a particle moves to its new position. In the proposed algorithm, to handle boundary constraints, a harmony search-based approach is used [29]. In this technique, there is a possibility like Harmony Memory Considering Rate (HMCR) that specifies whether the violating component must be changed with the corresponding component of the historically best position of a

Pseudo code of Vibrating Particles System (VPS)

```

Initialize algorithm parameters
Create initial positions randomly by Eq. (6)
Evaluate the values of objective function and store HB
While maximum iterations is not fulfilled
  for each particle
    The GP and BP are chosen
    if  $P < rand$ 
       $\omega_3 = 0$  and  $\omega_2 = 1 - \omega_1$ 
    end if
    for each component
      New location is obtained by Eq. (7)
    end for
    Violated components are regenerated by harmony search-based handling approach
  end for
end while
The values of objective function are evaluated and HB is updated
end

```

Figure 1. Pseudocode of the vibrating particles system algorithm [14].

random particle or it should be determined randomly in the search space. Moreover, if the component of a historically best position is selected, there is a possibility like Pitch Adjusting Rate (PAR) that specifies whether this value should be exchanged with the neighboring value or not;

Step 5: Terminating condition check. Steps 2–4 are repeated until a termination criterion is fulfilled. Any terminating condition can be considered; however, in this study, the optimization process is terminated after a fixed number of iterations. The pseudocode of the VPS is provided in Figure 1.

4. Chaotic Vibrating Particles System (CVPS)

Six types of scenarios are used for embedding chaotic maps into VPS. Chaotic maps are involved in manipulating the selection of *GP* and *BP* and tuning the amount of *R1*, *R2*, *R3*, and *p* operators of the VPS algorithm.

The chaotic *GP* and *BP* selection operators improve exploration, whereas the chaotic tuning of *R1*, *R2*, *R3*, and *p* operators enhances the exploitation of the VPS algorithm. Ten different chaotic maps are utilized for the VPS algorithm. The applications of chaotic maps are tested on each of the foregoing operators, singly.

4.1. Chaotic maps

Deterministic nonlinear systems can exhibit nonlinear behavior which is sensitive to initial conditions. Such systems are called chaotic and their relative equations are chaotic maps. Recently, chaos and metaheuristics have been combined for different purposes. Chaotic metaheuristics utilize chaotic maps to control the value of parameters or incorporate chaotic search into the procedures of the metaheuristics in order to enrich the searching behavior and avoid being trapped in local

optimums [21,30]. Thus, by reviewing the literature, ten most relevant one-dimensional chaotic maps to tackle CVPS have been used in the present work.

The selected chaotic maps employed in this research include Chebyshev [31], circle [32], gauss/mouse [33], iterative [34], logistic [35], piecewise [23], sine [36], singer [37], sinusoidal [35], and tent [38]. The required information about these maps is provided in Table 1. According to Table 1, deterministic systems can also have chaotic behaviors. There is no random component in the following chaotic maps, but the chaotic behaviors of the equations are quite evident in the related figures. This set of chaotic maps with different behaviors has been chosen, while all of the chaotic maps start from 0.7 ($x_0 = 0.7$). The starting point can take any number between 0 and 1 (or -1 and 1 based on the chaotic map range). However, it should be noted that the vibration pattern of the chaotic maps is sensitive to their relative initial value. This study uses similar initial values to those used in [24].

5. Numerical examples

In order to evaluate the performance of the proposed CVPS techniques in solving the RCPSP, three case studies previously treated by other researchers are investigated.

5.1. Example 1

This example is a project with 15 activities and one resource type, previously solved by Anagnostopoulos and Koulinas [39]. The duration, predecessors, and daily resource demand of activities are illustrated in Figure 2. The maximum availability level of renewable resource is 14 units per day.

5.2. Example 2

This study devised a real highway bridge construction project as a numerical case study, as presented in [40].

Table 1. Chaotic maps data.

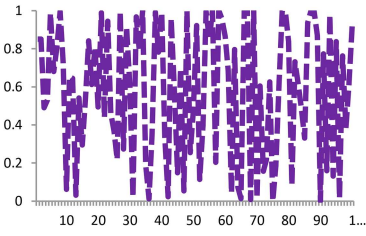
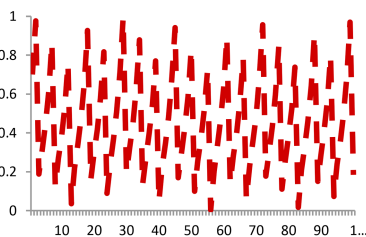
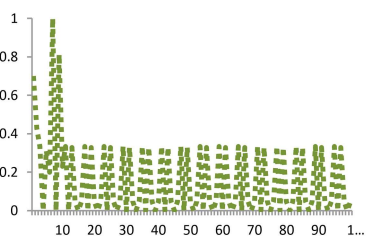
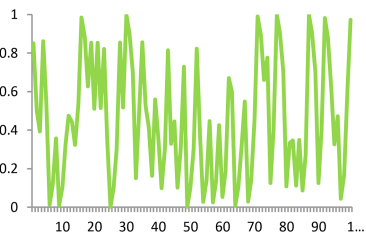
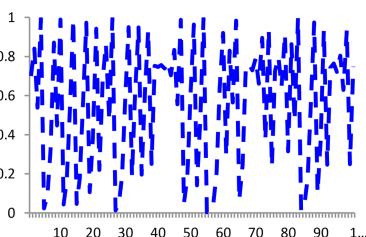
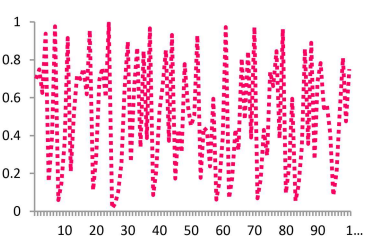
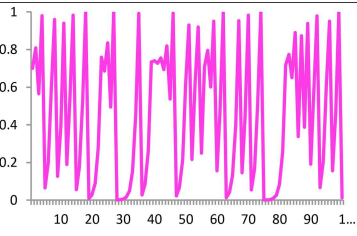
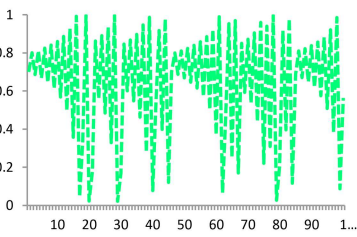
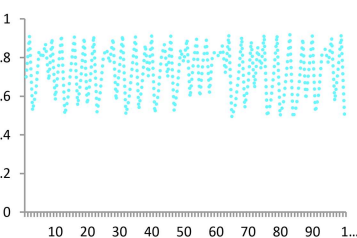
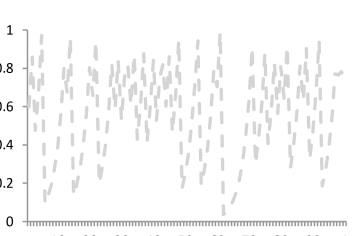
#	Map name	Map formula	Range	Distribution of the map between 0 and 1 after 100 iterations
1	Chebyshev [31]	$x_{i+1} = \cos(i \cos^{-1}(x_i))$	-1,1	
2	Circle [32]	$x_{i+1} = \text{mod}\left(x_i + b - \left(\frac{a}{2\pi}\right) \sin(2\pi x_i), 1\right)$ $a = 0.5, b = 0.2$	0,1	
3	Gauss / mouse [33]	$x_{i+1} = \begin{cases} 1 & x_i = 0 \\ \frac{1}{\text{mod}(x_i, 1)} & x_i \neq 0 \end{cases}$	0,1	
4	Iterative [34]	$x_{i+1} = \sin\left(\frac{a\pi}{x_i}\right), a = 0.7$	-1,1	
5	Logistic [35]	$x_{i+1} = ax_i(1 - x_i), a = 4$	0,1	
6	Piecewise [24]	$x_{i+1} = \begin{cases} \frac{x_i}{P} & 0 \leq x_i < P \\ \frac{x_i - P}{0.5 - P} & P \leq x_i < 0.5 \\ \frac{1 - P - x_i}{0.5 - P} & 0.5 \leq x_i < 1 - P \\ \frac{1 - x_i}{P} & 1 - P \leq x_i < 1 \end{cases}, P = 0.4$	0,1	

Table 1. Chaotic maps data (continued).

#	Map name	Map formula	Range	Distribution of the map between 0 and 1 after 100 iterations
7	Sine [36]	$x_{i+1} = \frac{a}{4} \sin(\pi x_i), a = 4$	0,1	
8	Singer [37]	$x_{(i+1)} = \mu(7.86x_i - 23.31x_i^2 + 28.75x_i^3 - 13.302875x_i^4),$ $\mu = 2.3$	0,1	
9	Sinusoidal [35]	$x_{i+1} = ax_i^2 \sin(\pi x_i), a = 2.3$	-1,1	
10	Tent [38]	$x_{i+1} = \begin{cases} \frac{x_i}{0.7} & x_i < 0.7 \\ \frac{10}{3}(1 - x_i) & x_i \geq 0.7 \end{cases}$	0,1	

This case study contains 44 activities by various daily resource demands. There are three types of renewable resources (e.g., labor, equipment, etc.) and maximum availability levels of resources are 12, 8, and 8 units per day. Figure 3 shows the precedence relationships of the project activities using arrow lines. The duration of the project activities is written on top of their respective circle nodes. The number of required resources is written below their respective circle nodes.

5.3. Example 3

The third case study was presented by Christodoulou [41] and studied by some other research studies, e.g., [10]. This project contains 17 activities and one type of resource with availability of 6 units per day. Precedence diagram of the project is shown in Figure 4.

The precedence relationships are illustrated by arrows. The duration of project activities is written on the top of their respective circle nodes. The number of required resources of each activity is written below the respective circle node.

6. Results and discussion

By employing ten chaotic maps, RCPSP examples are solved by MATLAB R2017a [42]. Based on the central limit theorem, when the sample size gets larger, the distribution of the sample mean converges to the normal distribution; the size of the sample must be equal to 30 or more [43]. Hence, each method has been run 30 times independently for all case studies. The Average (Avg.), Standard Deviation (SD), and

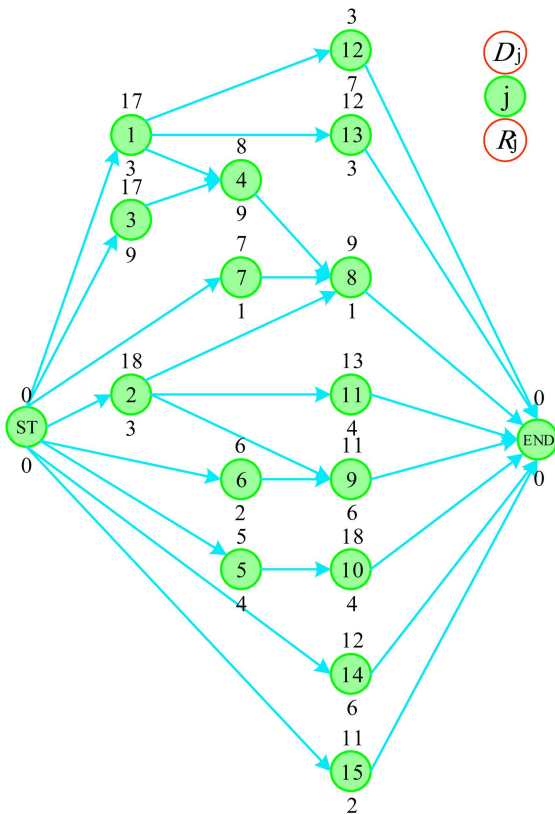


Figure 2. Project network of Example 1.

Success Rate (SR) of 30 runs for the cost of the last iteration are calculated to evaluate the stability and accuracy of algorithms. The performance of the VPS is dependent on the control parameters; thus, according to the performed sensitivity analysis and parameter settings in the literature [11,14,43,44], the values of parameters α , ω_1 , ω_2 , and p are set to 0.05, 0.3, 0.3, and 0.8, respectively. Here, the numbers of the population sizes are set to 50, 40, and 200 in Cases 1, 2, and 3 through 50, 100, and 500 iterations, respectively. In this paper, chaotic algorithms are coded as follows: (CVPS-operator name-map number). The second part of this code points to the operator's name. In the third part of this coding, 01 to 10 are assigned to Chebyshev, circle, gauss/mouse, iterative, logistic, piecewise, sine, singer, sinusoidal, and tent, respectively. For instance, "CVPS-GP-01" is a VPS algorithm with Chebyshev mapping of GP selection operator.

6.1. Results and discussion of Example 1

The convergence curves of VPS and CVPS techniques, which are applied to solving the second example, are shown in Figure 5. Results of using chaotic maps instead of VPS operators are depicted singly for each operator. The convergence curve of the VPS is shown in all figures for the simplicity of comparison. Figure 4 shows that except gauss/mouse and sinusoidal maps,

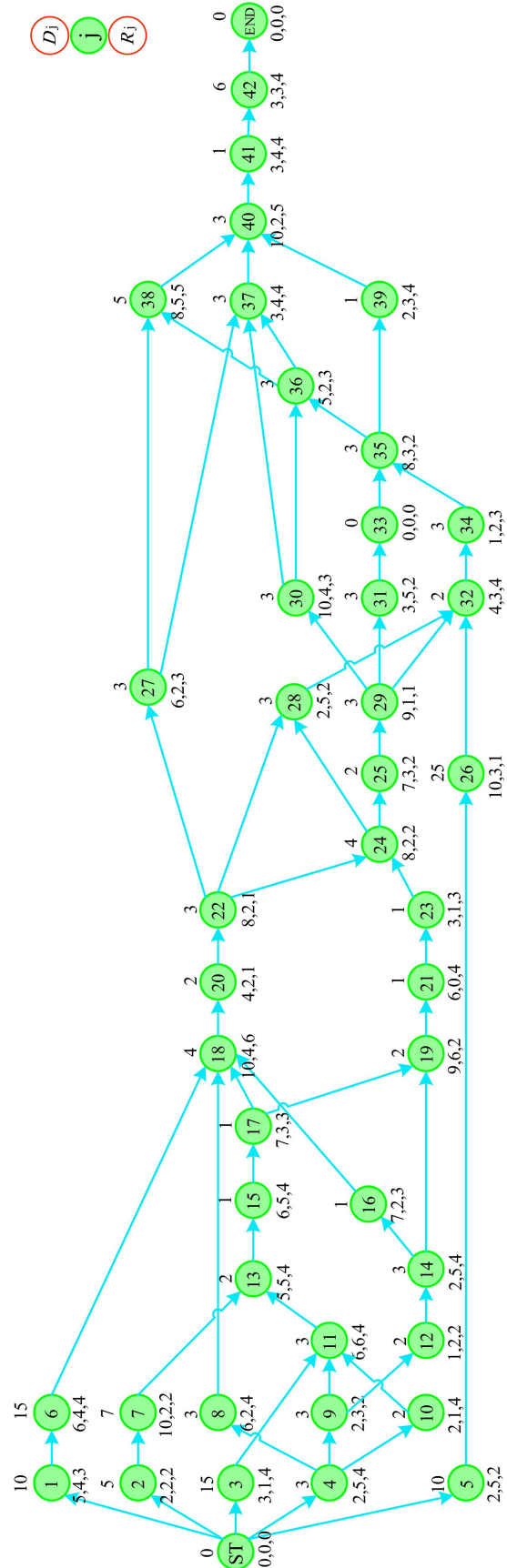


Figure 3. Project network of Example 2.

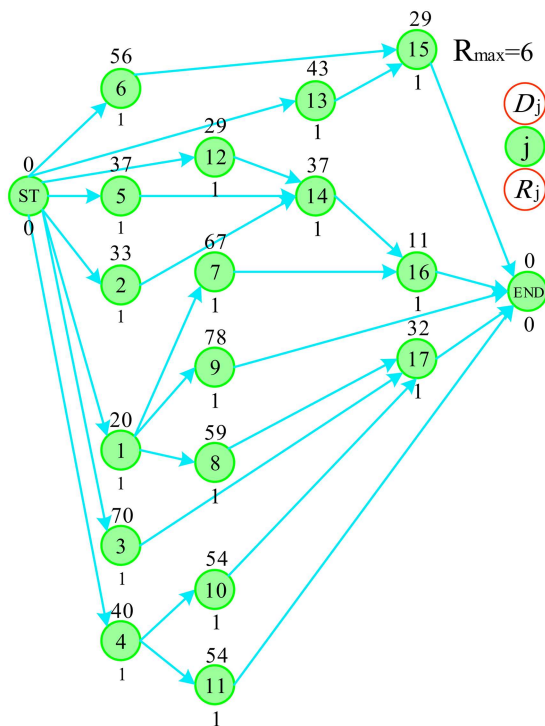


Figure 4. Project network of Example 3.

all other chaos-based GP selection operators are successful in improving the performance of the VPS in solving this example, and four Chebyshev, iterative, piecewise, and singer mappings present the best convergence of all other maps. Although gauss/mouse, piecewise, and sine chaotic BP selection operators do not improve the performance of the VPS in solving this

example, the application of Chebyshev, circle, iterative, logistic, singer, sinusoidal, and tent maps helps reduce the VPS cost and, also, iterative, singer, and tent mappings reach better solution than others. Besides, except gauss/mouse mapping, all other CVPS techniques with the chaotic operator $R1$ are able to outperform the VPS results and mapping with Chebyshev. In addition, piecewise maps have resulted in faster convergence. By using chaos-based maps as the operator $R2$, except the gauss/mouse mapping, other maps can improve the performance of VPS algorithm. Graphical curves of chaotic maps used as operators $R3$ in Figure 5 show that although gauss/mouse mapping cannot outperform the VPS results, other mapping methods have a better convergence speed than standard VPS. The best results have been achieved by using circle, logistic, and sinusoidal maps. Convergence curves of the CVPS- p show that all chaotic maps, except the gauss/mouse, are able to improve the VPS results by using p as a parameter and that the mapping of this key parameter by circle, logistic, singer, and sinusoidal maps improved the VPS algorithm more than other maps.

The statistical results of PSO, CBO, GA, HS, ICA, VPS, and CVPS algorithms for solving Example 1 are shown in Table 2. In this table, the best results of each parameter are highlighted in bold form. All of the algorithms could obtain the best duration in comparison to previous findings in the literature [8]. According to Table 2, except for gauss/mouse and sinusoidal maps, all other chaotic GP selection operators are able to improve the performance of the VPS, while Chebyshev, iterative, piecewise, and

Table 2. Statistical results of the chaotic Vibrating Particles System (VPS) for Example 1.

Algorithm	Avg.	SD	Worst	SR	Algorithm	Avg.	SD	Worst	SR	Algorithm	Avg.	SD	Worst	SR
PSO	117.77	1.1943	120	0.63	CBO	117.26	0.6397	120.00	0.80	GA	117.90	1.0289	120	0.43
HS	117.43	0.5040	118.00	0.57	ICA	117.56	0.8172	120.0	0.57	VPS	117.20	0.4068	118.00	0.80
CVPS-GP-01	117.00	0	117.00	1	CVPS-R1-01	117.00	0	117.00	1	CVPS-R3-01	117.13	0.3457	118.00	0.87
CVPS-GP-02	117.13	0.3457	118.00	0.87	CVPS-R1-02	117.13	0.3457	118.00	0.87	CVPS-R3-02	117.00	0	117.00	1
CVPS-GP-03	117.90	1.0289	120.00	0.43	CVPS-R1-03	117.43	0.8172	120.00	0.70	CVPS-R3-03	117.23	0.6261	120.00	0.83
CVPS-GP-04	117.00	0	117.00	1	CVPS-R1-04	117.13	0.3457	118.00	0.87	CVPS-R3-04	117.10	0.3051	118.00	0.90
CVPS-GP-05	117.03	0.1826	118.00	0.97	CVPS-R1-05	117.03	0.1826	118.00	0.97	CVPS-R3-05	117.00	0	117.00	1
CVPS-GP-06	117.00	0	117.00	1	CVPS-R1-06	117.00	0	117.00	1	CVPS-R3-06	117.13	0.3457	118.00	0.87
CVPS-GP-07	117.20	0.4498	118.00	0.73	CVPS-R1-07	117.13	0.3457	118.00	0.87	CVPS-R3-07	117.06	0.2537	118.00	0.93
CVPS-GP-08	117.00	0	117.00	1	CVPS-R1-08	117.10	0.3051	118.00	0.90	CVPS-R3-08	117.06	0.2537	118.00	0.93
CVPS-GP-09	117.23	0.6261	120.00	0.83	CVPS-R1-09	117.10	0.3051	118.00	0.90	CVPS-R3-09	117.00	0	117.00	1
CVPS-GP-10	117.17	0.3790	118.00	0.83	CVPS-R1-10	117.10	0.3051	118.00	0.90	CVPS-R3-10	117.10	0.3051	118.00	0.90
CVPS-BP-01	117.17	0.3790	118.00	0.83	CVPS-R2-01	117.00	0	117.00	1	CVPS-p-01	117.10	0.3051	118.00	0.90
CVPS-BP-02	117.03	0.1826	118.00	0.97	CVPS-R2-02	117.13	0.3457	118.00	0.87	CVPS-p-02	117.00	0	117.00	1
CVPS-BP-03	117.30	0.6513	120.00	0.77	CVPS-R2-03	118.33	1.2130	120.00	0.30	CVPS-p-03	117.56	0.8172	120.00	0.57
CVPS-BP-04	117.00	0	117.00	1	CVPS-R2-04	117.10	0.3051	118.00	0.90	CVPS-p-04	117.00	0	117.00	1
CVPS-BP-05	117.10	0.3051	118.00	0.90	CVPS-R2-05	117.03	0.1826	118.00	0.97	CVPS-p-05	117.10	0.3051	118.00	0.90
CVPS-BP-06	117.20	0.4498	118.00	0.73	CVPS-R2-06	117.16	0.3790	118.00	0.83	CVPS-p-06	117.13	0.3457	118.00	0.87
CVPS-BP-07	117.23	0.6261	120.00	0.83	CVPS-R2-07	117.03	0.1826	118.00	0.97	CVPS-p-07	117.20	0.4498	118.00	0.73
CVPS-BP-08	117.00	0	117.00	1	CVPS-R2-08	117.03	0.1826	118.00	0.97	CVPS-p-08	117.00	0	117.00	1
CVPS-BP-09	117.10	0.3051	118.00	0.90	CVPS-R2-09	117.00	0	117.00	1	CVPS-p-09	117.00	0	117.00	1
CVPS-BP-10	117.00	0	117.00	1	CVPS-R2-10	117.00	0	117.00	1	CVPS-p-10	117.33	0.6609	120.00	0.73

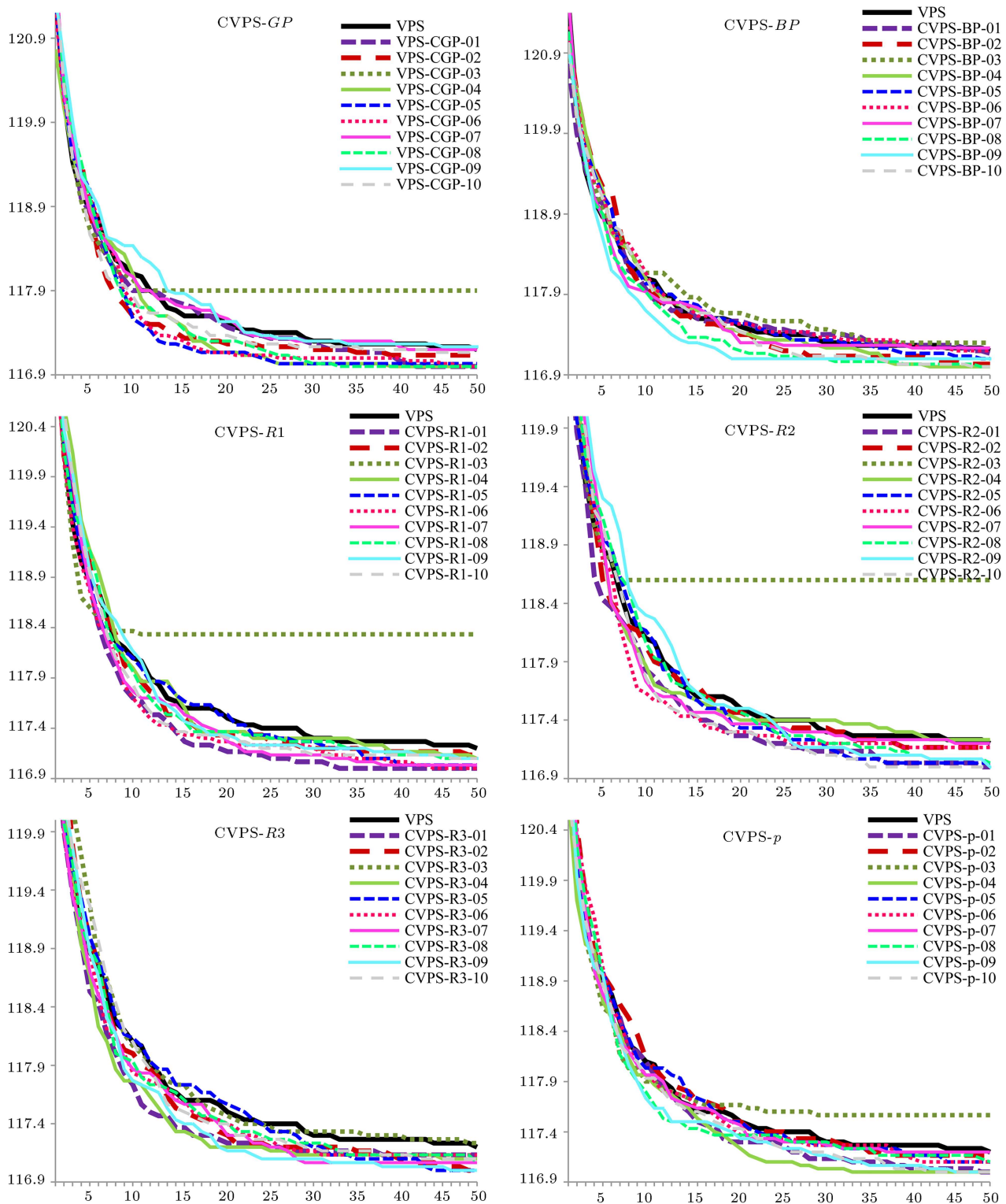


Figure 5. Convergence curves for the Chaotic Vibrating Particles System (CVPS) algorithms for solving Example 1.

singer *GP* selection operators show worse results than other maps in terms of average best cost (117.00), standard deviation (0), worst cost (117.00), and success rate (1.00). On the other hand, all of the chaotic *BP* selection operators, except gauss/mouse, have

successfully improved the performance of the VPS algorithm and iterative, singer, and tent *BP* selection operators have the best performance (Avg = 117.00, SD = 0, worst cost = 117.00, and SR = 1.00) for solving this example. Besides, except gauss/mouse

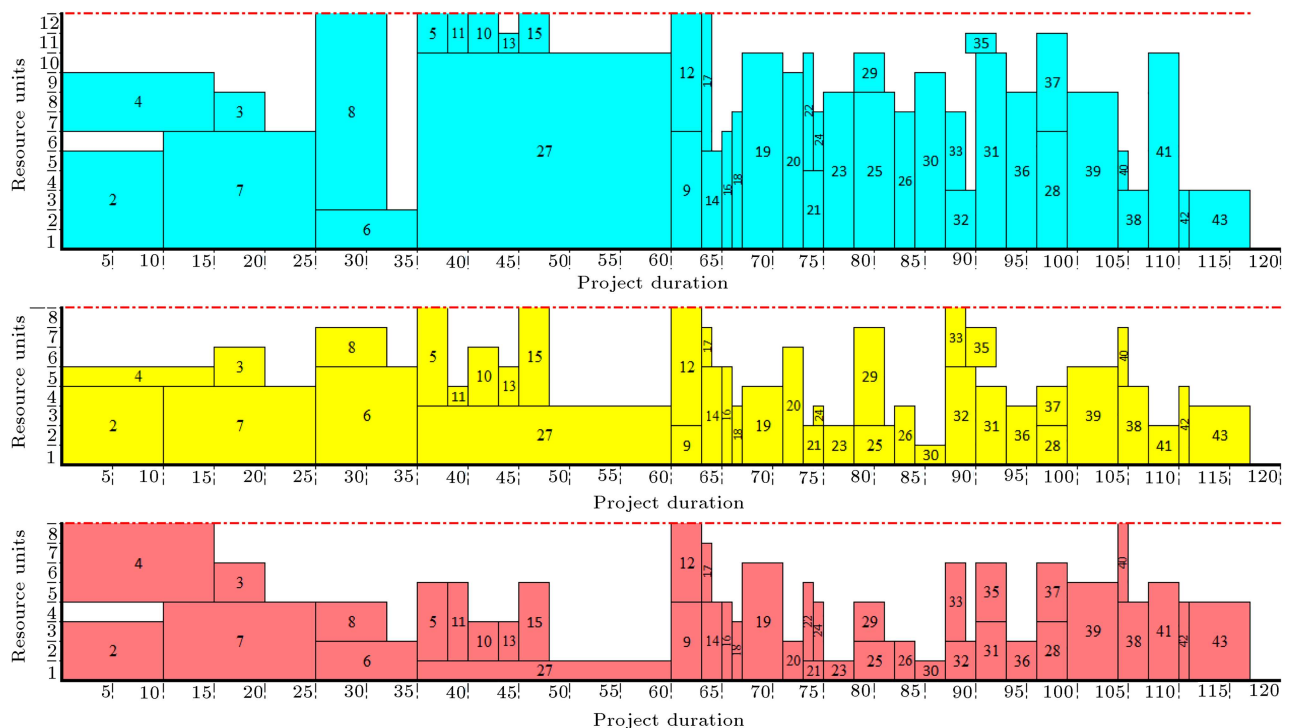


Figure 6. Resource allocation profile of Example 1.

mapping, all other CVPS algorithms with chaotic operators $R1$ were successful in outperforming the VPS and Chebyshev, and piecewise mappings showed the best performances in terms of statistical results (Avg = 117.00, SD = 0, worst cost = 117.00, and SR = 1.00). By using chaos-based maps as operators $R2$, except the gauss/mouse mapping, other maps could improve the performance of the VPS algorithm and Chebyshev, sinusoidal, and tent mappings presented the best possible results at different run times based on experimental results presented in Table 2 (Avg = 117.00, SD=0, worst cost=117.00, and SR=1.00). Although gauss/mouse mapping of $R3$ cannot outperform the VPS results, other mapping methods have good performance in enhancing the VPS and circle, logistic, and sinusoidal maps provide robustness for VPS at all run times (Avg=117.00, SD=0, worst cost=117.00, and SR=1.00). The statistical results of using chaotic maps as p parameter showed that all chaotic maps, except the gauss/mouse, could improve the VPS results and mapping of this parameter with circle, logistic, singer, and sinusoidal maps resulted in the robustness of the VPS algorithm.

Figure 6 illustrates the corresponding optimal solution schedule of the first example obtained by the chaos-based VPS algorithms for the first example. Moreover, sequences and start/finish times of all activities of the projects, with dummy activities not being included, are provided and the schedule describes the resource-allocation profile. The final project duration

was determined as 117 days by using the proposed method after resource leveling.

6.2. Results and discussion of Example 2

Figure 7 shows the convergence curves of VPS and CVPS techniques employed to solve the second example. Results of chaotic maps used as the VPS operators were drawn singly for all operators. The convergence curve of the VPS is shown in all figures for simplicity of the comparison. Although gauss/mouse, logistic, sine, and sinusoidal maps could not be as successful as GP selection operators, Chebyshev, circle, iterative, piecewise, sine, singer, and tent maps successfully improved the performance of the VPS in solving this example. In addition, Chebyshev and piecewise maps were more successful than others in using them as GP selection operators. All chaotic BP selection operators improved the performance of the VPS in solving this example and piecewise gave the best and fastest convergence result among all chaotic BP selection operators. Although gauss/mouse chaotic operator $R1$ did not improve the performance of the VPS in solving this example, other maps improved the VPS cost and piecewise mapping was better and faster than others in adjusting $R1$ parameter. Moreover, except gauss/mouse mapping, adjusting parameter $R2$ to all other CVPS techniques could outperform the VPS results and mapping with the sinusoidal map was successful in outperforming other algorithms. All chaos-based maps as the operator $R2$, except the gauss/mouse mapping, could improve

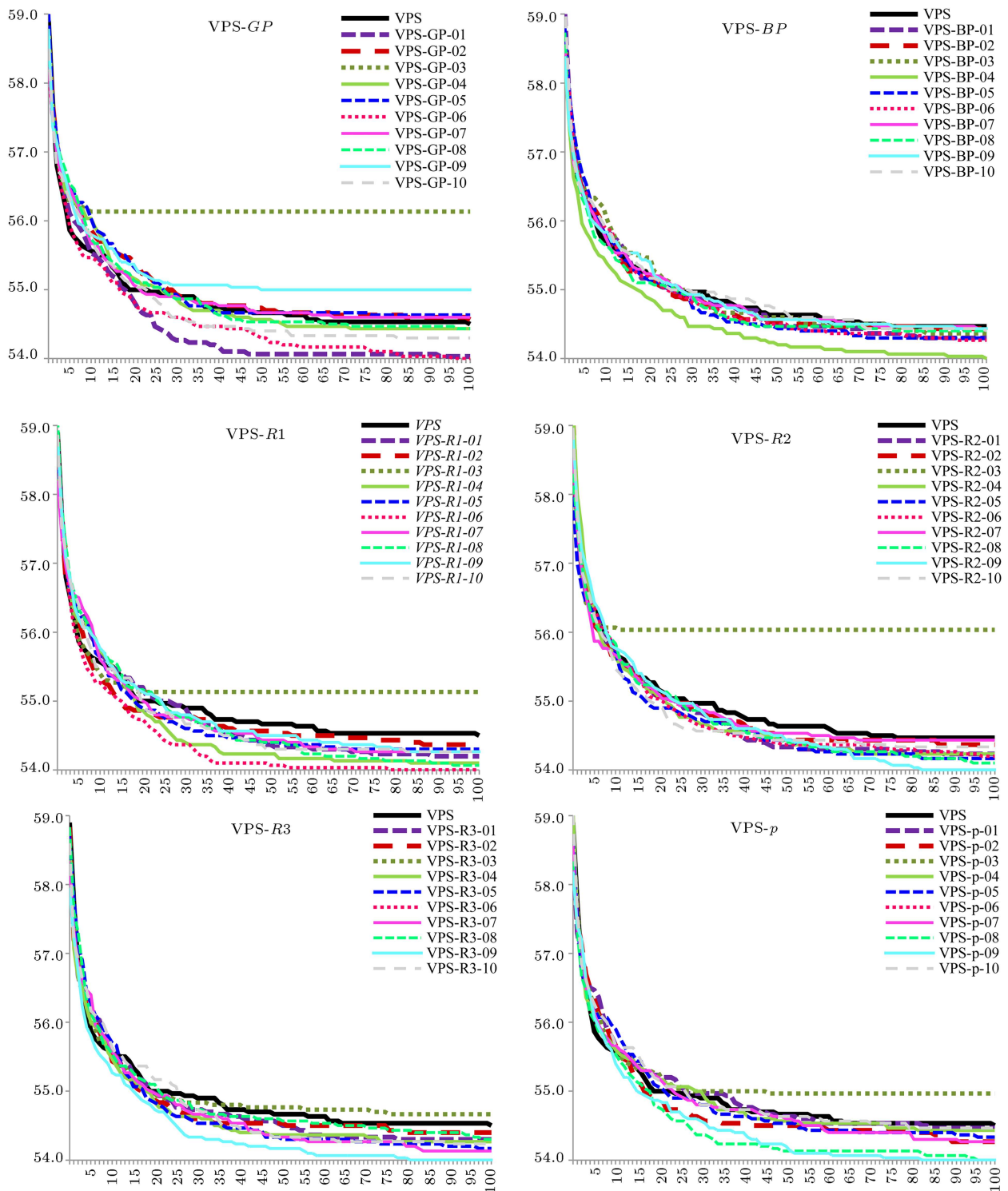


Figure 7. Convergence curves for the Chaotic Vibrating Particles System (CVPS) algorithms for solving Example 2.

the performance of the VPS algorithm and, also, mapping with sinusoidal led to more robust solutions. The graphical curve of chaotic maps used as the operator *R3* showed that while gauss/mouse mapping could not outperform the VPS results, other mapping methods had a better convergence speed than the standard VPS

and adjusting *p* values by the sinusoidal map led to higher robustness.

The statistical results of PSO, CBO, GA, HS, ICA, VPS, and different CVPS algorithms for solving Example 2 are shown in Table 3. Here, the best results of each parameter are highlighted in bold form.

Table 3. Statistical results of the chaotic VPS for Example 2.

Algorithm	Avg.	SD	Worst	SR	Algorithm	Avg.	SD	Worst	SR	Algorithm	Avg.	SD	Worst	SR
PSO	54.97	1.0981	58.00	0.53	CBO	54.57	0.8172	56.00	0.63	GA	54.63	0.8087	56.00	0.57
HS	54.60	0.8550	57.00	0.60	ICA	54.57	0.8172	56.00	0.63	VPS	54.53	0.7303	55.00	0.60
CVPS- <i>GP</i> -01	54.03	0.1826	55.00	0.97	CVPS- <i>R1</i> -01	54.20	0.5509	56.00	0.87	CVPS- <i>R3</i> -01	54.30	0.5350	56.00	0.73
CVPS- <i>GP</i> -02	54.57	0.8172	56.00	0.63	CVPS- <i>R1</i> -02	54.36	0.7649	57.00	0.77	CVPS- <i>R3</i> -02	54.40	0.7240	56.00	0.73
CVPS- <i>GP</i> -03	54.13	0.9371	57.00	0.07	CVPS- <i>R1</i> -03	55.13	0.8996	56.00	0.26	CVPS- <i>R3</i> -03	54.67	0.8841	57.00	0.57
CVPS- <i>GP</i> -04	54.43	0.1826	57.00	0.70	CVPS- <i>R1</i> -04	54.10	0.3051	55.00	0.90	CVPS- <i>R3</i> -04	54.27	0.6915	57.00	0.83
CVPS- <i>GP</i> -05	54.63	0.8087	56.00	0.57	CVPS- <i>R1</i> -05	54.30	0.7497	57.00	0.83	CVPS- <i>R3</i> -05	54.17	0.4611	56.00	0.87
CVPS- <i>GP</i> -06	54.00	0	54.00	1	CVPS- <i>R1</i> -06	54.00	0	54.00	1	CVPS- <i>R3</i> -06	54.13	0.3457	55.00	0.87
CVPS- <i>GP</i> -07	54.60	0.8550	57.00	0.60	CVPS- <i>R1</i> -07	54.26	0.5833	56.00	0.80	CVPS- <i>R3</i> -07	54.33	0.6065	56.00	0.73
CVPS- <i>GP</i> -08	54.43	0.7279	56.00	0.70	CVPS- <i>R1</i> -08	54.06	0.2537	55.00	0.93	CVPS- <i>R3</i> -08	54.30	0.6513	57.00	0.77
CVPS- <i>GP</i> -09	55.00	0.9826	57.00	0.43	CVPS- <i>R1</i> -09	54.26	0.5833	56.00	0.80	CVPS- <i>R3</i> -09	54.00	0	54.00	1
CVPS- <i>GP</i> -10	54.30	0.7022	57.00	0.80	CVPS- <i>R1</i> -10	54.26	0.5833	56.00	0.80	CVPS- <i>R3</i> -10	54.23	0.5040	56.00	0.80
CVPS- <i>BP</i> -01	54.30	0.5960	56.00	0.77	CVPS- <i>R2</i> -01	54.23	0.6789	57.00	0.87	CVPS- <i>p</i> -01	54.46	0.8193	56.00	0.73
CVPS- <i>BP</i> -02	54.43	0.8172	56.00	0.73	CVPS- <i>R2</i> -02	54.43	0.6789	56.00	0.67	CVPS- <i>p</i> -02	54.26	0.7397	57.00	0.86
CVPS- <i>BP</i> -03	54.37	0.7184	56.00	0.77	CVPS- <i>R2</i> -03	56.03	0.9643	58.00	0.10	CVPS- <i>p</i> -03	54.96	0.9643	57.00	0.43
CVPS- <i>BP</i> -04	54.00	0	54.00	1	CVPS- <i>R2</i> -04	54.23	0.5683	56.00	0.83	CVPS- <i>p</i> -04	54.43	0.6789	56.00	0.67
CVPS- <i>BP</i> -05	54.30	0.6513	57.00	0.80	CVPS- <i>R2</i> -05	54.16	0.3790	55.00	0.83	CVPS- <i>p</i> -05	54.33	0.5467	56.00	0.73
CVPS- <i>BP</i> -06	54.27	0.5833	56.00	0.80	CVPS- <i>R2</i> -06	54.20	0.5509	56.00	0.73	CVPS- <i>p</i> -06	54.26	0.5833	56.00	0.76
CVPS- <i>BP</i> -07	54.43	0.7279	56.00	0.70	CVPS- <i>R2</i> -07	54.43	0.7279	56.00	0.70	CVPS- <i>p</i> -07	54.46	0.7303	56.00	0.67
CVPS- <i>BP</i> -08	54.40	0.7240	56.00	0.73	CVPS- <i>R2</i> -08	54.10	0.3051	55.00	0.90	CVPS- <i>p</i> -08	54.03	0	54.00	1
CVPS- <i>BP</i> -09	54.43	0.7279	56.00	0.70	CVPS- <i>R2</i> -09	54.03	0	54.00	1	CVPS- <i>p</i> -09	54.03	0	54.00	1
CVPS- <i>BP</i> -10	54.40	0.6747	56.00	0.70	CVPS- <i>R2</i> -10	54.30	0.7022	57.00	0.80	CVPS- <i>p</i> -10	54.46	0.7761	56.00	0.70

All of the algorithms could obtain the best duration in comparison to those obtained in the literature [45]. As listed in Table 3, gauss/mouse, logistic, sine, and sinusoidal maps cannot obtain the results of the VPS, while other chaotic *GP* selection operators could improve the performance of the VPS and piecewise *GP* selection operator presented worse results than other maps. On the other hand, all chaotic *BP* selection operators successfully improved the performance of the VPS algorithm and the best influence as *BP* selection operator was presented by iterative map. Besides, except for gauss/mouse mapping, all other CVPS algorithms with chaotic adjusting operators *R1* were successful in outperforming the VPS and piecewise mappings had the best performances. According to the results shown in Table 3, adjusting *R2* to all maps provided better results than the standard VPS and tent provided the best performance for the second example. Further, as shown in this table, except gauss/mouse mapping, all other CVPS techniques with chaotic operators *R3* could outperform the VPS results and mapping of *R3* with sinusoidal map was successful at all run times. The statistical results of chaotic maps used as the operator *p* showed that all of the chaotic maps, except the gauss/mouse, were able to improve the VPS results and mapping of this parameter with singer and sinusoidal maps improved the robustness of the VPS algorithm.

Figure 8 illustrates the corresponding optimal solution schedule of Example 2 obtained by the chaos-based VPS algorithms for Example 2. Moreover, the schedule describes the resource-allocation profile, and

the sequences with start/finish times of all activities of the projects are provided with no dummy activities being included. The final project duration is 54 days after resource leveling by the proposed methods.

6.3. Results and discussion for Example 3

Figure 9 shows the convergence curves of all mapping techniques for solving the third example. For simplicity of the comparison, the convergence curve of the VPS for this case is shown in all figures. As can be seen from Figure 9, Chebyshev, iterative, piecewise, singer, and sinusoidal chaotic *GP* selection operators could improve the performance of VPS in solving the third example. On the other hand, piecewise mapping presented better convergence than other maps for the *GP* selection operator. All chaotic *BP* selection operators were able to improve the performance of VPS in solving the third example, and iterative mapping had the best efficiency in increasing the speed and reducing cost of this example. Besides, except gauss/mouse mapping, all other CVPS techniques with chaotic operators *R1* were successful in outperforming the VPS results. By using chaos-based maps as the adjusting operator *R2*, the gauss/mouse, logistic, and sinusoidal mappings cannot obtain the results of standard VPS and other maps could improve the performance of VPS algorithm as the adjusting operator *R2*. According to this table, the sinusoidal adjusting operator *R2* had the best convergence. However, adjusting parameter *R3* with gauss/mouse and singer cannot outperform the VPS results. Other mapping methods had better performances in enhancing the VPS and, among these

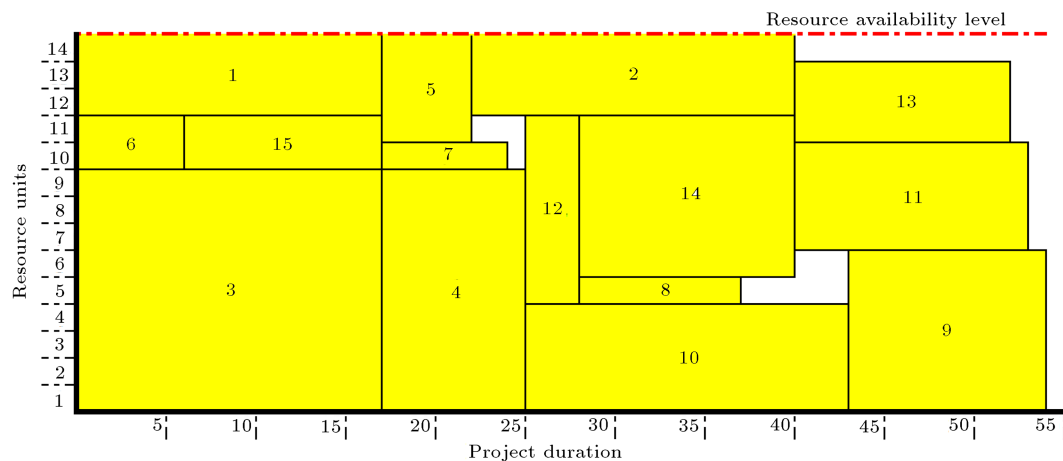


Figure 8. Resource allocation profile of Example 2.

Table 4. Statistical results of the chaotic VPS for Example 3.

Algorithm	Avg.	SD	Worst	SR	Algorithm	Avg.	SD	Worst	SR	Algorithm	Avg.	SD	Worst	SR
PSO	133.67	0.4795	134	0.33	CBO	133.57	0.5040	134	0.43	GA	133.73	0.4498	134	0.27
HS	133.73	0.4498	134	0.27	ICA	133.60	0.4983	134	0.40	VPS	133.63	0.4901	134	0.37
CVPS-GP-01	133.53	0.5074	134	0.47	CVPS-R1-01	133.43	0.5040	134	0.57	CVPS-R3-01	133.50	0.5085	134	0.50
CVPS-GP-02	133.67	0.4795	134	0.33	CVPS-R1-02	133.60	0.4983	134	0.40	CVPS-R3-02	133.53	0.5074	134	0.47
CVPS-GP-03	133.33	1.2685	140	0.10	CVPS-R1-03	133.73	0.4498	134	0.27	CVPS-R3-03	133.63	0.4901	134	0.37
CVPS-GP-04	133.60	0.4983	134	0.40	CVPS-R1-04	133.50	0.5085	134	0.50	CVPS-R3-04	133.50	0.5085	134	0.50
CVPS-GP-05	133.63	0.4901	134	0.37	CVPS-R1-05	133.47	0.5074	134	0.53	CVPS-R3-05	133.47	0.5074	134	0.53
CVPS-GP-06	133.50	0.5085	134	0.50	CVPS-R1-06	133.43	0.5040	134	0.57	CVPS-R3-06	133.53	0.5074	134	0.47
CVPS-GP-07	133.77	0.4302	134	0.23	CVPS-R1-07	133.50	0.5085	134	0.50	CVPS-R3-07	133.53	0.5074	134	0.47
CVPS-GP-08	133.53	0.5074	134	0.47	CVPS-R1-08	133.53	0.5074	134	0.47	CVPS-R3-08	133.67	0.4795	134	0.33
CVPS-GP-09	133.57	0.5040	134	0.43	CVPS-R1-09	133.50	0.5085	134	0.50	CVPS-R3-09	133.43	0.5040	134	0.57
CVPS-GP-10	133.67	0.4795	134	0.33	CVPS-R1-10	133.47	0.5074	134	0.53	CVPS-R3-10	133.50	0.5085	134	0.50
CVPS-BP-01	133.57	0.5040	134	0.43	CVPS-R2-01	133.43	0.5040	134	0.57	CVPS-p-01	133.67	0.4795	134	0.33
CVPS-BP-02	133.57	0.5040	134	0.43	CVPS-R2-02	133.47	0.5074	134	0.53	CVPS-p-02	133.57	0.5040	134	0.43
CVPS-BP-03	133.50	0.5085	134	0.50	CVPS-R2-03	134.13	0.8342	136	0.10	CVPS-p-03	133.80	0.4068	134	0.20
CVPS-BP-04	133.37	0.4901	134	0.63	CVPS-R2-04	133.53	0.5074	134	0.47	CVPS-p-04	133.50	0.5085	134	0.50
CVPS-BP-05	133.60	0.4983	134	0.40	CVPS-R2-05	133.43	0.5040	134	0.57	CVPS-p-05	133.60	0.4983	134	0.40
CVPS-BP-06	133.50	0.5085	134	0.50	CVPS-R2-06	133.57	0.5040	134	0.43	CVPS-p-06	133.57	0.5040	134	0.43
CVPS-BP-07	133.50	0.5085	134	0.50	CVPS-R2-07	133.57	0.5040	134	0.43	CVPS-p-07	133.70	0.4661	134	0.30
CVPS-BP-08	133.40	0.4983	134	0.60	CVPS-R2-08	133.40	0.4983	134	0.60	CVPS-p-08	133.26	0.4498	134	0.73
CVPS-BP-09	133.53	0.5074	134	0.47	CVPS-R2-09	133.37	0.4901	134	0.63	CVPS-p-09	133.53	0.5074	134	0.47
CVPS-BP-10	133.43	0.5040	134	0.57	CVPS-R2-10	133.60	0.4983	134	0.40	CVPS-p-10	133.60	0.4983	134	0.40

maps, the sinusoidal chaotic operator showed the best statistical results. Convergence curves of CVPS- p showed that Chebyshev, gauss/mouse, sine, and tent could improve the VPS results, while the application of circle, iterative, logistic, piecewise, singer, and sinusoidal maps as adjusting operator of p parameter improved the convergence speed of VPS and mapping of this parameter by the singer map achieved the best convergence results for the VPS among all results of this example.

The statistical results of PSO, CBO, GA, HS, ICA, VPS, and different CVPS algorithms in solving Example 3 are shown in Table 4. In this table, the best results of each operator are highlighted in

bold form. All of the algorithms could obtain better results than those obtained in the literature [8,39]. According to this table, Chebyshev, iterative, logistic, piecewise, singer, sinusoidal, and tent chaotic GP selection operators could improve the performance of VPS and piecewise GP selection operator showed worse results than others. On the other hand, all chaotic BP selection operators successfully improved the performance of the VPS algorithm and iterative map had the best influence as the BP selection operator. Besides, except gauss/mouse mapping, all other CVPS algorithms with chaotic operators R1 were successful in outperforming the VPS and Chebyshev, and piecewise mappings had the best performances. Considering

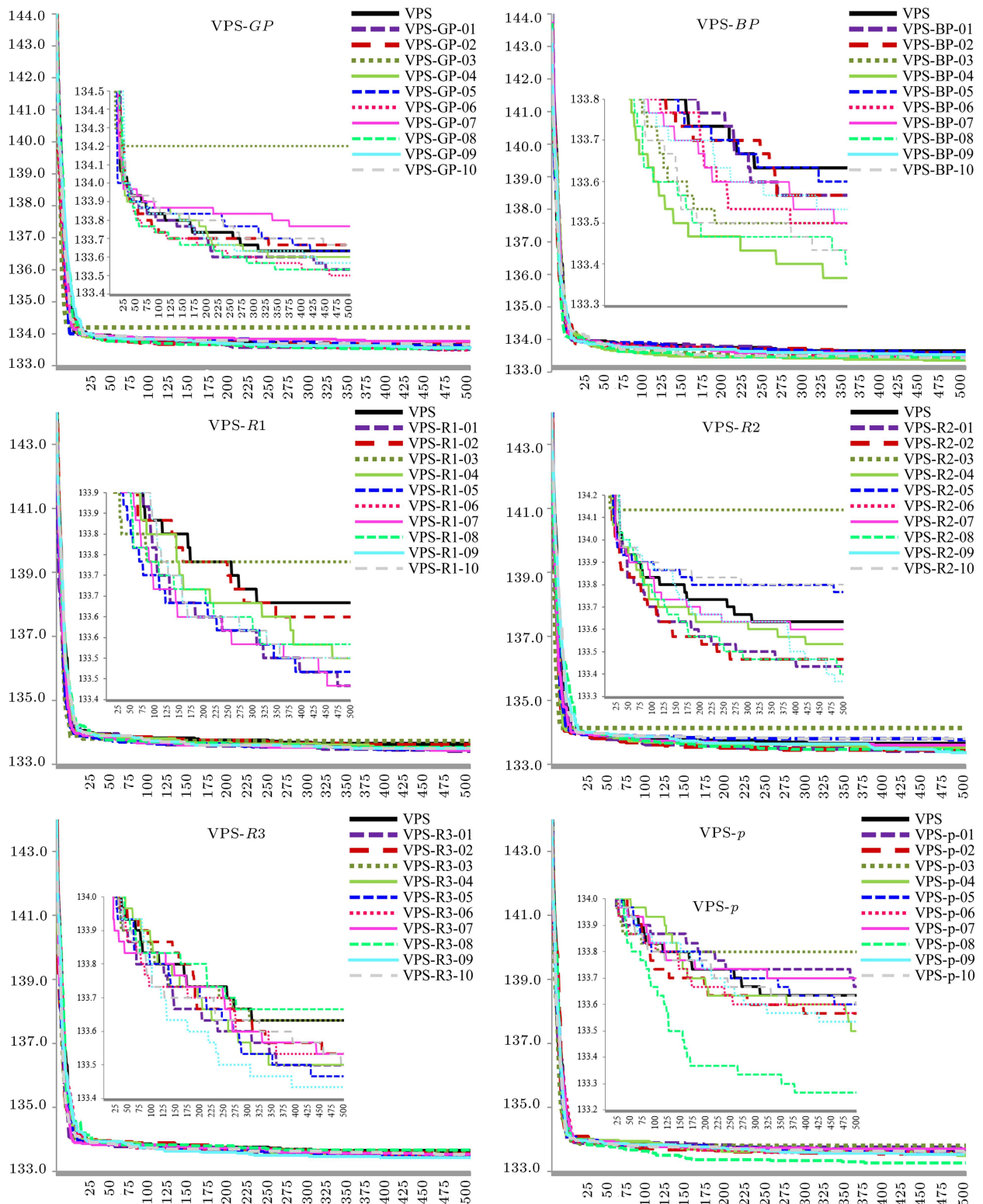


Figure 9. Convergence curves for the Chaotic Vibrating Particles System (CVPS) algorithms for solving Example 3.

the results presented in Table 4, the mapping of $R2$ with sinusoidal map provided the best performance for the third example in comparison with other adjusting methods of this parameter. According to this table, adjusting the $R3$ parameter by singer map produced

less satisfactory results in terms of statistics, while all other maps were able to outperform the standard VPS and sinusoidal map provided better performance than others. Moreover, adjusting the p parameter with Chebyshev, gauss/mouse, and sine maps cannot im-

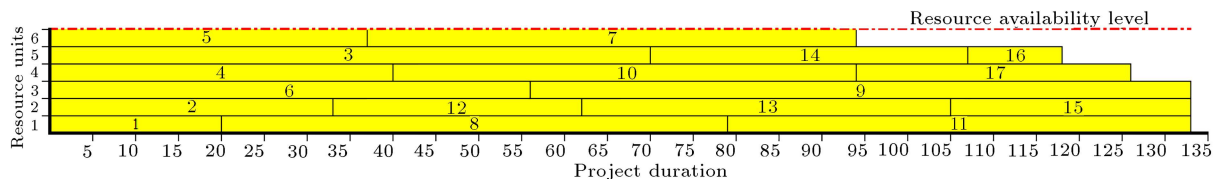


Figure 10. Resource allocation profile of Example 3.

prove the performance of VPS, while adjusting p value with other mapping methods enhances the performance of this algorithm and singer presents the best results for this algorithm.

The optimal schedule obtained by the CVPS algorithms for Example 3 is shown in Figure 10. Furthermore, the sequences and start/finish times of all activities of the projects whose dummy activities are not included are provided and the schedule describes the resource-allocation profile. The proposed method managed to compute the final project duration as 133 days after resource leveling.

7. Conclusions

Employing chaos as a technique for adjusting some parameters of the metaheuristic algorithms has become an interesting research topic among researchers in recent years. This study applied chaos to the standard VPS and developed a set of different chaotic VPSs. In this paper, six different Chaotic Vibrating Particles System (CVPS) algorithms using ten different chaotic maps were proposed. By comparing different chaotic VPS algorithms, the algorithm which used the piecewise map as its GP selection parameter (CVPS- GP -06) had the best results. Moreover, the results revealed that the application of iterative map was better than other mappings for BP selection parameter. Other results of this research showed that the application of Chebyshev and piecewise mappings as the operator $R1$ had additional improvement to the results. It was also found that both the operators $R1$ and $R2$ could be improved by replacement with sinusoidal chaotic maps. Finally, the results of using chaotic maps as the operator p revealed that singer map provided the best results of all other maps. Statistical results and the success rates of the Chaos-based VPS algorithms suggested that the adjusted algorithms could clearly improve the robustness of the global optimality and they also could enhance the quality of the results. For future works, it would be interesting to utilize chaos in enhanced vibrating particles system and solve engineering optimization problems using these algorithms.

References

1. Sebt, M.H., Zarandi, M.H.F., and Alipouri, Y. "Genetic algorithms to solve resource-constrained project scheduling problems with variable activity durations", *Int. J. Civ. Eng.*, **11**, pp. 189–198 (2013).
2. Kolisch, R. and Sprecher, A. "PSPLIB - A project scheduling problem library", *Eur. J. Oper. Res.*, **96**, pp. 205–216 (1996).
3. Ballestin, F. "When it is worthwhile to work with the stochastic RCPSP?", *J. Sched.*, **10**, pp. 153–166 (2007).
4. Hartmann, S. and Drexel, A. "Project scheduling with multiple modes: a comparison of exact algorithms", *Networks*, **32**, pp. 283–297 (1998).
5. Harvey, R.T. and Patterson, J.H. "An implicit enumeration algorithm for the time-cost tradeoff problem in project network analysis", *Found. Control. Eng.*, **4**, pp. 107–117 (1979).
6. Hindelang, T. and Muth, J. "A dynamic programming algorithm for decision CPM networks", *Oper. Res.*, **27**, pp. 225–241 (1979).
7. Chen, W., Shi, Y., Teng, H., et al. "An efficient hybrid algorithm for resource-constrained project scheduling", *Inf. Sci. (Ny)*, **180**, pp. 1031–1039 (2010).
8. Giran, O., Temur, R., and Bekdas, G. "Resource constrained project scheduling by harmony search algorithm", *KSCE J. Civ. Eng.*, **21**, pp. 479–487 (2017).
9. Kaveh, A., Khanzadi, M., and Alipour, M. "Fuzzy resource constraint project scheduling problem using CBO and CSS algorithms", *Int. J. Civ. Eng.*, **14**(5), pp. 325–337 (2016).
10. Khanzadi, M., Kaveh, A., Alipour, M., and Aghmiani, H.K. "Application of CBO and CSS for resource allocation and resource leveling problem", *Iran. J. Sci. Technol. - Trans. Civ. Eng.*, **40**, pp. 119–132 (2015).
11. Kaveh, A. and Vazirinia, Y. "Optimization of tower crane location and material quantity between supply and demand points: A comparative study", *Period. Polytech. Civ. Eng.*, **62**, pp. 732–745 (2018).
12. Kaveh, A., Hosseini Vaez, S.R., and Hosseini, P. "Enhanced vibrating particles system algorithm for damage identification of truss structures", *Sci. Iran., Trans. A, Civ. Eng.*, **26**(1), pp. 246–256 (2017).
13. Kaveh, A. and Zolghadr, A. "Comparison of nine metaheuristic algorithms for optimal design of truss structures with frequency constraints", *Adv. Eng. Softw.*, **76**, pp. 9–30 (2014).
14. Kaveh, A. and Ilchi Ghazaan, M. "A new metaheuristic algorithm: vibrating particles system", *Sci. Iran., Trans. A, Civ. Eng.*, **24**, pp. 1–32 (2017).

15. Golberg, D.E. "Genetic algorithms in search, optimization, and machine learning", *Addison-Wesley Publishing Company*, Reading, Massachusetts (1989).
16. Eberhart, R., and Kennedy, J. "A new optimizer using particle swarm theory", In: *Proceedings of the Sixth International Symposium on Micro Machine and Human Science.*, Nagoya, Japan, pp. 39–43 (1995).
17. Kaveh, A. and Farhoudi, N. "A new optimization method: Dolphin echolocation", *Adv. Eng. Softw.*, **59**, pp. 53–70 (2013).
18. Atashpaz-Gargari, E. and Lucas, C. "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition", In: *2007 IEEE Congress on Evolutionary Computation*, pp. 4661–4667 (2007).
19. Kaveh, A., Motie Share, M.A., and Moslehi, M. "Magnetic charged system search: a new meta-heuristic algorithm for optimization", *Acta Mech.*, **224**, pp. 85–107 (2013).
20. Kaveh, A. and Mahdavi, V.R. "Colliding bodies optimization method for optimum discrete design of truss structures", *Comput. Struct.*, **139**, pp. 43–53 (2014).
21. Geem, Z.W., Kim, J.H., and Loganathan, G.V. "A new heuristic optimization algorithm: Harmony search", *Simulation*, **76**(2), pp. 60–68 (2001).
22. Kaveh, A., *Advances in Metaheuristic Algorithms for Optimal Design of Structures*, 2nd Ed. Springer International Publishing, Cham, Switzerland (2017).
23. Mirjalili, S. and Gandomi, A.H. "Chaotic gravitational constants for the gravitational search algorithm", *Appl. Soft Comput. J.*, **53**, pp. 407–419 (2017).
24. Saremi, S., Mirjalili, S., and Lewis, A. "Biogeography-based optimisation with chaos", *Neural Comput. Appl.*, **25**, pp. 1077–1097 (2014).
25. Kaveh, A., Sheikholeslami, R., Talatahari, S., and Keshvari-Ilkhichi, M. "Chaotic swarming of particles: A new method for size optimization of truss structures", *Adv. Eng. Softw.*, **67**, pp. 136–147 (2014).
26. Gandomi, A.H. and Yang, X.S. "Chaotic bat algorithm", *J. Comput. Sci.*, **5**, pp. 224–232 (2014).
27. Fiori, S. "An improved chaotic optimization algorithm applied to a DC electrical motor modeling", *Entropy*, **19**, pp. 1–27 (2017).
28. Ditto, W. and Munakata, T. "Principles and applications of chaotic systems", *Commun. ACM*, **38**, pp. 96–102 (1995).
29. Kaveh, A. and Talatahari, S. "A novel heuristic optimization method: charged system search", *Acta Mech.*, **213**, pp. 267–289 (2010).
30. Kaveh, A., Dadras Eslamlou, A., and Montazeran, H. "Chaotic enhanced colliding bodies algorithms for size optimization of truss structures", *Acta Mech.*, **229**(7), pp. 2883–2907 (2018).
31. Tavazoei, M.S. and Haeri, M. "An optimization algorithm based on chaotic behavior and fractal nature", *J. Comput. Appl. Math.*, **206**, pp. 1070–1081 (2007).
32. Shenker, S.J. "Scaling behavior in a map of a circle onto itself: Empirical results", *Phys. D. Nonlinear Phenom.*, **5**, pp. 405–411 (1982).
33. Hilborn, R.C., *Chaos and Nonlinear Dynamics: an Introduction for Scientists and Engineers*, 2nd Ed. Oxford Univ. Press, New York (2004).
34. He, D., He, C., Jiang, L.G., et al. "Chaotic characteristics of a one-dimensional iterative map with infinite collapses", *IEEE Trans. Circuits. Syst. I Fundam. Theory Appl.*, **48**, pp. 900–906 (2001).
35. May, R.M. "Simple mathematical models with very complicated dynamics", *Nature*, **261**, pp. 459–467 (1976).
36. Devaney, R.L., *An Introduction to Chaotic Dynamical Systems*, Addison-Wesley (1987).
37. Peitgen, H., Jurgens, H., and Saupe, D., *Chaos and Fractals*, Springer-Verlag, Berlin, Germany (1992).
38. Tavazoei, M.S. and Haeri, M. "Comparison of different one-dimensional maps as chaotic search pattern in chaos optimization algorithms", *Appl. Math. Comput.*, **187**, pp. 1076–1085 (2007).
39. Anagnostopoulos, K. and Koulinas, G. "Resource-constrained critical path scheduling by a GRASP-based hyperheuristic", *J. Comput. Civ. Eng.*, **26**, pp. 204–213 (2012).
40. Sears, S.K., Sears, G.A., and Clough, R.H., *Construction Project Management, A Practical Guide to Field Construction*, 5th Ed., John Wiley & Sons, Inc., Hoboken, New Jersey (2008).
41. Christodoulou, S. "Scheduling resource-constrained projects with ant colony optimization artificial agents", *J. Comput. Civ. Eng.*, **24**, pp. 45–55 (2010).
42. MATLAB Software (2017).
43. Kaveh, A. and Vazirinia, Y. "Construction site layout planning problem using metaheuristic algorithms: a comparative study", *Iran. J. Sci. Technol. - Trans. Civ. Eng.*, **43**(2), pp. 105–115 (2019).
44. Kaveh, A. and Ilchi Ghazaan, M. "Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints", *Acta Mech.*, **228**, pp. 307–322 (2017).
45. Tran, D., Cheng, M., and Cao, M. "Solving resource-constrained project scheduling problems using hybrid artificial bee colony with differential evolution", *J. Comput. Civ. Eng.*, **30**, pp. 1–10 (2016).

Biographies

Ali Kaveh was born in 1948 in Tabriz, Iran. After graduation from the Department of Civil Engineering at the University of Tabriz in 1969, he continued his studies on Structures at Imperial College of Science and

Technology at London University and received his MSc, DIC, and PhD degrees in 1970 and 1974, respectively. He then joined the Iran University of Science and Technology. Professor Kaveh is the author of 660 papers published in international journals and 150 papers presented in national and international conferences. He has authored 23 books in Persian and 15 books in English published by Wiley, Research Studies Press, American Mechanical Society and Springer. Professor Kaveh is the fellow of the Iranian Academy of Science,

the fellow of the World Academy of Sciences, and the fellow of the European Academy of Sciences and Arts.

Yasin Vazirinia was born in Isfahan, Iran, 1992. He completed his MSc at Iran University of Science and Technology, and he is currently a PhD student in the field of Construction Management at Iran University of Science and Technology. His main research interests are optimization and applications in construction management.