Asian Journal of Computer Science and Technology
ISSN: 2249-0701 (P) Vol.8 No.S3, 2019, pp.105-108
© The Research Publication, www.trp.org.in
DOI: https://doi.org/10.51983/ajcst-2019.8.S3.2074

An Efficient Novel Load Balancing Algorithm to Improve the Performance of the System in Cloud Environment

P. Neelima[1] and A. Rama Mohan Reddy[2]
[1]Research Scholar, [2]Professor, [1&2]Department of Computer Science and Engineering,
[1]Jawaharlal Nehru Technological University (JNTUA), Anantapura, Andhra Pradesh, India
[2]Sri Venkateshwara University, Tirupati, Andhra Pradesh, India
E-Mail: neelima.pannem@gmail.com, ramamohansvu@yahoo.com

(Received 22 April 2019; Revised 3 May 2019; Accepted 15 May 2019; Available online 22 May 2019)

***Abstract* - Distribution of workload in a balanced manner is a main challenge in cloud computing system. It distributes workload among multiple nodes, hence resources are properly utilized. This is an optimization problem and a good load balancer should be involved for this strategy to the types of tasks and dynamic environment. To overcome load balancing problem here a Novel Load balancing Algorithm is develop i.e. Dragonfly Algorithm is design and developed, to execute the entire task with shortest completion time and load balanced. Our algorithm will be presented with efficient solution representation, derivation of efficient fitness function (ormulti-objective function) along with the usual Dragonfly operators. The performance of the algorithm will be analyzed based on the different evaluation measures. The algorithms like particle swarm optimization (PSO) and Genetic algorithm (GA) will be taken for the comparative analysis.**

***Keywords:* Cloud Computing, Load Balancing, Evolutionary Algorithms, Architecture**

I. INTRODUCTION

Cloud computing is a computing model used everywhere and provides convenient, on-demand access to a shared pool of computing resources such as networks, servers, storage, applications, etc. These resources can be dynamically assigned and released with minimal management effort [3].

A. Cloud Computing Service Models: Cloud computing provides various types of services to its users.

1. Infrastructure as a Service (IaaS): It consists services that permit its consumers to request storage and computational resources on demand.

2. Platform as a Service (PaaS): It contains high levels of services that provide a platform to develop and manage the software infrastructure. In PaaS developer can built and deploy different types of applications using libraries, languages and tools handled by the cloud service providers.

3. Software as a Service (SaaS): SaaS comprises end users applications that are delivered to consumers' as a network services. So, this eliminates the need to install and run different applications on consumer's computers. An example of SaaS is SalesForce.comand Google mail. Scheduling tasks in cloud process setting is being exploited so as to correctly and exactly schedule users' requests consistent with on the market resources. A decent planning rule should offer some solutions at failure-state times to cover these failures from user and additionally to settle the task with given conditions the maximum amount as potential. As a result of importance, quality and extensiveness of programming algorithms, providing ways to optimize parameters like load balance, response time, execution time, tasks migration, and etc. is of nice importance. So, here dragonfly optimization algorithm is planned for proper allocation of resources to tasks and also establishing load balance. This paper is organized as follows.
Section II- Related Work
Section III- Load Balancing Algorithms
Section IV- Proposed Dragon Fly Algorithm
Section V- Simulation Results and Analysis
Section VI- Conclusion

II. RELATED WORK

Load Balancing is a mechanism which distributes the workload on the resources of a node to respective resources on the other node in a network without eliminating any of the running tasks. So balancing the load between various nodes of the cloud system [2] became a main challenge in cloud computing environment.

A. Load Balancing Measurement Parameter

There are some measurement parameters to evaluate the load balancing techniques which allow us to check whether the given technique is good enough to balance the load or not.

1. Throughput: It is the amount of work to be done in the given amount of time.
2. Response time: It is the amount of time used to start fulfilling the demand of the user after registering the request.
3. Fault tolerance: It is the ability of the load balancing algorithm that allows system to work in some failure condition of the system.
4. Scalability: It is the ability of the algorithm to scale itself according to required conditions.
5. Performance: It is the overall check of the algorithms working by considering accuracy, cost and speed.
6. Resource Utilization: It is used to keep a check on the utilization of various resources.

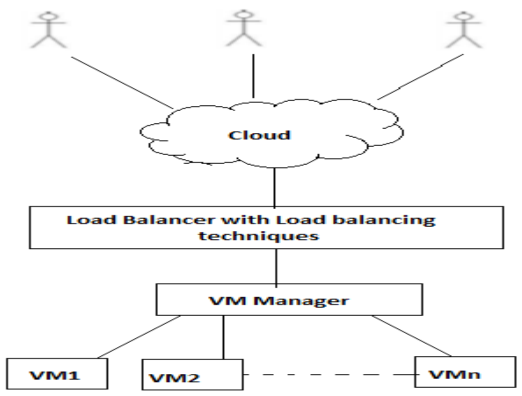

Fig. 1 Load Balancing process

B. Need of Load Balancing

1. Load equalization is used to distributing a bigger process load to smaller process nodes for enhancing the general performance of system [9]. In cloud computing atmosphere load balancing is needed distribute the dynamic native work evenly between all the nodes.
2. Load equalization helps in fair allocation of computing resource to realize a high User satisfaction and correct Resource utilization.
3. Load balancing may be a technique that helped networks and resources by providing a most throughput with minimum latent period. Load equalization is dividing the traffic between all servers, thus knowledge may be sent and received without any delay with load equalization [11].

C. Types of Load Balancing Algorithms

1. Static Load Balancing: This algorithmic program needs previous information concerning system resources. Therefore, the choice of load distribution doesn't rely on the present (present) state of the system. In this setting performance of processors is explained at the beginning of the execution and it doesn't amendment the execution process at run time for creating changes within the system load. This algorithmic program is appropriate for homogenized system environment [6].

2. Dynamic Load Balancing: This rule doesn't need any previous data regarding system resources as a result of the load distribution decision are predicated on the present state of the system. This can be appropriate for heterogeneous system [7]. Dynamic load leveling create changes in load at run time. This rule provides outstanding improvement in performance than static rule.

III.LOAD BALANCINGALGORITHMS

A. Static Algorithms

1. Round Robin Algorithm: It uses the principle of your time slices. Here time is split into slices and every node is given a measure, within which the node must perform the task. If the process isn't completed inside the time quantum, it's to attend for following slot.

2. Min-Min Algorithm: In this algorithmic rule first the minimum completion time of all the nodes is calculated. A task with minimum completion time is chosen and assigned to the corresponding node. This process is continual till all the tasks are assigned to the nodes.
3. Max-Min Algorithm: It a static algorithm, in this first minimum completion time of all the tasks is calculated and a task with maximum completion time is selected and it is assigned to the corresponding node.
4. Opportunistic Load Balancing Algorithm: This formula doesn't calculate the execution time and therefore the current load of the node. It assigns the tasks willy-nilly to the nodes. The task process takes while as a result of it doesn't calculate the execution time of the node.

B. Dynamic Algorithms

1. Ant Colony Optimization Technique: In this rule, when a call for participation is initiated the ant starts its movement in forward direction visiting the nodes one by one and checking whether or not a node is overloaded or under loaded and records the information. If the ant finds a full node it starts backward movement to the previous underneath loaded node to share knowledge [15].
2. Honey Bee Foraging Algorithm: It's a nature inspired decentralized load equalization technique that helps to balance the load across heterogeneous nodes of the cloud. In this formula initial current load of the nodes is calculated then it decides whether or not the node is over loaded [12], under loaded or balanced. A task from the significant loaded node is removed and by considering its priority it's allotted to a gently loaded node.
3. Biased Random Sampling Algorithm: In this rule the network is represented as a virtual graph. The server's area unit represented as nodes and therefore the in-degree represents the free resources on the market to the node. On the idea of in-degree the load balancer assigns the tasks to the node. Once a task is allotted the in-degree is decremented and it is incremented once the job gets dead.
4. Resource Allocation Scheduling Algorithm (RASA): In this algorithm first virtual nodes are created. The expected response time is calculated for all the virtual nodes. According to least loaded node criteria, efficient virtual node is found. If the number of resources are odd then Min-Min strategy is applied, otherwise Max-Min strategy is applied.

IV. PROPOSED DRAGON FLYALGORITHM

Dragonfly formula (DA) is a multi-objective optimization technique galvanized from normal behavior of dragonflies, ideally obsessed with exploration and exploitation [16]. Dragonfly [15] creates sub-warms around numerous areas for navigating, food searching and survival from enemies that is helpful for convergence towards pare to-optimal answers and coverage of optimum solution on the objectives. Architecture of proposed methodology for Load balancing exploitation dragonfly optimization algorithm is illustrated in figure a pair of.

A. Dragon Fly Algorithm

Step 1: Datacenters start hosts and virtual machines to be able to execute users' request.
Step 2: In this step requests will enter Cloud Computing and each request will be divided into series of tasks to be performed by virtual machines.
Step 3: In this step virtual machines and a number of task will be selected for running.
Step 4: Then it will be inspected that processing of tasks by virtual machine and using dragonfly optimization algorithm has finished or not.
Step 5: 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.

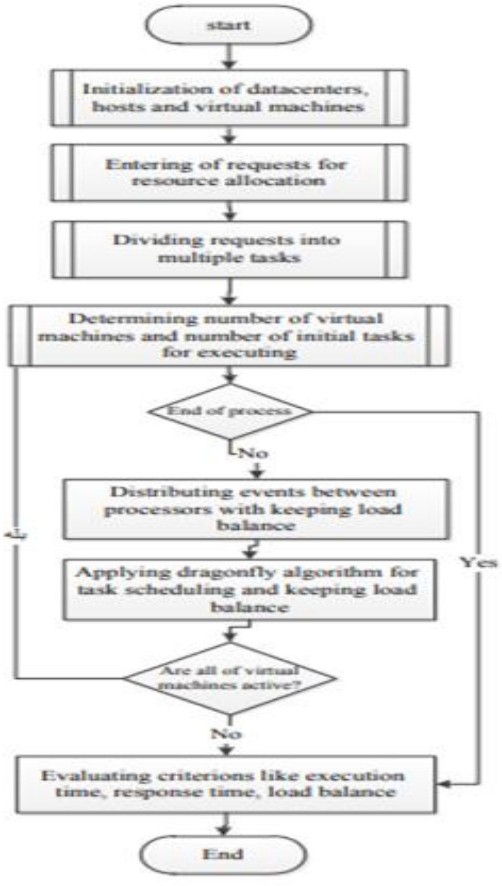
start
Initialization of datacenters, hosts and virtual machines
Entering of requests for resource allocation
Dividing requests into multiple tasks
Determining number of virtual machines and number of initial tasks for executing
End of process
No
Distributing events between processors with keeping load balance
Applying dragonfly algorithm for task scheduling and keeping load balance
Yes
Are all of virtual machines active?
No
Evaluating criterions like execution time, response time, load balance
End

Fig. 2 Architecture of Proposed Method

According to Reynolds point of view, 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: It 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.

Separation operator is formulated in the below model

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

In the above formula X denotes location of a particle or a virtual machine. Xj 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 operator is also modeled using this formula:

$$A_i = \frac{\sum_{j=1}^{N} V_j}{N}$$

In the above formula, Vj 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.
Integrity operator is modeled using this formula

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} - X$$

In the above formula X denotes the position of a particle. Xj 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.

V. SIMULATION RESULTS ANDANALYSIS

The proposed Dragonfly algorithm has been simulated by using the parameters: Resource utilization and load balancing level.

*Resource Utilization***:** Resource utilization means to serve more users during specific operation time. The more the utilization of resources means more the balancing of load i.e. it must be large. Here, the numbers of virtual machine are 5 for all the experiments and numbers of tasks have been varied like 50, 100 and 200.

TABLE I RESOURCE UTILIZATION VALUES

Number of Tasks	Dragonfly	GA
50	44	93.96
100	46.79	96.21
200	69.03	99.02

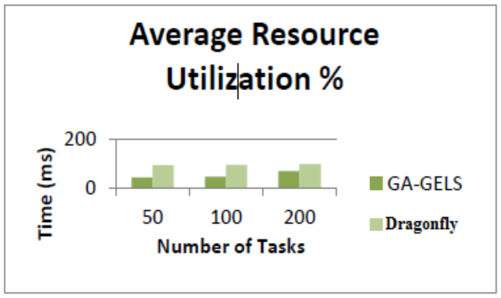

Fig. 3 Resource Utilization Analysis

Load Balancing Level: It includes the balancing level of load on the VMs or how the load is distributed evenly on the machines. The load is distributed efficiently, if this parameter is high.

TABLE II LOAD BALANCING VALUES

Number of Tasks	Dragonfly	GA
50	33.84	95.19
100	42.08	96.83
200	74.35	99.01

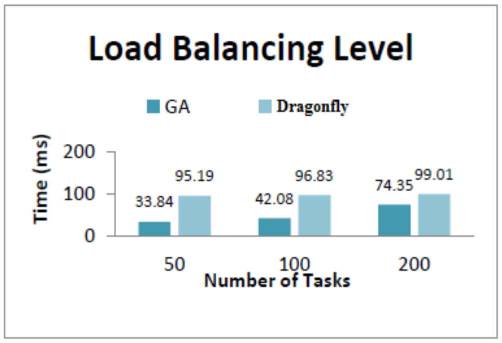

Fig.4 Load Balancing Analysis

Generally, by simulating the projected methodology the ascertained shows that result of exploitation dragonfly improvement rule provides higher results as a result of its speed and precision in programming operations for maintaining load balance once allocating resources to virtual machines and programming them. Investigations has dispensed with several criterions so as to judge dragonfly improvement algorithm potency that are as follows: execution time, response time, and load balance. After the end of simulation, we obtained the following results:

TABLE III COMPARISON OF DRAGONFLY WITH OTHER OPTIMIZATION ALGORITHMS

Criteria	Dragonfly	Particle swarm Optimization	Genetic Algorithm
Execution Time	0.79	1.22	1.45
Response Time	1.09	2.45	2.01

So, from these obtained results it is evidenced that dragonfly formula provides additional tidy enhancements in task programming, load equalization and resource allocations once comparison to different strategies.

VI. CONCLUSION

Load equalization is one in every of the foremost necessary issue of cloud computing. It is a mechanism that distributes work equally across all the nodes within the whole cloud. Through efficient load equalization, we are able to win a high user satisfaction and resource utilization. Dragonfly algorithmic program is style and developed, to execute the complete task with shortest completion time and load balanced. Our algorithmic program are given with economical resolution illustration, derivation of economical fitness operate (or multi-objective function) together with the standard dragonfly operators. The performance of the algorithmic program are analyzed supported the various analysis measures. The algorithmic programs like particle swarm improvement (PSO) and Genetic algorithm (GA) are taken for the comparative analysis.

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, and T. Grance, The NIST Definition of Cloud Computing, (Draft NIST, 2011).

[4] A. K. Sidhu, and S. Kinger, “Analysis of Load Balancing Techniques in Cloud Computing”, *(International Journal of Computers& Technology*, Vol. 4, No. 2, pp. 737-741,2013

[5] X. Evers, A Literature Study on Scheduling in Distributed Systems, (National Institute voorKernfysicaenHoge-EnergieFysica P.O. Box 14882, 1009 DB Amsterdam, The Netherlands, 2000).

[6] A. A. Rajguru, and 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, and I. Chana, Cloud Load Balancing Techniques: A Step towards Green Computing”, *IJCSI International journal of Computer Science*, Vol. 9, 2012.

[8] S. Sethi, A. Sahu, and S. Kumar Jena, Efficient load balancing in Cloud Computing using Fuzzy Logic, *IOSR Journal of Engineering (IOSRJEN*), Vol. 2, No.7 pp. 65-71, 2012.

[9] J. James, and B. Verma, “Efficient VM Load Balancing Algorithm for a Cloud Computing Environment”, *International Journal on Computer Science and Engineering (IJCSE)*,Vol. 4, No, 3, pp.1658-1663, 2012

[10] M. Brototi, “Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach”, *sciver science direct, C3IT- Procedia Technology*, pp. 783-789z, 2012

[11] N. Sranand N. Kaur, “Comparative Analysis of Existing Load Balancing Techniques in Cloud Computing”, *International Journal of Engineering Science Invention*, Vol. 2, No. 1, 2013.

[12] D. Babu, and P. V. Krishna, “Honey bee behavior inspired load balancing of tasks in cloud computing environments”,(*Applied Soft Computing, 2013*), pp. 292-2303.

[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*, pp. 7-13,2013.

[14] S. Mirjalili, Dragonfly algorithm, “A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems”, *(Neural Computer & Applications*, 2015.

[15] Misha Goyal, and MehakA ggarwal, “Optimize Workflow Scheduling UsingHybrid 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, and K. S. Chatrapati, “Dragonfly optimization and constraint measure-based load balancing in cloud computing”, *Cluster Computing*, Vol. 1, No. 2, pp. 1-13, 2017