IJICTR
International Journal of Information & Communication Technology Research

Volume 12- Number 3 – 2020 (16 -26)

A Modified Multi-Verse Optimizer Based Workflow Scheduling in Cloud Computing Using Trust-Based Mechanism

Fatemeh Ebadifard
Department of Computer,
University of Kashan
Kashan, Iran
ebadifard.fatemeh@gmail.com

Seyed Morteza Babamir*
Depertment of Computer
University of Kashan
Kashan, Iran
babamir@kashanu.ac.ir

Received: 10 March 2020 - Accepted: 1 June 2020

Abstract—**The problem of task scheduling on Virtual Machines is selecting appropriate resources for a task so that its associated tasks have already been executed. Since the workflow contains a set of tasks, the likelihood of failure increases with the failure of a task throughout the workflow. The allocation of tasks on virtual machines with higher reliability improves workflow-scheduling efficiency. Therefore, Trust relationship is an important factor of resource allocation and job scheduling, and in this paper, we have presented a good method to estimate the trust of virtual machines on which the workflow is run. In addition to the trust, which is an important factor in the workflow scheduling, there are other criteria for the satisfaction of service providers and customers. By increasing the number of requests and the diversity of virtual machines as well as the contradiction between objectives, finding the optimal Pareto front is more challenging. Therefore, multi-objective evolutionary algorithms face a large space of permutations to find an optimal tradeoff of objectives. In this paper, we present a multi-objective workflow-scheduling algorithm using Multi-Verse Optimizer algorithm with the aim of increasing diversity and convergence, so that the proposed method can consider Quality of Services requirements for service providers and customers simultaneously. In order to evaluate our proposed method, we have developed WorkflowSim tools. We have extended the original core of these tools to present our algorithm and then compared our proposed method with previous algorithms such as Pareto-Based Grey Wolf Optimizer, Parallel genetic and Strength Pareto Evolutionary Algorithm. The simulation results show that the proposed approach has a good improvement in service quality factors compared to previous methods.**

***Keywords*- cloud computing; workflow; scheduling; Trust.**

I. INTRODUCTION

Workflow is a common model for modeling most scientific applications in distributed systems. Typically, a workflow is represented as a Directed Acyclic Graph (DAG), in which each task is represented by a node and the relationship among the tasks is shown using edges. Given the importance of workflow applications, in recent years, extensive investigations have been done on workflow scheduling in the cloud environment.

The workflow scheduling problem on the VMs is the optimal selection of a VM for each task, such that its relevant tasks have been already performed. This selection of resources and assignment of tasks on them depends on the quality requirements of the considered

* Corresponding Author

service for users and service providers, so that the issue of scheduling is a NP-hard problem [1].

A workflow consists of several tasks. If a task fails properly in a workflow (for example, when it is terminated by unexpected events), this workflow is demonstrated as an unsuccessful workflow, even if all tasks are executed successfully in the same process. Different methods have been devised to reduce the failure of workflow implementation. One of the most common measures is the repetition of work that has previously failed. This, however, may cause execution to be successful, but it wastes resources due to the need to re-run some of the other tasks. Moreover, it is probable that the repetition of that task is also unsuccessful. In order to solve this problem and reduce the amount of energy waste, cloud service providers are preferred to place their requests on machines with higher trust capability when scheduling tasks in a workflow. Due to the dependency of tasks on the workflow, trust-based scheduling for these types of requests is needed more than other tasks.

In the workflow scheduling problem in the real world, in addition to the need for trust-based scheduling, which is one of the most important criteria in workflow scheduling; we often encounter several objectives in providing proper service quality, which, in many scenarios, contradict each other. Therefore, the scheduling algorithm should be able to achieve balance among the conflicting objectives [2]. This issue is referred to as multi objective scheduling and there are various approaches for solving it. One of these methods makes use of multi objective evolutionary algorithms using Pareto optimizers. These algorithms help the user to find a near optimal trade of among the conflicting objectives by finding a set of near optimal solutions called non-dominant solutions. Each solution introduces a permutation of the tasks on virtual machines. When there is an increase in the number of virtual machines and variety of tasks, we face a huge amount of permutations. In such situations, it is difficult to search the entire space of permutations and find a set of optimal solutions. Once the set of conflicting objectives in scheduling algorithm are taken into account, the problem becomes even more complicated. In such situations, the scheduling algorithm might be bounded by local optima and not have the capability to find a proper and diverse set of solutions.

In this paper, we presented a new multi-objective workflow scheduling based on the trust mechanism by focusing on the diversity and convergence of solutions. To this end, we extended the Multi-Verse Optimizer algorithm (MVO) [3] using Grid dominance [4] that leading to increasing the hypervolume and more increase of the diversity of non-dominant solutions.

To evaluate our proposed method, we used the WorkFlowSim Toolkit, which is an extension of the Open Source CloudSim. We developed the initial core of this tool to provide our algorithm, and then compared our proposed method with the popular multi objective algorithms such as SPEA2 [5] and PGWO [6]. The contributions of this research compared to related work are as follows:

(1) Increasing the diversity and convergence of non-dominant solutions, (2) using the capability of the MVO algorithm which has not previously been used for workflow scheduling in the cloud environment and (3)transforming it to a trust-based multi objective algorithm using a proper dominance equation.

The rest of the paper consists of the following sections. Section 2 gives an overview of the related work. In Section 3, the mathematical model of the workflow scheduling problem and the details of the objective optimization are presented. In Section 4, the details of the proposed method are explained. In Section 5, the conditions for the evaluation of the proposed method are given together with the results of the presented algorithm and finally, the conclusions and future work are presented in Section 6.

II. Related Work

Task scheduling is one of the most common optimization problems that play a key role in improving the flexibility and reliability in distributed systems. The problem of task scheduling means mapping and determining the order of tasks on resources so that one or more performance metric is optimized. A good scheduling mechanism should satisfy both the user service quality requirements and be able to achieve effective productivity on the resources simultaneously. Much work has been done in task scheduling with different objectives in the cloud domain [7-9].

Since our paper provides a meta-heuristic method for workflow scheduling with increased trust capability, we examine related work in the area of trust-based meta-heuristic scheduling algorithms.

In 2012, Wang et al. [10] have presented the dynamic trust scheduling algorithm (Cloud - DLS) with inspiration from the Bayesian cognitive model and sociological relationships model. Theoretical analysis and simulations in this article prove that the Cloud-DLS algorithm can efficiently satisfy the need for reliable, cloud-based computing at a lower cost and guarantee the execution of requests in a secure environment. The proposed method in this paper is for independent tasks and is not applicable to workflow; we are inspired by this model in our proposed method to provide the trust model in the workflow request.

In 2015, Xie et al. [11] presented a task scheduling model based on the trust mechanism in the cloud environment. They use a Shuffled Frog Leaping Algorithm (SFLA) and estimate the amount of trust for optimal allocation of requests on virtual machines. The experimental results show that the proposed method [8] can effectively improve the average reliability and the success rate of task scheduling and needs of user's quality of service more efficiently compared to the min-min algorithm and traditional genetic algorithm. Since this method is single objective, it does not have applications for workflow scheduling with the aim of maintaining the simultaneous satisfaction of users and service providers.

In 2018, Gupta et al.[12], proposed the fault-aware Big-Bang-Big Crunch (BBC) algorithm for task scheduling in cloud environment. This algorithm is made up of the Big-Bang-Big Crunch (BBC) on the

idea of creating a world in astrological. In this method, they have proposed a task-scheduling algorithm to find the best solution from a large set of solutions, where a generation of the universe is referred to as the Big Bang phase and dissipation of the universe in the black hole near the center is said to be a Big Crunch phase. The proposed algorithm aims to improve the performance of the task scheduling algorithm and reduce the number of request failure. It also improves the system reliability and finds the overall schedule for requests. The simulation results show that the proposed method offers a better QoS quality by increasing the number of requests and resources with the probability of failure.

In 2016, Gupta et.al [13] proposed the power and Fault Awareness of Reliable Resource scheduling for Cloud Infrastructure . The proposed method is based on fitness value, that is assessed using the probability of the data center failure and power efficiency . the experiment results shows that the proposed algorithm performs better than DVFS in terms of the failed demand , energy efficiency, and the number of completed requests.

In 2014, Wu et al. [14] have proposed a new approach of trust-based workflow scheduling in cloud computing. In this paper, using fuzzy clustering, the trust-based workflow scheduling has proposed aims to increase user reliability, reduce costs, and work completion time. This algorithm provides a set of weights for both direct and indirect trust. This method, like other previous methods, satisfies only the user's QoS requirements.

In 2020, Shukri et al. [15] have Proposed a method for task scheduling in the cloud environment using a single objective multi-verse optimizer algorithm with the objective of reducing execution time and cost and increase resource utilization. They have compared their proposed method with the basic PSO and MVO algorithms. The simulation results show that the proposed method has better results compared to the basic PSO and MVO algorithms in the compared objectives.

Although in [15] like our method, the MVO algorithm has been used for the task scheduling problem, the main difference is in the form of its use, so that in the [15], the single-objective form of MVO, and in our proposed method the multi-objective form of MVO has been used. In addition, our differences and innovations in this paper compared to previous work and [15] are:

- Providing a suitable method for estimating the reliability of virtual machines on which workflow is performed, with the objective of increasing the reliability of workflow execution.
- Using of Grid dominance Relationship for Converting the single-objective MVO to multi-objective MVO to Find Near-Optimal Solutions with conflicting objectives.
- Increasing diversity and convergence in search of non-dominated solutions in the sample space.
- Increasing the hypervolume criteria that lead to the selection of near-optimal solutions despite the vastness of the search space and the existence of conflicting objectives.

Contrary to the earlier works mentioned above, in this paper, we have proposed a Modified MVO-Based Multi-Objective Workflow Scheduling in Cloud Computing Using Trust Based mechanism to maintain the satisfaction of users and service providers simultaneously.

III. Problem Formulation

A. *Workflow Scheduling Problem Formulation*

The workflow is represented as a non-directed graph (N, E) where N represents a set of tasks and E represents a set of edges that reveal the dependence between tasks. There is an edge E (1,2) from node N1 to node N2. If N1 is in the graph before N2, the N1 task must be completed before the N2 task can start running. The weight on the edges E (1, 2) shows the cost of the output transfer from the node N1 to node N2. The node without the input edge indicates the arrival task, and the node without any output edge indicates the departure task. Figure 1 shows an example of a workflow.

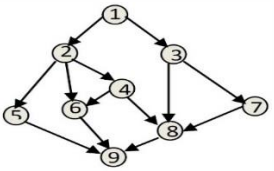

Fig. 1. An example of a workflow.

In the proposed method, two constraints are defined for each task in the workflow. The longest starting time (D_{ti}^{s}) of the task and the longest finishing time (D_{ti}^{e}) of the task.

Workflow scheduling can be modeled in both single-objective and multi-objective ways. In single-objective scheduling, we are faced with a definite optimal solution, but unlike single-objective algorithms, in multi-objective methods, a set of conflicting objectives must be optimized simultaneously. As a result, the multi-objective scheduling method produces one or more solutions; each solution is a permutation from tasks to virtual machines and does not dominate one another. The objectives of this study are (1) to reduce Makespan time, (2) to increase the degree of load balancing, (3) to increase resource utilization, and (4) to increase trust capability. in the following, we describe each objective by detail.

1) *Makespan*

Suppose VM = {VM_1, VM_2 , … , VM_m} is a set of virtual machines and Task = {t_1, t_2,…, t_n} is a set of tasks that can be executed on resources if the Completion time of request ti on VMj is illustrate by CT_i^j , Makespan is defined using Eq. 1 [16].

$$Makespan = \max_{1 \le j \le m} \sum_{i=1}^{n} CT_i^j \times x_{ij} \quad (1)$$

If request t_i is executed on VM_j, the value of x_{ij} is one; otherwise, its value is zero. For example, if we have three virtual machines and the requests are executed on the machines in Figure 2, the value of the makespan is specified below.

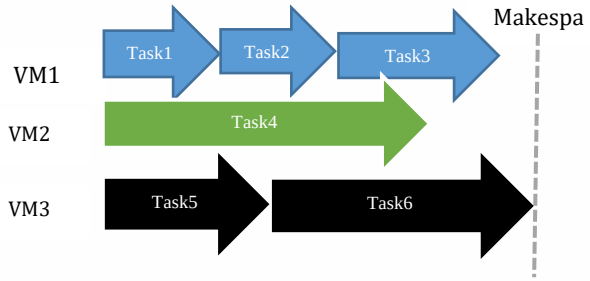

Fig. 2. The value of Makespan for 6 tasks and 3 VMs.

2) Degree of Imbalancing

The degree of Imbalancing is a measure that shows the degree of fairness in the distribution of workloads among virtual machines based on the capabilities of the virtual machines. Eq. 2 defines this measure [17].

$$DI = \frac{T_{\max} - T_{\min}}{T_{avg}} \quad (2)$$

where, T_{max} and T_{min} are the most and least time for performing tasks among virtual machines, and T_{avg} is the average execution time of all virtual machines.

3) Resource Utilization

Resource utilization refers to the amount of virtual machine usage compared to the time it takes to complete the task. If the utilization value of each virtual machine is calculated from Eq.3, the average utilization is calculated by Eq.4. [18].

$$U_i = \frac{\sum_{j=1}^{n} PT_{ij}}{MS} \quad (3) \qquad \overline{U} = \frac{\sum_{i=1}^{m} U_i}{m} \quad (4)$$

where PT_{ij} indicates the processing time of task i on VMj, n is the number of total tasks and MS is the makespan. In fact, the utilization denotes the amount of resource usage when all requests on VMs are executed.

4) Trust Capability

In the proposed method, if the virtual machine can meet the requirements of Eq.5 for each task assigned to it, its trust capability is increased; otherwise, its trust capability is reduced.

$$ST_{t_i} \le D^s_{t_i} \,\&\&\, FT_{t_i} \le D^e_{t_i} \quad (5)$$

The above equation states that the start time of request (t_i) should not exceed the deadline for the start time (D^s_{ti}) and the completion time of the request, ti, should not exceed the legal deadline of completion (D^e_{ti}).

B. Basic Multi-Verse Optimizer algorithm

The MVO algorithm is inspired by the cosmological structure and behaves according to three concepts in cosmology (white hole, black hole and wormhole): The mathematical models of these three concepts are designed for local exploration, exploitation and local search operations, respectively.

This algorithm randomly generates random populations (Universe) similar to other meta-heuristic algorithms and then divides the search process into two stages based on the initial population: exploration and exploitation. In this algorithm, the concept of a white hole and the black hole is used to discover search spaces. In contrast, wormholes help the MVO to exploit search spaces. Every solution in this algorithm is the Universe. In addition, the algorithm sets an inflation rate for each solution, which is proportional to the corresponding fitness function value of the solution. During optimization, the following rules apply to the MVO Universe:

1. The higher inflation rate, the higher the probability of having a white hole.
2. With the rising inflation rate, the possibility of black holes is low.
3. Universes with higher inflation rates tend to send objects through the white hole.
4. Universes with lower inflation tend to get more objects through the black hole.
5. Objects in all the worlds can randomly go to the best of the world via a wormhole regardless of inflation.

In the MVO algorithm, the optimization process begins with the creation of a set of a random universe. The inflation rate is then calculated for each solution based on the objective function. After calculating the inflation rate, we choose the black hole and white hole solutions. A universe with a higher inflation rate is considered to have a white hole, whereas the universe with less the inflation rate is assumed to own black holes. In each iteration, objects in the universe with a high inflation rate through white/black holes tend to move toward the universe that have a low inflation rate. In the meantime, every single universe randomly moves its objects to the best of the universe through wormholes. This process is repeated as long as the termination condition is met.

C. Basic Trust Estimation Method

In this paper, we have inspired the trust estimation model in paper [10] to estimate trust and justify it for the workflow scheduling problem. For this purpose, we discuss the basic method of trust estimation in this section. In this model, trust in the cloud environment is based on assessing the ability of nodes to provide services with respect to nodes' behaviors under different conditions, using observations of previous behaviors and other nodes' recommendations. The relationship of the proposed trust is based on the probability of successful direct cooperation and the probability of indirect successful cooperation.

Direct Trust:

Suppose x, y are two nodes in the cloud system that interact directly with each other and the results of their interaction are mutual (success/failure), in which case the probability of successful interaction between these two nodes is achieved by Eq.6. In other words, when the number of direct interactions between two nodes x,

y is 'n' and the number of successful and unsuccessful interactions respectively are 'u' and 'v'; (θdt) is the probability of successful direct cooperation between x, y at interaction n + 1 and is obtained by using Eq.6.

$$\theta_{dt} = \frac{u+1}{u+v+2} \quad (6)$$

$$where \;\; 0 < \theta < 1, and \;\; u, v > 0$$

Recommended Trust:

In addition to direct trust, if there is a node Z corresponding to Y and X, and there is a direct relationship between (X, Z) and (Y, Z), we can, therefore, we can obtain an indirect probability of success, the co-operation between X, Y, Called the recommended trust (θrt), which is obtained by Eq.7.

$$\theta_{rt} = \frac{u_1 + u_2 + 1}{n_1 + n_2 + 2} \quad (7)$$

where the number of interactions between (X, Z) and (Z, Y) are n_1, n_2, and u_1,u_2 are successful interactions and v_1,v_2 are unsuccessful interactions between them.

When there are several recommended nodes, the above formula is extended and the degree is obtained using Eq.8.

$$\theta_{rt} = \frac{\sum_{\gamma \geq \gamma_0} u+1}{\sum_{\gamma \geq \gamma_0} (u+v)+2} \quad (8)$$

where γ_0 is the threshold value of the number of indirect interactions with a node so that the number of interactions with the node to compute the recommended trust must be greater than this threshold. Finally, the degree of total trust is calculated based on the recommended trust (Eq.8) and direct trust (Eq.6) for a node by Eq.9.

$$trust = \lambda_0 \times \theta_{dt} + (1-\lambda_0) \times \theta_{rt}, \lambda_0 \in (0,1) \quad (9)$$

The value of λ0 is selected based on the user's preference for direct or recommended trust.

IV. PROPOSED METHOD

Multi-objective scheduling of workflow in distributed systems has been considered highly in recent years. It is almost impossible to find the best solutions for the scheduling problem due to its NP-hardness, so the purpose of the existing algorithms is to provide a near-optimal solution [19]. Many algorithms have been proposed with the aim of finding suitable solutions to meet the quality requirements of the service. In real-world scheduling issues, we are often faced with several objectives of service quality. Therefore, proper scheduling algorithms should be able to maintain a balance between some of the service quality objectives.

Increasing requests and dependency between them, as well as the existence of different objectives, cause the complexity of the scheduling problem on virtual machines because there are many permutations of solutions and it is difficult to search the entire sample space and find optimal permutations. Therefore, Meta heuristics algorithms have many help to solve such problems. One of the most well known meta-heuristic algorithms is the MVO algorithm. This algorithm has not been used in the cloud scheduling domain before. We have used this algorithm to solve the workflow scheduling problem in cloud computing and have improved it according to our objectives. In this section, we first present a proposed method for estimating the amount of trust to virtual machines in workflow scheduling and then discuss the proposed trust-based workflow scheduling method in detail.

The proposed model for scheduling requests on virtual machines in the current approach consists of two parts: the first part is the calculation of service quality parameters such as increasing the resource efficiency, decreasing response time, and reduction of makespan time, which is calculated based on the permutation of requests on virtual machines. The second part is based on the amount of trust in every virtual machine, which is the result of virtual machines' behavior in responding to previous requests in a workflow.

This factor is calculated based on the two capabilities of direct trust and recommended trust .Direct trust is in relation to the machines that a virtual machine depends on to execute its request and receive the required files. The recommended trust is related to machines that indirectly provide the required files of a machine needs to execute its current request. Figure 3 shows the architecture of the proposed method.

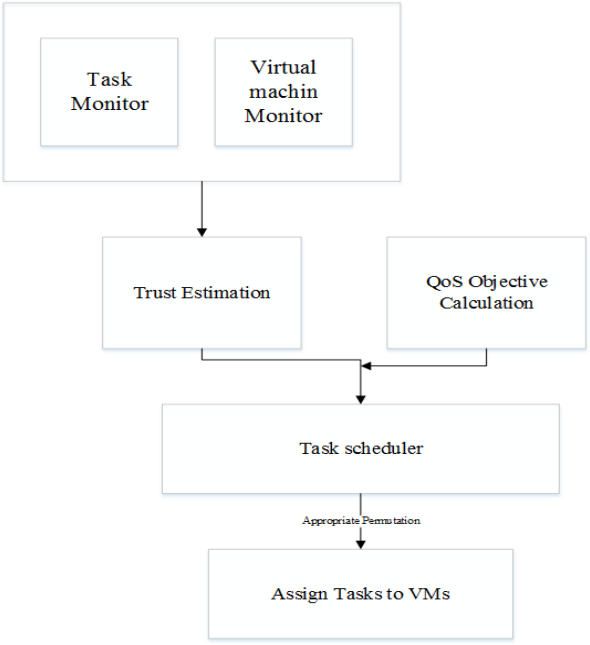

Fig. 3. Proposed method architecture.

A. The proposed method of estimating trust capability of virtual machines in workflow scheduling

We have used two factors of direct and recommended trust to calculate trust capability. The amount of direct trust of VMi is derived from the interactions between other machines with VMi in previous interactions.

For example, task eight in Figure 1 for execution, needs the execution of three tasks 4, 3, and 7. While these three tasks are executed in machines 1 and 2 and 3 respectively and task 8 performs on machine 5, machine 5 is directly related to these three machines.

For example, while Eq.10 for t_4 and t_8 is satisfied, the value of the successful direct trust (u) for machine 1 is increased, otherwise, v is increased.

$$FT_{t_4} \le D^S_{t_8} \,\&\&\, FT_{t_4} \le D^e_{t_4} \quad (10)$$

In Eq.10, if the request t_4 is completed in its completion deadline and also when request t4 is completed before the start deadline of request t_8, machine 1 (executor of t_4) has a successful direct connection to machine 5 (executor of t_8). We calculate all direct interactions with this virtual machine to calculate successful direct trust for it and then compute its direct trust level using Eq.6.

Indirect trust for virtual machines is also based on indirect interactions. For example, if VM_4 executes t_9 request, the interaction between the machines (4, 1) is achieved indirectly by using the middle node 5, which executes request t_8. We use the Eq.8 to gain trust for interaction (4, 1). After calculating the direct and recommended amount of trust for each virtual machine, the overall trust amount for that virtual machine is achieved using Eq.10.

B. Proposed multi-objective scheduling method based on trust mechanism using MVO algorithm

The proposed algorithm in this paper is based on the basic MVO algorithm that we have converted to a multi-objective algorithm using G (Grid-dominance) optimizer. Given the prevalence of data centers at the geographic level and as a result of the vastness of the search space and a large number of optimization objectives, we are faced with a large number of permutations of solutions, in such a case, an algorithm with high diversity is needed to increase efficiency . For this reason, we have improved this algorithm and we use our meta-algorithm to increase the diversity in the search as well as optimal solution selection. The pseudo-code of the proposed method is presented in algorithm1. To this end, we address this algorithm.

Initially, in the first step (instructions 1 and 2), the initial population of solutions (P_0) is randomly selected by the number of N members. Each solution (universe) is a permutation of requests on virtual machines. In order to generate the initial solution, we first generate an array of tasks and assign a virtual machine to each task randomly. For example, suppose that we have three virtual machines VM_0, VM_1, VM_2, and five task t_0, t_1, t_2, t_3, t_4. A random solution is created, such as in Figure 4.

T_0	T_1	T_2	T_3	T_4
0	1	1	0	2

Fig. 4. An example of a random solution.

In addition to the initial solutions, we create an empty set of archive solutions (P_t). we deal with two kinds of the population each time: (1) the current generation population, which we want to select the most suitable members (solutions) from it (P_t) and (2) the archive of selected members from previous generations (A_t).

In the second step (instruction 4), the objective function value is calculated for each solution from the initial and the archive population according to the objectives (Makespan, degree of imbalance, resource utilization). In addition to the above factors, the trust factor is also considered as one of the objectives of the workflow scheduling. We use Eq.9 to estimate the trust of the workflow scheduling for each virtual machine (see Section 4-1) and then obtain the average trust for all virtual machines.

After calculating the values of objective functions for each solution, in the third step (instructions 5, 6 and 7), we must select the undesirable solutions from the set of solutions, for this purpose, we have used the grid dominance. In the Grid Dominance, if x and y are two solutions of the population and y is dominated by x, it is shown as $x \prec_{grid} y$ and is defined by Eq.11.

$$\forall i \in \{1,2,....,m\} : G_i(x) \le G_i(y) \; and \; \exists j \in \{1,2,....,m\} : G_j(x) < G_j(y) \quad (11)$$

where m is the number of objectives. The $G_i(x)$ is a normalization of the objective value i for the solution x $value_i(x)$. The amount of $G_i(x)$ for negative objectives (objectives that less values of them are acceptable such as Makespan, Response time) is obtained by Eq.12.a, and for positive objectives (objectives that larger amount of which are desirable such as Trust and Utilization) is obtained from Equation (12.b). The values of di, lb_i and ub_i are obtained from Eq.13, Eq.14 and Eq.15.

$$G_i(x) = \left\lfloor \frac{(value_i(x) - lb_i)}{d_i} \right\rfloor \quad (12.a)$$

$$G_i(x) = \left\lceil \frac{(ub_i - value_i(x))}{d_i} \right\rceil \quad (12.b)$$

$$d_i = \frac{(ub_i - lb_i)}{div} \quad (13)$$

$$lb_i = \min_i(x) - \frac{\max_i(x) - \min_i(x)}{2 \times div} \quad (14)$$

$$ub_i = \max_i(x) - \frac{\max_i(x) - \min_i(x)}{2 \times div} \quad (15)$$

Where $\max_i(x)$ and $\min_i(x)$ are the lowest and highest values of the objective function i for solution x. The div value is equal to the number of target space divisions in each dimension, and the value is chosen by the user and given the size of the solution population.The distance between two solutions: if x and y are two solutions, the distance between them is obtained from Eq.16.

$$GD(x,y) = \sum_{i=1}^{m} |G_i(x) - G_i(y)| \quad (16)$$

Considering that the purpose of the proposed algorithm is to cover diversity and convergence, the following metrics are used to determine the value of solutions. (1) Grid Ranking (GR), (2) Grid Crowding Distance (GCD).

Grid Rank (GR): This criterion determines the ranking of solutions based on their location in the grid. For each solution, the value of GR is equal to the sum of its grid coordinates (Eq.17).

$$GR(x) = \sum_{i=1}^{m} G_i(x) \quad (17)$$

where m indicates the number of objectives. Given that the goal is to minimization, if a solution is better than most of its competitors in most objectives, it will have a lower GR. The GR metric in the proposed algorithm is used to evaluate the convergence of solutions.

Grid Crowding Distance (GCD): Eq.18 can be used for the distribution of solutions so that the farthest neighbor is selected as the next-generation candidate. This metric will help to distribute the solutions properly in the solution space. The use of the GCD metric in the proposed algorithm causes increase the diversity to select the best solutions.

$$GCD(x) = \sum_{y \in N(x)} (m - GD(x,y)) \quad (18)$$

In Eq.18 N (x) is the set of neighbors of solution x. The solution y is the neighbor of solution x if $GD(x,y) < m$.

In the next step, we find non dominated Grady solutions. To this end, we consider the R-value fitness for each solution, and we create a dominated matrix to compare all solutions to each other. Then by using Eq.11, each solution of the population is compared with other solutions. If the solution i is dominated by solution j, the value of the dominated matrix in row i and column j will be equal to one. If the solutions do not overcome each other, the comparison is made in terms of Grade Rank (GR) and, if appropriate, based on GCD distance. Finally, the sum of each column of the dominated matrix (R-value) is considered as the number of times that each solution is dominated.

For example table 1 shows a sample of values for the solutions x and y (indicated by S_x and S_y) that were computed in our method for a case study. Each solution consists of four objective values (i=1..4). $G_i(x)$ and $G_i(y)$ show the normalization values for S_x and S_y, respectively where each value of $G_i(x)$ is less than or equal to its corresponding value in $G_i(y)$. In solution S_x, the objective values should be minimized (Makespan, Response time) are less than the corresponding values in S_y and the objective values should be maximized (Utilization, and Trust) are equal or more than the corresponding values in S_y; therefore, S_x dominates S_y.

TABLE I. A SAMPLE OF VALUES OF THE THREE OBJECTIVES AND THOSE OF THE RELATED RELATIONS

	Makespan	Utilization	Response time	Trust Value
S_x	361.82	0.47123	370.54	0.98
S_y	370.23	0.3923	381.40	0.98
Lbi	337.5583	0.349433	360.925	0.983333
Ubi	379.4483	0.459633	387.835	0.963333
di	13.96333	0.036733	8.97	-0.00667
Gi(x)	1.737527	-0.3157	1.071906	2.5
Gi(y)	2.339819	1.833031	2.282609	2.5
max	386.43	0.478	392.32	0.96
min	344.54	0.3678	365.41	0.98

In this example, for the two solutions x, y, the values of the dominance matrix are set as follows.

$$domiated\ \ matrix = \begin{matrix} \\ S_x \\ S_y \end{matrix} \begin{matrix} \begin{matrix} S_x & S_y \end{matrix} \\ \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \end{matrix}$$

In the above example, the value of R for solution S_x (the sum of the data in the first column) is equal to one, and for solution S_y is equal to zero, thus the solution x dominates solution y.

In the next step, the obtained R-values are arranged. The zero value for R merit indicates the effectiveness of that solution. At this step, the solution by lower R-value is selected as the Pareto front and put at the archive. If a solution exceeds the upper bound of each of the objectives, the amount of R-value is given a very large number.

After selecting the archive set in step 4 (instructions 9, 10 and 11), we create a new solution using the MVO algorithm update function. To do this, first, the values of black hole, white hole, and TDR and WEP values are obtained (instructions 10) then the values of each solution are updated based on the WEP values. (Instruction 11)

Finally, the number of steps of the algorithm is increased and steps 3 to 11 are repeated until it reaches the maximum number of steps (termination conditions). Once the algorithm terminates, the output of the final step (instructions 14) will be the archive set, which includes an optimal tradeoff between the user and the service provider objectives. We will select the solution with the lowest possible amount of GR in the archive as an optimal solution. While the GR value of the two solutions is the same, the solution is selected with the minimum value of GCD as a suitable permutation. Figure 5 shows the flowchart of the proposed method.

Algorithm 1: the Pseudo-Code of the Proposed Algorithm

1. $t \leftarrow 0$;
2. Initialize random population of solution (P_t) and create empty archive set (A_t)
3. *While (t< Max number of iterations)*
4. *Calculate the fitness vector of all solution based on objective functions*
5. *Calculate grid setting for each solution;*
6. *Calculate R value for each solution*
7. A_t=*Get grid Non-dominated Solution*
8. *Copy Solution from* A_t *to the archive*
9. *For each Solution in* $(P_t \cup A_t)$
10. *Select Blackhole_whitehole_wormhole* $(P_t \cup A_t)$
11. *New_pop* $\leftarrow$ *Update_position* $(P_t \cup A_t)$
12. $t \leftarrow t + 1$;
13. *End while*
14. ***Return the archive***

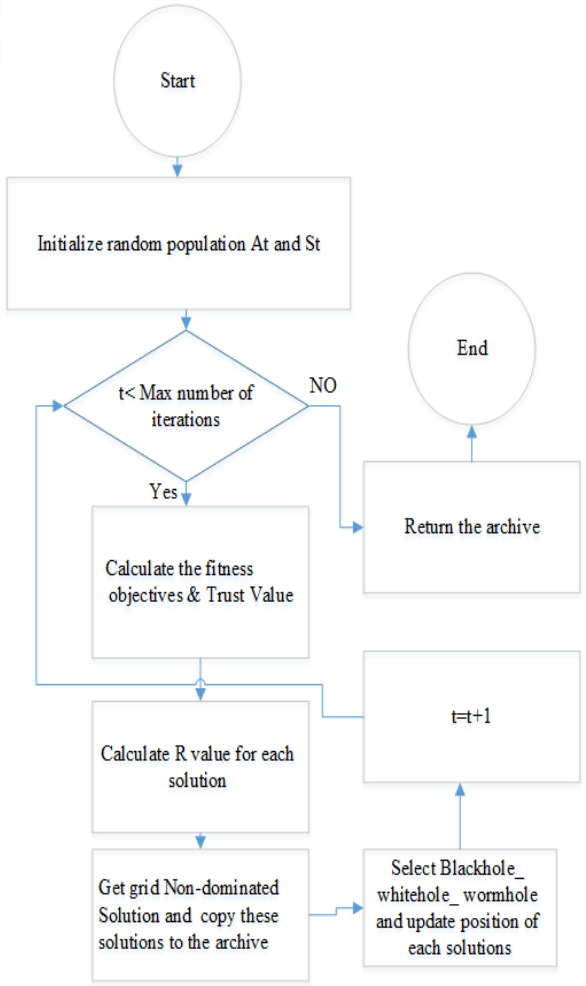

Fig. 5. Flowchart of the proposed method

V. RESULT EVALUATION

In this section is stated the details of the simulation of the proposed method, which is trust-based workflow scheduling using the MVO algorithm. Initially, the simulation environment specifications, simulation details, and workflow characteristics are presented and then the performance of the proposed algorithm is compared with the gray wolf optimization algorithms [6] , parallel genetic[20] and SPEA2 [5] and finally we analyze the results.

A. Simulation Settings

To evaluate the proposed method, we selected a data center consisting of 20 physical machines with similar characteristics. Each physical machine has Xen virtualization middleware and thus has the ability to share resources like virtual machines. The characteristics of the physical machines are listed in Table 2. It should be noted that Sharp notation has been used to indicate the number.

TABLE II. PHYSICAL MACHINES SPECIFICATIONS

Host #	Core#	CPU speed (MIPS)	RAM (MB)	Storage (MB)	BW (Mbps)
1-20	8	2000000	20480000	1000000	10000

Since resource allocation is done by assigning tasks on virtual machines, it is necessary to have a simulated cloud environment with virtual machines that these machines are located on the physical machines. As a result, we have placed 40 virtual machines on this data center; the features of these virtual machines are listed in Table 3.

TABLE III. VIRTUAL MACHINES SPECIFICATIONS

Vm #	Core#	CPU speed (MIPS)	RAM (MB)	Storage (MB)	BW (Mbps)
1-40	1	1000	512	10000	1000

We have also listed the parameters needed to simulate the proposed method, gray wolf algorithm and spea2 algorithm in table 4.

TABLE IV. THE INITIAL PARAMETER OF THE ALGORITHMS

Parameter	Value
Population Size (Proposed Method,PGWO, SPEA2)	50
Archive Size (Proposed Method,PGWO, SPEA2)	10
Maximum Iteration (Proposed Method,PGWO, SPEA2)	100
Maximum Generation (SPEA2)	100
Mutation Probability (SPEA2)	0.5
Crossover Probability (SPEA2)	0.9

Since our proposed method is based on workflow, we have used the real workflow to evaluate our proposed method. Bharathi et al. [21] have introduced the real workflow library, and we used this library to evaluate our proposed method. This library has studied the structure of five actual workflows, which include: (a) Montage (b) Cybershake (c) Epigenomics (d) LIGO (gravitational physics) (e) Sipht (Biology). Figure 6 shows a small sample of each workflow.

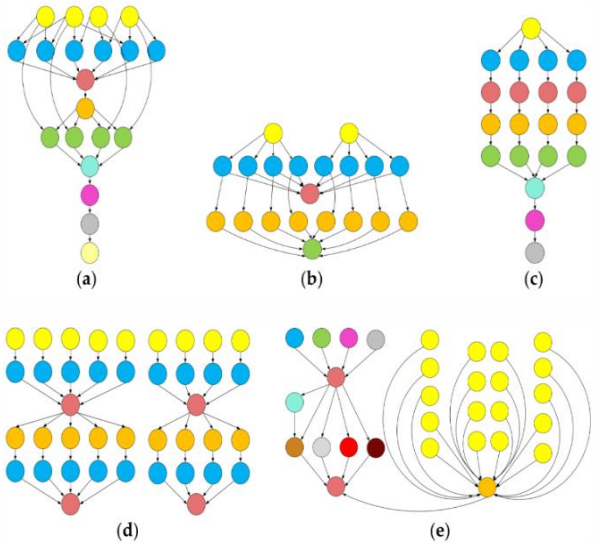

Fig. 6. An example of workflow types [22]

We have selected two balanced (Epigenomics) and unbalanced (Montage) workload categories to evaluate our algorithm.

One type of Input/Output-based workflow is Montage workflow, which creates a large mosaic image of many smaller astronomical images [21]. Depending

on the size of the area of sky of the mosaic, it can have different sizes. The size of a montage workflow depends on the number of images used in the construction of the desired sky mosaic. A simple model of montage workflow is shown in Fig. 6. The colors indicate the level of each task in a general workflow.

Another workflow we have used is Epigenomics workflow; which it is a CPU-dependent bioinformatics workflow with eight levels of requests. The Epigenomics workflow essentially has a data processing line using the Pegasus workflow management system to automate the different sequences of the genome operations. Table 5 shows the characteristics of each workflow.

TABLE V. THE WORKFLOW CHARACTERISTICS USED IN THIS PAPER

Workflow	Jobs	CPU hours	I/O Read (GB)	I/O Write (GB)	Peak Memory (MB)	CPU Utilization
Montage	10429	4.93	146.01	49.93	16.77	31.04 %
Epigenomics	529	7.45	24.14	5.36	197.47	95.91 %

Each of the montage and Epigenomics workload is selected in three sizes of 50,100, 1000.

B. Simulation results

We have repeat our experiments ten times and have calculated the mean of each result for the four objectives of Makespan time, average response time, the average resource utilization, and imbalancing degree for two workflows, in small, medium, and large sizes. Tables 6 and 7 show the objective function values for the two workflows in the three small, medium, and large sizes. From the objective function values for the Unbalanced Workflow (Montage) in Table 6 and the Balanced Workflow (Epigenomics) in Table 7 we can conclude:

1. For large and medium workflow in montage workload and for large workflow in Epigenomics workload, the Makespan is lower in the proposed algorithm than in the other algorithms and in other cases, the Makespan value in the PGWO algorithm is better than the proposed algorithm.
2. The average resource utilization of montage workload in medium and large mode and for Epigenomics workload in large states in the proposed method is better than other methods.
3. The average response time at Montage workload in all three modes and Epigenomics loading at medium and large modes in the proposed algorithm is better than the other algorithms.

As the results of Tables 6 and 7 show, the proposed method has a good improvement in makespan compared to other methods, as Eq.1 shows the Makespan's reduction causes improvements in resource utilization; as a result, the proposed method has good improvement in resource utilization. Figure 7, 8 show the degree of imbalance for montage and Epigenomics workflow in three small, medium, and large sizes.

TABLE VI. VALUES OF THE OBJECTIVES AT THE MONTAGE

Response time	Utilization	Makespan	Montage	Size
135.43	0.1312	105.61	PGWO	small
154.23	0.1231	101.55	SPEA2	
128.3	0.1211	109.1	Parallel GA[21]	
131.5	0.1302	109.51	Proposed Method	
129.45	0.3123	125.12	PGWO	Medium
134.65	0.32765	121.31	SPEA2	
113.54	0.3123	125.6	Parallel GA[21]	
114.76	0.31765	126.11	Proposed Method	
405.67	0.46128	391.61	PGWO	large
442.1	0.45432	400.01	SPEA2	
379.34	0.4621	387.2	Parallel GA[21]	
371.54	0.47123	361.82	Proposed Method	

TABLE VII. VALUES OF THE OBJECTIVES AT THE EPIGENOMICS

Response time	Utilization	Makespan	Epigenomics	Size
2540	0.2312	27450	PGWO	small
2676	0.1108	31054	SPEA2	
2543	0.1902	31121	Parallel GA[21]	
2729	0.2101	31180	Proposed Method	
91765	0.4321	88654	PGWO	Medium
91923	0.4127	91234	SPEA2	
89810	0.4147	88021	Parallel GA[21]	
89132	0.4498	87123	Proposed Method	
164345	0.7054	165234	PGWO	large
165941	0.6821	167312	SPEA2	
161324	0.6945	160757	Parallel GA[21]	
158371	0.7231	155987	Proposed Method	

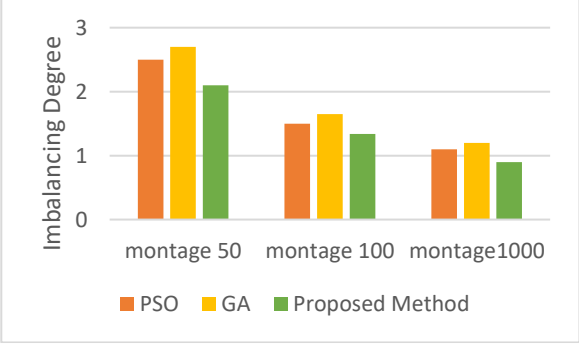

Fig. 7. Comparison of the load imbalance for montage

As the results of Tables 6 and 7 show, the proposed method reduces makespan time compared to other methods. Since the Makespan decrease reduces the Tmax in Eq.1, the Tmax-Tmin difference is reduced and the degree of imbalancing is reduced. As the number of requests increases, the amount of overload

on virtual machines increases, resulting in an improvement in imbalancing degree.

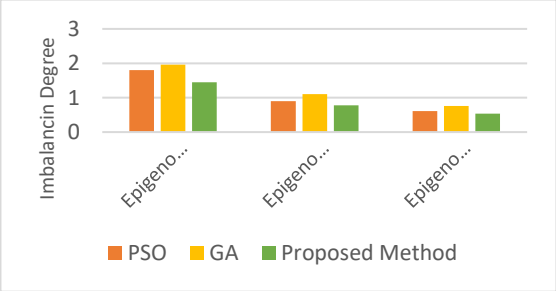

Fig. 8. Comparison of the load imbalance for Epigenomics

We show "Average Ratio of Successful Execution" metric to evaluate the impact of trust in the proposed method compared to when the trust factor was not used in the fitness function.

Successful execution of the proposed method means maintaining a legal deadline for the beginning and end of each task in each type of execution. Figures 9 and 10 show the "Average Ratio of Successful Execution" metric for each montage, Epigenomics for the three small, medium and large sizes.

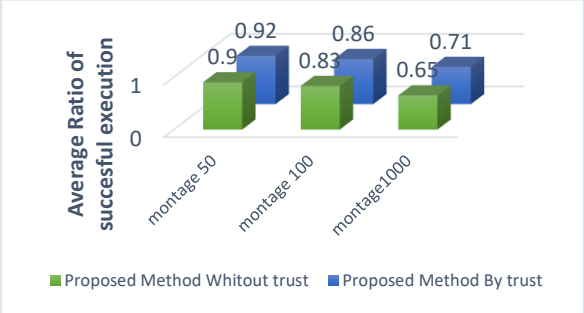

Fig. 9. Comparison of the Average Ratio of Successful Execution for montage workflow.

Fig. 10. Comparison of the Average Ratio of Successful Execution for Epigenomics workflow.

As the simulation results show, by increasing the number of requests, the number of successful execution in the proposed method has better results compared to the method in which trust is not considered. This improvement is due to the choice of virtual machines that have been more successful in accepting applications.

As the simulation results show, the proposed method often leads to better results in each of the service quality objectives, at different workloads, compared to other methods. One of the main reasons for increasing the performance of the proposed method is to use the Grid dominance relationship in converting the single objective MVO algorithm to a multi-objective MVO algorithm. The Grid Rank (Grid) and Grid Crowding Distance (GCD) equations were have used in the Grid dominance relation.

Using the GR equation converges solutions to the best solution. In such cases, the selected optimal solution may be local optimum. For this purpose, the GCD equation is used. Using this factor causes a solution to be chosen that has a greater distance to its neighbors. As a result, the degree of diversity in the selection of solutions increases. The use of GR in conjunction with GCD increases diversity and convergence in the selection of optimal solutions. These solutions have more appropriate values in service quality factors compared to other solutions.

In PGWO [6] and SPEA2 [5] methods, the Pareto dominance relationship is used to find the optimal tradeoff. This relationship increases the possibility of choosing the local optimal solution compared to the Grid dominance relationship. As a result, in these methods, with the increase in the number of requests and virtual machines, as well as with the increase in the number of conflicting optimization objectives, the sample space expands, and thus, the possibility of getting stuck in the local optimization increases.

VI. Conclusion and Future Work

The main purpose of this paper is to present a new multi-objective heuristic algorithm to solve the workflow-scheduling problem in the cloud environment, which on the one hand can satisfy the users by fulfilling the service quality requirements and on the other hand can increase provider profitability by providing the quality requirements of the service providers. One of the important factors in scheduling tasks in the workflow is to increase the trust to the virtual machines; increasing this criterion increases the efficiency of the scheduling algorithm and so the degree of failure is reduced in the performance of executing the workflow.

In this paper, we propose a new workflow scheduling approach by presenting a new model for estimating trust capability. We have also used the Grid dominance Relationship for a purposeful search in the environment. Using the grid dominance in the proposed method increases the convergence and diversity of solutions and thus increases the efficiency of the proposed approach compared to the previous algorithms. In the future, we intend to provide dynamic scheduling for workflow applications using this algorithm, as well as we plan to apply this method to scheduling algorithms in the cloud environment with many objectives.

References

[1] F. Ebadifard and S. M. Babamir, "A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment," Concurrency and Computation: Practice and Experience, vol. 30, no. 12, p. e4368, 2018.

[2] F. Ebadifard and S. M. Babamir, "Scheduling scientific workflows on virtual machines using a Pareto and

hypervolume based black hole optimization algorithm," The Journal of Supercomputing, 2020/02/06 2020.

[3] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization," Neural Computing and Applications, vol. 27, no. 2, pp. 495-513, 2016/02/01 2016.

[4] S. Yang, M. Li, X. Liu, J. Zheng, A grid-based evolutionary algorithm for manyobjective optimization, IEEE Trans. Evol. Comput. 17 (5), Pp.721–736, 2013.

[5] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm, "Technical Report 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 2001.

[6] A. Khalili And S. M. Babamir, "Optimal Scheduling Workflows In Cloud Computing Environment Using Pareto-Based Grey Wolf Optimizer," Concurrency And Computation: Practice And Experience, Vol. 29, No. 11, Pp. E4044-N/A, 2017, Art. No. E4044.

[7] F. Ebadifard, S. M. Babamir, and S. Barani, "A Dynamic Task Scheduling Algorithm Improved by Load Balancing in Cloud Computing," in 2020 6th International Conference on Web Research (ICWR), 2020, pp. 177-183.

[8] F. Ebadifard and S. M. Babamir, "Federated Geo-Distributed Clouds: Optimizing Resource Allocation Based on Request Type Using Autonomous and Multi-objective Resource Sharing Model," Big Data Research, vol. 24, p. 100188, 2021/05/15/ 2021.

[9] F. Ebadifard and S. M. Babamir, "Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment," Cluster Computing, vol. 24, no. 2, pp. 1075-1101, 2021/06/01 2021.

[10] W. Wang, G. Zeng, D. Tang, and J. Yao, "Cloud-DLS: Dynamic trusted scheduling for Cloud computing," Expert Syst. Appl., vol. 39, no. 3, pp. 2321–2329, 2012.

[11] X. Xie, R. Liu, G. Zhou, and J. Ni, "Research of Job Scheduling With Cloud Based On Trust Mechanism And SFLA," vol. 8, no. 1, pp. 93–100, 2015.

[12] P. Gupta, S.P. Ghrera, Fault tolerant big bang-big crunch for task allocation in cloud infrastructure. Int. J. Adv. Intell. Paradig. 10(4), 329–343 (2018).

[13] P. Gupta, S.P. Ghrera, Power and fault aware reliable resource allocation for cloud infrastructure.Proc. Comput. Sci. 78, 457–463 (2016).

[14] Li, W., Wu, J., Zhang, Q., Hu, K., & Li, J. (2014). Trust-driven and QoS demand clustering analysis based cloud workflow scheduling strategies. Cluster Computing, 17(3), 1013-1030.

[15] S. E. Shukri, R. Al-Sayyed, A. Hudaib, and S. Mirjalili, "Enhanced multi-verse optimizer for task scheduling in cloud computing environments," Expert Systems with Applications, vol. 168, p. 114230, 2021/04/15/ 2021.

[16] F. Ebadifard, S. Doostali, and S. M. Babamir, "A Firefly-based Task Scheduling Algorithm for the Cloud Computing Environment: Formal Verification and Simulation Analyses," in 2018 9th International Symposium on Telecommunications (IST), 2018, pp. 664-669.

[17] F. Ebadifard and S. M. Babamir, "Dynamic task scheduling in cloud computing based on Naïve Bayesian classifier," Proceedings of the International Conference for Young Researchers in Informatics, Mathematics and Engineering Kaunas, Lithuania, April 28, 2017, vol. 1852, 2017.

[18] F. Ebadifard and S. M. Babamir, "Optimizing multi objective based workflow scheduling in cloud computing using black hole algorithm," in 2017 3th International Conference on Web Research (ICWR), 2017, pp. 102-108.

[19] F. Ebadifard and S. M. Babamir, "A modified black hole-based multi-objective workflow scheduling improved using the priority queues for cloud computing environment," in 2018 4th International Conference on Web Research (ICWR), 2018, pp. 162-167.

[20] M. Sardaraz and M. Tahir, "A parallel multi-objective genetic algorithm for scheduling scientific workflows in cloud computing," International Journal of Distributed Sensor Networks, vol. 16, no. 8, p. 1550147720949142, 2020/08/01 2020.

[21] Bharathi, S., Chervenak, A., Deelman, E., Mehta, G., Su, M.H. and Vahi, K.''Characterization of scientific workflows'', The 3rd Workshop on Workflowsim Support of Large Scale Science, Austin, TX, USA, pp. 1–10 (2008).

[22] Pegasus Workflow Management System, available at:http://pegasus.isi.edu/workflow_gallery/index.php.

Fatemeh Ebadifard received her M.Sc. degree in Software Engineering from Iran University of Science & Technology and now she is a Ph.D. candidate of Software Engineering under supervision of Dr. Seyed Morteza Babamir at University of Kashan. Her interests are Cloud Computing and Multi-Objective Optimization. She has already published four journal and six conference papers on Cloud Computing and Multi-Objective Optimization.

Seyed Morteza Babamir received his B.Sc. degree in Software Engineering from Ferdowsi University of Mashhad, Mashhad, Iran and M.Sc. and Ph.D. Degrees from Tarbiat Modares University, Tehran, Iran. He is Associate Professor and faculty member at University of Kashan, Kashan, Iran. He has published more than 60 journal and 80 conference papers. He is a researcher working on Distributed Computing, Cloud Computing and Multi-Objective Algorithms.