ATLANTIS PRESS International Journal of Networked and Distributed Computing, Vol. 6, No. 1 (January 2018) 35-42

Providing a load balancing method based on dragonfly optimization algorithm for resource allocation in cloud computing

Zahra Amini [1], Mehrdad Maeen [2], Mohammad Reza Jahangir [3]

[1,2,3] Department of Computer Engineering, Yadegar-e-Imam Khomeini (RAH), Shahre Rey Branch
Shahre Rey, Tehran, Iran
[1] z.amini@iausr.ac.ir, [2] m.maeen@iausr.ac.ir, [3] jahangir@iausr.ac.ir

Abstract

Nowadays cloud computing is being highly considered by many of researchers, organizations, governments and so on. According to processing happening inside of cloud computing, some of the most important problems and challenges in cloud computing are load balancing, managing resource allocations, scheduling running of tasks. Load balancing on the surface of virtual machines on the internal surface of datacenters, scheduling and resource allocations in hosts and over virtual machines. Thus, by considering existing challenges in cloud computing, in this paper by the help of dragonfly optimization algorithm because of speed and preciseness in scheduling tasks, the process of allocating resources to virtual machines in cloud computing has been done. The proposed method has multiple steps that are as follows: initialization of algorithm and cloud computing, setting number of virtual machines and tasks, running dragonfly optimization algorithm, allocating resources and scheduling tasks with maintaining load balance in virtual machines. By simulation of the proposed method in this research, we observed that the rate of improvement in dragonfly optimization algorithm for resource allocation and keeping load balance between virtual machines is much higher than other methods when considering criterions like execution time, response time, number of migrated tasks and load balance.

Keywords: scheduling tasks, load balance, dragonfly optimization algorithm, resource allocation, cloud computing.

1. Introduction

Cloud computing can be considered as the ability of sharing computational resources between different users. In developing datacenters there is a need for designing some centers which programs run independent of hardware infrastructure and to make it possible to transfer resources from a virtual machine to another virtual machine without stopping the task and source virtual machine [1]. Cloud computing, is computations distributed over internet that makes computational resources reachable for users in a sharing model [2] [3]. Existing challenges and problems in cloud computing are as follows: resource allocations, load balancing, managing energy consumption, optimized task scheduling and etc.

Load balancing is one of the main important challenges in cloud computing that gets done on the surface of datacenters. Mechanism of dynamically distributing tasks equally on each virtual machine in cloud computing will be performed in such a way to be able to prevent virtual machines from being idle, under small number of tasks or being laborious and to establish a load balance between them [4]. Proper using of resources and making load balance between components helps achieving smaller response time, improvement in being fault-tolerant, scalability, maximum user satisfaction, least heat production, optimized electricity consumption, decreasing gas emission and least operational costs. Load balancing algorithms are classified mainly in two types: 1) Static algorithms 2) Dynamic algorithms. Static scheduling methods are

non-exclusive and assigning tasks to processors gets done before start of the program. Scheduled decision making is based on information about execution time of tasks, resource processing and etc. [5] [6]. In cloud environment, it is required to schedule computational resources in such a way that service providers get most out of their resources and also users manage their functional programs with lowest cost. In fact, big scale of functional programs, heterogeneity and dynamism of resources features over virtual machines and existence of various requests in cloud processing environment leads to complexity in gaining required precision for predictions in this environment. Therefore, in this scope by using time estimation techniques and optimization algorithms we can guarantee improvements in performance and efficiency of cloud processing networks. Scheduling tasks in cloud processing environment is being exploited in order to correctly and precisely schedule users' requests according to available resources. A good scheduling algorithm should provide some solutions at failure-state times to hide these failures from user and also to finalize the task with given conditions as much as possible. Because of importance, complexity and extensiveness of scheduling algorithms, providing methods to optimize parameters like load balance, response time, execution time, tasks migration, and etc. is of great importance. So, in this paper, dragonfly optimization algorithm is used for proper allocation of resources to tasks and also establishing load balance. The rest of this paper is categorized this way: in section 2 previous works are discussed, in section 3 proposed method is presented by describing its architecture. In sections 4 and 5 achieved results and in section 6 the conclusion and future works are presented.

2. Related works

Mr. kansal et al in 2012, proposed an active clustering algorithm based on classification of similar nodes in order to provide a solution for balancing the load in cloud computing. One of important benefits of this method is the great speed of it in execution and processing tasks [7]. Seti et al in 2012 presented a method based on fuzzy logic for iterative turn-based load balancing in virtual machine environments in cloud computing, in order to achieve improvements in response time and processing time. One of the benefits of this method is that load balancing algorithm gets done before the arrival of processing servers [8]. Jasmine et al in 2012, proposed a method with name of "Iterative Turns". This algorithm, assigns requests to virtual machines iteratively [9]. Mr. mondal et al in 2012, presented a method inspired by random hill climbing algorithm for keeping load balance and task scheduling in cloud computing datacenters. Random hill climbing algorithm, expresses a general state of load balancing with optimizing resources as much as possible [10]. Asran et al in 201, proposed a method based on Min-Min algorithm in order to scheduling task with keeping load balance [11]. Mr. dinesh et al in 2012, proposed a method for task scheduling and load balancing in cloud computing datacenters that was based on load balancing algorithm inspired by honey bee colony algorithm. The main purpose of this algorithm is to quickly respond to requests and efficiently manage virtual machines as much as possible [12]. Mr. dizmity et al in 2013, proposed an optimized method based on honey bee algorithm for maintaining load balance in cloud computing and task scheduling [13]. Mr. sabhadra et al in 2014, presented a method named ESCE in order to schedule task with maintaining load balance [2]. Mr. yu et al in 2016, presented a parallel method for scheduling task in datacenters in cloud computing. One of the most important problems that they put into consideration in their research was the problem of quick running of task and the total cost for cloud owners [10]. Generally, by considering the reason of speed and preciseness in task scheduling operations, capabilities of dragonfly algorithm [12] and studying previous works it seems that until now problem of scheduling tasks and keeping load balance in cloud computing data centers, dragonfly algorithm has not been used that in this paper it is used in order to improve task scheduling problem and keep load balance in cloud computing datacenters.

3. Proposed Method

Architecture of proposed method for resource allocation and task scheduling using dragonfly optimization algorithm is illustrated in figure 1. As you can see in figure 1, in order to run dragonfly optimization algorithm first, it is required to simulate and prepare cloud computing infrastructure for running. Therefore, in the first step datacenters start hosts and virtual

machines to be able to execute users' request. In the next step requests will enter cloud computing and each request will be divided into series of tasks to be performed by virtual machines. In the next step number of virtual machines and a number of task will be selected for running. Then it will be inspected that processing of tasks by virtual machine and using dragonfly optimization algorithm has finished or not. If the answer was yes simulation will be ended and if the answer was no the dragonfly optimization algorithm will be executed and also tasks will be scheduled.

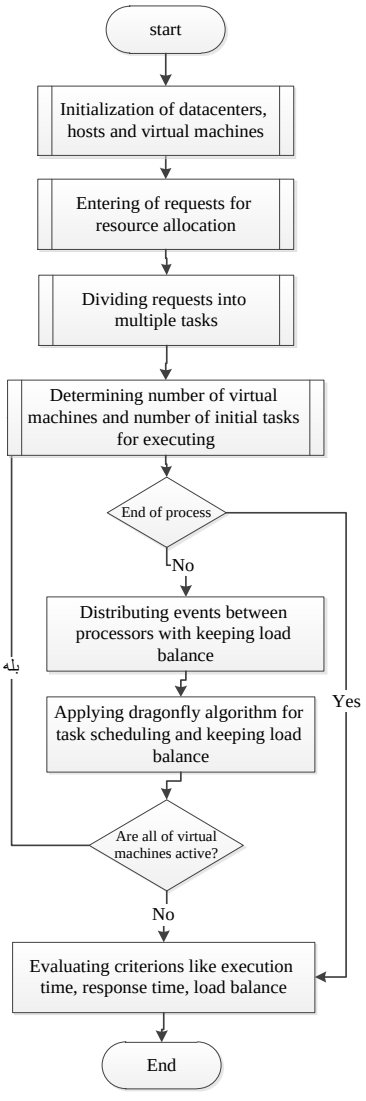

Fig. 1. Architecture of proposed method

According to Reynolds point of view [14], behavior of particles includes three important principles that are as follows:

- Separation: this operator refers to lack of contact between two or multiple individuals or particles in one space.
- Alignment: not exceeding particles or things speed from other neighbors in one-unit space.
- Integrity: this operator also refers to this fact that people should move toward goal of unit or center

The main goal of each particle in the above scenario is surviving. So, all people or particles should hop toward food resources and avoid from affiliations to other cases. By considering these two behaviors of particles, there are 5 important factors in updating particles location that are illustrated in figure 3.

In the following flowchart the way of executing dragonfly algorithm is illustrated.

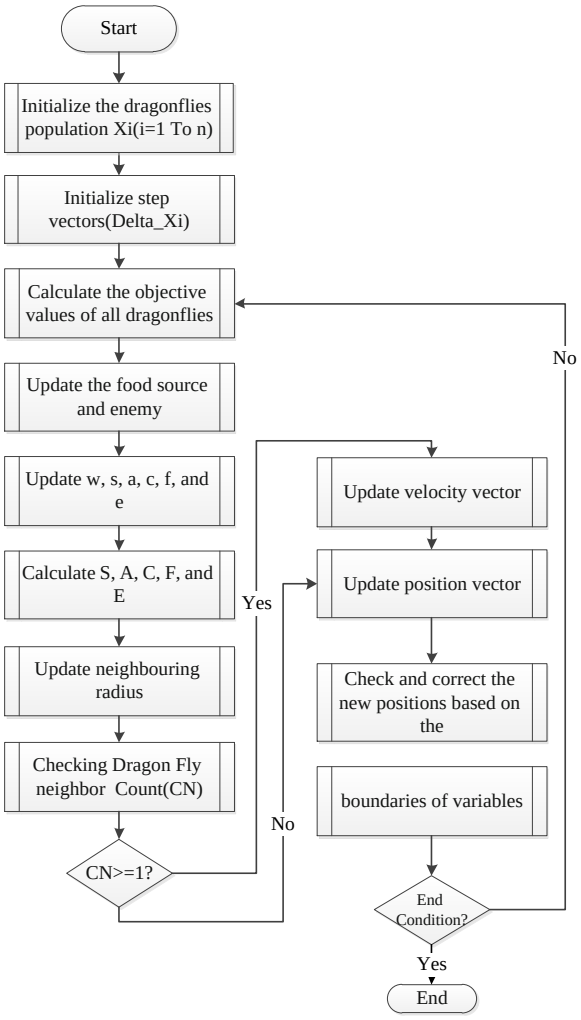

Fig. 2. Dragonfly algorithm flowchart

According to Reynolds point of view [14], behavior of particles includes three important principles that are as follows:

- Separation: this operator refers to lack of contact between two or multiple individuals or particles in one space.
- Alignment: not exceeding particles or things speed from other neighbors in one-unit space.
- Integrity: this operator also refers to this fact that people should move toward goal of unit or center

The main goal of each particle in the above scenario is surviving. So, all people or particles should hop toward food resources and avoid from affiliations to other cases. By considering these two behaviors of particles, there are 5 important factors in updating particles location that are illustrated in figure 3.

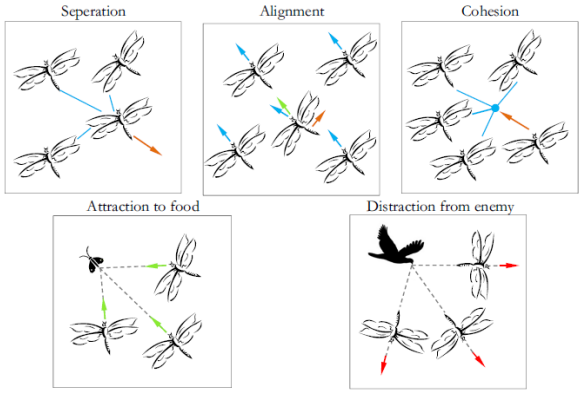

Fig. 3. Primary corrective samples between individuals in dragonflies group [14]

Each of expressed behaviors of particles is formulated using some models. These models are discussed in the following part. Separation operator is formulated in the below model [14]:

$$S_i = \sum_{j=1}^{N} X - X_j \quad (1)$$

In the above formula X denotes location of a particle or a virtual machine. X_j denotes 'j'th neighbor in the environment. N is number of neighbors of a particle or virtual machine. Therefore, generally in the proposed method this formula will be used to find most proper virtual machine for executing tasks with high priority. In fact between existing virtual machines around intended task, location of virtual machine that has less information load is detectable that this detection will be done by dragonflies. Alignment is also modeled using this formula:

$$A_i = \frac{\sum_{j=1}^{N} V_j}{N} \quad (2)$$

In the above formula, V_j denotes speed of a particle. In fact in proposed method, above formula can be used for obtaining average speed of tasks execution by virtual machines and also can be used to calculate average speed of information exchange between virtual machines.

The whole above calculations can be done by dragonflies. Integrity operator is modeled using this formula [1]:

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} - X \quad (3)$$

In the above formula X denotes the position of a particle. X_j also denotes location of jth neighbor in the space. N is the number of neighbors of a particle. The above formula has a very close relation with separation formula and is used for finding average of difference between locations of current virtual machine with other virtual machines that exist in the proposed method.

Particles hopping toward food resource are formulated using the following equation:

$$F_i = X^+ - X \quad (4)$$

In the above formula X denotes location of particle and X^+ also denotes location of food resources (tasks). In equation 4 after finding virtual machine that has low amount of load for executing tasks with higher priority by dragonflies, hopping will be done toward this virtual machine that is handled in form of equation. Distance from margins is denoted by this equation:

$$X_{t+1} = X_i + \Delta X_{t+1} \quad (5)$$

Generally, by using equation 5 virtual machines that are not suitable for selecting and running tasks will be deleted from selection list. Behavior of dragonfly is a combination of these five presented samples. For updating particles in a scope, dragonfly algorithm considers two vectors: 1) Step (X delta) and 2) Location of particle or X. Step vector is like speed vector in PSO optimization algorithm and dragonfly algorithm is also developed based on PSO framework. Step vector (or the same step related to dragonfly algorithm) is defined in one dimension that is surveyed in the following equation:

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i - w\Delta X_t) \quad (6)$$

That S is separation weight, S_i refers to separation of ith particle and 'a' denotes alignment weight. A is alignment of ith particle or individual, C is integrity weight and C_i is integrity of ith integration. F is food criterion; F_i denotes food resource of ith particle. E is enemy factor and E_i is the location of ith enemy. W

denotes the related weight and finally t denotes number of iterations of algorithm. After computation of Step Vector, Location vectors will be calculated using this formula:

$$X_{t+1} = X_i + \Delta X_{t+1} \quad (7)$$

t denotes current location. Generally mentioned vectors are used for managing neighbors and presenting a report about location and way of information exchange of virtual machines by dragonflies. For separation criterions, alignment, integrity, food and enemy can be determined during optimization. In the following figure a model of particles behavior in dragonfly algorithm is illustrated.

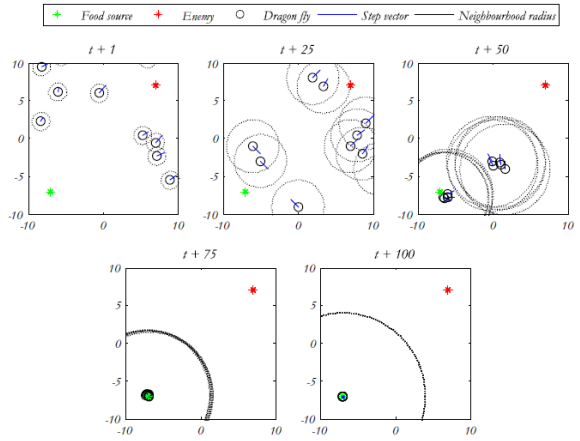

Fig. 4. Model of particles behavior in dragonfly algorithm [14]

Therefore, by using this algorithm the most proper virtual machine for executing required processes can be used at every moment in order to allocate resources in datacenters, maintain load balance and present a task scheduling mechanism.

4. Evaluating results

The method that is proposed in this paper is implemented using Matlab 2013a. The specifications of used system are as follows:

- Operating system: windows 7
- Type of operating system: 32 bit os
- RAM memory: 4 gigabytes – 3.06 usable
- Processor: Intel processor- number of cores 7 ((Core™) i7 CPU) - Q 720@1.60GHz 1.60 GHz

So, all of the results are based on a system with above specifications. All of the obtained results in this paper are compared with articles of [12] [15] [16]. In order to compare results some scenarios and criterions has been inspected. Comparing criterions are as follows:

- Execution time: is the amount of time that it takes for a request to be received and executed by a virtual machine.
- Response time: is the amount of time that it takes for a request to be received by virtual machine and a response to this request will be delivered.
- Number of migrated tasks: this process happens when a virtual machine doesn't have the ability to run a task and that task has to migrate to another virtual machine.

According to above, some scenarios are defined for executing dragonfly algorithm that is as follows:

- 10 tasks and 40 virtual machines
- 20 tasks and 40 virtual machines
- 100 tasks and 40 virtual machines
- 200 tasks and 40 virtual machines
- 500 tasks and 40 virtual machines

According to above scenarios, results will be inspected and also will be compared. In this section, we will compare our proposed method with the ACO (Ant Colony Optimization), PSO (Particle Swarm Optimization), ACO-PSO [15] and HBB-LB (honey bee behavior inspired load balancing) [12] algorithms. This algorithm is one of the most important algorithms recently used to schedule tasks in cloud computing.

4.1. Evaluating execution time of tasks criterion

Execution time of tasks is one of the most important parameters that is highly considered in task scheduling problems and resource allocation in cloud computing. This criterion will be computed using CPU time and proofs level of performance and execution time of task in used algorithm. In table 1 comparison of execution time of dragonfly optimization algorithm for scheduling tasks between virtual machines and resource allocation with maintaining load balance in cloud computing is illustrated with 10 to 500 numbers of task and with 5 to 40 numbers of virtual machines.

Table 1. Comparison of execution time of dragonfly optimization algorithm for scheduling tasks

	VMs=5					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	0.27	0.5	0.31	1.36	1.59	1.4
ACO	0.45	0.75	0.81	1.54	1.84	1.9
ACO-PSO	0.54	0.87	0.98	1.63	1.96	2.07

Table 1 (continued)

	VMs=20					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	0.27	0.5	0.31	0.92	1.15	0.96
ACO	0.45	0.75	0.81	1.1	1.4	1.46
ACO-PSO	0.54	0.87	0.98	1.19	1.52	1.63
	VMs=40					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	0.27	0.5	0.31	0.71	0.94	0.75
ACO	0.45	0.75	0.81	0.89	1.19	1.25
ACO-PSO	0.54	0.87	0.98	0.98	1.31	1.42

As it can be seen from table 1, execution time for dragonfly algorithm in proposed method is very less than other methods and in a better amount of time tasks are executed by virtual machines in cloud computing. In figure 5 a summary of execution time results is presented. The results of the proposed method are illustrated in the figure (5).

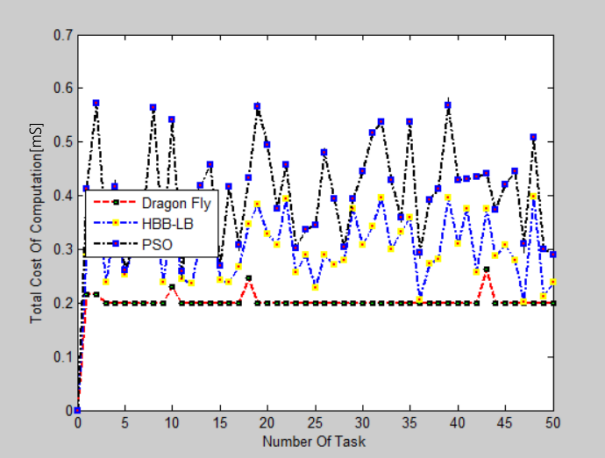

Fig.5. Comparing execution time for dragonfly optimization algorithm in order to schedule tasks

In Figure (5), mS means milliSeconds. As can be seen, the number of tasks in the same conditions is set to [12]. In general, the improvement of the proposed method is about 0.3 milliseconds compared to the PSO method. The improvement in the proposed method is about 0.2 milliseconds compared to the HBB-LB method.

4.2. Evaluating criterion of response time to requests

Response time criterion is amount of time that a request will be sent to a virtual machine in form of task until the time virtual machine responds to that request. As the response time gets smaller the efficiency of algorithm is higher and virtual machines will run tasks with a reasonable load balance. In table 2 comparison of response time for dragonfly optimization algorithm and other algorithms in order to schedule tasks between virtual machines and allocate resources in cloud computing is illustrated with 10 to 500 numbers of tasks and 5 to 40 numbers of virtual machines.

Table 2. Comparison of response time for dragonfly optimization algorithm and other algorithms in order to schedule tasks

	VMs=5					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	0.44	0.54	0.81	1.79	1.89	2.16
ACO	1.02	1.75	2.81	2.37	3.1	4.16
ACO-PSO	1.24	2.87	4.98	2.59	4.22	6.33
	VMs=20					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	0.44	0.54	0.81	1.16	1.26	1.53
ACO	1.02	1.75	2.81	1.74	2.47	3.53
ACO-PSO	1.24	2.87	4.98	1.96	3.59	5.7
	VMs=40					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	0.44	0.54	0.81	1	1.1	1.37
ACO	1.02	1.75	2.81	1.58	2.31	3.37
ACO-PSO	1.24	2.87	4.98	1.8	3.43	5.54

As it can be seen in table 2, sum of response time of virtual machines to tasks for executing them decreases by incrementing number of virtual machines but with unnecessary increasing number of virtual machine system cost will be increased.

4.3. Number of migrated tasks

This criterion is also of great importance in scheduling and maintaining load balance and as a result in optimized resource allocation in cloud computing. As the number of the tasks that migrate during task scheduling and resource allocation gets smaller we can say that load balance is maintained better and smaller number of tasks will be lost. When number of tasks migrating is bigger, response time and execution time will be increased. In table 3 comparison of number of migrated tasks in virtual machines in dragonfly optimization algorithm with other algorithms in scheduling tasks between virtual machines and resource allocation with maintaining load balance in cloud computing is illustrated with 10 to 500 numbers of tasks and 5 to 40 numbers of virtual machines.

Table 3. Comparison of number of migrated tasks in virtual machines in dragonfly optimization algorithm with other algorithms in scheduling tasks

	VMs=5					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	1	1	2	4	5	9
ACO	2	2	3	7	10	19
ACO-PSO	3	3	4	10	13	28
	VMs=20					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	0	1	1	2	4	6
ACO	1	2	2	5	7	15
ACO-PSO	2	3	3	8	10	17
	VMs=40					
	10	**20**	**40**	**100**	**200**	**500**
dragonfly	0	0	1	1	2	5
ACO	1	2	2	4	5	12
ACO-PSO	2	2	3	7	8	15

As it can be observed from presented results, level of task migration from a virtual machine to another virtual machine in proposed method is less than other methods by considering number of different iterations. As the number of virtual machines gets higher, accordingly number of task migration will be decreased but system cost will be increased. In figure 6 reduction level of task migrations in virtual machines for proposed method that leads to proofs of load balance maintenance in cloud computing, is illustrated.

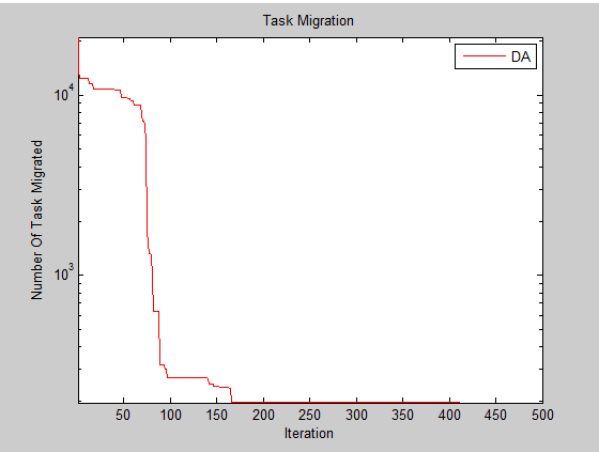

Fig.6. Reduction of task migration level using dragonfly algorithm with maintaining load balance between virtual machines

As it can be observed from figure 5, number of migrated task after maintaining load balance will be zero. In first iterations number a big number of tasks will be migrated and gradually this number gets closer to Zero. In [16], dragonfly algorithm is used to maintain the load balancing between virtual machines in cloud computing. This article evaluates the number of migratory tasks. Accordingly, we give a general comparison of the proposed method in this article and article [16].

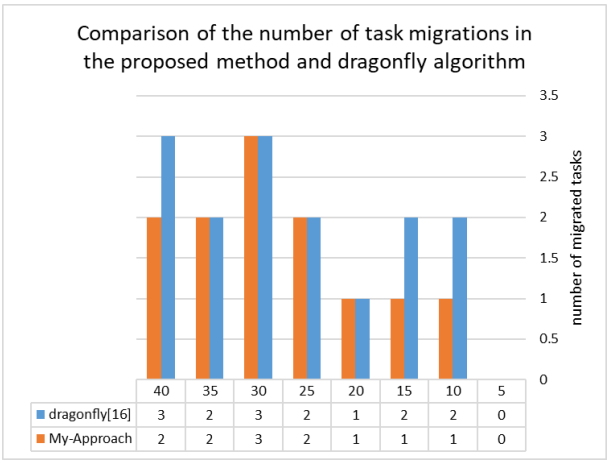

Fig. 7. Comparison of the number of task migrations in the proposed method and dragonfly algorithm

As can be seen from Fig. 9, the total number of migrating tasks in [16] is 15, if the total number of migrations in the proposed method is equal to 12. In general, the proposed method, with a difference of 3 goals in the same conditions, works better than the method described in [16].

5. Conclusion

Generally, by simulating the proposed method in this paper we observed that results of using dragonfly optimization algorithm provides better results because of its speed and precision in scheduling operations for maintaining load balance when allocating resources to virtual machines and scheduling them. In this paper we have inspected many criterions in order to evaluate dragonfly optimization algorithm efficiency that are as follows: execution time, response time, tasks migration and load balance. After the end of simulation, we obtained the following results:

Table 4. Final comparison of dragonfly algorithm with other methods

criteria	**dragon-fly**	**ACO [15]**	**ACO-PSO[15]**	**DA-ACO**	**DA-PSO**
Execution time	0.723	1.033	1.16	41%	**47%**
Response time	1.035	2.298	3.468	31%	**39%**
Tasks migration	2	6	8	25%	**42%**

So, by these obtained results it can be proved that dragonfly algorithm provides more considerable improvements in task scheduling, load balancing and resource allocations when comparing to other methods.

References

1. K. Ren, C. Wang and Q. Wang, *Security Challenges for the Public Cloud*, (IEEE Computer Society, 2012), pp.77-96.
2. A.K. Singh, S. B. Shaw, *A Survey on Scheduling and Load Balancing Techniques in Cloud Computing Environment*, (International Conference on Computer and Communication Technology (ICCCT) ,2014).
3. P. Mell, T. Grance, *The NIST Definition of Cloud Computing*, (Draft NIST, 2011).
4. A. K. Sidhu, S. Kinger, *Analysis of Load Balancing Techniques in Cloud Computing*, (International Journal of Computers & Technology, Vol. 4, No. 2, 2013), pp. 737-741.
5. X. Evers, *A Literature Study on Scheduling in Distributed Systems*, (National Institute voor Kernfysica en Hoge-EnergieFysica P.O. Box 14882, 1009 DB Amsterdam, The Netherlands, 2000).
6. A. A. Rajguru, S.S. Apte, *A Comparative Performance Analysis of Load Balancing Algorithms in Distributed System using Qualitative Parameters*, (International Journal of Recent Technology and Engineering, Vol. 1, No. 3, 2012).
7. N. J. Kansal, I. Chana, *Cloud Load Balancing Techniques: A Step towards Green Computing"*, (IJCSI international journal of Computer Science, Vol. 9, No. 1, 2012).
8. S. Sethi, A. Sahu, S. Kumar Jena, *Efficient load balancing in Cloud Computing using Fuzzy Logic*, (IOSR Journal of Engineering (IOSRJEN), Vol. 2, No. 7, 2012), pp. 65-71.
9. J. James, B. Verma, *Efficient VM Load Balancing Algorithm for a Cloud Computing Environment*, (International Journal on Computer Science and Engineering (IJCSE), Vol. 4, No, 3, 2012), pp.1658-1663.
10. M. Brototi, *Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach*, (sciverse sciencedirect, C3IT- Procedia Technology, 2012), pp. 783-789z
11. Sran.N and Kaur.N, *Comparative Analysis of Existing Load Balancing Techniques in Cloud Computing*, (International Journal of Engineering Science Invention, Vol. 2, No. 1, 2013).
12. D. Babu, P. V. Krishna, *Honey bee behavior inspired load balancing of tasks in cloud computing environments*, (Applied Soft Computing, 2013), pp. 2292-2303.
13. D. Kliazovich, S. T. Arzo, F. Granelli, P. Bouvry and S. U. Khan, *e-STAB: Energy-Efficient Scheduling for Cloud Computing Applications with Traffic Load Balancing*, (IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, 2013), pp. 7-13.
14. S Mirjalili, *Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems*, (Neural Comput & Applic, 2015).
15. Misha Goyal, Mehak Aggarwal, *Optimize Workflow Scheduling Using Hybrid Ant Colony Optimization (ACO) & Particle Swarm Optimization (PSO) Algorithm in Cloud Environment*, (International Journal of Advance research, Ideas and Innovations in Technology, 2017).
16. V. Polepally, K. S. Chatrapati, *Dragonfly optimization and constraint measure-based load balancing in cloud computing*, (Cluster Computing, Vol. 1, No. 2, 2017), pp. 1-13.