Deadline constrained task scheduling in the cloud computing using a discrete firefly algorithm

Mina Hoseinnejhad and Nima Jafari Navimipour*
Department of Computer Engineering, Tabriz Branch, Islamic Azad University, Tabriz, Iran
Jafari@iaut.ac.ir

Cloud computing is used to provide convenient and quick access to a shared pool of configurable computing resources. In cloud computing, information technology (IT) related capabilities are provided as services, accessible cloud computing is a model to provide convenient and on-demand access without requiring detailed knowledge of the underlying technologies, and with minimal management effort. There are many challenges in cloud computing. Tasks scheduling is considered as one of these challenges. The concept of scheduling as one of the famous NP-Hard problems is an optimal allocation of suitable resources to tasks. This study presents a new deadline–aware scheduling approach using discrete firefly algorithm. The makespan is improved compared to FCFS, A2DJS, ELPR, and HLBA scheduling algorithm based on the results of simulation in the Cloudsim environment. Also, missed tasks are decreased using the suggested method compared to FCFS, SPN, HRRN, and PSO.

Keywords: Aerial Cloud computing, Scheduler, Deadline, Discrete firefly algorithm

1. INTRODUCTION

Cloud computing is developed through the recent progress in hardware, virtualization technology, distributed computing, and service delivery over the Internet [Fouladi and Navimipour [2017] ; Hazratzadeh and Jafari Navimipour [2017]]. Cloud computing certainly represents a new way of managing IT, although it does not contain a lot of new technologies [Ashouraie and Jafari Navimipour [2015] ; Hajizadeh and Navimipour [2017]]. In many cases, this will change the workflow within the IT organization, and it will result in a complete reorganization of the IT department. Cost savings and scalability can be highly achieved from cloud computing [Sheikholeslami and Navimipour [2017] ; S.Manvi and Shyam [2014]]. The cloud metaphor is a reference to the ubiquitous availability and accessibility of computing resources via Internet technologies [Oliveira et al. [2014]]. Generally, cloud services can be categorized into three main types of services, including Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) [Singh and Chana [2016]]. These services can be accessed through a cloud client, which could be a web browser, mobile app, and so on [Chong et al. [2014] ; Mohammadi and Navimipour [2017]]. The agility, lower capital expenditure, location independence, resource pooling, broad network access, reliability, scalability, elasticity, and ease of maintenance are all provided by cloud computing [Gurkok [2014]].

On the other hand, task scheduler as an NP-Hard problem [Chiregi and Navimipour [2016]] is the important part of any distributed system like grid [Navimipour et al. [2014]], cloud [Laili et al. [2013] ; Wu et al. [2013]] and P2P networks . Jobs are assigned to suitable resources in order to have high performance. Minimizing the overall execution time of a collection of tasks is considered as the goal of task scheduler . Reducing response time and improving the service providers' utilization are considered as the main goals of cloud scheduler.

Evolutionary algorithms like genetic, ant colony, bee colony and particle optimization algorithm (PSO) are used in many published papers for optimization problems. However, the task scheduling is done by discrete firefly algorithm (DFA) due to its advantages. Yang [2008] has developed one of the meta-heuristic algorithms. High convergence speed, flexibility, and insensitivity to basic values and high fault tolerance are considered as its features. For this reason, this algorithm used in optimized multi-faceted Yang [2009] and optimized Random testing functions [Sayadi et al. [2010]]. Firefly algorithm can be used to solve different optimization issues. In practice, any optimization issue that can be solved with genetic algorithm and particle swarm optimization can be also solved by the Firefly algorithm. Yang [2009] proposed a firefly algorithm for multi-modal optimization applications. Lukasik and Zak [2009] have presented a further study on the firefly algorithm for constrained continuous optimization problems. Yang [2010] has applied the firefly algorithm for the optimization of pressure vessel design. Also, a few new test functions are presented to validate the firefly optimization algorithm. Bean [1994] has described the smallest position value (SPV) rule. It enables the continuous firefly algorithm to be applied to discrete scheduling problems. A discrete firefly algorithm (DFA) is proposed in this regard. Researchers have already used the SPV rule to solve the scheduling problems [Marichelvam et al. [2014] ; Tasgetiren et al. [2007]]. The primary goals of this paper can be summarized as follows:

(1) Reducing the total running time jobs that are considered as the completion of jobs time

(2) Reducing missed task that is considered to calculate the number of missed tasks, the start time and the deadline for task execution.

The rest of this paper is organized as follows: the overview and background are reviewed in Section 2. Section 3 presents the DFA algorithm for task scheduling in the cloud environments. Section 4 shows the computational experiments and the results of comparisons. The paper is concluded in the last section and some indications for future researches are suggested.

2. RELATED WORK

The maximization of revenue both on the part of the cloud provider and the user is one of the main goals of cloud computing. Task scheduling has got the main focus in cloud computing [Tsai et al. [2014]] since inefficient task scheduling can lead to revenue loss, performance degradation, and breach of service level agreement (SLA). Therefore, efficient scheduling algorithms are required to minimize both computation-based metrics such as response time, system utilization, makespan, system throughput and network-based metrics such as network communication cost, traffic volume, round trip time, data communication cost [Aceto et al. [2013]]. These metrics are very important in cloud activities in order to address some issues like load balancing, energy efficiency, SLA and quality of service (QoS) guarantee, and fault tolerance. Some important papers about task scheduling in the cloud are reviewed in the rest of this section.

Zeng et al. [2015] have proposed a model, which considers data management to obtain satisfactory makespan on multiple data centers. At the same time, their adaptive data-dependency analysis can reveal parallelization opportunities. An adaptive strategy is suggested for workflow applications. It consists of a set-up stage, which builds the clusters for the workflow tasks and datasets, and a run-time stage, which makes the overlapped execution of the workflows. The proposed method can effectively improve the workflow completion time and utilization of resources in a cloud environment through the rigorous performance of evaluation studies.

Furthermore, Liu et al. [2015] have proposed a fuzzy clustering method to effectively pre-process the cloud resources. A new directed acyclic graph based on scheduling algorithm called earliest finish time duplication algorithm has been proposed for heterogeneous cloud systems by combining the list scheduling with the task duplication scheduling scheme. Earliest finish time duplication attempts to insert suitable immediate parent nodes of the currently selected node in order to reduce its waiting time on the processor. The proposed algorithm is better than the popular heterogeneous earliest finish time algorithms regarding the experimental results.

A genetic based algorithm has been proposed in Navimipour et al. [2014] to schedule the jobs on human resources (HRs) in the expert cloud. The received jobs can be scheduled by the proposed method in appropriate time with high accuracy. Common methods, including (First Come First Served (FCFS), Shortest Process Next (SPN) and Highest Response Ratio Next (HRRN)) are compared with the proposed method. The results show the best performance of the proposed method in term of TET, service?wait time, failure and human resource (HR) utilization rate. But, in term of failure rate, SPN acts slightly better.

It applies a selection method that selects one of the N chromosomes using search algorithms of O(N) complexity.

Many parameters like makespan, latency, and load balancing are considered in quality of QoS-driven. But allocation cost parameter is not considered in QoS-driven scheduling algorithm. Minimizing the total allocation cost is an important issue in cloud computing. The cost is calculated by QoS-driven task scheduling algorithm and compare with traditional task scheduling algorithm in cloud computing environment by Bansal et al. [2015]. The QoS-driven achieves good performance in cost parameter based on cloudsim 3.0 toolkit with NetBeans IDE8.0.

Jena [2015] has presented Multi-Objective Particle Swarm Optimization (MOPSO) based on optimization algorithm, which can solve the task scheduling problem under the computing environment, where the number of data center and user job changes dynamically. But, in changing the environment, cloud computing resources need to be operated in an optimal manner. Therefore, multi-objective nested Particle Swarm Optimization (PSO) based on the algorithm is suitable for cloud computing environment because of its ability to reduce energy and makespan by using the system resources. The experimental results show that the proposed method has better performance in choosing the best resources and random scheduling algorithm.

Finally, Navimipour [2015] has proposed a new artificial bee colony algorithm to schedule the tasks on service providers in the cloud environments because of the advantages of an artificial bee colony algorithm. The results show a better operation of the proposed method in terms of task execution time and waiting time compared to Kumar and Palaniswami [2012] and Keshk et al. [2014].

3. PROPOSED METHOD

This section offers the firefly algorithm and its parameters. A discrete firefly algorithm is proposed for task scheduling in cloud computing. The considered parameters are defined to evaluate the method and finally describe a case study at the same time.

3.1 Firefly Algorithm (FA)

The firefly algorithm is a recently developed nature-inspired metaheuristic algorithm. The firefly algorithm is inspired by the social behavior of fireflies, which are called as lightning bugs. The firefly has 2000 species in the world. Most of the firefly species produce short and rhythmic flashes. These species have a unique pattern of flashes. A firefly's flash mainly acts as a signal to attract mating partners and potential prey. Flashes also serve as a protective warning mechanism. The following three idealized rules are considered in [Yang [2008]] to describe the firefly algorithm.

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 each pair of fireflies, the less bright one will move toward the brighter one. The attractiveness is proportional to the brightness. They both decrease by increasing their distance. 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 may be proportional to the objective function value. For the minimization problem, the brightness may be the reciprocal of the objective function value. The pseudocode of the firefly algorithm has been proposed by Marichelvam et al. [2014] and Yang [2008]. The pseudocode of the firefly algorithm is given in Algorithm 1.

A. Attractiveness

The attractiveness of a firefly is determined by its light intensity. The attractiveness may be calculated by using the equation:

$$(r) =_0 e^{-\gamma r^2} \tag{1}$$

Algorithm 1 Pseudocode of the Firefly Algorithm

Objective function f (x), x =($x_1, \ldots, x_d)^T$
Generate initial population of fireflies x_i= (i = 1, 2, ..., n)
Light intensity I_i at $x_i$is determined by f (x_i)
Define light absorption coefficient
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 toward j in d-dimension
Attractiveness varies with distance r via exp (−r)
Evaluate new solutions and update light intensity
end for
end for
Rank the fireflies and find the current best
end while
Post process results and visualization

B. Distance
The distance between any two fireflies i and j atx_i and x_j is the Cartesian distance as follows:

$$r_{i,j} = x_i - x_j = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{2}$$

C. Movement
The movement of a firefly i that is attracted to another more attractive firefly j is determined by:

$$x_i\,(t+1) = x_i\,(t) +_0 e^{-r_{i,j}^2}\,(x_i - x_j) + \in_i \tag{3}$$

3.2 Discrete Firefly Algorithm

The firefly algorithm has been originally developed for solving continuous optimization problems. The firefly algorithm cannot be applied directly to solve the discrete optimization problems. Bean [1994] has described the smallest position value (SPV) rule, which is used and extended in this study. The SPV rule enables the continuous firefly algorithm to be applied to discrete scheduling problems. A discrete firefly algorithm (DFA) is proposed in this regard. The SPV rule has already been applied by the researchers to solve the scheduling problems [Marichelvam et al. [2014] ; Tasgetiren et al. [2007]].
A. Implementation of the method
1) Solution Representation: Solution representation is one of the most important issues in designing a DFA. The solution search space consists of n dimensions as n number of jobs are considered in this letter. Each dimension represents a job. The vector x_i^t= (x_{i1}^t, x_{i2}^t, ...,x_{in}^t) represents

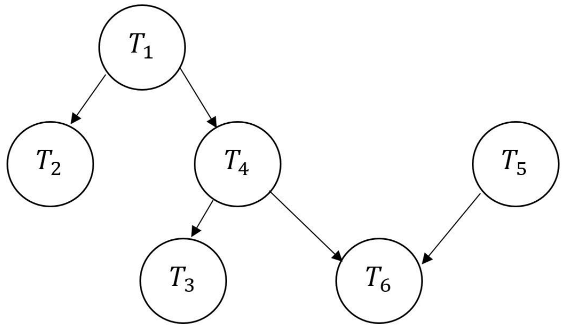

Fig. 1 Acyclic graph for 6 jobs

Table I: Solution representation of a firefly for 6 jobs

Dimension j	**1**	**2**	**3**	**4**	**5**	**6**
x_{ik}^{t}	0/81	0/9	0/12	0/09	0/71	0/63
jobs	5	6	2	1	4	3

the continuous position values of fireflies in the search space. Changing the continuous position values of the fireflies to the discrete job permutation is done by the SPV rule. Table 1 shows the solution representation of a firefly with six jobs.

The smallest position value is x_{i4}^{t}= 0/09, and the dimension j = 4 is assigned to be the first job in the permutation according to the SPV rule. The second smallest position value is $x_{i3}^{t}=\frac{0}{12}$, and the dimension j = 3 is assigned to be the second job in the permutation. Similarly, all the jobs are assigned in the permutation.

2) Population Initialization: In most of the meta-heuristics and also in the DFA, the initial population is generated at random. The continuous values of positions are generated randomly using a uniform random number between 0 and 1.

3) Solution Update: By using the permutation, each firefly is evaluated to determine the objective function value. The objective function value of each firefly is associated with the light intensity of the corresponding firefly. A firefly with less brightness is attracted and moved to a firefly with more brightness. The attractiveness of the firefly is determined using (1). The distance between each pair of fireflies is determined by (2). The SPV rule is applied to obtain the job permutation. The attractiveness is calculated for each firefly. Then, the movement of the firefly is determined by (3) depending on the attractiveness of the firefly. The above steps are repeated until the termination criterion is met. Fig.2. presents the steps on the proposed DFA algorithm by flowchart and the main function of this algorithm is shown in Algorithm 1.

3.2.1 *Fitness function.* Fitness value helps the DFA to decide which firefly would survive to generate the next-generation population. The proposed method is based on tasks scheduling according to deadline and makespan**.** The fitness of the proposed methods to reduce makespan and missed task is calculated by using the equation:

$$Fitness = \mathrm{w}_1 * \text{ Makespane} + w_2 * \text{Missed task} , \quad \mathrm{w}_1 + w_2 = 1 \tag{4}$$

For normalization, the fitness function using the equation

$$\text{Normalized}\,(\mathrm{m}) = \frac{\mathrm{m} - \mathrm{M_{min}}}{\mathrm{M_{max}} - \mathrm{M_{min}}} \tag{5}$$

In the proposed method for makespan, completion of jobs time is determined by

$$makespan = min(c_{max}) \text{ where } \mathrm{c}_{max} \geq \mathrm{c}_t,\ \mathrm{t}=1,2,\ldots.,\ \mathrm{n} \tag{6}$$

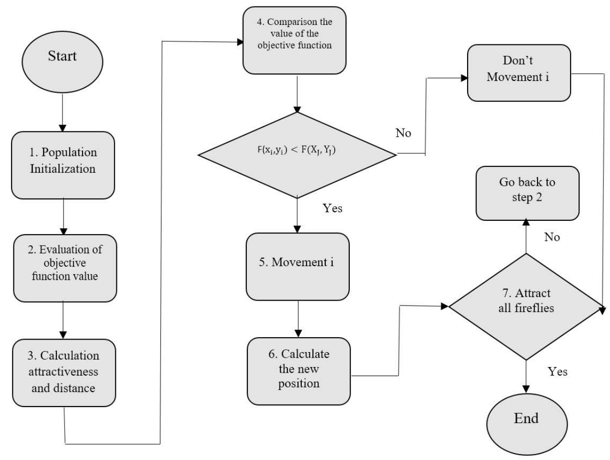

Fig. 2 The DFA flowchart

In the equation (6), t represents tasks, c_{max} represents maximum completion time and completion time of t is c_t. The start time and the deadline for task execution are considered to calculate the number of missed tasks.

3.3 A case study

This section will show the process of implementing the proposed method. The considered values in this example are as follows:

The number of tasks equals 30 and 6 virtual machine, tasks length equal 100 and 200 and the deadline according to the table 2 is considered. The completion time is equal 1.58 second and the number of lost jobs is equal 1. It should be noted that initial attractiveness is generated based on Gaussian distribution and initial dimension is generated at random. The objective function value of each firefly is associated with the light intensity of the corresponding firefly. A firefly with less brightness is attracted and moved to a firefly with more brightness. The attractiveness of the firefly is determined using (1). The distance between each pair of fireflies is determined by (2). The SPV rule is applied to obtain the job permutation. The attractiveness is calculated for each firefly. Then, the movement of the firefly is determined by (3) depending on the attractiveness of the firefly. The above steps are repeated until the termination criterion is met. Fig.3 illustrates this example's Gantt chart.

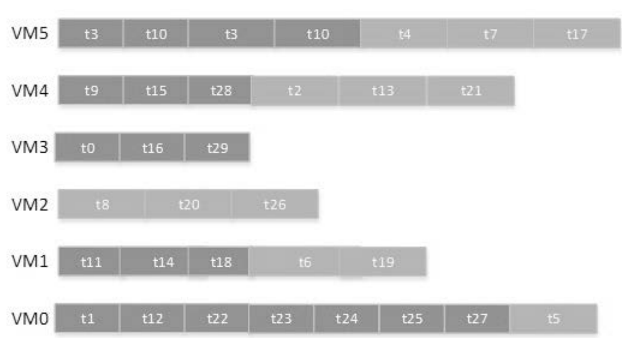

Fig. 3 Gantt chart for 30 tasks and 6 virtual machine

4. EXPERIMENTAL RESULTS

The proposed algorithm is compared with some algorithms, including FCFS, SPN, HRRN, PSO [Navimipour and Milani [2015]] and FCFS, HILBA (Hierarchical Load Balancing Algorithm), ELPR (Enhanced IC-PCP), A2DJS(Adaptive Deadline Based Dependent Job Scheduling) [Komarasamy and Muthuswamy [2015]] to show the behavior of DFA. The discrete firefly algorithm was coded in Java and run on a PC with an Intel Core Duo 3 GHz CPU and 3 GB RAM. The data of Table (2) are used for simulation and comparison of the proposed algorithms with existing algorithms. The proposed algorithm is tested with different types of parameter settings, including the attractiveness of fireflies, light absorption coefficient, and randomization parameter. Table (3) shows the used parameters in this letter. Finally, the jobs features are depicted in Table (4), virtual machines features are depicted in Table (5) and data centers features are depicted in Table (6).

Table II: Service time, deadline and dependency of 40 tasks

Task #	**Service**	**deadline**	**Dependency**
1	4	8	——
2	2	4	——
3	5	10	1
4	8	16	1
5	9	18	——
6	2	4	2
7	1	2	5
8	8	16	——
9	3	6	1,7
10	5	10	2,5,7
11	10	20	6
12	6	12	10
13	3	6	7,3
14	10	3	8,10
15	9	18	5,3
16	9	18	10
17	7	14	——
18	3	6	11
19	5	10	16
20	4	8	15,2
21	3	6	15,11
22	1	2	——
23	7	14	8,9,13
24	8	16	5
25	8	16	1,14
26	3	6	——
27	2	4	7,11
28	10	20	22,26
29	9	18	8
30	6	12	21
31	6	12	19,29
32	6	12	16,19,27
33	4	8	16
34	7	14	23
35	9	18	17
36	4	8	16
37	1	2	——
38	4	8	31
39	7	14	12,24
40	2	4	6,37

Table III: Parameters of the Discrete Firefly Algorithm

parameters	Levels		
	0.0 (low)	0.5 (medium)	1.0 (high)
	0.5 (low)	0.75 (medium)	1.0 (high)
A	0.0 (low)	0.5 (medium)	1.0 (high)

Table IV: Jobs features

Jobs length	Randomly between 1-1000
Input size	300
Output Size	300
Number of processors	1

Table V: Virtual machines features

Speed (mips)	{100,750,500,80}
Number of processors	1
RAM	128
Bandwidth	2500

Table VI: Data centers features

Speed	500000
physical machine Capacity	10000
Storage Capacity	1000000
Bandwidth(mips)	100000

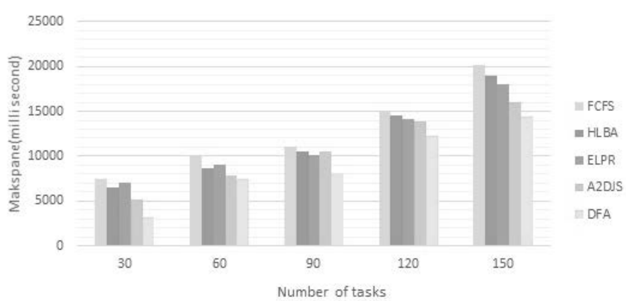

Fig. 4 Results based on the makespan.

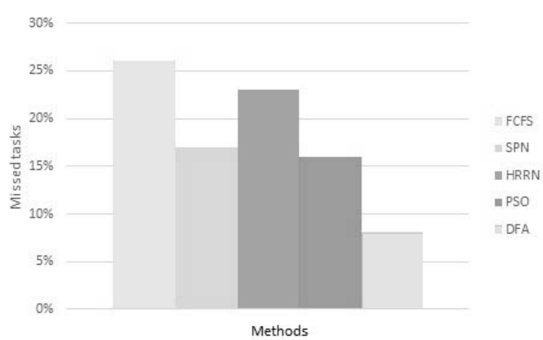

Fig. 5 Result based on the average of missed tasks.

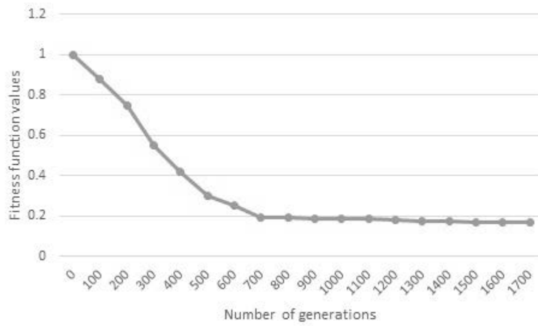

Fig. 6 Fitness function values.

The DFA is compared with other algorithms in terms of makespan as shown in Fig.4. The number of tasks is selected from 30, 60, 90,120 and 150 that the DFA has better performance. The proposed scheduling method has been compared with the commonly used methods, including FCFS, HLBA, ELPR, and PSO to investigate the missing tasks in a heterogeneous environment. The simulation results based on the average of missed tasks show the better performance of DFA, which is illustrated in Fig.5. The simulation results based on the proposed method fitness function values in the different number of repetitions are illustrated in Fig.6.

5. CONCLUSION

There should be a new powerful algorithm for scheduling input tasks in the cloud to achieve the minimum amount of makespan by reducing the number of missed tasks. This paper applies the firefly algorithm, which is inspired by nature and collective intelligence algorithms to solve this problem. The proposed method is evaluated by using a cloudsim simulation environment and the Java programming language. The makespan and missed tasks are reduced according to the simulation results. The proposed method has better performance compared with some other algorithms. Some evolutionary algorithms will be applied in the future for scheduling problem in the cloud environment in which more parameters such as cost and load balancing are optimized.

References

Aceto, G., Botta, A., de Donato, W., and Pescap. 2013. Cloud monitoring. a survey. *Computer Networks*, 2115–2093.

Ashouraie, M. and Jafari Navimipour, N. 2015. Priority-based task scheduling on heterogeneous resources in the expert cloud. *Kybernetes 44*, 1455–1471.

Bansal, N., Maurya, A., Kumar, T., Singh, M., and Bansal, S. 2015. Cost performance of qos driven task scheduling in cloud computing. *Procedia Computer Science 57*, 126–130.

Bean, J. C. 1994. Genetic algorithm and random keys for sequencing and optimization. *ORSA J. Comput. 6*, 154–160.

Chiregi, M. and Navimipour, N. J. 2016. Trusted services identification in the cloud environment using the topological metrics. *Karbala International Journal of Modern Science 2,* 3, 203–210.

Chong, H.-y., Wong, J. S., and Wang, X. 2014. An explanatory case study on cloud computing applications in the built environment. *Automation in Construction 44*, 152–162.

Fouladi, P. and Navimipour, J. N. 2017. uman resources ranking in a cloud-based knowledge sharing framework using the quality control criteria. *Kybernetes 46*, 5.

Gurkok, C. 2014. Securing cloud computing systems. *Network and System Security(Second Edition)*, 83–126.

Hajizadeh, R. and Navimipour, J. N. 2017. *A method for trust evaluation in the cloud environments using a behavior graph and services grouping*. Kybernetes.

Hazratzadeh, S. and Jafari Navimipour, N. 2017. *Colleague recommender system in the Expert Cloud using the features matrix*. Kybernetes.

Jena, R. 2015. Multi objective task scheduling in cloud environment using nested pso framework. *Procedia Computer Science 57*, 1219–1227.

Keshk, A. E., El-Sisi, A. B., and Tawfeek, M. A. 2014. *International Journal of Intelligent Systems and Applications 6*, 25.

Komarasamy, D. and Muthuswamy, V. 2015. Adaptive deadline based dependent job scheduling algorithm in cloud computing. *Seventh International Conference on the Advanced Computing (ICoAC)*.

Kumar, V. V. and Palaniswami, S. 2012. A dynamic resource allocation method for parallel dataprocessing in cloud computing. *Journal of computer science 8*, 780.

Laili, Y., Tao, F., Zhang, L., Cheng, Y., Luo, Y., and Sarker, B. R. 2013. A ranking chaos algorithm for dual scheduling of cloud service and computing resource in private cloud. *Computers in Industry 64*, 448–463.

Liu, Z., Qu, W., Liu, W., Li, Z., and Xu, Y. 2015. Resource preprocessing and optimal task scheduling in cloud computing environments. *Concurrency and Computation: Practice and Experience 27*, 3461–3482.

Lukasik, S. and Zak, S. 2009. Firefly algorithm for continuous constrained optimization tasks. In *Paper presented at the International Conference on Computational Collective Intelligence*.

Marichelvam, M. K., Prabaharan, T., and Yang, X. S. 2014. A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems. *IEEE transactions on evolutionary computation 18*, 301–305.

Mohammadi, S. Z. and Navimipour, J. N. 2017. *Invalid cloud providers' identification using the support vector machine*. International Journal of Next-Generation Computing.

Navimipour, J. N., Masoud Rahmani, A., Habibizad Navin, A., and Hosseinzadeh, M. 2014. Job scheduling in the expert cloud based on genetic algorithms. *Kybernetes 43*, 1262–1275.

Navimipour, N. J. 2015. Task scheduling in the cloud environments based on an artificial bee colony algorithm. In *Paper presented at the International Conference on Image Processing,* *Production and Computer Science*. Istanbul(Turkey).

Navimipour, N. J. and Milani, F. S. 2015. A comprehensive study of the resource discovery techniques in peer-to-peer networks. *Peer-to-Peer Networking and Applications 8*, 474–492.

Oliveira, T., Thomas, M., and Espadanal, M. 2014. Assessing the determinants of cloud computing adoption: An analysis of the manufacturing and services sectors. *Information Management 51*, 497–510.

Sayadi, M., Ramezanian, R., and Ghaffari-Nasab, N. 2010. A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. *International Journal of Industrial Engineering Computations 1*, 1–10.

Sheikholeslami, F. and Navimipour, J. N. 2017. Service allocation in the cloud environments using multi-objective particle swarm optimization algorithm based on crowding distance. *Swarm and Evolutionary Computation*.

Singh, S. and Chana, I. 2016. A survey on resource scheduling in cloud computing: Issues and challenges. *Journal of Grid Computing 14*, 217–264.

S.Manvi, S. and Shyam, G. 2014. Resource management for infrastructure as a service (iaas) in cloud computing: a survey. *Journal of Network and computer and applications 41*, 424–440.

Tasgetiren, M. F., Liang, Y.-c., Sevkli, M., and Gencyilmaz, G. 2007. A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. *European journal of operational research 177*, 1930–1947.

Tsai, C.-W., Huang, W.-C., Chiang, M.-H., Chiang, M.-C., and Yang, C.-S. 2014. *A hyper-heuristic scheduling algorithmfor cloud*. IEEE Trans.Cloud Comput.

Wu, Z., Liu, X., Ni, Z., Yuan, D., and Yang, Y. 2013. A market-oriented hierarchical scheduling strategy in cloud workflow systems. *The Journal of Supercomputing*, 1–38.

Yang, X.-s. 2008. *Nature-Inspired Metaheuristic Algorithm*. Luniver, Bristol, U.K.

Yang, X.-s. 2009. Firefly algorithms for multimodal optimization. In *Paper presented at the International symposium on stochastic algorithms*.

Yang, X.-s. 2010. Firefly algorithm, stochastic test functions and design optimisation. *International Journal of Bio-Inspired Computation 2*, 78–84.

Zeng, L., Veeravalli, B., and Zomaya, A. Y. 2015. An integrated task computation and data management scheduling strategy for workflow applications in cloud environments. *Journal of Network and Computer Applications 50*, 39–48.

Mina Hoseinnejhad received her B.S. in computer science, from Rey Payam Noor University , Tehran, Iran, in 2014 and the M.Sc. in computer engineering, software engineering, from Tabriz Branch, Islamic Azad University, Tabriz, Iran, in 2016. Her research interest is Cloud Computing.

Nima Jafari Navimipour received his B.S. in computer engineering, software engineering, from Tabriz Branch, Islamic Azad University, Tabriz, Iran, in 2007; the M.S. in computer engineering, computer architecture, from Tabriz Branch, Islamic Azad University, Tabriz, Iran, in 2009; the Ph.D. in computer engineering, computer architecture, from Science and Research Branch, Islamic Azad University, Tehran, Iran in 2014. He is an assistance professor in the Department of Computer Engineering at Tabriz Branch, Islamic Azad University, Tabriz, Iran. He has published more than 100 papers in various journals and conference proceedings. His research interests include Cloud Computing, Social Networks, Fault-Tolerance Software, QCA, Evolutionary Computing, and Network on Chip.