International Journal of Security and Its Application
Vol.11, No.3 (2017), pp.41-58
http://dx.doi.org/10.14257/ijsia.2017.11.2.05

Secure and Efficient SOS-based Workflow Scheduling in Cloud Computing

[1]Moazam Bidaki, [2]Seyed Reza Kamel Tabbakh, [3]M.Yaghoobi, [4]Hassan Shakeri

[1]Department of Computer Engineering, Neyshabur Branch, Islamic Azad University, Neyshabur, Iran, mbidaki@Iau-neyshabur.ac.ir

[2]Department of Software Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran, rezakamel@computer.org

[3]Department of Electrical Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran, Yaghoobi@mshdiau.ac.ir

[4]Department of Information Technology, Mashhad Branch, Islamic Azad University, Mashhad, Iran, shakeri@mshdiau.ac.ir

Abstract

Security is one of the main concerns in the cloud computing platforms. To improve the cloud security various secure scheduling algorithms have been proposed in the literature which tries to improve the assignment of the cloud VMs to the workflow tasks. In this paper, a secure workflow scheduling scheme is provided based on the newly proposed Symbiotic Organism Search (SOS) meta-heuristic algorithm designed to solve numerical optimization problems. Our proposed secure workflow scheduling solution mainly tries to mitigate the makespan and the execution cost of the workflow execution and reduces the difference between the tasks requested security level and the VMs security level assigned for them. The extensive simulations exhibit the effectiveness of the proposed SOS-based secure workflow scheduling scheme against the PSO-based workflow scheduling in various scenarios.

Keywords: *Cloud computing, Workflow, scheduling, Security, SOS, PSO.*

1. Introduction

Cloud computing is a computing model, in which various virtual processing and storage resources are provided to the users based on the pay for use basis. In cloud computing, users can lease, use and release the required resources through internet or other networks in on-demand and pay-for-use model [1]. As indicated in figure1, the cloud computing services can be categorized into Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). In Software as a Service or SaaS consumers purchase the ability to access and use an application or service that is hosted in the cloud and is delivered as a service. In Platform as a Service or PaaS consumers buy administrator access to the platforms to be able to deploy their software and applications in the cloud to make it finally available to the users. However, in Infrastructure as a Service or IaaS customers try to control and manage the applications, operating systems, network and storage of the systems, except they do not control the cloud infrastructure [2].

Cloud computing requirements of dynamic cutting and distribution of computing resources are owing to virtualization. The term virtualization refers to the abstraction of compute resources from applications and end users consuming the service. Virtualization creates plural subsets logically from the complete set of machine. These subsets that behave same as real machines are called Virtual Machine (VM) [3]. In a cloud environment, a large number of users submit their computing tasks to the cloud system and efficient and reliable scheduling of these tasks is a challenge to cloud computing environment.

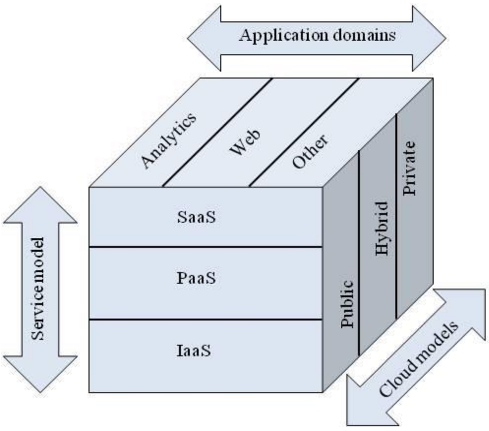


Figure 1. Cloud Computing Architecture

Task scheduling system is responsible to select the best suitable VMs (Virtual Machines) for the user's tasks. Actually, a task scheduling system plays a very important role in how to meet cloud computing users' task QoS requirements and use the cloud resources efficiently in an economic way to reduce both customer and service provides costs [4,5].

Cloud computing paradigm has gained interest in the scientific community for workflow simulation, image processing, data analysis and etc. Scientific workflow tasks are commonly represented as a directed acyclic graph (DAG) in which nodes represent application tasks and edges represent dependencies between these tasks [3]. One well-known challenge for executing workflow applications on cloud is tasks allocation and scheduling, that is, making decisions for mapping computations to resources to optimize some performance metric, such that completion time, execution cost and Service Level Agreements (SLA) are satisfied [4].

The workflow scheduling problem in heterogeneous computing systems is an NP-hard optimization problem [5], because the amount of computation required to find optimum solutions increases exponentially as the problem size increases. Therefore, meta-heuristic methods such as particle swarm optimization (PSO) and genetic algorithm (GA) are used to solve the issue. Metaheuristic algorithms are approximate and non-deterministic. They are not problem-specific and their goal is to efficiently explore the search space in order to find near optimal solutions. PSO or Particle Swarm Optimization developed by the Kennedy and Eberhart [6], is a swarm-based optimization algorithm which operates based on the social behavior of flocks. Various PSO-based scheduling algorithms have been proposed in the literature which are analyzed in [7].

SOS or Symbiotic Organism Search algorithm is another population-based metaheuristic algorithm which is newly presented for numerical optimization problems [8]. SOS is aimed to mimic the symbiotic relations such as mutualism, commensalism, and parasitism between organisms in the nature. It uses a population to converge to an optimal position and for this purpose applies mutualism, commensalism, and parasitism mechanisms to update the positions of the population.

Security concerns are one of the important issues in the cloud computing which they can be the target of various internal and external security attacks[9, 10]. Secure scheduling tries to improve the scheduling security by assigning workflow tasks to the VMs based on the security level requested by the user which have submitted the workflow[11].

In this paper, a security-aware SOS-based workflow scheduling algorithm is presented for the cloud computing data centers. The proposed scheme tries to optimize the makespan, security and execution cost of the workflow execution in cloud. On the other hand, it is aimed to allocate the workflow tasks to the VMs according to the security levels requested while reducing the cost and makespan of the workflow execution. Extensive simulations results conducted in the various scenarios indicates the effectiveness of our SOS-based secure workflow scheduling solution.

The rest of this article is organized as follows: Section 2 presents related work in the meta-heuristic and secure scheduling, and section 3 and 4, respectively describe the PSO and SOS optimization algorithms. Section 5 presents the proposed SOS-based workflow scheduling algorithm. Experimental results are presented in section 6 and finally, section 7 provides the concluding remarks.

2. Related Works

Many metaheuristic-based scheduling schemes are proposed in the literature, which some of them are aimed to provide a secure scheduling for cloud environment and the other try to optimize other objectives such as scheduling makespan, resource utilization and etc.

For example in [12] Pandey et al. presented a scheduling heuristic based on PSO to minimize the total cost of execution of scientific application workflows on cloud computing environments. Moreover, they analyzed the cost savings and compared the result of their proposed solution PSO against Best Resource Selection' (BRS) algorithm.

Another PSO-based scheduling scheme called Revised Discrete PSO or RDPSO algorithm has been proposed by Wu et al. in [13] which tries optimize the workflow scheduling in cloud computing environment. In this algorithm, each particle not only learns from different exemplars, but also learns from the other feasible pairs for different dimensions. The applied algorithm can greatly reduce the search space which this enhances the algorithm performance.

In [14] authors formulate the workflow applications scheduling problem in distributed data-intensive computing environments and designed a security constraint model for it. They performed several meta-heuristic adaptations to the PSO algorithm and provided a variable neighborhood version of PSO algorithm for producing efficient schedules. In this scheme, three security modes for data-intensive job scheduling are considered which are: Secure mode, Risky mode and c-risky mode. In secure mode which is a conservative approach to workflow scheduling, operations are scheduled only on computing resources which can truly have the security requirements. A job is assigned to an available computing unit, only if the condition security demand of the workflow operation be less than the security rank of the computing resource or data resource. In risky mode which is an aggressive method to workflow scheduling, operations are scheduled on any available computing or data resource and accept all possible risks. In c-risky mode, operations are scheduled to available resources taking maximum c risk, where c is a probability measure.

In [15] authors incorporate trust factor in schedule, to manage the life cycle of a workflow and improve the robustness and predictability of the overall scheduling system. This scheme consists of three main stages: plan evaluation, scheduling and monitoring. In the evaluation stage, each workflow is first decomposed into a set of plans which are capable of achieving the workflow goals. Then plans are evaluated to find the best probability it can succeed by applying the median values of cost, time and trust. Finally, the plan with the highest value is selected for scheduling. The scheduling stage applies genetic algorithm to find the optimized assignment of each task in the plan to an autonomous service agent, thereby maximizing the robustness of the schedule while meeting the constraints of deadline and budget. During the schedule execution, the Coordinator monitors for any exceptions that occur. If one is detected, the Coordinator actively respond by following the exception handling rules, result in task reassignment, plan rescheduling or even reevaluation.

In [16] authors try to integrate the trustworthiness of the nodes into a dynamic scheduling algorithm called Cloud-DLS. This algorithm is aimed to decrease the failure probability of the task assignments, and assures the tasks execution in a secure environment. It uses a Bayesian trust model to evaluate the trust degree of resource for general purpose Cloud computing and meets the requirement of user tasks in trust. This solution is a compiled time, static list scheduling heuristic, which has been developed to allocate a DAG - structure application to a set of heterogeneous machines to minimize the execution time of the application. At each scheduling step, the DLS algorithm chooses the next task to schedule and the machine on which that task is to be executed by finding the ready task and machine pair that have the highest dynamic level.

In [17] a trust-based dynamic task scheduling model is proposed for the cloud computing environment. In this scheme the trust relationship between computing nodes is built, and the trustworthiness of nodes is computed using the Bayesian cognitive method. This scheme improves the traditional DLS algorithm by considering the trustworthiness of the resource nodes.

In the next sections, we provide our proposed SOS-based secure workflow scheduling solution and compare its result with the PSO-based workflow-scheduling. For this purpose in the rest of this paper, we briefly describe both PSO and SOS algorithms.

3. PSO Algorithm

A metaheuristic algorithm employs iterative strategies to find solutions in a reasonable time and is aimed to provide better results than deterministic algorithms. These algorithms generally find approximate solutions faster than traditional exhaustive algorithms in terms of the computation time [18]. PSO or Particle Swarm Optimization which is introduced by Kennedy and Eberhart in 1995, is a population-based swarm intelligence metaheuristic algorithm. It is a powerful optimization technique for solving multimodal continuous optimization problems[19]. In PSO algorithm, in each iteration, each particle advances to a certain speed and particle tries to get to the optimal location after some iterations [20]. As indicated in fig.3, each particle changes its searching direction based on the best searching experience of individual so far is called pBest, and the best result obtained so far by all particles in the population is called gBest [21].

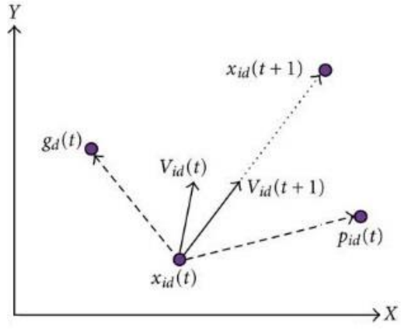


Figure 2. Factors Affecting the Particle's Next Location

$$v_{id}^{t+1} = \omega v_{id}^{t} + c_1 r_1 \times (p_{id}^{t} - x_{id}^{t}) + c_2 r_2 \times (p_{gd}^{t} - x_{id}^{t}) \quad (1)$$

In equation 1, ω is the inertia weight, c_1 and c_2 are the acceleration constants, r_1 and r_2 are the random numbers ranging between 0 and 1, in *t-th* iteration, p_{id}^{t} is the best position of particle and p_{gd}^{t} is the best position of the whole particles in the population[22]. Also, iIn equation 1, $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})^t$ is the original position of *i-th* particle, t is iteration number, x_{id}^{t+1} is the new position of *i-th* particle, $v_i = (v_{i1}, v_{i2},\ldots,v_{iD})^t$ is the original velocity of *i-th* particle, and v_{id}^{t+1} is the new velocity of *i-th* particle.

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \qquad (2)$$

As indicated in equation 3, after new position is computed, its fitness is compared with the best personal fitness and is replaced with the best personal fitness is its fitness is better:

$$p_{id} = \begin{cases} x_{id}^{t+1}, & if\ f\left(x_{id}^{t+1}\right) < f(p_{id}) \\ p_{id}, & otherwise \end{cases} \qquad (3)$$

In equation 3, p_{id} is the best position of *i-th* particle presently.

4. SOS Algorithm

In [8] Cheng et al. propose the SOS (Symbiotic Organisms Search) algorithm which simulates the interaction among different organisms in the nature and consist of the three phases called Mutualism phase, Commensalism phase and Parasitism phase. Generally, organisms rarely live in isolation and often relies on the other species for sustenance and even survival which is known as symbiosis. Like other population-based algorithms, the SOS algorithm applies a population of the candidate solutions to promising areas in the search space in the process of seeking the optimal global solution. Three phases of the SOS algorithm is indicated in the figure 3.

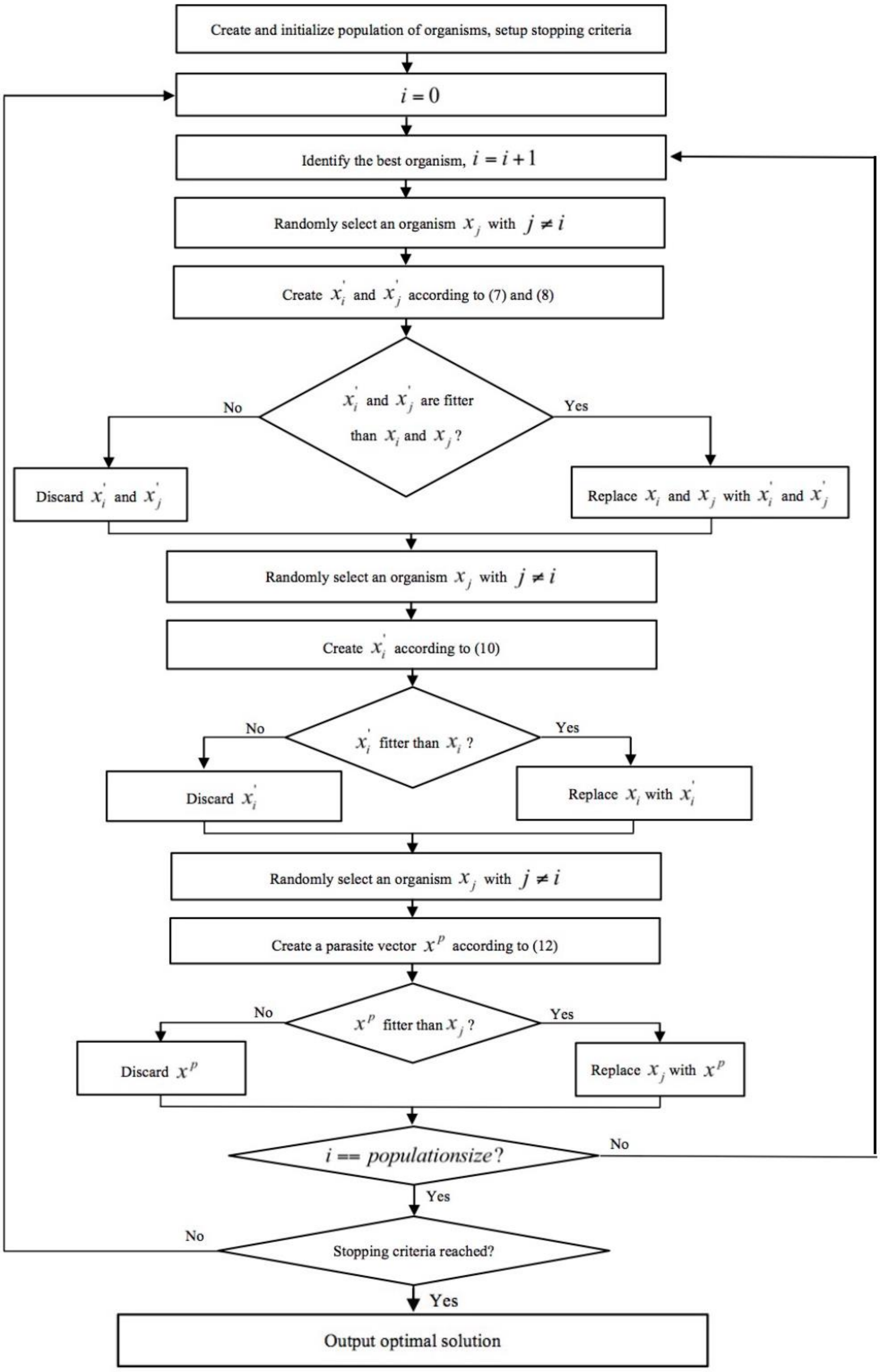

Create and initialize population of organisms, setup stopping criteria
$i = 0$
Identify the best organism, $i = i + 1$
Randomly select an organism x_j with $j \neq i$
Create $x_i^{'}$ and $x_j^{'}$ according to (7) and (8)
$x_i^{'}$ and $x_j^{'}$ are fitter than x_i and x_j ?
No
Yes
Discard $x_i^{'}$ and $x_j^{'}$
Replace x_i and x_j with $x_i^{'}$ and $x_j^{'}$
Randomly select an organism x_j with $j \neq i$
Create $x_i^{'}$ according to (10)
$x_i^{'}$ fitter than x_i ?
No
Yes
Discard $x_i^{'}$
Replace x_i with $x_i^{'}$
Randomly select an organism x_j with $j \neq i$
Create a parasite vector x^p according to (12)
x^p fitter than x_j ?
No
Yes
Discard x^p
Replace x_j with x^p
$i == populationsize?$
No
Yes
Stopping criteria reached?
No
Yes
Output optimal solution

Figure 3. Symbiotic Organisms Search

SOS begins with a randomly created initial population called the ecosystem and each of the organism indicates one candidate solution to the problem. Each organism has a fitness value, which shows the adaptation degree of the organism to the desired objective. Each organism in the ecosystem randomly interacts with the other organisms through all phases. The interaction defines the main principle applied in each phase, for example the interactions which benefit both sides are performed in the mutualism phase. Also, the interactions which benefit one side and do not impact the other are conducted in the commensalism phase. However, the interactions which benefit one side and actively harm the other side is in the parasitism phase. The phases of the SOS algorithm have very simple mathematical operations and unlike the other meta-heuristic algorithms, SOS does not use parameters tuning which this enhances the performance stability.

5. Proposed Workflow Scheduling Solution

This section presents the proposed SOS-based workflow scheduling for cloud computing. This scheme is aimed to assign the workflow tasks to the best avaiable VM's, to minimize the workflow makespan and the execution cost of the workflow. Moreover, our solution considers the workflow tasks security requirrements in assignment to the VMs. For this purpose, the before mentioned objectives are considered in the fitness function. In each organism, the fitness value indicates the success level of each the existing solutions which determines the best solution. An optimal solution is a feasible solution which optimizes the objective function. In the proposed workflow scheduling solution, the fitness value of each organism can be computed by equation 4:

$$Fitness = \alpha_1 * Makespan + \alpha_2 * ExecutionCost + \alpha_3 * 1/\ SecurityLevel \quad (4)$$

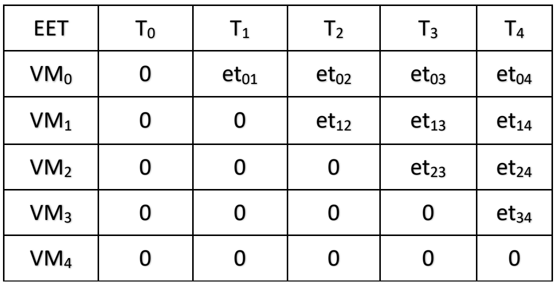
$$\sum_{i=1}^{3} \alpha_i = 1 \quad (5)$$

Moreover, as indicated in the equation 5 the total value of the objective's coefficients must be one and each coefficients control the impact of each objective in the fitness value.

To compute the makespan, the Expected Execution Time of each task on each VM should be determined. In this scheme, the Expected Execution Time of each task on each VM is indicated in the EET matrix. In this matrix, as indicated in equation 5, the et_{ij} parameter shows the execution time of the *jth* task on the *ith* VM.

$$et_{ij} = \frac{Tj}{VMi} \quad (6)$$

Figure 4 indicates the Expected Execution Time matrix for 5 tasks and 5 VMs.



EET	T_0	T_1	T_2	T_3	T_4
VM_0	0	et_{01}	et_{02}	et_{03}	et_{04}
VM_1	0	0	et_{12}	et_{13}	et_{14}
VM_2	0	0	0	et_{23}	et_{24}
VM_3	0	0	0	0	et_{34}
VM_4	0	0	0	0	0

Figure 5. Execution Time Matrix

In this scheme, we assume that there is a security management component in the cloud that can determine the security level of each VM applied in the data center to the cloud workflow scheduler. The security level of the *jth* VM can be specified as equation 7:

$$VMSec(j)\ \varepsilon\ \{\ 1, 2, 3, 4, 5\ \} \quad (7)$$

The number which each VM can achieve dpends on the security mechanism that it presnets and when more secirity options can be granted then VM can accpet higher security values. Also, users which submit a workflow for execution should provide a security requirenment vector as indicated below, which specifies the tasks security needs:

2	3	1		2
TreqSec (1)	*TreqSec* (2)	*TreqSec* (3)	…	*TreqSec* (n)

The security parameter of the fitness function which can be computed by equation 8 indicates how much the VMs assigned to the tasks satisfy the user’s required security level.

$$SecurityLevel = \sum_{i=1}^{N_{task}} TaskSecurity_i \qquad (8)$$

In this equation, N_{task} is the number of workflow tasks and $Task\ Security_i$ indicates the security degree of the *ith* task assignment to a VM which can be computed using equation 9:

$$TaskSecurity_i = \begin{cases} 0 & VMSec(j) - TreqSec(i) \geq 0 \\ \frac{1}{TreqSec(i) - VMSec(j)} & VMSec(j) - TreqSec(i) < 0 \end{cases} \qquad (9)$$

In this equation, $VMSec_j$ denotes the security level of the *jth* virtual machine and the *TreqSec* denotes the *ith* task requested security level.

Our propose workflow scheduling scheme tries to allocate each task to a VM that has the security level demanded by the user. When such allocation is not possible it tries to minize the *Degree of Insecurity* or the difference between the tasks requested security level and the assigned VM’s security level. Equation 11 indicates how the degree of Insecurity can be computed.

$$Degree\ of\ Insecurity = \sum_{i=1}^{N_{task}} (TreqSec(i) - VmSec(j)) \qquad (10)$$

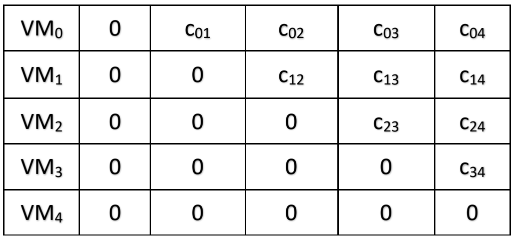
Where in this equation $VMSec_j$ denotes the security level of the *jth* virtual machine *and* the $TSec_i$ denotes the security request of the *ith* task. Also, in this equation, N_{Task} exhibits the number of workflow tasks.

As specified in equation 11, the workflow execution cost is considered as the total cost of each task’s execution on the elected VM:

$$ExecutionCost = \sum_{i=1}^{N_{task}} Cost_{ij} \qquad (11)$$

In this equation, $Ntask$ is the number of tasks in the workflow and $Cost_{ij}$ is the cost of executing the *ith* task on the *jth* VM elected for execution. In this scheme, the execution cost of each task on each VM is specified in the cost matrix indicated in the figure 6.

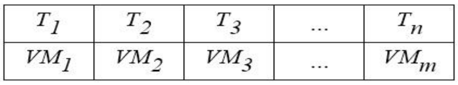
Cost	T_0	T_1	T_2	T_3	T_4



VM_0	0	c_{01}	c_{02}	c_{03}	c_{04}
VM_1	0	0	c_{12}	c_{13}	c_{14}
VM_2	0	0	0	c_{23}	c_{24}
VM_3	0	0	0	0	c_{34}
VM_4	0	0	0	0	0

Figure 6. Execution Cost Matrix

In the proposed scheduling scheme, figure 7 indicates the structure of an organism in the ecosystem.



T_1	T_2	T_3	…	T_n
VM_1	VM_2	VM_3	…	VM_m

Figure 7. Task to VM Assignment in an Organism

6. Simulation Results

This section presents the simulation results of the proposed scheduling algorithm conducted in MATLAB software in the various scenarios. In each simulation scenario, different number of the workflow tasks are assigned to different number of the VMs. Moreover, the following items are evaluated and measured in each scenario:

- Execution cost
- Makespan
- Degree of insecurity

Table 1 indicates the simulation parameters applied in the first scenario. In this scenario, 100 tasks are scheduled respectively on the 3VMs and 5VMs. Also, in both PSO and SOS-based workflow scheduling, 40 particles and organisms are applied respectively.

Table 1. Simulation Parameters of the First Scenario

Parameter	**Value**
Number of the tasks	100 tasks
Number of the VMs	3VMs and 5 VMs
Number of the particles	40
c1 (PSO)	2.05
c2 (PSO)	2.05
w (PSO)	0.725
Number of the organisms	40
Number of the iterations	200
α1	0.3
α2	0.2
α3	0.5

Figure 8 indicates the execution cost of the workflow with 100 tasks in the first scenario in the 200 iterations. Also, the results of the proposed SOS-based workflow scheduling are compared with cost of the PSO-based workflow scheduling. As exhibited in the figure 8, the proposed scheduling solution can mitigate the execution costs better than the PSO-based scheduling. It is obvious that when more VMs are applied in the scheduling, more cost is incurred. Figure 9 exhibits the degree of insecurity in the first scenario. On the other hand, it shows the difference between requested security by user tasks and the security level of the VMs assigned for the workflow tasks.

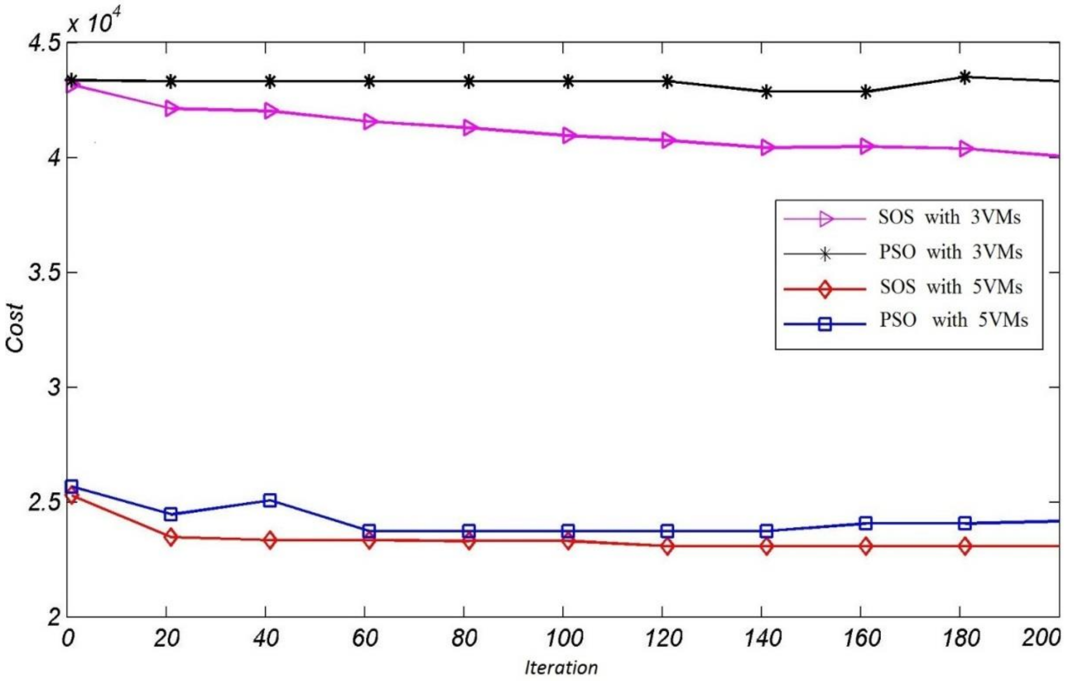


Figure 8. Execution Cost in the First Scenario

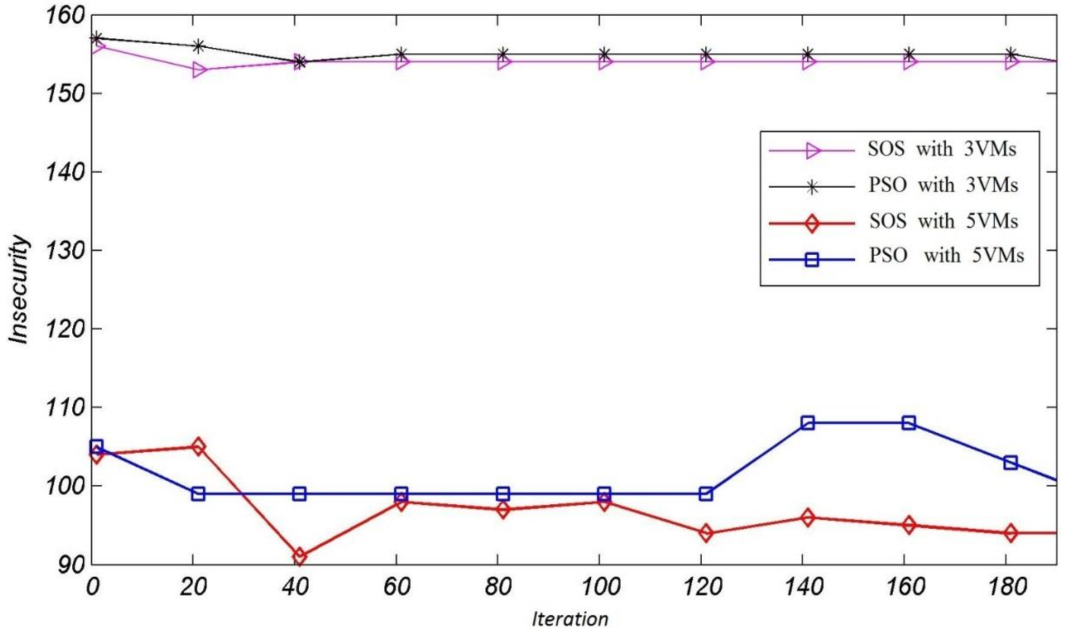


Figure 9. Degree of Insecurity in the First Scenario

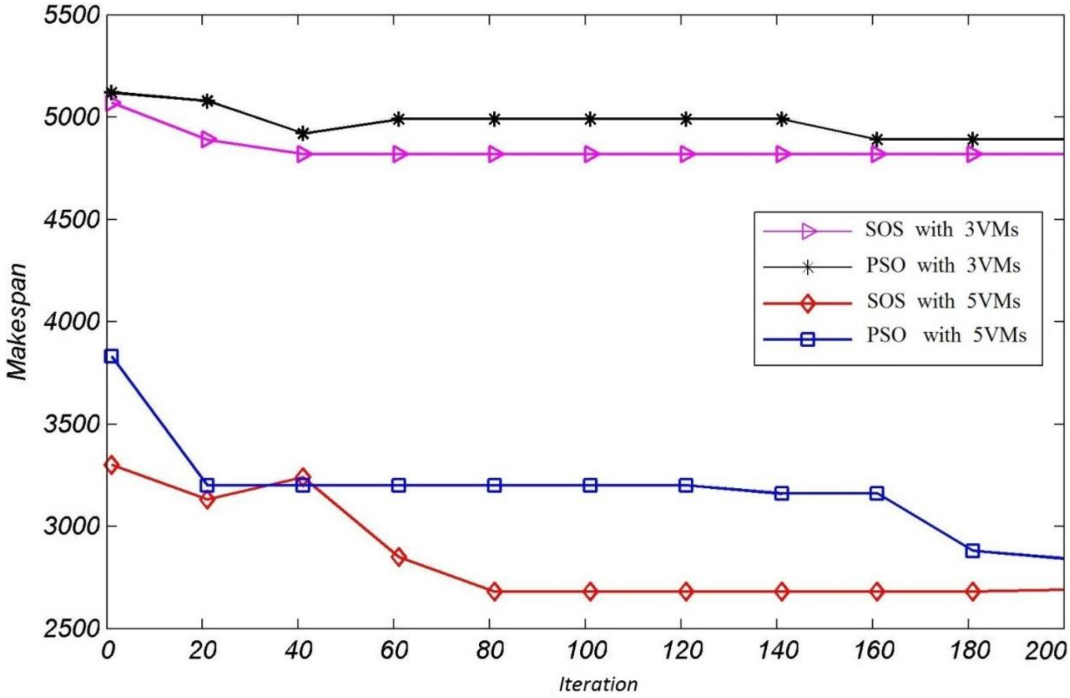


Figure 10. Makespan in the First Scenario

As shown in figure 9 our solution can effectively reduce the insecurity in the workflow scheduling Figure 10 shows the makespan of the workflow execution in PSO-based workflow scheduling and SOS-based workflow scheduling in the first scenario. As shown in this figure our solution can reduce the makespan of a 100 tasks workflow execution on 3 and 5 VMs better than PSO-based workflow scheduling.

Table 2 indicates the simulation parameters applied in the second simulation scenario. In this scenario, a smaller workflow with 50 tasks is executed on the 4 VMs and 6 VMs. Again, 40 particles and organisms are applied in the PSO and SOS-based workflow scheduling algorithms respectively.

Figure 11 indicates the execution cost of the workflow with 50 tasks in the second scenario which is executed for 200 iterations. The result of the proposed SOS-based workflow scheduling is compared with the PSO-based workflow scheduling. As exhibited in this figure, the proposed scheduling solution can reduce the execution costs of the workflow execution, better than the PSO-based scheduling.

Figure 12 shows the degree of insecurity in the second scenario. On the other hand it indicates the total difference of the security level asked by the workflow tasks and assigned by the scheduling algorithm which can be computed using equation 10.

Table 2. Simulation Parameters of the Second Scenario

Parameter	Value
Number of the tasks	50 tasks
Number of the VMs	4VMs and 6 VMs
Number of the particles	40
c1 (PSO)	2.05
c2 (PSO)	2.05
w (PSO)	0.725
Number of the organisms	40
Number of the iterations	200
α1	0.3
α2	0.2
α3	0.5

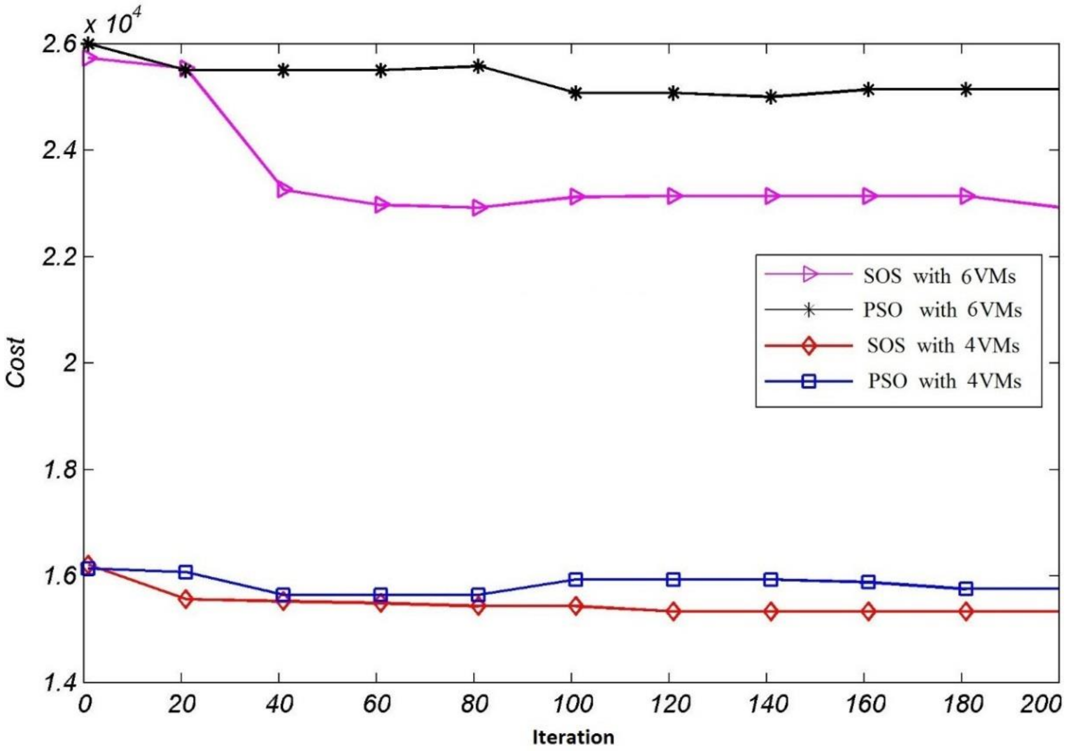


Figure 11. Execution Cost in the Second Scenario

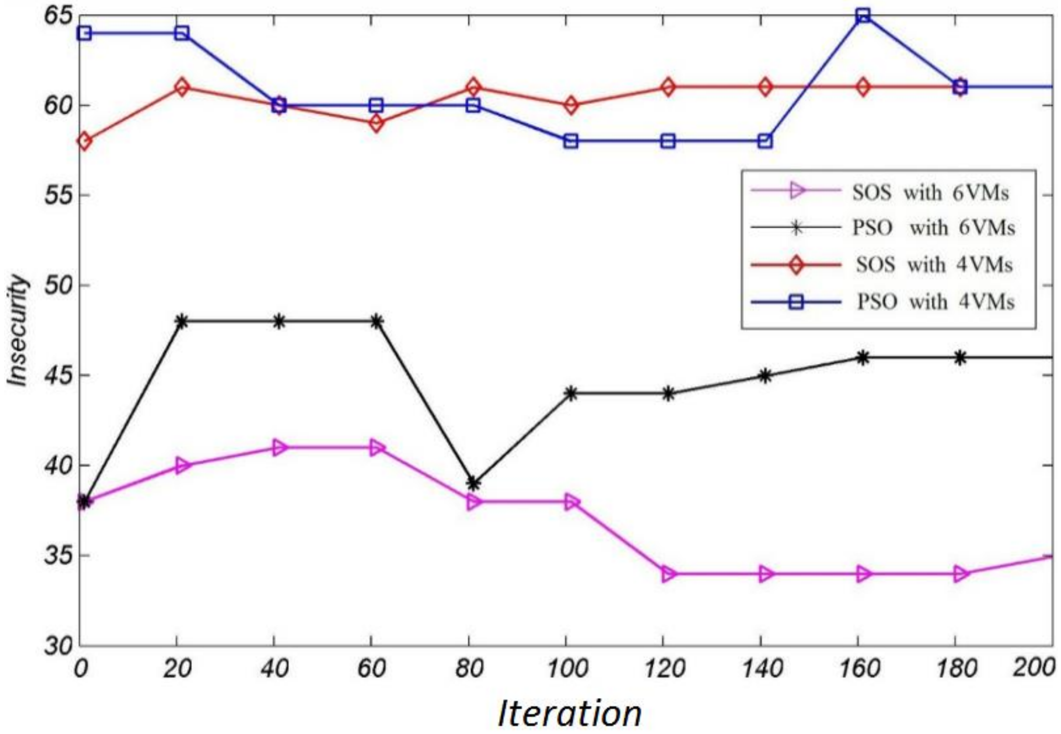


Figure 12. Degree of Insecurity in the Second Scenario

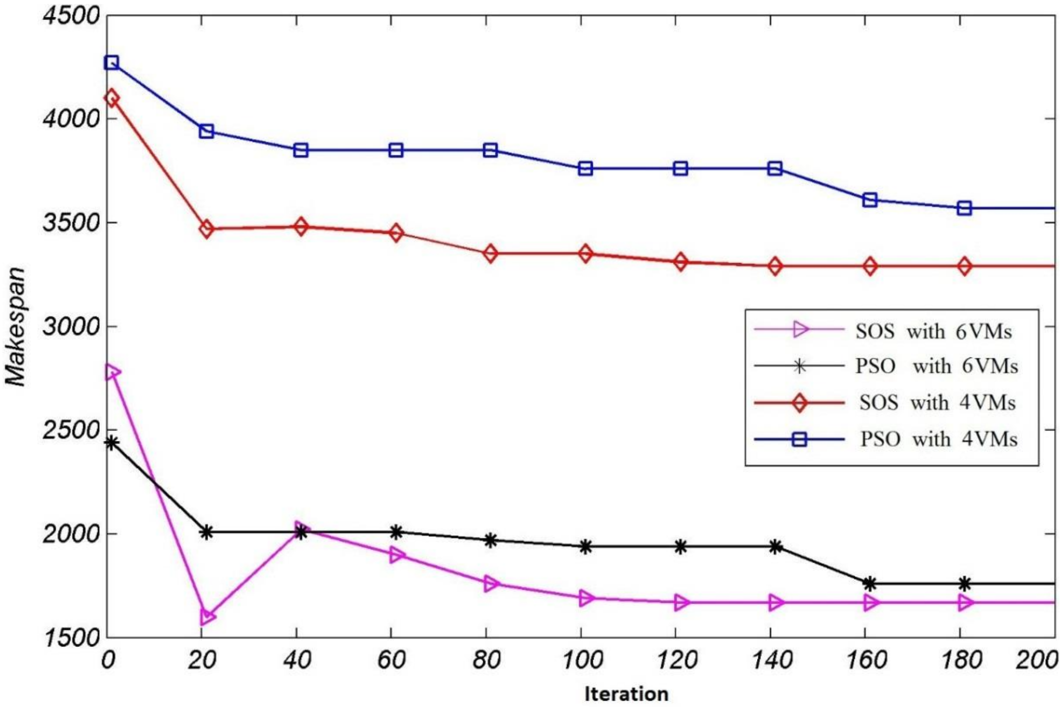


Figure 13. Makespan in the Second Scenario

Table 3. Simulation Parameters of the Third Scenario

Parameter	**Value**
Number of the tasks	25 tasks
Number of the VMs	4VMs and 5 VMs
Number of the particles	40
c1 (PSO)	2.05
c2 (PSO)	2.05
w (PSO)	0.725
Number of the organisms	40
Number of the iterations	200
α1	0.3
α2	0.3
α3	0.4

Table 3 indicates the simulation parameters applied in the third simulation scenario which a workflow with 25 tasks is used. Moreover, these tasks are scheduled to be placed on the 4 and 5 VMs. Also, 40 particles and organisms are utilized in both PSO-based workflow scheduling and SOS-based workflow scheduling. Figure 15 shows the execution cost of the 25 tasks workflow in the third scenario. As it is indicated in this figure, approximately both PSO and SOS algorithms achieve the same cost for the execution of the workflow. However, the cost of using 5 VMs in both algorithms is more than the 4 VMs.

Figure 16 exhibits the degree of Insecurity in the third scenario computed using equation 10 in 200 iterations. Also, figure 14 indicates the workflow execution makespan in the third scenario in 200 iterations of the SOS-based and PSO-based workflow scheduling algorithms.

Considering the results of the figures 8 to 17, it can be concluded that the proposed SOS-based workflow scheduling solution, can effectively reduce the cost, degree of insecurity and makespan of the workflow execution in various scenarios which various VMs are allocated to the different size workflow.

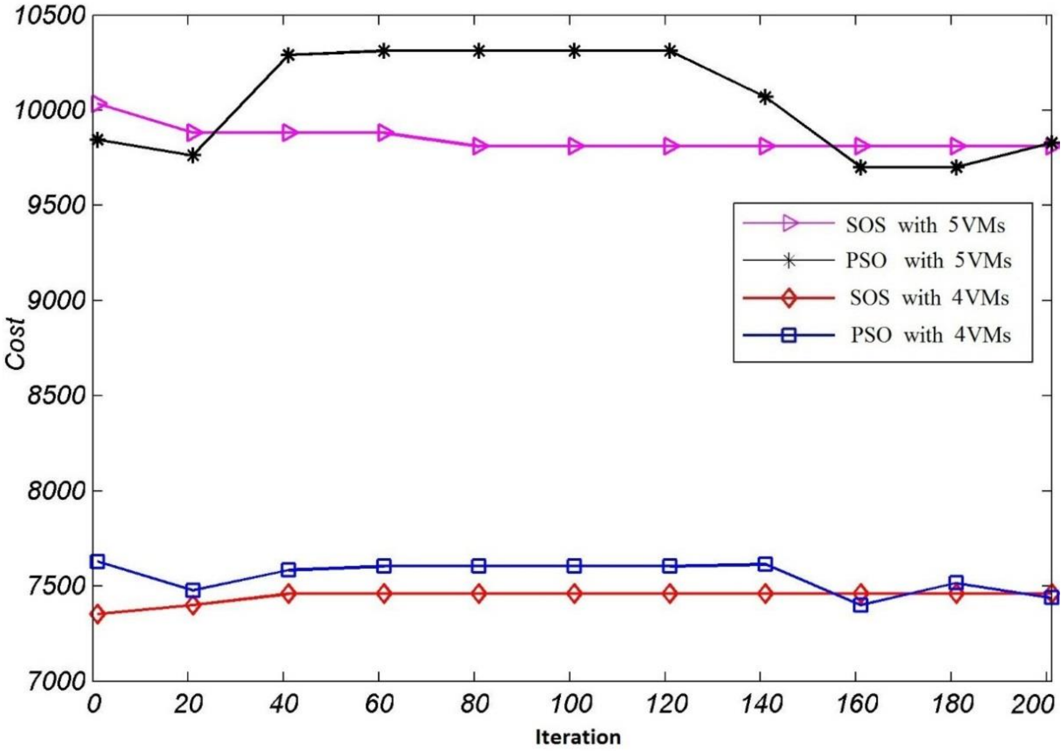


Figure 15. Execution Cost in the Third Scenario

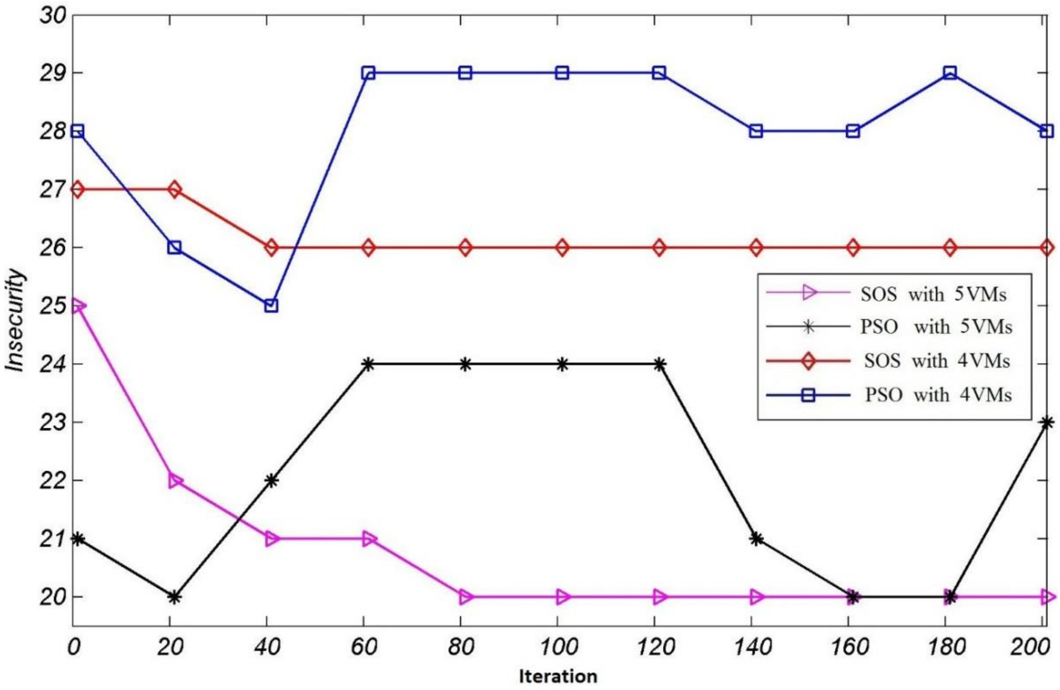


Figure 16. Degree of Insecurity in the Third Scenario

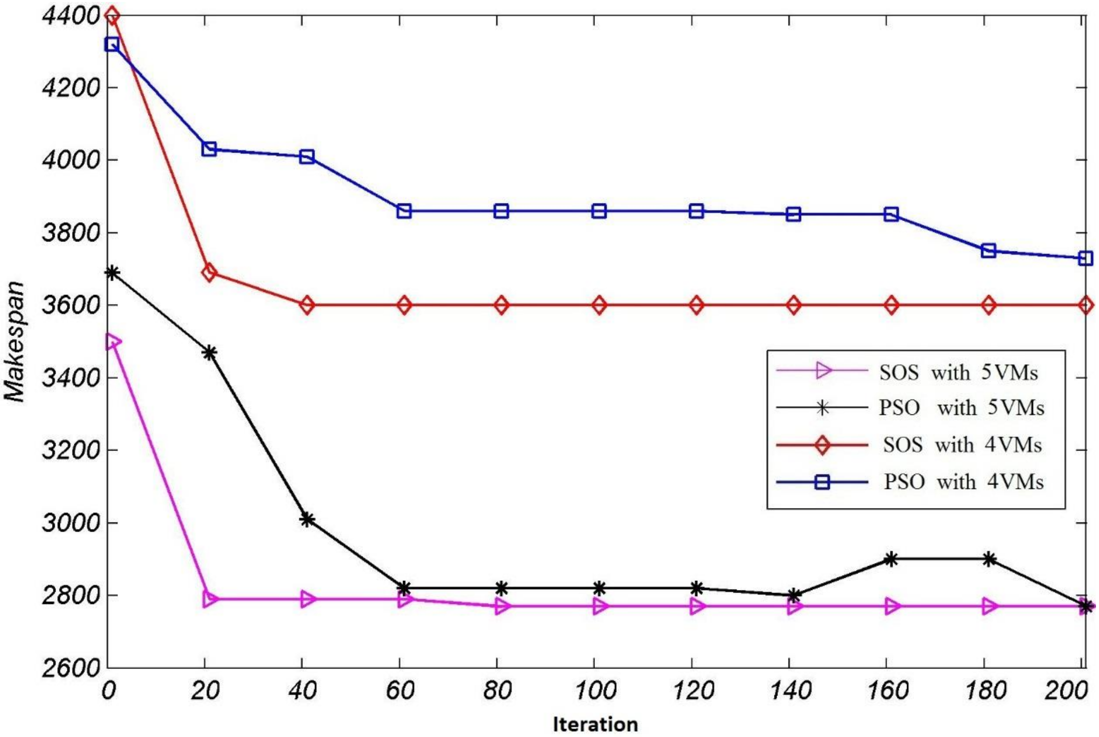


Figure 17. Makespan in the Third Scenario

7. Conclusion

Cloud computing is aimed to provide high amount of virtualized computing and storage resources for the end users. These resources can be allocated dynamically based on the requirements and preferences of the consumers. For this purpose various scheduling algorithms have been proposed in the literature which considering the NP-completeness of the problem, apply meta-heuristic algorithms. The ever increasing security attacks on computer networks and cloud data centers, increased the security concerns in the workflow scheduling problem which each tasks should be executed based on its security requirements.

This paper presents a SOS-based secure workflow scheduling algorithm for cloud data centers which tries to optimize the cost, makespan and degree of insecurity of workflow execution. The results of the proposed workflow scheduling algorithm, is compared against the PSO-based workflow scheduling in several scenario which various number of workflow tasks are assigned to the different number of the VMs. The simulation results indicate the effectiveness of the proposed SOS-based secure workflow scheduling in decreasing the cost and makespan of the workflow and improving the security level of task to VM assignments.

In the future research and studies, we will study the secure workflow scheduling problem in hybrid clouds which a private cloud tries to dynamically achieve some resources from the public cloud.

References

[1] M. Masdari, S. S. Nabavi, and V. Ahmadi, "An overview of virtual machine placement schemes in cloud computing," *Journal of Network and Computer Applications,* vol. 66, pp. 106-127, 2016.

[2] G. Da Cunha Rodrigues, R. N. Calheiros, V. T. Guimaraes, G. L. d. Santos, M. B. de Carvalho, L. Z. Granville, *et al.*, "Monitoring of cloud computing environments: concepts, solutions, trends, and future directions," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, 2016, pp. 378-383.

[3] S. Bharathi, A. Chervenak, E. Deelman, G. Mehta, M.-H. Su, and K. Vahi, "Characterization of scientific workflows," in *Workflows in Support of Large-Scale Science, 2008. WORKS 2008. Third Workshop on*, 2008, pp. 1-10.

[4] C. Jianfang, C. Junjie, and Z. Qingshan, "An optimized scheduling algorithm on a cloud workflow using a discrete particle swarm," *Cybernetics and Information Technologies,* vol. 14, pp. 25-39, 2014.

[5] M. Masdari, S. ValiKardan, Z. Shahi, and S. I. Azar, "Towards workflow scheduling in cloud computing: a comprehensive analysis," *Journal of Network and Computer Applications,* vol. 66, pp. 64-82, 2016.

[6] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm intelligence,* vol. 1, pp. 33-57, 2007.

[7] M. Masdari, F. Salehi, M. Jalali, and M. Bidaki, "A Survey of PSO-Based Scheduling Algorithms in Cloud Computing," *Journal of Network and Systems Management,* pp. 1-37.

[8] M.-Y. Cheng and D. Prayogo, "Symbiotic organisms search: a new metaheuristic optimization algorithm," *Computers & Structures,* vol. 139, pp. 98-112, 2014.

[9] B. Grobauer, T. Walloschek, and E. Stöcker, "Understanding cloud computing vulnerabilities," *Security & privacy, IEEE,* vol. 9, pp. 50-57, 2011.

[10] T. Islam, D. Manivannan, and S. Zeadally, "A Classification and Characterization of Security Threats in Cloud Computing," *INTERNATIONAL JOURNAL OF NEXT-GENERATION COMPUTING,* vol. 7, 2016.

[11] W. Li, J. Wu, Q. Zhang, K. Hu, and J. Li, "Trust-driven and QoS demand clustering analysis based cloud workflow scheduling strategies," *Cluster Computing,* vol. 17, pp. 1013-1030, 2014.

[12] S. Pandey, L. Wu, S. M. Guru, and R. Buyya, "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments," in *2010 24th IEEE international conference on advanced information networking and applications*, 2010, pp. 400-407.

[13] Z. Wu, Z. Ni, L. Gu, and X. Liu, "A revised discrete particle swarm optimization for cloud workflow scheduling," in *Computational Intelligence and Security (CIS), 2010 International Conference on,* 2010, pp. 184-188.

[14] H. Liu, A. Abraham, V. Snášel, and S. McLoone, "Swarm scheduling approaches for work-flow applications with security constraints in distributed data-intensive computing environments," *Information Sciences,* vol. 192, pp. 228-243, 2012.

[15] M. Wang, K. Ramamohanarao, and J. Chen, "Trust-based robust scheduling and runtime adaptation of scientific workflow," *Concurrency and Computation: Practice and Experience,* vol. 21, pp. 1982-1998, 2009.

[16] W. Wang, G. Zeng, D. Tang, and J. Yao, "Cloud-DLS: Dynamic trusted scheduling for Cloud computing," *Expert Systems with Applications,* vol. 39, pp. 2321-2329, 2012.

[17] W. Wang, G. Zeng, J. Zhang, and D. Tang, "Dynamic trust evaluation and scheduling framework for cloud computing," *Security and Communication Networks,* vol. 5, pp. 311-318, 2012.

[18] C.-W. Tsai and J. J. Rodrigues, "Metaheuristic scheduling for cloud: A survey," 2014.

[19] P. Shelokar, P. Siarry, V. K. Jayaraman, and B. D. Kulkarni, "Particle swarm and ant colony algorithms hybridized for improved continuous optimization," *Applied mathematics and computation,* vol. 188, pp. 129-142, 2007.

[20] X. Wen, M. Huang, and J. Shi, "Study on Resources Scheduling Based on ACO Allgorithm and PSO Algorithm in Cloud Computing," in *Distributed Computing and Applications to Business, Engineering & Science (DCABES), 2012 11th International Symposium on*, 2012, pp. 219-222.

[21] H. Zhang, P. Li, Z. Zhou, and X. Yu, "A PSO-Based Hierarchical Resource Scheduling Strategy on Cloud Computing," in *Trustworthy Computing and Services*, ed: Springer, 2013, pp. 325-332.

[22] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, pp. 69-73.