Load Balancing in Cloud Complex Systems using Adaptive Fuzzy Neural Systems

Mohammad Taghi Sadeghi
Bachelor of Computer, Faculty of Engineering
Birjand University
Jajarm, North Khorasan, Iran

***Abstract*—Load balancing, reliability, and traffic are among the service-oriented issues in software engineering, and cloud computing is no exception to this rule and has put many challenges ahead of experts in this field. Considering the importance of the load balancing process in cloud computing, the purpose of this paper is to provide an appropriate solution for load balancing load in complex cloud systems using an adaptive fuzzy neural system. This system consists of four layers, and a particular operation is performed on each layer. The results of the experiments show that the system has better performance in the criteria mentioned above (balancing, traffic and reliability).**

Keywords*—Load balancing; cloud computing; adaptive fuzzy neural system

I. Introduction

Cloud computing has caused many changes in the world of information and communication technology because of its efficiency. It is gaining more and more popularity due to its easy access, low costs, and comfortable usage. Small companies can conduct business in the field of technology and information processing, and bigger companies can significantly cut their expenses in maintenance, manpower, and processing equipment by using cloud computing [1], [2]. The majority of the physical servers in cloud computing work with the virtualization technique. Each virtual machine needs a specific amount of materials such as central processing unit, memory, bandwidth, sending and receiving data, etc. to maintain the application Function Segmentation and security. Furthermore, virtualization technology enables a number of virtual servers to work on the same physical machine used to improve the efficiency of Physical server resources and reduce energy consumption. Therefore, virtualization can help managers achieve an efficient solution for the flexible management of resources by using virtualization Technique, we can enable more than one virtual machine to work on one physical machine, and each of them will provide their required resources by cutting a section of the machine Physical resources. Data centers of virtual machines in cloud computing have broadened to such an extent that the quality of positioning the virtual machines has become an important issue for producers of cloud computing in order to prevent interference in load balance between virtual machines and losses due to non-compliance with the quality of the virtual machine based on the Service Level Agreements (SLAs) in cloud computing [5]-[8].

II. Literature Review

In [9], the authors have claimed that I/O virtualization is a big challenge, and there is no ideal solution. When virtual machines want to access these non-sliceable resources, an interference in their performance occurs, and the SLA is violated. Hence, one of the main problems of cloud computing is the interference between the virtual machines and their load imbalance of the SLA, and the effective placement of virtual machines greatly reduces or increases the profitability of the cloud computing of virtual machines. In [10], researchers have only considered the strategy of placing virtual machines but have not considered the quality requirements of user applications in load balancing of virtual machines [11]. They have provided a framework for load balancing strategies in the cloud computing environment and have proposed a method to evaluate the resource allocation strategies in the cloud computing environment, seeking to focus on optimizing the awareness and compatibility of network load balancing strategies. The framework for network load balancing in cloud computing is based on active criteria. Network topology, taking into account traffics, and the optimal change of the criteria corresponding to the user's dynamic needs, which plays a major role in determining the Internet architectures and protocols and shaping load balancing management strategies in cloud computing, are the most important results of the research [12].

It is possible to provide a linear scheduling strategy for load balancing in the cloud environment. The separate scheduling of resources and tasks includes the waiting time and the response time. In this research, a linear scheduling algorithm for scheduling tasks and resources called LSTR is designed which schedules tasks and resources, respectively. Here is a combination of Nimbus and Cumulus services to create a server-provider. IaaS cloud environment, KVM/Xen virtualization, and LSTR scheduler have been used to balance load and maximize operational capacity and resource utilization [13]. In [14], dynamic load balancing is done using virtual machines for the cloud computing environment. This research adopts a system which uses virtualization technique to balance the dynamic load of the data center based on applied demands and service resource optimization. In this research, this concept is introduced as a rough criterion for exploiting multidimensional resources in the service. In addition, this research effectively develops a series of innovations to prevent overload in the system and use it in

energy storage. The experimental results show that the proposed algorithm is desirable.

In the distributed type, dynamic load balancing algorithms are implemented by all the nodes in the system, among which the task of load balancing is divided. The interaction between nodes to achieve load balancing can be in two forms of cooperation and without cooperation. In the first form, the nodes work along with each other to achieve a common goal (for example, to improve the overall response time). In the second form, each node works independently towards a local goal, such as improving the response time of a local work. Dynamic load balancing algorithms with distributed feature generate more messages than the non-distributed type. An advantage of this method is that if one or more nodes fail within this system, this does not cause the entire load balancing process to stop. To create resources in cloud computing, an efficient model provides two interactive performance evolutionary classes to determine the minimum number of servers required for SLAs. For both classes, the probability of a response time smaller than x is considered to be y. Two server allocation strategies are used; the first one is the shared allocation and the other is exclusive allocation. The FCFS scheduler determined an allocation strategy for distributing response time to develop an innovative algorithm that required the least number of servers. This algorithm was used in operational conditions and yielded favorable results [15].

In [6], the researcher presented a game theory approach to load balancing for cloud computing services. The purpose of this research is to solve the problem of the service quality by limited load balance. Service seekers require that their complex parallel computing problem be provided by requesting the resources they need. In this research, the game theory has been used to solve the load balancing problem and a suitable two-step solution has been proposed. First, each customer independently solves his/her problem without considering the load balancing multiplex. An appropriate binary programming method is proposed for the independent optimal solution. Second, an evolutionary mechanism is designed that changes the multiplex strategy of the initial solutions of different customers. The overall result is that an appropriate solution can always be found [16]. Moreover, in another study, the dynamic load balancing used in cloud computing was studied by using multi-criteria distributed analysis. In this research, a two-step process method is proposed for managing dynamic autonomous resources in cloud computing in two stages. First, a distributed architecture divides resource management into independent tasks, each of which is run by agent nodes that are physical machines of firm connection at the data center. Second, automated agent nodes are configured through a number of decision-making criteria using the fuzzy configuration method. Simulation results show that the proposed method is flexible [17].

In different studies, load balancing was done to provide services in cloud computing environments. These load balancing algorithms are provided for Software-as-a-Service (SaaS) to minimize the cost of infrastructure. The proposed algorithms designed are a safe way enabling SaaS providers to manage client changes, map client requests to infrastructure layer parameters, and set up virtual machines [18]. This research analyzes and validates the algorithms for minimizing the costs of SaaS providers in the cloud computing environment. The optimal allocation algorithms are investigated in cloud computing clusters and a possible model for tasks (requests such as CPU, memory, and storage) in cloud computing is presented. The proposed model can split the source allocation problem into two balancing and scheduling problems and consider the join-the-shortest-queue and power-of-two-choice algorithms with the maxweight scheduling algorithm. This research shows that algorithms optimize operational power and limit the optimal queue length in heavy traffic [19]. In [20] a tool is presented for modeling and simulating the real-time assignment of virtual machines in the cloud data center. The innovative method was applied to schedule resources dynamically in a cloud data center that has runtime constraints on RMs and PMs. This method focuses on timing simulation in the Infrastructure-as-a-Service (IaaS) layer. Simulations indicate that the multidimensional information source designed and implemented in real time shows that the results have improved compared to previous approaches. In [21], load balancing and automatic load balancing for cloud services are surveyed with fuzzy logic and ant colony, and a method is developed to support cloud agents for an optimal configuration in using dependencies on cloud-based applications that require high security. This method uses the model-driven principles of UML and the Bayesian networks to store, analyze, and optimize the configuration in the cloud space.

Therefore, the load balancing of the whole system is performed by the central nodes of each cluster. Centralized dynamic load balancing receives fewer messages. Hence, the total number of interactions within the system is reduced compared to that of the distributed type. However, centralized algorithms can cause a limiting operation (bottlenecks) on the central node, and the load balancing process fails when the central node collapses. Therefore, this algorithm is more suitable for networks with a smaller size.

III. Proposed Approach

For the virtual machine load balancing problem, the virtual machines are embedded in the physical machines in such a way that the cloud provider's profit is maximized. Each physical machine limits sources for hosting a number of virtual machines. Each virtual machine has its own resource requirement and rental rates. In the virtual machine load balancing problem, each physical and virtual machine can be considered as a backpack and an item in the multi-backpack problem, respectively. The physical machine limits sources to host a number of virtual machines [9]. A virtual machine has its own demand for resources and rental price. The purpose of the virtual machine load balancing problem is to balance the load of virtual machines in the physical machine as much as possible, while not exceeding the resources of each physical machine. In addition, the cloud provider can maximize profits by load balancing of the virtual machines in the physical machines. With the above mappings, the problem of the virtual machine load balancing can be turned into a multi-backpack problem.

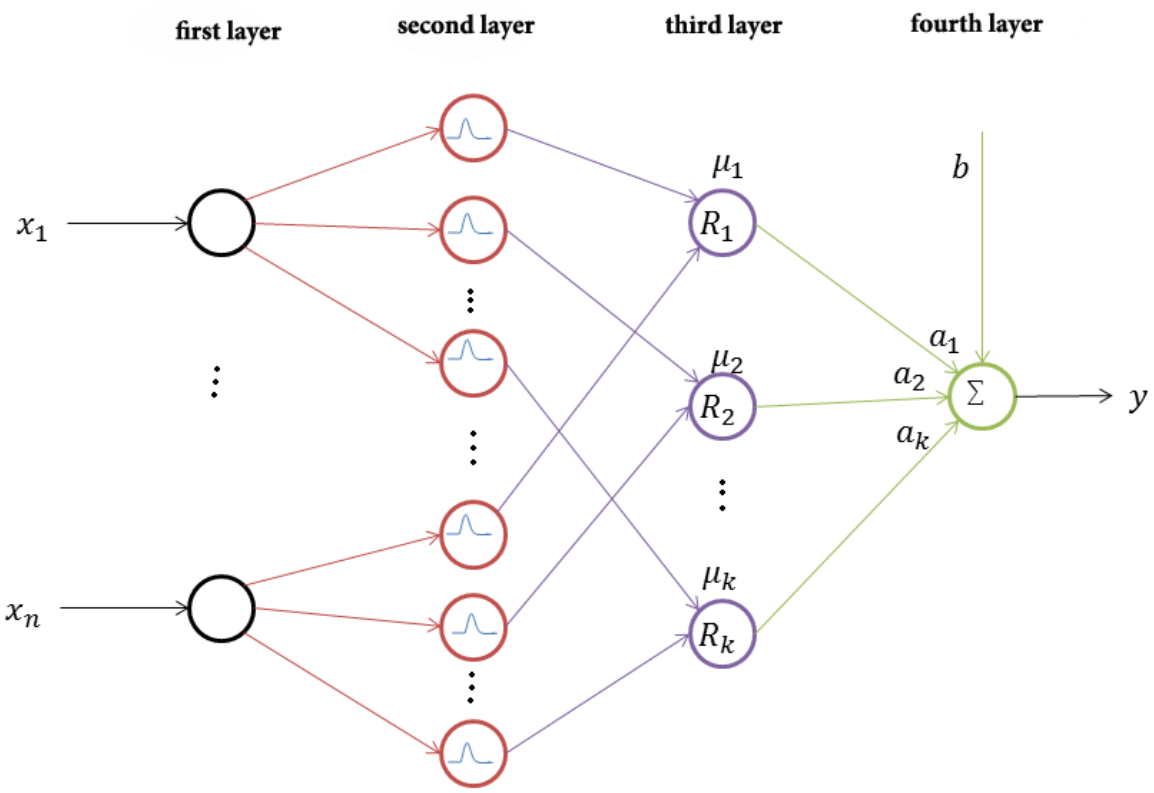

Fig. 1. Proposed system.

Without considering the interference of the virtual machine in the virtual machine load balancing, the requirements of the virtual machine's quality of running applications in the virtual machine may be violated. Different from the problem of the usual placement of the virtual machine, the load balancing problem aware of the quality of the virtual machine considers the following three factors in the virtual machine load balancing: 1) demands for resource from a virtual machine, 2) requirements for the virtual machine's quality of applications, and 3) interference in virtual machines. To integrate these three factors, we consider the problem of virtual machine load balancing aware of the quality of service to the virtual machine load balancing based on the prediction of the cloud provider's profit for these factors. Then, the load balancing problem aware of the quality of service can be formulated as an integer linear programming model to obtain an optimal solution. However, solving the integer linear programming requires noticeable computing time. In this section, a load balancing algorithm is proposed in a complex cloud system using an adaptive fuzzy neural system.

Fig. 1 shows the structure of the proposed fuzzy system with four layers. Each layer has its own operations, each of which is described below.

A. Definition of the Rules

In this fuzzy network, each rule in the proposed method is defined as follows:

The K[th] rule: If x_1 is equal to A_{k1} and x_n is equal to A_{kn}, then $y' = a_k$

Where a_k refers to a single fuzzy rule and A_{kn} refers to a set of fuzzy rules. Here are four network layers.

B. The First Layer (Input Layer)

In the first layer, the values for each node (input variable) are transmitted directly to the next layer. Therefore, no calculations are required. In other words, each node is associated with an input variable, and as a result of this layer, each input is transmitted exactly to the next layer. The trained dataset tag (S) will be described in relation (1):

$$= \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_N, y_N)\} \quad (1)$$

C. The Second Layer (Calculation Of Membership Function)

In the second layer, each node is calculated corresponding to a fuzzy set and membership value. In this layer, the fuzzy set A_{kj} is obtained by the Gaussian membership function and from equation (2):

$$M_{kj}(x_j) = \exp\left(-\frac{(x_j - m_{kj})^2}{\sigma_k^2}\right) \quad (2)$$

In this equation, m_{kj} shows the center of the fuzzy set and σ_k^2 shows the width of the fuzzy set. This Gaussian membership function is widely used in neural fuzzy systems. In other words, in the second layer, each node corresponds to

a fuzzy rule and the value of its membership function is calculated, and the Gaussian function is used for this network.

D. The Third Layer (Fuzzy Rules)

In the third layer, each node represents a fuzzy logic rule, and the *AND* operation of their set is done by multiplying the output of the membership functions.

$$\mu_k(\vec{x}) = \prod_{j=1}^{n} M_{kj}(x_j) = \exp\{-\sum_{j=1}^{n}[\frac{(x_j - m_{kj})^2}{\sigma_k^2}]\} = \exp\{-\frac{\|\vec{x}-\vec{m}_k\|^2}{\sigma_k^2}\} \quad (3)$$

$\mu_k(\vec{x})$ is used as a criterion for deciding whether to produce a new fuzzy rule. In this equation, $\vec{x}$ is obtained from equation (4):

$$\vec{x} = [x_1, \dots, x_n]^T, \vec{m}_k = [m_{k1}, \dots, x_{kn}]^T \quad (4)$$

E. Fourth Layer (Output Layer)

In the fourth layer, each node corresponds to an input variable, and the defuzzification operation is performed in this section. In this layer, the total weight of the outputs of the previous layer with a bias value is considered for calculating the output of the layer. The output of the fourth layer is obtained using equation (5):

$$y' = \sum_{k=1}^{r} a_k . exp\left\{-\frac{\|\vec{x}-\vec{m}_k\|^2}{\sigma_k^2}\right\} + b = \sum_{k=1}^{r} a_k . \mu_k(\vec{x}) + b \quad (5)$$

In the layer *b*, the bias is related to the values computed by the system in this layer.

The purpose of training the network structure is to partition the input space generated under the influence of a number of fuzzy rules. As mentioned above, $\mu_k(\vec{x})$ is used as a criterion for deciding whether to produce a new fuzzy rule. For the first input data $\vec{x}(0)$, a new fuzzy rule is generated to which the membership function with the Gaussian center and Gaussian width are allocated using equation (6):

$$m_{1j} = x_j(0), j = 1, \dots, n, \sigma_1 = \sigma_{init} \quad (6)$$

In this equation, σ_{init} is a predetermined value that defines the initial Gaussian width in the first cluster. For successful input data $\vec{x}(t)$, the value of K is obtained using equation (7):

$$K = arg \max_{1 \le k \le r(t)} \mu_k(\vec{x}(t)) \quad (7)$$

Where r(t) is the number of rules available at time t, and as long as $\mu_k < \mu_{th}$, the new rules are created.

F. Parameters of the Proposed Method

In the experiments conducted in this paper, it is assumed that the cloud computing system has 250 physical machines randomly distributed in 10 clusters. Each of the clusters is located in a local area on a 100 × 100 square plate. Of the 10 clusters, one as the central cluster organizes all of the 10 clusters of the physical machine and is determined as a tree topology. In each cluster, the physical machines are randomly located in a local area in the cluster. In addition, there is a switch in each cluster of the physical machine to provide intra-cluster and inter-cluster communications between physical machines.

Based on the architecture of the cloud computing system, simulation experiments have been performed on the following parameters:

1) In each physical machine, there are a number of virtual machines available. The number of virtual machines available is randomly assigned between 0 and 10.

2) The amount of resources available (the available central processing in GHz, the available memory space in gigabytes, the available storage space in gigabytes) is triple. The source interval [(12,129,200), (96, 3000, 9600)] is randomly used to decide the resources available to each physical machine.

3) The bandwidth is assumed to be in the range of 10 gigabytes/second to 40 gigabytes/second.

4) The number of virtual machines in each simulation run is 200 to 1000.

5) To simulate the profit metric, the cost of a virtual machine (revenue) and the payment of a fine resulting from a service quality violation in a virtual machine are allocated.

IV. EXPERIMENTS

Some parameters are considered for load balancing oudtechniques in the cl environment. Some algorithms create and improve these parameters the optimal load balance parameter over other algorithms. Thes of traffic, ,reliability efficiency, etc. are used to calculate the number of tasks executed. These factors need to be optimized to improve These .system performance factorneed to be improved at a s The .reasonable cost factor of resource efficiency and usefulness of resources used to control the is usefulness of for This factor should be optimized .resources useful load balance. Scalability demonstrates the ability of an algorithm to perform load balancing on a system with any limited number of nodes. This parameter should be improved. time Response is the eamount of tim spent by an algorithm to answer specific balancing load in distributed systems.This parameter should be minimized. Failover Tolerance Parameters show the ability of an algorithm to perform load balancing when a node loses its connection . This section indicates results related to criteria such as the overall the traffic of the adaptive fuzzy neural system, computational benefits, points sensitive to delays, and running time. Each of these criteria a discussed in is graph.

A. Overall Traffic

Fig. 2 shows the outcomes of overall traffic in three different methods. As the diagram shows, the adaptive fuzzy neural system has a better effect on the overall traffic, the ant colony algorithm is somehow similar to clustering. Also, the overall traffic of the topology of the tree is less than that of the FAT tree.

Moreover, Fig. 3 shows the comparison of efficiency using the proposed method, the clustering method, and the ant colony base algorithm in different topologies.

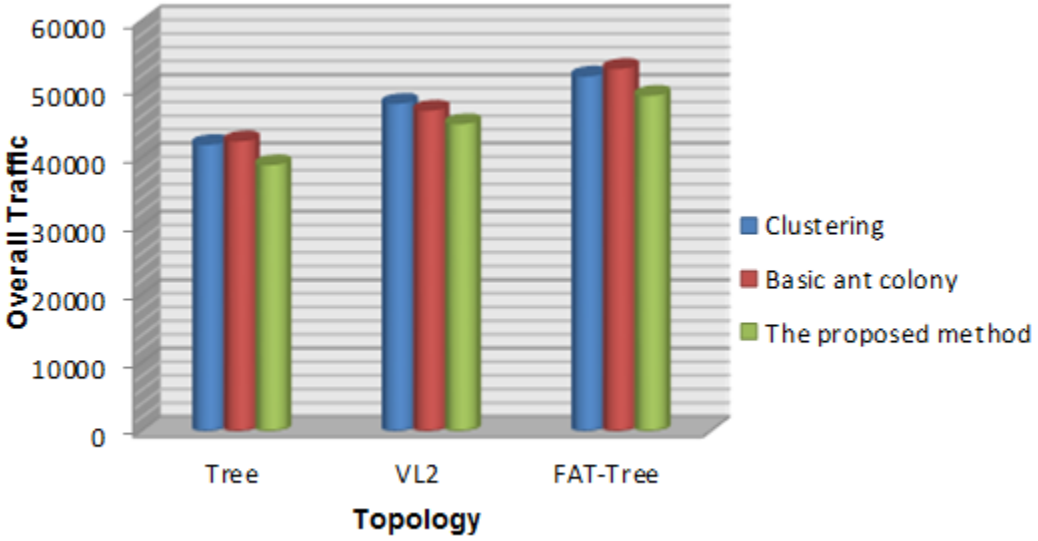

Fig. 2. Change in overall traffic generated by the proposed method.

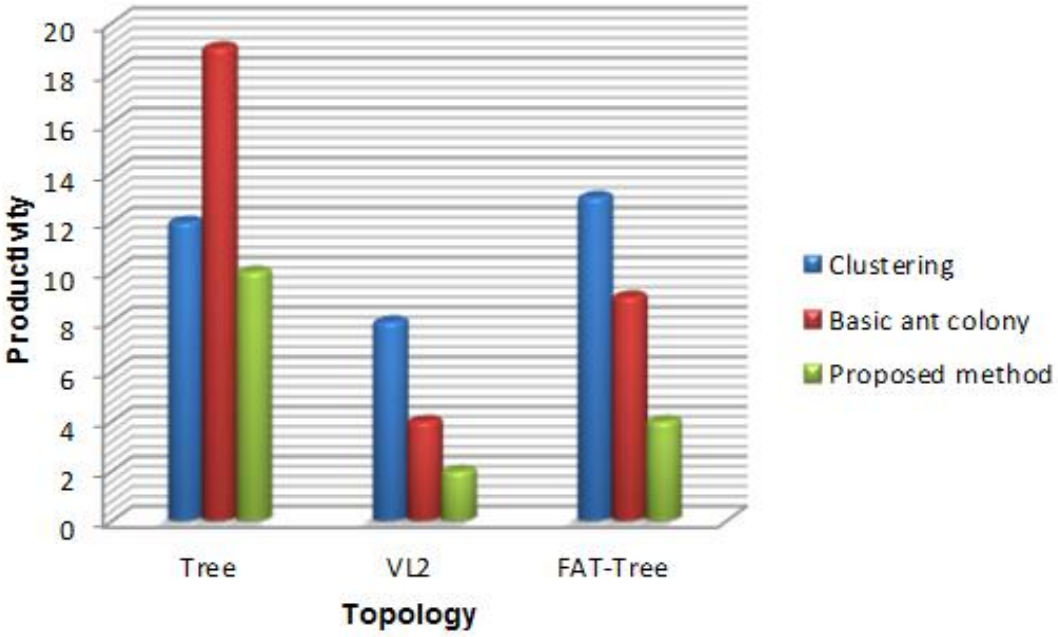

Fig. 3. Minimum productivity.

The proposed algorithm not only leads to the optimization of overall network traffic, but it is also for network productivity. The adaptive fuzzy neural system refers to a situation in which the method of overall traffic and productivity is optimized. By using the data in this paper, traffic has been used among the same virtual machines for different network topologies. As seen in Fig. 2 and 3, although the proposed method slowly increases overall traffic, it further reduces overall traffic and improves network performance. Also, when the FAT tree and VL2 use multiple paths, their effects are more significant.

B. *Patterns of Critical Points*

When the traffic matrix of the virtual machine is created, if the system wants to overlap some traffic, the network path generates Critical points, leading to aggregation. Fig. 4 shows the change in the number of Critical points created in the three methods, including the adaptive fuzzy neural system, clustering, and ant colony in different topologies. Compared to the two methods, in the proposed method, the number of paths of Critical points has decreased significantly, and the number of paths of the Critical points in the FAT tree has been to 8%. Although the proposed method cannot completely avoid

network aggregation, it greatly improves network performance.

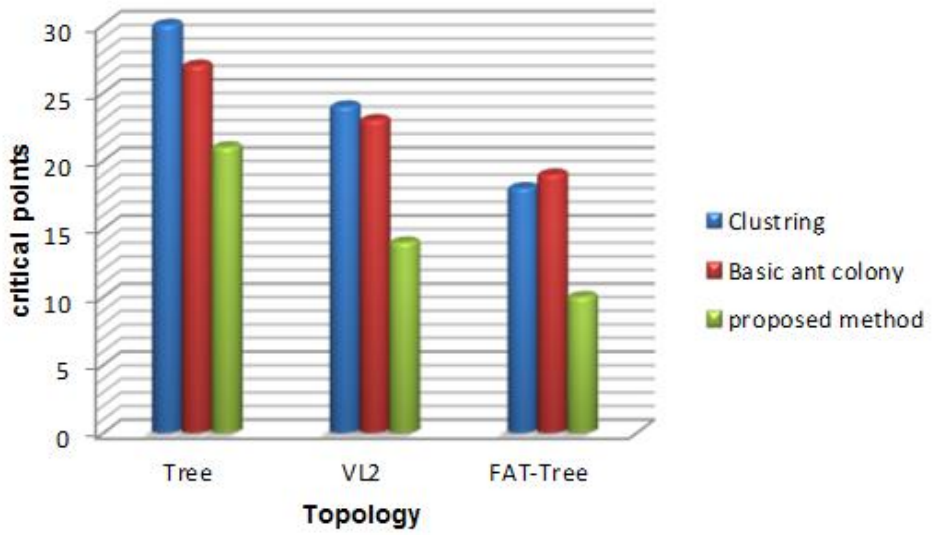

Fig. 4. Change in the number of critical points created.

C. Changes in Profit

The following simulation results are shown for 50 simulation runs on average. The benefits obtained are shown by creating different numbers of the virtual machines from 200 to 1000. As seen in the figure, the profit from all algorithms increases along with the number of virtual machines. However, random fit algorithms, the first fit, do not consider the minimum fit of virtual machine interference. By creating more virtual machines, the virtual machine interference increases. The virtual machine interference influences the quality requirements of the virtual machine of the applications. If the effect of virtual machine interference is not controlled, violating the quality of the virtual machine will result in fines being imposed to reduce the profitability of the cloud provider. In an adaptive fuzzy neural system, the virtual machine interference has dropped as much as possible. There is a linear process in profit growth by increasing the number of virtual machines created. Compared to the two algorithms, the proposed algorithm can increase their profits by approximately 12%, 9%, and 21%.

D. Reliability and Load Balance

Reliability should be calculated by a timetable that calculates the time associated with different activities such as sending and receiving packet. The reliability of the initial phase (when the node enters the network) is ignored because the lifespan of the network is very high and is considered about tens or even hundreds of days, and can be ignored due to the very low initial phase (Fig. 5 & 6).

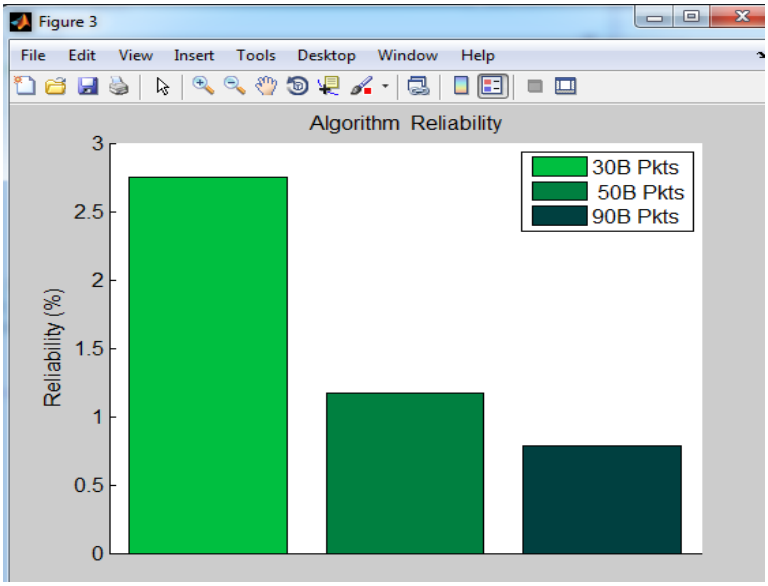

Fig. 5. Comparison of the reliability based on the size of packages.

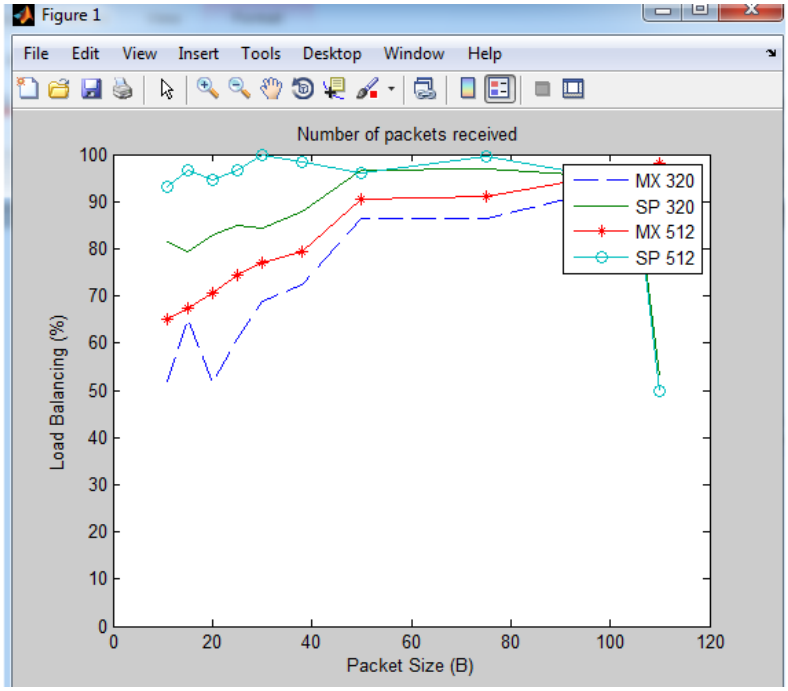

Fig. 6. Comparison of load balances with packages received.

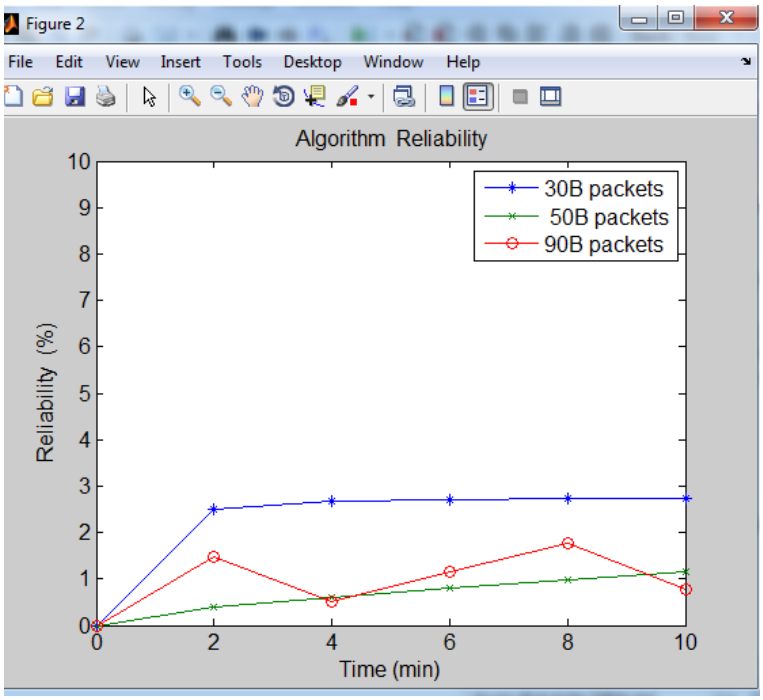

Fig. 7. Comparison of reliability towards time.

In Fig. 7, the most important comparison has been made in terms of reliability which is calculated based on the number of packages received. As can be seen, the reliability of the proposed method for the number of packages received is higher than that of other methods.

E. Execution Time

The execution time of the proposed method is measured in seconds. Due to the parallel characteristics of the adaptive fuzzy neural system, the algorithm can be applied to memory machines and parallel calculations can be used to reduce time and improve performance. The execution time in seconds is acceptable for analysis in the data center, especially for solving the NP-Hard problem. Although the time performance of the adaptive fuzzy neural system is slightly weak, the algorithm shows more accurate results. Experiments confirm that the use of the adaptive fuzzy neural system is suitable for analyzing problems.

V. CONCLUSION

In general, load balancing can be effective by dividing the flow of performance between all nodes. This paper investigated the load balancing in complex cloud systems using the adaptive fuzzy neural system. As can be seen in the figures, tree topology has a better overall optimization and also contains the largest amount. It can be concluded that the tree topology is more likely to generate aggregation among similar virtual machines. Although the FAT tree has a large overall traffic, it can smooth traffic due to its multi-path connections. There is a great difference between the minimum path efficiency and the optimal amount for VL2. Therefore, the distribution of traffic is non-uniform. According to the results of the graphs, the efficiency of the proposed method is greater than that of the clustering and base ant colony method. The delays in the Critical points of the proposed method are approximately 7% better than that of the other two methods.

REFERENCES

[1] Einollah Jafarnejad Ghomia, Amir Masoud Rahmania, Nooruldeen Nasih Qaderb, Load-balancing algorithms in cloud computing: A survey, Journal of Network and Computer Applications, Volume 88, 15 June 2017, Pages 50-71.

[2] Tang Linlin, Li Zuohua, Ren Pingfei, Pan Jengshyang, Lu Zheming, Jingyong Su, Meng Zhenyu, Online- and offline-based load balance algorithm in cloud computing, Knowledge-Based Systems, Volume 138, 15 December 2017, Pages 91-104.

[3] AvnishcThakur, Major Singh Goraya, A taxonomic survey on load balancing in cloud, journal of Network and Computer Applications, Volume 98, 15 November 2017, Pages 43-57

[4] Mohit Kumar, S.C. Sharma, Dynamic load balancing algorithm for balancing the workload among virtual machine in cloud computing, Computer Science, Volume 115, 2017, Pages 322-329.

[5] Ren et al., "The load balancing algorithm in cloud computing environment," in International Conference on Computer Science and Network Technique, Changchun, China, 2012.

[6] J. Bhatia et al., "HTV Dynamic Load Balancing Algorithm for Virtual Machine Instances in Cloud," in International Symposium on Cloud and Services Computing, Mangalore, KA, 2012.

[7] Aarti Singha, Dimple Junejab, Manisha Malhotra, Autonomous Agent-Based Load Balancing Algorithm in Cloud Computing, Computer Science, Volume 45, 2015, Pages 832-841.

[8] B. Shaoo et al., "Analyzing the Impact of Heterogeneity with Greedy Resource Allocation Algorithms for Dynamic Load Balancing in Heterogeneous Distributed Computing System," Int J Comput Appl, Jan. 2013.

[9] P. Samal and P. Mishra, "Analysis of variants in Round Robin Algorithms for load balancing in Cloud Computing," International Journal of Computer Science and Information Technologies, 2013.

[10] A. Lakra, and D. Yadav, "Multi-Objective Tasks Scheduling Algorithm for Cloud Computing Throughput Optimization." Procedia Computer Science, 2015.

[11] A. Thomas et. al., "Credit-Based Scheduling Algorithm in Cloud Computing Environment." Procedia Computer Science, 2015.

[12] X. Ren et al., "A Dynamic Load Balancing Strategy for Cloud Computing platform based on Exponential Smoothing Forecast," in International Conference on Cloud Computing and Intelligence Systems., Beijing., China, 2011.

[13] H. Chen et al., "Towards energy-efficient scheduling for real-time tasks under uncertain cloud computing environment," J.Syst. Software, Jan. 2015.

[14] K. Navdeep and K. Kaur, "Improved Max-Min Scheduling Algorithm.," IOSR Journal of Computer Engineering, vol. 17, May 2015.

[15] E. Pacini et al., "Balancing throughput and response time in online scientific Clouds via Ant Colony Optimization (SP2013/2013/00006)", Advances in Engineering Software, June 2015.

[16] F. Ramezani and F. Khadeer Hussain, "Task-based System Load Balancing in cloud computing using Particle Swarm Optimization" Int JParallel Prog, Oct. 2013.

[17] D. Babu and P. Venkata, "Honeybee behavior inspired load balancing of tasks in cloud computing environments," Appl Soft Comput, May 2013.

[18] N. Tziritas et al., "On minimizing the resource consumption of cloud applications using process migrations." Journal of Parallel and Distributed Computing, 2013.

[19] Himani and H. Sindhu, "Comparative analysis of scheduling algorithms of Clouds in cloud computing." *International Journal of Computer Applications*, 2014.

[20] Ranesh Kumar, Naha Mohamed Othman, Cost-aware service brokering and performance sentient load balancing algorithms in the cloud, Journal of Network and Computer Applications, Volume 75, November 2016, Pages 47-57.

[21] A. Saffar, R. Hooshmand, A. Khodabakhshian, A new fuzzy optimal reconfiguration of distribution systems for loss reduction and load balancing using the ant colony search-based algorithm, Applied Soft Computing, Volume 11, Issue 5, July 2011, Pages 4021-4028