VOL 3 (2019) NO 3

e-ISSN : 2549-9904
ISSN : 2549-9610

INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

Workflow Scheduling in Cloud Environment Using Firefly Optimization Algorithm

Shahin Ghasemi1[#], Asra Kheyrolahi [*], Abdusalam Abdulla Shaltooki [**]

[#] *Kermanshah University of Medical Sciences, Kermanshah, Iran*

[*] *Department of Computer Engineering, Kermanshah Branch, Islamic Azad University, Kermanshah, Iran*

[**] *Department of Information Technology, University of Human Development, Sulaymaniyah, Iraq*
E-mail: shghasemi@kums.ac.ir, asrakheyrolahi@gmail.com, salam.abdulla@uhd.edu.iq

***Abstract*— One of the issues in cloud computing is workflow scheduling. A workflow models the process of executing an application comprising a set of steps and its objective is to simplify the complexity of application management. Workflow scheduling maps each task to a proper resource and sorts tasks on each resource to meet some efficiency measures such as processing and transmission costs, load balancing, quality of service, and etc. Task scheduling is an NP-Complete problem. In this study, meta-heuristic firefly algorithm (FA) is used to present a workflow scheduling algorithm. The purpose of the proposed scheduling algorithm is to explore optimal schedules such that the cost of processing and transmission of the whole workflow are minimized while there will be load balancing among the processing stations. The proposed algorithm is implemented in MATLAB and its efficiency is compared with cat swarm optimization (CSO) algorithm. The evaluations show that the proposed algorithm outperforms CSO in finding better solutions.**

***Keywords*— cloud computing, workflow scheduling, optimization algorithm, Firefly.**

I. INTRODUCTION

Cloud computing has been recently presented as a paradigm for hosting and delivering services over the Internet. Cloud computing is a paradigm for distributed computation comprising a number of resources and requests aiming to share resources as service over the Internet [1, 2]. Workflow technology is business processes able to manage new requirements of the firms and improve their productivity. Transferring workflows to the cloud environment enable organizations to use various cloud services to facilitate their workflows [3].

In fact, workflow models executive steps of an application. Workflow scheduling maps each task to a proper resource and sorts tasks on each resource to meet some efficiency measures. Workflow scheduling in a distributed environment of the cloud means allocating each task to distributed processing resources (cloud virtual machines) such that rules governing the workflow are not broken. These rules might include QoS parameters like cost and deadline in addition to dependencies among tasks of the workflow. Since workflow scheduling problem is an NP-complete problem, using evolutionary algorithms is the most appropriate approach for solving that [3, 4].

Already, various evolutionary algorithms like particle swarm optimization (PSO), genetic algorithm (GA), imperialist competitive algorithm (ICA), CSO, ant colony optimization (ACO), artificial bee colony (ABC), cuckoo optimization algorithm (COA), and FA have been presented so far to solve different optimization problems. Amidst, PSO [5], CSO [6], and COA [7] have been used to solve workflow scheduling. These three studies have addressed to same objectives, decreasing processing and transmission costs and increasing load balancing among the processing stations.

In this study, a workflow scheduling algorithm based on FA [8, 9] is proposed. FA models behavior of a set of fireflies and allocates a value corresponding to position fitness of each firefly as a model for firefly pigments and updating the position of fireflies in subsequent iterations to search for the optimal solution. The aim of the proposed scheduling algorithm, like existing studies [5-7], is to optimize both cost and load balancing metrics.

The rest of this study is organized as follows: Section II reviews related work and FY. Also, it presents the proposed scheduling algorithm. Section III discusses the performance evaluation and simulation results. The paper is concluded in Section IV.

II. Material And Method

In this section, we first review some existing workflow scheduling algorithms. Then, we present the FA in details. Finally, the proposed scheduling algorithm is presented.

A. Related Work

Pendia et al. [5] have proposed a heuristic algorithm based on PSO for task scheduling in cloud resources. In this algorithm, processing and transmission costs and load balancing among working stations has been considered. The efficiency of this algorithm has been compared with best resource selection (BRS) algorithm and the results showed the superiority of PSO.

Bilgayan et al. [6] have proposed a heuristic algorithm based on CSO for workflow scheduling in the cloud environment. The efficiency of CSO has been compared with PSO and the results showed that CSO is able to offer better solutions compared to PSO.

Ghasemi et al. [7] have proposed a COA-based workflow scheduling algorithm. The cuckoo search algorithm is a meta-heuristic optimization methodology that has an evolutionary approach in finding and exploration of optimized solutions. The cuckoo optimization algorithm is inspired by the amazing behavior of cuckoo breeding and combines it with a Levy flight method that is a random patrol.

Alkhanak et al. [10] discussed the cost-aware workflow scheduling. The researchers initially provided an overview of concepts and research related to cost-aware scheduling, and then, classified cost-aware challenges based on service quality, system throughput, and system architecture for workflow scheduling in the cloud computing. The focus of this research is on the execution cost of the workflows, and the load balancing is not considered here.

Zhang et al. [11] proposed a method for scheduling grid systems using the birds' movement algorithm. Increasing resource throughput and reducing make-span are two objectives of authors in this study. The authors of this paper implemented their method using the genetic algorithm and after comparing the practical results of the algorithm of birds' movement with the genetic algorithm, it was concluded that the algorithm of the birds' movement is to produce better results.

Rimal et al. [12] used a cyclic & directed graph model to solve the problem of workflow scheduling in the cloud environment. Load balancing is the main purpose of this algorithm, so that more freely accessible workstations can be used. This algorithm relies on the best effort and tries to make the best possible decision.

Xave et al. [13] presented a multi-objective optimization method for workflow scheduling in a cloud environment. Make-span and productivity are the main criteria for optimization in this algorithm. A genetic method was used to design this algorithm. The throughput of this algorithm was compared with the algorithm of birds. The results show that the genetic algorithm has better throughput.

Vorma et al. [14] also focused on the multi-objective optimization problem of workflow scheduling in the cloud environment. The proposed scheduling algorithm in this study is based on a hybrid PSO method whose goal is to optimize two criteria: make-span and cost.

Goyal et al. [15] presented a hybrid algorithm based on two methods PSO and ACO. In this hybrid algorithm, parameters such as processor power, processor memory, the cost of running a task on a specific processor, the cost of communication links between processors and the cost of communication between the tasks are used.

Rodriguez et al. [16] presented a workflow scheduling algorithm for scientific workflows, based on short-term budgeting. In this algorithm, a detailed pricing model has been used that enables the user to avoid unnecessary use of the Internet. The main purpose of this algorithm is to reduce the make-span. To achieve this goal, criteria such as budget constraints and quality of service were used.

B. Firefly Algorithm (FA)

The flashing light of fireflies is an amazing sight in the summer sky in the tropical and temperate regions. There are about two thousand firefly species, and most fireflies produce short and rhythmic flashes. The pattern of flashes is often unique for a particular species. The flashing light is produced by a process of bioluminescence, and the true functions of such signaling systems are still debating. However, two fundamental functions of such flashes are to attract mating partners (communication) and to attract potential prey. In addition, flashing may also serve as a protective warning mechanism. The flashing light can be formulated in such a way that it is associated with the objective function to be optimized, which makes it possible to formulate new optimization algorithms.

Yang [10] used the following three idealized rules for simplicity in describing the FA:

1. All fireflies are unisex so that one firefly will be attracted to other fireflies regardless of their sex;
2. Attractiveness is proportional to their brightness, thus for any two flashing fireflies, the less bright one will move towards the brighter one. The attractiveness is proportional to the brightness and they both decrease as their distance increases. If there is no brighter one than a particular firefly, it will move randomly;
3. The brightness of a firefly is affected or determined by the landscape of the objective function. For a maximization problem, the brightness can simply be proportional to the value of the objective function. Other forms of brightness can be defined in a similar way to the fitness function in genetic algorithms.

Based on these three rules, the basic steps of the FA can be summarized as the pseudo-code shown in Fig. 1.

Firefly Algorithm

Objective function $f(\mathbf{x})$, $\mathbf{x} = (x_1, ..., x_d)^T$
Generate initial population of fireflies $\mathbf{x}_i$ $(i = 1, 2, ..., n)$
Light intensity I_i *at* $\mathbf{x}_i$ *is determined by* $f(\mathbf{x}_i)$
Define light absorption coefficient γ
while *(t <MaxGeneration)*
for $i = 1 : n$ *all n fireflies*
  for $j = 1 : i$ *all n fireflies*
    if *($I_j > I_i$), Move firefly i towards j in d-dimension;* **end if**
    Attractiveness varies with distance r via $\exp[-\gamma r]$
    Evaluate new solutions and update light intensity
  end for *j*
end for *i*
Rank the fireflies and find the current best
end while
Postprocess results and visualization

Fig. 1 Pseudo code of the firefly algorithm (FA) [9]

C. The Proposed Scheduling Algorithm based on FA

The novelty of the proposed scheduling algorithm is as follows:

- Modeling the workflow scheduling problem in a cloud environment using FA.
- Solving the workflow scheduling optimization problem using FA.

To this end, steps of FA are modified such that it can solve workflow scheduling problem in the cloud environment which is a multi-objective and discrete problem. The main objective of the proposed scheduling algorithm is to optimize two metrics:

1) Execution cost of the whole workflow;
2) Load balancing among workstations to find optimal scheduling.

C.1 Map Between Workflow Scheduling and FA

Before describing the proposed scheduling method, it is required to present a map between workflow scheduling and FA. Assuming that $VM=\{PC_1, PC_2, .., PC_m\}$ is the set of workstations and $T=\{t_1, t_2, .., t_n\}$ is the set of tasks in the workflow model, the proposed mapping would be as follows:

- The number of dimension of the problem, *d*, is mapped to the number of tasks in the workflow.
- The location of each firefly, x_i, is mapped to a possible solution to the workflow scheduling problem.

$$x_i = \left[\overbrace{PC_i}^{t_1} \quad \overbrace{PC_j}^{t_2} \quad \overbrace{\dots}^{\dots} \quad \overbrace{PC_k}^{t_n}\right]$$

- Intensity (I) is mapped to the fitness of each solution of the workflow scheduling. Schedules with less cost and higher load balancing are equal to fireflies with higher intensity.
- Movement of low-intensity fireflies towards fireflies with higher intensity is mapped to changing non-optimal schedules to more optimal schedules.

C.2 The Proposed Algorithm Description

Considering basic FA in Fig. 1, the proposed scheduling algorithm is as follows:

1. The total number of processing stations is *m*, the number of tasks in the workflow model is *n*, and the population of fireflies is *MaxPop*.
2. Generating an initial set of fireflies. Each firefly is equal to a possible solution in the workflow scheduling problem. The location of each firefly is defined as a vector with *d* members where each member might be a value in the definition domain of $VM=\{PC_1, PC_2, .., PC_m\}$. For instance, firefly $x_i=\{1, 2, 2, 1, 3, 4, 1, 5\}$ indicates a scheduling in which, task t_1 is performed by PC_1, tasks t_2 and t_3 are performed by PC_2, t_4 is performed by PC_1, t_5 is performed by PC_3, t_6 is performed by PC_4, t_7 is performed by PC_1, and t_8 is performed by PC_5.
3. Defining intensity I_i for fireflies at position x_i (calculating the fitness of the initial population).
4. Determining brightness attraction coefficient $\gamma=1$
5. Repeating the following steps to the number of *MaxGeneration*.
   a. The following steps are repeated for each pair of fireflies' *i* and *j* ($i \leq j$):

If fitness(*j*) is higher than fitness(*i*), firefly *i* should move towards firefly *j*. In the proposed movement method, first, *k* random members of the location of firefly *i* are selected and then, they are replaced with the corresponding items of the location of firefly *j*. In the following example, a simple instance of movement of firefly *i* towards firefly *j* is shown in Fig 2. In this example, items 3, 5, and 6 are selected randomly from firefly *i* and they are replaced with corresponding items of firefly *j*. Thus, firefly *i* moves towards firefly *j*

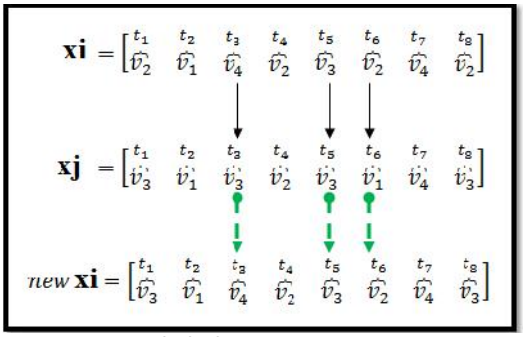

Fig. 2 An example of a firefly movement in the proposed algorithm

   - varying attractiveness using exp[- r]. Where r is the distance of two fireflies i and j. In the proposed algorithm, the distance of two fireflies is obtained based on non-identical items in their locations.
   - Evaluate new solutions and update light intensity.

   b. Best fireflies (best schedules with low cost and high load balancing) are selected for the next iteration.
6. Present the best firefly as an optimal solution

C.3 Fitness Evaluation

At this stage of the proposed algorithm, the fitness of each firefly is calculated. Different cloud service providers have provided several pricing policies to determine the cost of services in the cloud environment. For example, service provider Amazon has provided the Amazon Web Services AWS calculator to calculate costs for its users. If each firefly's position is represented by a vector M, then, according to the equations (1) ~ (4) in the researches [5] and [6], the fitness of each firefly is obtained by equation (4). Equation (1) and equation (2) compute the processing cost and transmission cost imposed on each VM by scheduling M, respectively. Equation (3) calculates the total cost (the plural of the processing and transmission costs) imposed on each VM. Finally, the fitness of M is calculated by equation (4). In fact, the fitness of each firefly is equal to the maximum processing and transmission costs imposed on VMs

$$C_{exe}(M)_j = \sum_k w_{kj} \qquad \forall M(k) = j \tag{1}$$

$$C_{tx}(M)_j = \sum_{k1 \in T} \sum_{k2 \in T} d_{M(k1),M(k2)} \times e_{k1\,k2} \qquad \forall M(k1) = j \text{ and } M(k2) \neq j \tag{2}$$

$$C_{total}(M)_j = C_{exe}(M)_j + C_{tx}(M)_j \tag{3}$$

$$Cost(M) = max\left(C_{total}(M)_j\right) \qquad \forall j \in P \tag{4}$$

III. Results And Discussion

In this section, the efficiency of the proposed algorithm is evaluated. The proposed algorithm is implemented in MATLAB and its efficiency is compared with CSO [6]. In order to evaluate the proposed method and compare it with the basic algorithm, the following measures are used:

- **Total cost**: this measure describes the total cost imposed on workstations for scheduling all tasks of a workflow. This cost includes the sum of processing and transmission costs as evaluated in equation (4).
- **Load balancing**: this measure indicates load distribution among workstations using the scheduling algorithm. For each scheduler, load (processing load or data transfer) imposed on each workstation might be different. If the load imposed in processing stations is balanced, load balancing increases as a result of which, the efficiency of the cloud system is increased.

In order to do the experiments, the data set presented in [6] is used. This data set is selected due to the followings:

- This data set is valid and it is referred in many studies.
- Data set is complete and includes all data and inputs required for implementing the proposed method.
- The basic algorithm, CSO, is also evaluated on this data set.

This data set is a workflow comprising 17 tasks which its dependency graph is shown in Fig. 3. In addition, in this data set, there are 5 VMs and their connectivity is shown in Fig. 4. In this connectivity, the links between VMs and storage resources (DR_i) is shown. This connectivity is a complete graph; that is, there is an independent link between each two VMs.

Also, Table 1. shows the processing cost of each task on each VM. In addition, the transmission cost between VMs is shown in Table 2

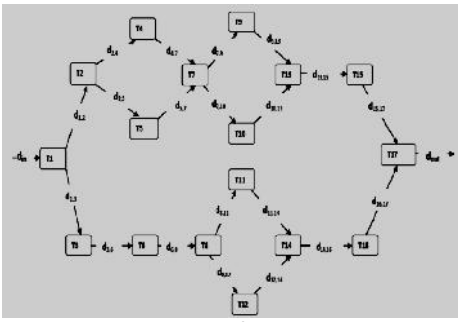

Fig. 3 The workflow model [6

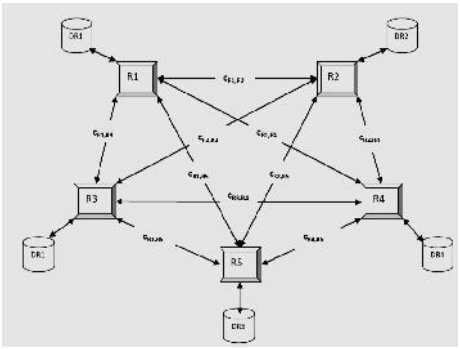

Fig. 4 The connectivity of VMs [6]

Table I
The processing cost of each task on each VM (cent) [6]

	VM1	*VM2*	*VM3*	*VM4*	*VM5*
T1	1.23	1.12	1.15	1.13	1.16
T2	1.17	1.17	1.28	1.14	1.17
T3	1.13	1.11	1.11	1.13	1.16
T4	1.26	1.12	1.14	1.15	1.15
T5	1.19	1.14	1.22	1.15	1.15
T6	1.23	1.12	1.15	1.13	1.15
T7	1.13	1.11	1.11	1.13	1.11
T8	1.26	1.12	1.14	1.15	1.12
T9	1.11	1.11	1.13	1.15	1.13
T10	1.26	1.12	1.14	1.15	1.17
T11	1.17	1.17	1.28	1.14	1.17
T12	1.26	1.12	1.14	1.15	1.15
T13	1.23	1.12	1.15	1.13	1.14
T14	1.26	1.12	1.14	1.15	1.17
T15	1.26	1.12	1.14	1.15	1.15
T16	1.11	1.11	1.13	1.15	1.13
T17	1.17	1.17	1.28	1.14	1.17

Table II
The communication cost between VMs (cents per megabyte) [6]

	VM1	*VM2*	*VM3*	*VM4*	*VM5*
VM1	0	0.01	0.15	0.19	0.2
VM2	0.01	0	0.15	0.19	0.05
VM3	0.15	0.15	0	0.2	0.19
VM4	0.19	0.19	0.2	0	0.15
VM5	0.2	0.01	0.11	0.15	0

In the experiments, the size of files transferred among tasks is varied from 64MB to 1024MB. The maximum population size is *Max_Pop*=50 (for the proposed and the basic algorithms). Each experiment is executed 1000 times and the final results are obtained by averaging over these 1000 times. In the experiments, the efficiency of the proposed algorithm and CSO are evaluated for 50 to 400 iterations.

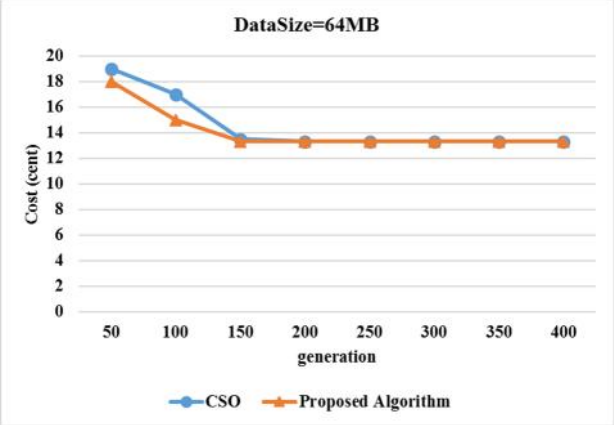

Fig. 5 Comparison of the proposed algorithm and CSO in terms of the total cost for *DataSize*=64MB

Fig. 5 shows the results of experiments for *DataSize*= 64MB. As can be seen, after 50 iterations, the best solution obtained by the proposed algorithm has a cost of 18 while the best solution of CSO has a cost of 19. For iterations less than 150, the proposed algorithm performs better than CSO. The proposed algorithm has obtained the best solution after 150 iterations but CSO has obtained the best solution after 200 iterations

In addition, Fig. 6 shows the results for DataSize=128MB, Fig. 7 shows the results for DataSize=256MB, Fig. 8 shows the results for DataSize=512MB, and Fig. 9 shows the results for DataSize=1024MB. The results of these experiments indicate the desired performance of the proposed algorithm. By changing the size of files, the proposed algorithm has explored better solutions in fewer iterations compared to CSO.

The proposed algorithm is superior over CSO because CSO tends to explore local optimal solutions while the proposed algorithm which is based on FA tends to explore global optimal solutions. In addition, in the proposed algorithm, the whole population moves towards optimal points at each iteration. Thus, its convergence rate is higher.

Fig. 6 Comparison of the proposed algorithm and CSO in terms of the total cost for *DataSize*=128MB

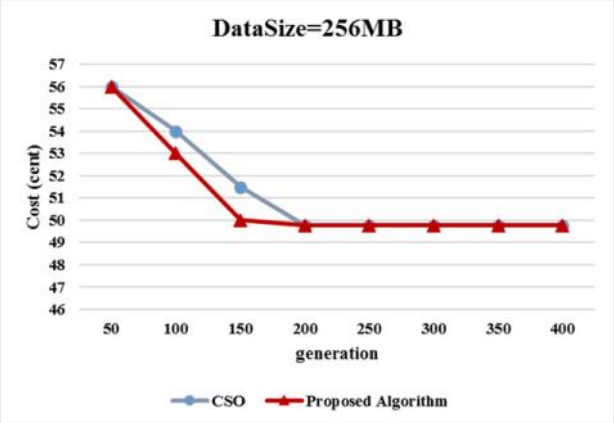

Fig. 7 Comparison of the proposed algorithm and CSO in terms of the total cost for *DataSize*=128MB

Fig. 8 Comparison of the proposed algorithm and CSO in terms of the total cost for *DataSize*=128MB

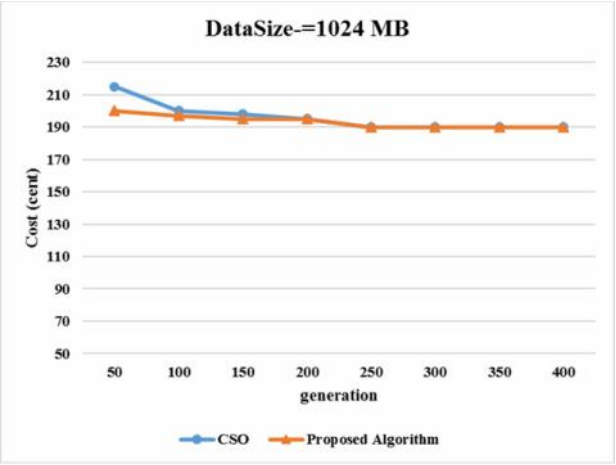

Fig. 9 Comparison of the proposed algorithm and CSO in terms of the total cost for *DataSize*=128MB

In addition, in another experiment, the load balancing among VMs is evaluated. This experiment is implemented for 200 iterations of the proposed algorithm. This experiment is implemented for different values of *DataSize* and the results are shown in Fig. 10. As can be seen, the proposed algorithm establishes the desired load balancing among VMs.

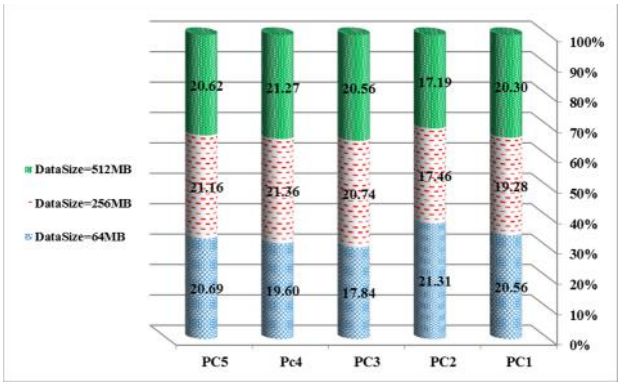

Fig. 10 The load balancing of the proposed algorithm

IV. CONCLUSION

In this study, workflow scheduling in the cloud environment is investigated. Various algorithms have been presented for solving workflow scheduling in cloud computing. In this study, the firefly algorithm is used to present a new workflow scheduling algorithm in cloud computing. In the design of the proposed algorithm, the main purpose is to optimize the cost of executing the whole workflow and load balancing among workstations. The proposed algorithm is implemented and its efficiency is evaluated in terms of total cost and load balancing among VMs. Simulation results of the proposed algorithm are compared with the results of CSO and the results show that the proposed algorithm finds better solutions.

REFERENCES

[1] S Abolfazli, S., Sanaei, Z., Sanaei, M.H., Shojafar, M. and Gani, A., 2015. Mobile cloud computing: The-state-of-the-art, challenges, and future research.

[2] Ranjbari, M. and Torkestani, J.A., 2018. A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers. Journal of Parallel and Distributed Computing, 113, pp.55-62.

[3] Wang, J., Korambath, P., Altintas, I., Davis, J. and Crawl, D., 2014. Workflow as a service in the cloud: architecture and scheduling algorithms. Procedia computer science, 29, pp.546-556.

[4] Bala, A. and Chana, I., 2011, November. A survey of various workflow scheduling algorithms in cloud environment. In 2nd National Conference on Information and Communication Technology (NCICT) (pp. 26-30).

[5] Pandey, S., Wu, L., Guru, S.M. and Buyya, R., 2010, April. A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In Advanced information networking and applications (AINA), 2010 24th IEEE international conference on (pp. 400-407). IEEE.

[6] Bilgaiyan, S., Sagnika, S. and Das, M., 2014, February. Workflow scheduling in cloud computing environment using cat swarm optimization. In Advance Computing Conference (IACC), 2014 IEEE International (pp. 680-685). IEEE.

[7] Ghasemi, S., Hanani, A., 2019. A Cuckoo-based Workflow Scheduling Algorithm to Reduce Cost and Increase Load Balance in the Cloud Environment. JOIV: International Journal on Informatics Visualization, 3(1), pp. 79-85.

[8] Yang, X. S., 2010. Firefly algorithm, Levy flights and global optimization. In Research and development in intelligent systems XXVI (pp. 209-218).

[9] Yang, X. S., 2009. Firefly algorithms for multimodal optimization. In International symposium on stochastic algorithms, pp. 169-178, Springer, Berlin, Heidelberg.

[10] Alkhanak, E.N., Lee, S.P. and Khan, S.U.R., 2015. Cost-aware challenges for workflow scheduling approaches in cloud computing environments: Taxonomy and opportunities. Future Generation Computer Systems, 50, pp.3-21.

[11] Zhang, L., Li, K., Li, C. and Li, K., 2017. Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems. Information Sciences, 379, pp.241-256.

[12] Rimal, B.P. and Maier, M., 2017. Workflow scheduling in multi-tenant cloud computing environments. IEEE Transactions on Parallel and Distributed Systems, 28(1), pp.290-304.

[13] Zhu, Z., Zhang, G., Li, M. and Liu, X., 2016. Evolutionary multi-objective workflow scheduling in cloud. IEEE Transactions on parallel and distributed Systems, 27(5), pp.1344-1357.

[14] Verma, A. and Kaushal, S., 2017. A hybrid multi-objective Particle Swarm Optimization for scientific workflow scheduling. Parallel Computing, 62, pp.1-19.

[15] Goyal, M. and Aggarwal, M., 2017. Optimize workflow scheduling using hybrid ant colony optimization (ACO) & particle swarm optimization (PSO) algorithm in cloud environment. Int. J. Adv. Res. Ideas Innov. Technol, 3(2).

[16] Rodriguez, M.A. and Buyya, R., 2017. Budget-driven scheduling of scientific workflows in IaaS clouds with fine-grained billing periods. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 12(2), pp.5.