Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

Load balancing in cloud environment using enhanced migration and adjustment operator based monarch butterfly optimization

R. Kaviarasan[a], P. Harikrishna[a,*], A. Arulmurugan[b]

[a] Department of Computer Science and Engineering, Rajeev Gandhi Memorial College of Engineering and Technology, Nandyal, Andhra Pradesh, India
[b] Department of CSE, Vignan's Foundation for Science, Technology & Research (Deemed to be University), Guntur, Andhra Pradesh, India

A R T I C L E I N F O

Keywords:
Cloud computing
Meta heuristic
Bio inspired
Load balancing

A B S T R A C T

In the decades before the advent of computers, humans tend to make mistakes while calculating and remembering tasks. Distributed computing helped to reduce the workload of each computer by distributing the workload evenly among computers connected in the network. Cloud computing have eradicated most of the problems that occurred in distributed computing but were also prone to different types of issues. Major issues in cloud computing relate to security and load balancing. Load balance of a node relates to two important parameters namely request time and response time. Meta heuristics algorithms can be used to provide proper load balancing techniques in cloud. This paper provides a mechanism namely EMAMBO to ensure that each node is properly load-balanced in cloud. Based on different metrics considered, it could be inferred that the proposed system fares better when compared to different benchmarked existing systems.

1. Introduction

Before the invention of computers, the life of the human beings was found to be challenging. It is a well- known fact that computer has drastically improved its user's capability in doing complex calculation and storing data. Even computers have limitation in storage and computation power which led to distributed computing. In case of distributed computing [1,2], a huge task is divided into sub tasks. These sub-tasks are given to computers that are connected over a network. So, some of the advantages of using distributed computing is Flexibility, Scalability, Fault tolerance and Reliability. But there were also several drawbacks like security, multiple points of failure. One of the most commonly used variant of distributed is Cloud computing.

Cloud computing [3] provides different types of services to user based on his demand/ request. So these resources requested by the users are shared to them with the help of network (most preferably Internet). As soon as the task with that resource is completed, the user may return the resource and he/she will be charged only for the duration of the use of resource. This technique is similar to our electricity bill and is known as pay-per-use technology. Companies like Google, Amazon, and Microsoft provides different types of Cloud based services. These companies are collectively known as Cloud Service Providers [4,5]. The advantages of cloud computing [6] are Flexibility, Accessibility, easier implementation and lower cost. The disadvantages of cloud computing are over dependence on internet, security, compliance concern and limited control. Some of the challenges faced by companies can be solved when the shift their data from their Classical Data Center [7] to Cloud. These challenges include Storage growth, cost of ownership, Globalization and ageing data centers.

The following are the characteristics of Cloud computing[8]

1. On Demand Self Service: In the case of cloud computing, if a user requires any computing resource he/she can just demand it. Based on the user's request the resource will be allocated by the CSP.
2. Broad Network Access: Cloud computing is not restricted to desktops, but to different types of devices like laptop, tablets and smart phones as long as they have a proper internet connection.
3. Resource Pooling: The resources are pooled so as to able to cater the needs of multiple users at the same time. This is known as multi-tenant model. These resources will be allocated based on the user's demand.
4. Rapid Elasticity: Resource can be allocated and de-allocated to a user in a simple and faster manner. There are some Cloud Service Providers (CSP) where this process is done automatically based on the demand.

* Corresponding author.
E-mail address: pillutlaharikrishna@yahoo.co.in (P. Harikrishna).

https://doi.org/10.1016/j.advengsoft.2022.103128
Received 27 January 2022; Received in revised form 26 March 2022; Accepted 20 April 2022
Available online 7 May 2022

5. Measured Service: In cloud, the CSP will monitor or measure the service provided to the user. This measuring can be for billing as well as to make sure that the resource is being properly utilized.

There are three types of cloud namely public, private and hybrid. In case of a public cloud [9], the services are available for the public users. Companies like Amazon, Google and Microsoft can be used to avail this type of cloud. Private Cloud [10] is opposite of public cloud. The private cloud is available to users of a particular organization. Hybrid cloud [11] is the combination of both public and private cloud. Then there are several services provided by Cloud out of which there are three which are mostly commonly used. They are SaaS (Software as a Service), PaaS (Platform as a Service) and IaaS (Infrastructure as a Service). In SaaS [12], the service provided is the software. The main difference between a traditional software and a SaaS is that there is no need to install and purchase the software in SaaS. The best example of SaaS is Google docs. In PaaS [13], the service delivered is a platform in which a user can create, design and manage his/her own software. The example of PaaS is Windows Azure. In IaaS [14], the service provided is physical infrastructure like compute, storage and networking components as well as load balancing. The example of IaaS is Rackspace.

Security and optimization play an important role in better utilizing the cloud [15]. Some of the security issues in Cloud are Denial of Service attacks, Distributed Denial of service attack [16], Advanced Persistent Threat and misusing of cloud resources. To overcome these threats there are several countermeasures that are available like Authentication, Authorization, IDPS (Intrusion Detection and Prevention System), Trust modeling and Firewall.

The advent of Cloud computing has benefited the mankind in many ways in terms of less cost which minimizes capital expenditure of a company, provides better security when compared with other computing platforms and the business data that can be stored in the cloud will be a good source of back up during disaster and the organization need not depend upon the hardware or software utilities for their company exclusively when they migrate to cloud.

One of the objectives of cloud computing is to make sure that the computing resources are properly utilized. One method to perform it is by Optimization methods. There are several bio inspired optimization algorithms like Ant Colony optimization [23], Bee Hive optimization, Particle Swarm Optimization, Monarch Butterfly optimization and Kill Herd optimization. This paper main objective is to make sure that the cloud resources are properly utilized to the fullest extent.

The task scheduling is the major concern in cloud computing environment which degrades the system performance. To improvise the system performance we must make use of effective load balancing algorithms. The major setbacks faced in task allocation are discussed below:

Unpredictable workloads: This is a significant setback in cloud computing as the workloads are generally unpredictable and the fluctuation in the workload can happen in a planned or unplanned manner. In case of a planned fluctuation the excessive workload can be forecasted well in advance and the allocation of resources will be done in a smooth manner.

Guaranteed resource utilization: In spite of, an unplanned demand, the resources must be allocated whenever a demand is created. This is auto scaling mechanism in cloud environment. The incoming workload should be allocated to the VM for effective resource utilization. This can be achieved by effective scheduling methods to allocate the tasks to the VM by analyzing the under loaded and overloaded nodes.

Presence of Hetro nodes in Data Center (DC): The nodes that are distributed in different locations will vary in terms of computation capacity, memory and network performance. The incoming tasks that are allocated to various available hetro nodes and different tasks are performed by different capacity VMs.

Scheduling problem: The problems with respect to scheduling have grown from processing a simple task in classical computing systems to handling complex problems in VM in terms of resource scheduling and migration in cloud environment.

The proposed work has been designed based on the inspiration of the bio inspired behavior of the Monarch Butterflies. The highlight of this paper includes:

a) The proposed work which is designed being a Meta-heuristic approach doesn't get struck into local optimum during the search process and to find an optimal solution.
b) Monarch Butterfly being a population based search performs the search process with random initial population and is enhanced over the course of time.
c) Being population based search, the proposed work can move into promising areas of search space thus the exploration rate is found to be greater when compared to single solution based search algorithms.
d) The shift in convergence is found to be uniformly maintained during the exploration and exploitation.
e) The major improvement of this approach the throughput, response time is found to improve and migration time, fault tolerance and energy consumption is found to be minimized when compared to the bench marks.

The organization of the paper is as follows: Section 2 describes about the literature survey with various load balancing techniques with their merits and demerits. Section 3 depicts the proposed work design. Section 4 tells about the simulation environment and the results inferred. The last section tells about the conclusion and the future direction.

2. Literature survey

Cloud computing has gained the attention of the users in the recent years. As this is a digital era and the industries consideration has been grabbed by cloud computing environment because of sharing of computing resources over the internet at a minimum cost. Luthur et al designed a computing model based on the internet that helps in sharing of hardware and software resources [17]. The cloud computing facilitates virtualized sharing of files that laid the basics of load balancing and sharing of the resources in an optimized manner. Cloud registered users, access the stored files and resources through a methodology called virtualization. So, an effective resource sharing algorithm and load balancing scheme has to be designed to handle effective file sharing

Several researchers have tried to afford load balancing methods in the Hadoop environment. The Hadoop map reduces splits the given file into two fixed blocks and each divided block has replicas on three different nodes. Balancer a built in tool is used to move the data blocks from the overloaded node to the less overloaded node. The purpose of this tool is for balancing the cluster which consumes higher amount of system resources. For this primary reason several new load balancing approaches has been evolved.

A block based load balancing algorithm had been proposed by Kolb et al. for minimizing the search space in the Entity-Resolution [18]. This approach divides the input data into multiple blocks and prevents the successive matching of the entities to the same block. This approach takes the size of the block into consideration and assigns the entire block to minimize the task without violating the load balancing limitations. It had been experimented with the real time data set of Amazon EC2 cloud computing environment and is found to be robust when compared with data skew. It uses different size sub blocks which may create imbalance in the reduce phase and it is also time consuming as it is using multiple blocking keys. Another scheduling approach had been designed by Bok et al. to handle multimedia data in MapReduce framework [19]. It also utilizes replica of data approach if the load is found to be high for a node, the replica node refers to other node that has the needed block available to process the job. If the job is not able to be completed with the deadline, then the most important job is selected and temporarily

Algorithm
Algorithm for EMAMBO.

```
Calculate total number tasks in all nodes
  Calculate the number of tasks in every node.
  Response time "$t_{res}$" for the received packet is calculated is Equ. (1).
  Calculate the sum of workload of all the nodes for a time period "$L_{C_i\ t_n}$" using Equ. (2)
  Requested task response time is to be calculated using Equ. (4)
  Threshold is calculated and compared with response time
  If response time is found to be higher
  Node is overloaded
  The task will be migrated to the under loaded node using migration operator using Equ.(6)
  The position of task where node is present is updated using Equ.(7)
  Else
  Node is under loaded.
```

suspending the ongoing process in order to minimize time deadline. This approach is found to have an added advantage of minimized completion time. But the major concern is that it doesn't have implementation in real map environment.

Another pioneer load balancing method in MapReduce had been proposed by Kulkarni et al in which they had designed a scheduling model to work in a heterogeneous environment [20]. It has a classification algorithm that is much more aware about the needs of the resources of the clusters and the job necessities which is needed for the scheduler. The jobs in this approach are classified into two approaches executable and non-executable. The executable jobs are assigned to the proper nodes for successful execution and to evade failure rate. This mechanism has added advantages like minimizing the overhead faced by the master nodes and smaller job starvation has been avoided. Identifying the content information is found to be more time consuming with this approach.

Biological adopted phenomena's are also been adopted by the researchers for balancing the load in the cloud environment [36]. These algorithms had been created by mimicking the behavior of the Ant [37], Honey bees, Cuckoo and Genetics.

Cloud computing has gained demand among the users because of the merits of less cost and high availability. But still it has certain demerits in terms of resource management and power utilization. Yakhchi, M et al. has proposed a mechanism using a Bio inspired approach called Cuckoo Search Algorithm (CSA) [21]. This algorithm is constructed based on the behavioral inspiration of cuckoo bird. This bird has a peculiar behavior of laying eggs in other birds nest. With this approach the population of the cuckoo bird will increase in that particular area. At certain cases the host birds identifies the cuckoo eggs and eliminate it. Generally increasing in the number of eggs in an area would come up with significant profit value. The CSA has three different phases namely: (i) detection of over utilized hosts (ii) detection of underutilized nodes and (iii) selection policy regarding the nodes.

The over utilized node will not be able to handle the entire request and as a result the time spent in replying the request will be high. So CSA has been deployed to handle this situation. For solving the discrete optimization problem the states has been changed from continuous state to discrete state. Initially the host is selected randomly and the characteristic of the hosts are stored in an array called habitat. Then the CSA is used for calculating the resource utilization using the profit utilization. Then for each selected slots, the eggs are laid between 2 to 4 which is the upper and lower bound which has been generated using Egg Laying Radius (ELR). When a host is under loaded it means it can handle more request and has more energy to serve the request. The under loaded nodes are marked as per the selection policy, their request are migrated to other nodes making them free and it is set to sleep, as a result the energy consumption is minimized. The Minimum Migration Time policy is used to select the nodes. The major con of this approach is SLA avoidance and it does not concentrate more on the security policies for VM migration.

Load balancing in cloud can be achieved using the foraging behavior of the honey bees. The behavior in identifying the food source and reaping it based on which the Honey Bee Algorithm is been proposed by Dhinesh Babu et al. [22]. Generally in Bee hives, there will be Queen Bee, Drone Bee and worker Bee. The function of the worker bee is used to identify the food source and reach to the hive and intimate to other worker bees through vibration dance. The dance has a message regarding the quantity and quality of the food source and the distance from the bee hive. Other foraging bees move in the direction of the worker bees and reap the food. After returning to the hive it again makes a vibration dance stating the information of how much food is left out which results in further exploitation of the food source. Dinesh Babu et al. relates the tasks as honey bees and the VM as food source. When a task is submitted to a under loaded VM, the task will be updated with other tasks and priority level is updated and the current load of the VM is calculated and made available to other tasks which is waiting in the queue. This information is more vital for other task in choosing the VM which is less loaded. And whenever a high priority request is submitted a particular VM some things has to be taken into account like the VM should have minimum number of high priority tasks. The VMs will be sorted in the ascending order depending on the number of task that is handling. Whenever the VM is overloaded the task is moved over to the under loaded node. The removed task updates the details of the VM in terms of number of task handling by the machine and the higher priority details. This updating helps in identifying the VM based on the load and availability factor. But the selection of head node is a vital one which helps in forward and backward movement but no effective mechanism is used in selecting the approach which is a major drawback in this approach.

Agent based technique is been adopted by certain researchers for load balancing in the cloud environment. The agent is software that is designed to satisfy the needs of the objectives.

Singh et al. [24] have proposed Autonomous based Agent Load Balancing Algorithm (A2LB) to improve the throughput, resource utilization as well as scalability and reliability. Whenever a Virtual Machine is found to be overloaded, the CSP should be able to distribute the resources in a way so as to make sure that the resources that are currently available are properly utilized while also to maintain all the VMs balanced. There are three types of agents that are available in A2LB, they are 1) Load Agent, 2) Channel Agent and 3) Migration Agent. The main purpose of a local agent is to retain each and every specifics of the data center while also controlling the information policy. The channel agent generally receives the request from the load agent. Based on these requests, this agent will begin some migration agents to other data centers which will be used for exploring all other VM's in order to find similar configuration. The channel agents have the ability to maintain the location, selection and transfer policies. Channel agents are helpful in initializing the Migration agents that are helpful for shifting to other data centers as well as used for communicating with the load agent of that data center in order to enquire about the status of the Virtual Machines that are available there. The major drawback of this approach is time consumption as the migration agent consumes higher time in identifying a VM.

An application aware load balancing scheme was proposed by Tasquier et al. it uses multi agent concepts where the agents are used to identify the status of the node whether it is over utilized or underutilized [25]. It uses three agent namely executor agent which signify the application in the cloud scenario, the provisioned agent is responsible for including and removing the resources and the monitor agent is useful in analyzing the overloaded and under loaded conditions that prevail in the cloud environment. This approach doesn't insist more upon the Quality of Service.

General load balancing scheme had been proposed by Chien et al, this approach initially calculated the size of the jobs and the processing capacity of the VM [26]. It has two factors one which relies on the selection of VM that has the capacity to finish the required job and the load balancing algorithm that is present. It calculates the overall time required for processing the job that is present in the queue and also the new jobs. The VM which communicates first will be selected to distribute the job. From the results it is observed that the algorithm has improved response time. This approach has power processing complexity in it which is considered to be a hinder.

Sarood et al has eradicated the gap which arose between the application on cloud and super computers [27]. It uses object migration method for running parallel application in virtual scenarios that undergo from impede jobs. The load balancing approach not only reduces the time which caused by interfering jobs it also minimizes the energy utilization. This approach always utilizes the load balancing approach and doesn't has any type of decision making system which is considered to be an added drawback.

This section has discussed about the various load balancing approaches for balancing the load in the cloud environment. This section has discussed some noteworthy approaches proposed by the various researchers and it has been analyzed with their working their advantages and disadvantages. Based on the literature survey we were able to conclude the drawbacks of existing systems are consumes higher amount of resources for load balancing, implementation with real map environment, time complexity is high, security policies of the VM is not followed during migration of tasks, services doesn't insisted upon quality of service, lack of decision making and lack in better exploration and exploitation in assigning the tasks. It is quite evident that there arose a need for a methodology for improved load balancing in cloud. The proposed methodology will be designed based on the properties of the Meta heuristic approach as it has added merits like exploration and exploitation which would play a vital role in load balancing [38].

3. Outline on the working of the proposed Enhanced Migration and Adjustment operator Based Monarch Butterfly optimization (EMAMBO)

- Whenever a new work arrives it is subjected to the analysis of preprocessor
- The pre-processor computes arrived work based on the number of tasks and length of the task.
- The tasks which are ready are sent to the VM load balancer
- The VM load balancer makes the first task to wait in the queue. The tasks are generally performed in the order of FCFS (First Come First Serve).
- In order to identify which task is allocated to which VM we must know the details of allocation and de-allocation of the last task. This can be related to how butterfly is identifying the food source. Then the VM threshold is identified and the priority of the task must be taken into consideration for processing the task.
- The Host limit is identified if the VM is found to be overloaded the task has to be removed and placed in waiting queue and a suitable host has to be found and allocated. It finds the VM machine based on the threshold data's and priority of the tasks.
- Once a suitable VM is found the task is removed from the waiting queue
- The next task is obtained from the waiting queue.
- There may be situations when one or more VM is available in that case the minimum migration time can be taken into consideration for selecting a VM.
- The tasks which is allocated is generally updated on which VM that is handling and whenever new tasks has to be processed with the help of the proposed EMAMBO the new VM machine is been identified and its threshold is cross verified for the purpose of the VM is not overloaded.

The below section discusses about the proposed EMAMBO which is designed based on the inspiration of the behavior of the Monarch Butterflies.

Fig. 1. Monarch butterfly.

3.1. *Enhanced Migration and Adjustment operator Based Monarch Butterfly optimization (EMAMBO)*

Based on the inspiration of migration behavior of Monarch Butterflies this Bio inspired algorithm has been designed based on [31–33]. The monarch butterflies have orange pattern and black pattern in their wings which is found to be different from other butterflies. The male and female can be easily identified with their pattern in wing. Initially they were native of north and South America but they are wide spread across other warm places wherever the milkweed grows.

Monarch butterflies are famous for their seasonal migration which migrates from United States and Canada to California and Mexico during winter seasons. They even travel nearly 3000 miles. Generally they use sun to stay on course and their gene has efficient muscles which help in travelling larger distance. They major reason for migration is to lay eggs in milk weed and to increase their population. But now in the current scenario the conservation group has suggested the US government to include the Monarch butterflies in the endangered species as due to climatic changes and reduced amount of milk weed available. Fig. 1 depicts the image Monarch Butterfly [34,35].

3.2. *Modeling of Migration and Adjustment operator Based Monarch Butterfly optimization*

The migration behavior of butterfly is been adopted in migrating the task of the overloaded nodes into the node which has been less loaded. The general analogy of working of Monarch Butterfly is initially discussed. The monarch butterflies exhibit the behavior of migration from one region to another region. Based on this operation the migration operator is designed. The search process is effectively carried only with the help of migration operator and the adjustment operator. The adjustment operator helps to identify the position of the node. Both these operators can be used in parallel manner. It helps in making the transition between exploration and exploitation.

The total number of task is analyzed in all the nodes and total number of tasks handled by the nodes is initially calculated.

The periodical statistics is used for the response time and is initially calculated for period of time interval 't'. The t_{arr} denotes the packet arrival message and t_{rep} denotes the reply time taken for replying the received message. The response time for the received packet is calculated using Eq. (1).

$$t_{res} = t_{rep} - t_{arr} \tag{1}$$

The number of task request is denoted by t_{req} in t_n is denoted by $f_{treqtn.}$ to denote the workload brought by t_{req} int$_n$ is denoted by $L_{treq\ tn.}$ To calculate the sum of load of these nodes in the set N_{Ci} can be used to represent the total number of received task request L_{Citn} where C_i denotes the node in the n_{th} time period which is shown in Eq. (2).

$$L_{C_i t_n} = \sum_{t_{req} \mathrm{euro} N_{C_i}} f_{t_{req} t_n} \tag{2}$$

Next phase involves in calculating the response time ($t_{response}$)of the requested task. The total response time of the accepted task in C_i in t_n can be obtained. The response time for the task which is sent by the controller is denoted by t_{req} in t_n. So the response time is calculated using the Eq. (3)

$$t_{c_i\ t_n} = \frac{\sum_{t_{req}\ \mathrm{euro} N_{ci}} t_{response}}{L_{C_i\ t_n}} \tag{3}$$

The threshold has to be calculated during the next phase of the work as once the nodes crosses beyond the threshold the task must not be allocated to the respective node. So response time plays a major role in identifying the overloaded node. If the response time is found to be high it denotes that the node is overloaded. The appropriate threshold $t_{threshold}$ is calculated using the extreme response time variation where $r(t_i\text{-}1)$ denote the average response time in the (i-1) time and $r(t_i)$ represents the average response time in the i^{th} period. Using the Eq. (4) and (5) the threshold can be predicted based on the first and second order.

$$r'(t_i) = \frac{r_{t_i} - r_{t_{i-1}}}{t_i - t_{i-1}} \tag{4}$$

$$r\prime\prime(t_i) = \frac{r'_{t_i} - r'_{t_{i-1}}}{t_i - t_{i-1}} \tag{5}$$

$r(t_i)$ is the average response time for the i^{th} period $r(t_i\text{-}1)$ denote the average response time in the (i-1). $r'(t_i)$ and $r''(t_i)$ are the first and second order prediction

If the response time increases it denotes that the node is overloaded and further task should not be allocated to that node.

The task in the overloaded node has to be migrated to the under loaded node, for this the migration operator is been used. Initially the total number of tasks handled by the cluster can be calculated using ┌(p * NT) (NT1) and NT–NT1(NT2) where NT is the total number of tasks and p is the ratio of tasks in a node1. The migration process is done using Eq. (6)

$$x^{t+1}_{i,k} = x^{t}_{r_1,k} \tag{6}$$

Where $x^{t+1}_{i,k}$represents the k^{th} task of $x_i$and'$t+1$'represents the old nodes position of the task and $x^{t}_{r_1,k}$denotes the newly migrated node position of the task.$r_1$represents the task which is randomly chosen from the node.

$$r = rand * peri$$

peri denotes the migration time period and rand is the random number selected from the uniform distribution.

The position of the task in which node is present is been updated using the adjusting operator. For task 'j' the randomly generated rand number is smaller or equal to the value of p it is updated using the Eq. (7).

$$x^{t+1}_{j,k} = x^{t}_{best,k} \tag{7}$$

Where $x^{t+1}_{j,k}$represents the k^{th} task of x_i and '$t+1$'represents the position of the task in the 'j' node. $x^{t}_{best,k}$ denotes the best kth element of x_{best}.

If rand > p it will be updated using Eq. (8).

$$x^{t+1}_{j,k} = x^{t}_{r_3,k} \tag{8}$$

IfBar> p, where is the adjusting rate. It can be updated using Eq. (9)

$$x^{t+1}_{j,k} = x^{t+1}_{j,k} + \beta\ (\mathrm{dy}\ - 0.5) \tag{9}$$

*where*β denotes the weighted factor which influences the exploration when β is found to be small it decreases exploration and influence exploitation and Bar indicates the adjusting rate. The algorithm below portrays the working of the proposed work.

Table 1
Simulation parameters.

Simulation Parameters	Value
Number of Physical Machines(PM)	2-6
Number of Processing Units (PM)	4
Scheduling Interval (PM)	30ms
Monitoring Interval (PM)	180ms
No of Virtual Machines	50
Total number of Tasks	100-1000
Length of the Task	2500* Simulation limit
Number of iterations	100
Cluster size	1-65
Size of the task	500
Average RAM	512 MB
Average Bandwidth	1,00,000 Mbps

4. Simulation set up

The proposed method has been implemented using CloudSim 3.03 environment. The CloudSim software is been widely used by the researchers for implementing the cloud environment [28,29]. For implementing the data centers and virtual machines and the policies for provisioning host resources to the VM. The CloudSim is much more flexible between the sharing of spaces and time shared allocation of processing cores to the virtualized servers [30].

The two data centers which is simulated has a configuration of HP ProLiant ML110 G4 (1860 MIPS, 4 GB) and HP ProLiant ML110 G5 (2660 MIPS, 4 GB) and it has a storage capacity of 1GB and a bandwidth of 1GBps. For this experiment 50 VM has been designed to have storage capacity of 2.5 GB and bandwidth of 100 mbps. Table 1 highlights the additional simulation parameters used for implementing in the proposed work.

The main role of a performance metrics is to measure the performance of the proposed system against benchmarked systems like Cuckoo Search Algorithm (CSA), Honey Bee Algorithm (HBA) and Autonomous Agent Based Load Balancing Algorithm (A2LB). The following metrics were found to be suitable and were considered for measuring the performance of EMAMBO with existing approaches mentioned above.

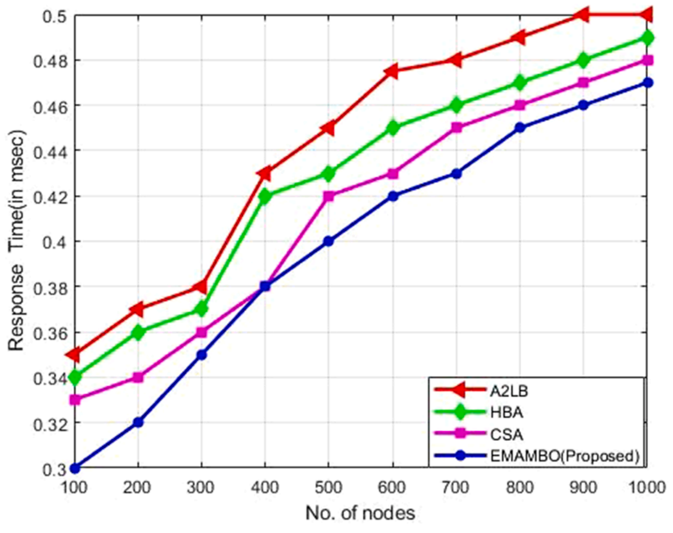

Fig. 2. EMAMBO response time vs no. of nodes.

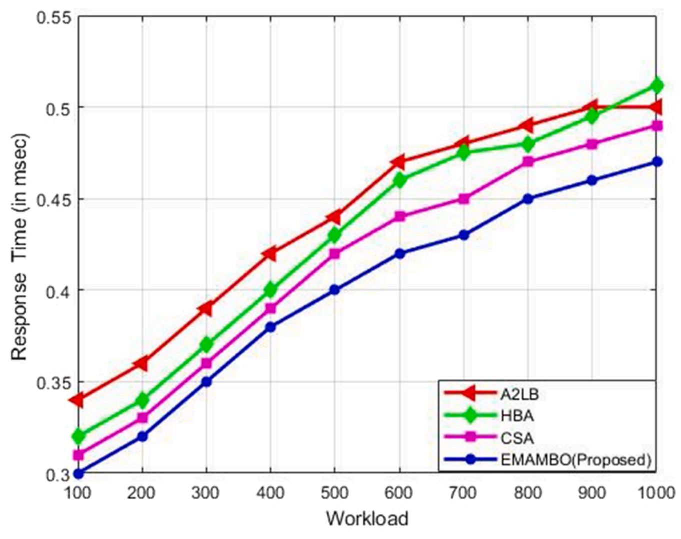

Fig. 3. EMAMBO response time vs workload.

1. **Response Time**: The response time is the total time taken by system to serve a put forward task
2. **Throughput**: The throughput calculates number process which is completed at a given unit time
3. **Fault tolerance**: The fault tolerance states that the algorithm continues to balance the load in the cloud in spite of the occurrence of fault in the nodes present
4. **Migration**: The migration time is the time incurred in migrating a task from the overloaded node to the node which has fewer loads
5. **Performance**: The performance of the system is generally used to measure the efficiency of the system after performing the load balancing algorithm
6. **Energy Consumption**: Energy consumption denotes the amount of energy consumed by the proposed system.
7. **Transmission Time**: It is the time taken by a task has to reach a particular VM. It also depends upon the size of the task and the bandwidth of the VM

Figs. 2 and 3 highlights the response time in accordance with the proposed work and it is been studied under no of tasks and the no of nodes. The response time is the total time taken by system to serve a put forward task. The response time of the proposed EMAMBO algorithm is found to be increased to a greater extend when compared to the existing methods namely Cuckoo Search Algorithm (CSA), Honey Bee Algorithm (HBA) and A2LB when it is varied against the number of nodes and also with respect to the no of tasks. In Fig. 2, the response time is found to be minimized by the proposed algorithm by 2%, 4% and 5 % respectively against the benchmarks chosen and Fig. 3 the response time is considerably minimized by even under different number of task by 3%, 4% and 6% when compared against CSA, HBA and A2LB respectively.

Figs. 4 and 5 depicts the throughput which is achieved by the proposed work and it is further studied against the number of nodes and increased number of task and the performance is compared with the bench mark chosen. The throughput plays a vital role in analyzing the efficiency of the proposed which is designed. The throughput generally calculates number process which is completed at a given unit time. In Fig. 4, the throughput of the proposed system EMAMBO is found to be improved under increasing number of workloads by 6%, 9% and 12% against the benchmarks CSA, HBA and A2LB. In Fig. 5, the proposed work EMAMBO dominates the existing work like CSA, HBA and A2LB by

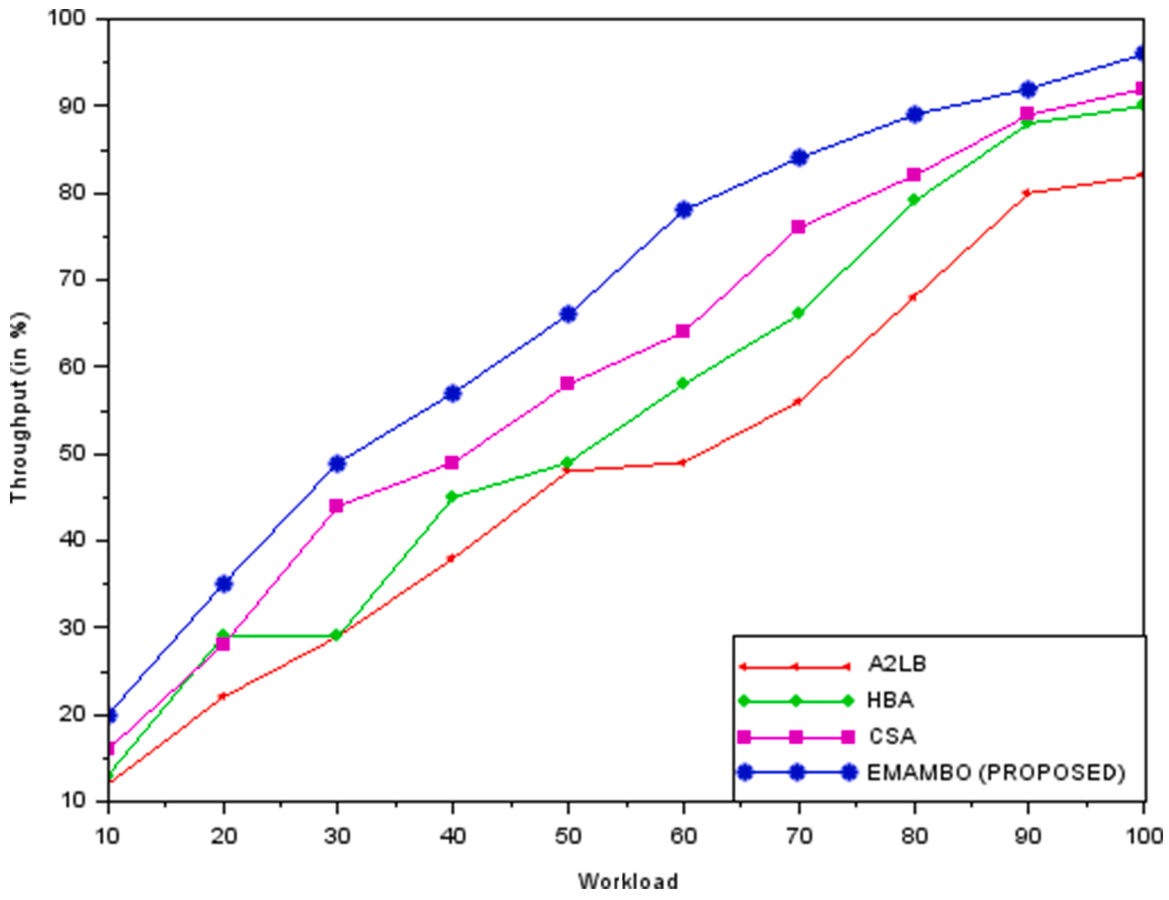

Fig. 4. EMAMBO throughput vs workload.

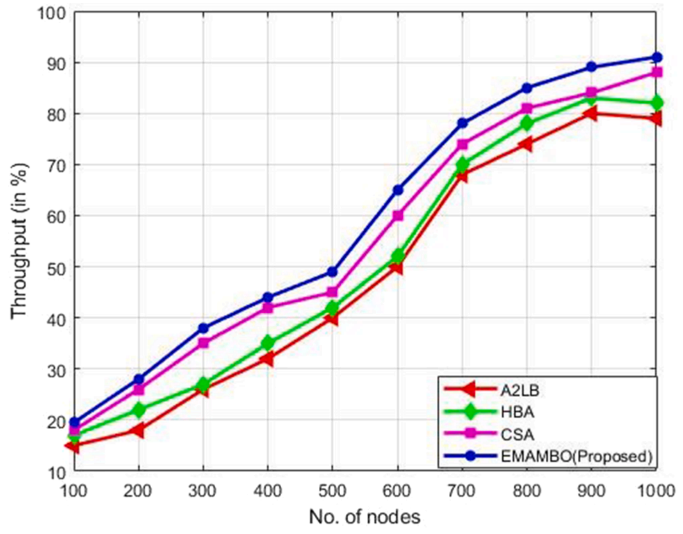

Fig. 5. EMAMBO- throughput vs no. of nodes.

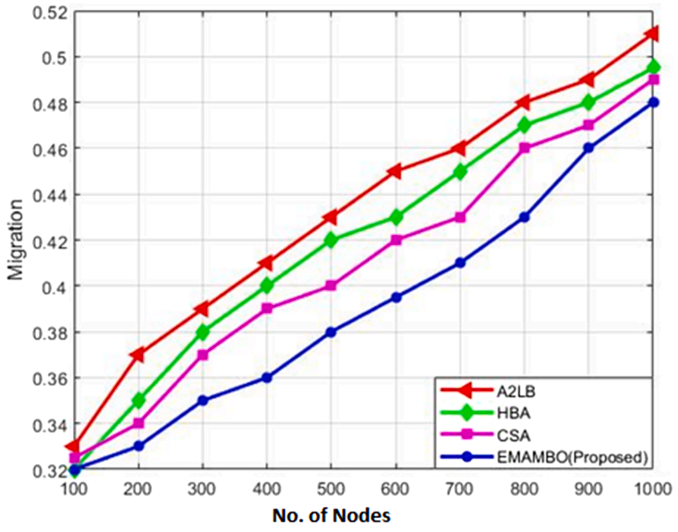

Fig. 8. EMAMBO migration vs no. of nodes.

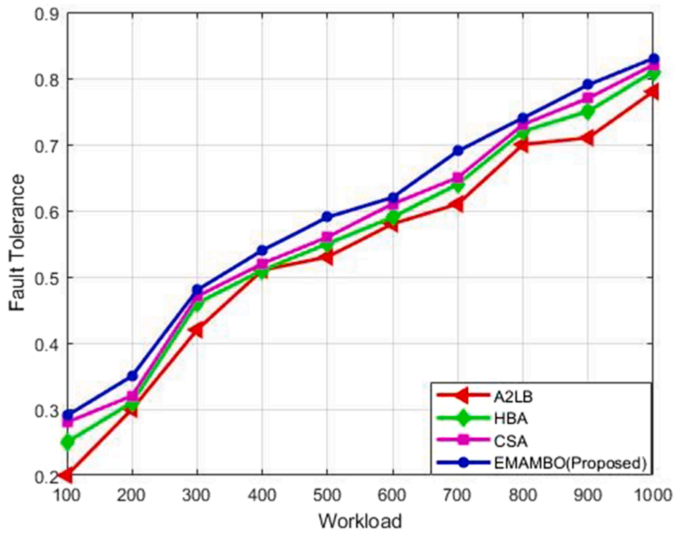

Fig. 6. EMAMBO fault tolerance vs workload.

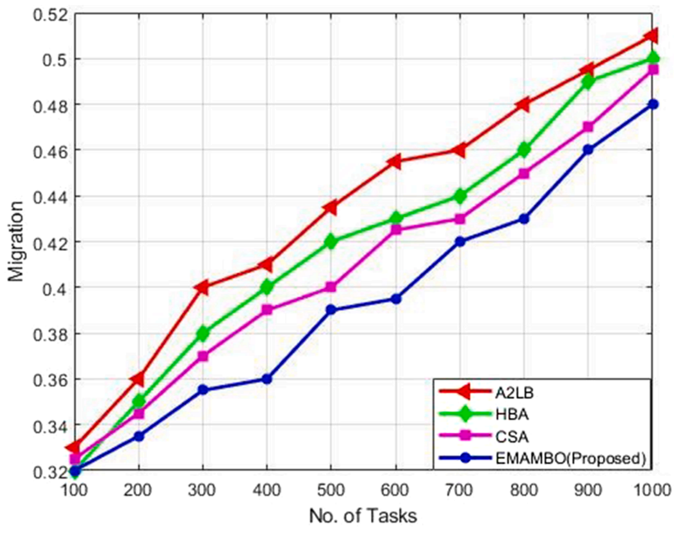

Fig. 9. EMAMBO migration vs no. of tasks.

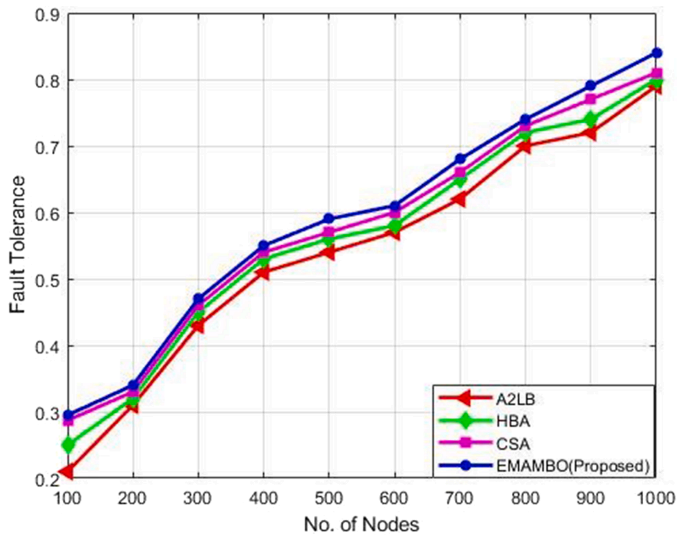

Fig. 7. EMAMBO fault tolerance vs no. of nodes.

5%, 8% and 10%improved throughput. The throughput is considered to be important parameter always in optimization.

In Figs. 6 and 7 portrays hoe the fault tolerance is minimized by the proposed work EMAMBO when it is investigated against the increased number of tasks and the number of nodes. The fault tolerance is found to be minimized when compared to the existing works chosen namely CSA, HBA and A2LB. The fault tolerance generally states that the algorithm continues to balance the load in the cloud in spite of the occurrence of fault in the nodes present. Fig. 6 depicts that the proposed work EMAMBO has maximized the fault tolerance rate under increased work load by 3%, 5% and 6% respectively when compared to CSA, HBA and A2LB. The Fig. 7 depicts that proposed work has maximized fault tolerance rate drastically by 4%, 6% and 9% respectively when compared to its bench mark chosen. The fault tolerance rate is crucial parameter that has to be taken into consideration as the occurrence of fault in the node is found to be quite evident. The fault should not affect the Quality of Service which is rendered to the users. The proposed work EMAMBO incorporates the features Meta heuristic approach and has added advantages when compared to Cuckoo and Honey Bee optimization.

Figs. 8 and 9 evaluates the proposed work EMAMBO in terms of the parameter migration time under increased load and number of nodes

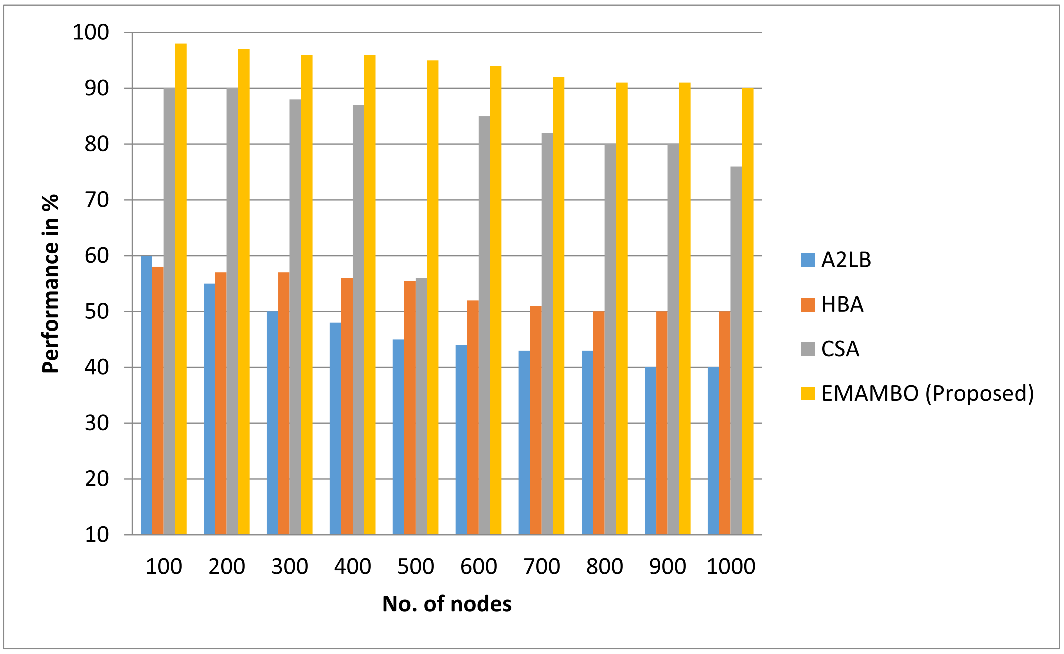

Fig. 10. EMAMBO performance vs workload.

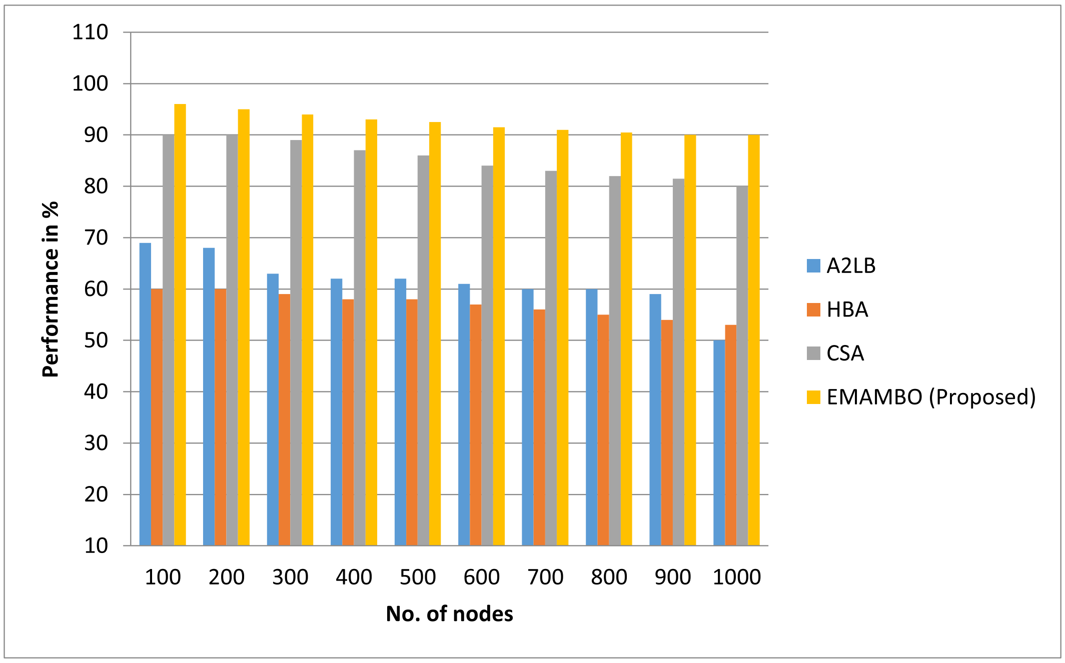

Fig. 11. EMAMBO performance vs no. of nodes.

and make as comparative analysis on the CSA, HBA and A2LB. The migration time is the time incurred in migrating a task from the overloaded node to the node which has fewer loads. The migration time of the proposed work EMAMBO is found to minimum which eventually reduces the computational overhead of the network. The Proposed work makes use of exploration and exploitation at a better rate when compared to cuckoo and honey bee. In Fig. 8 the migration time of the proposed work EMAMBO is found to be minimized by 3%, 6% and 9% respectively when compared to the bench marks CSA, HBA and A2LB chosen. In Fig. 9, the proposed work is found to work better by minimizing the time taken for migrating the task under increased number of nodes by when compared to the existing work.

Figs. 10 and 11 depicts performance of the proposed work EMAMBO on the metrics performance when varied with number of tasks and Number of nodes. The performance of the system is generally used to measure the efficiency of the system after performing the load balancing algorithm. The proposed work is found to have improved efficiency when compared to the existing work namely CSA, HBA and A2LB. In Fig. 10 the proposed Meta heuristic work EMAMBO is found to have improved efficiency by 11%, 13% and 15% respectively when compared to CSA, HBA and A2LB. In Fig. 11 the efficiency is improved even though improvised when varied under varied number of nodes the performance is improved by 12%, 13% and 17% respectively when compared against the bench marks chosen.

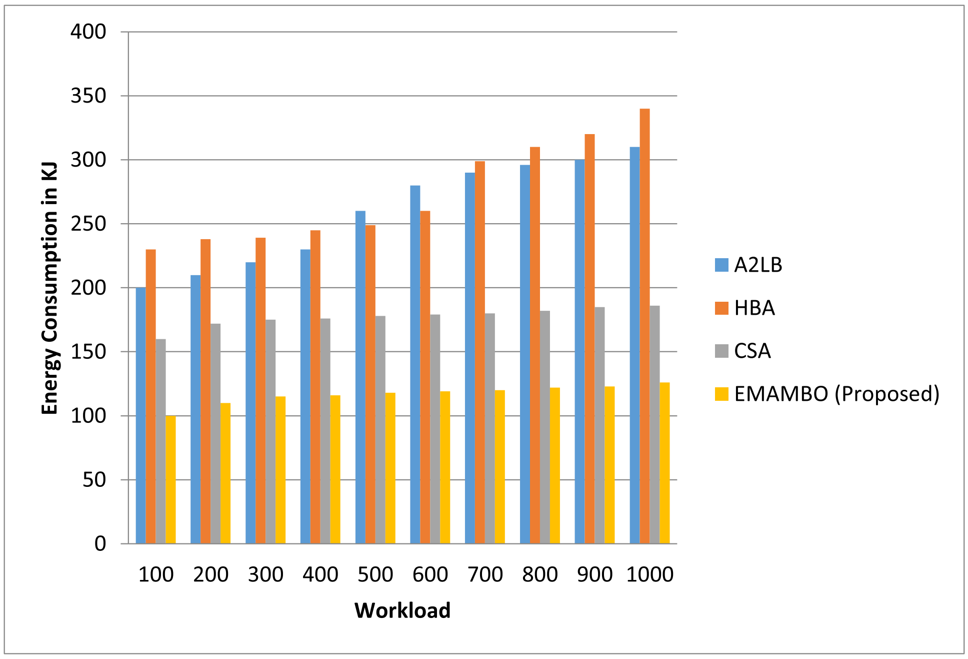

Fig. 12. EMAMBO energy consumption vs workload.

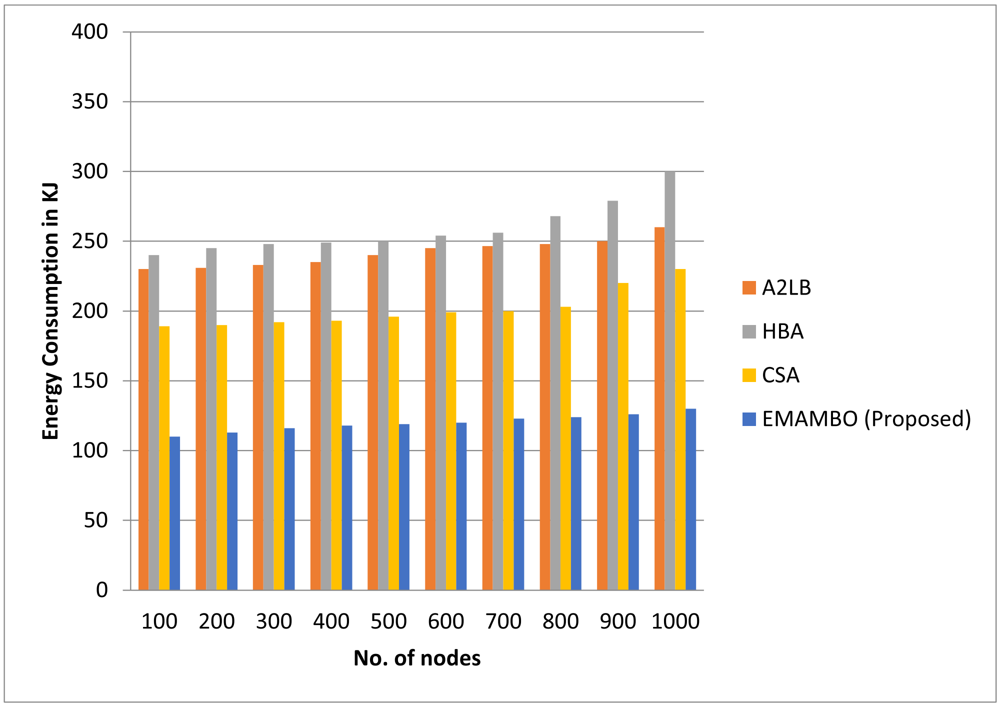

Fig. 13. EMAMBO energy consumption vs no. of nodes.

In Figs. 12 and 13 the energy consumption parameter plays a crucial role in load balancing approach. Generally the load balancing approach eventually reduces the overheating and therefore minimizes the energy consumption of the nodes. The usage of an effective load balancing approach helps in minimizing the energy consumption. In Fig. 12 the usage of proposed methodology EMAMBO helps in minimizing the energy utilization to a greater extend by 6%, 8% and 12% respectively when compared to its counterparts CSA, HBA and A2LB by. In Fig. 13 the energy utilization is found to be minimized even under different nodes by 7%, 1% and 14% respectively when compared against the benchmarks chosen.

The Fig. 14, uses transmission time as a metric to compare the proposed system namely EMAMBO with benchmarked systems namely A2LB, HBA and CSA. The transmission time for EMAMBO is lower by 10%, 8% and 7% when compared with A2LB, HBA and CSA respectively.

5. Conclusion and Future Enhancement

It is well known that computers tend to make the life of its users simple. There are several instances where a single computer cannot handle tremendous workload. So this lead to the birth of distributed computing in which a complex task is divided into much simpler task

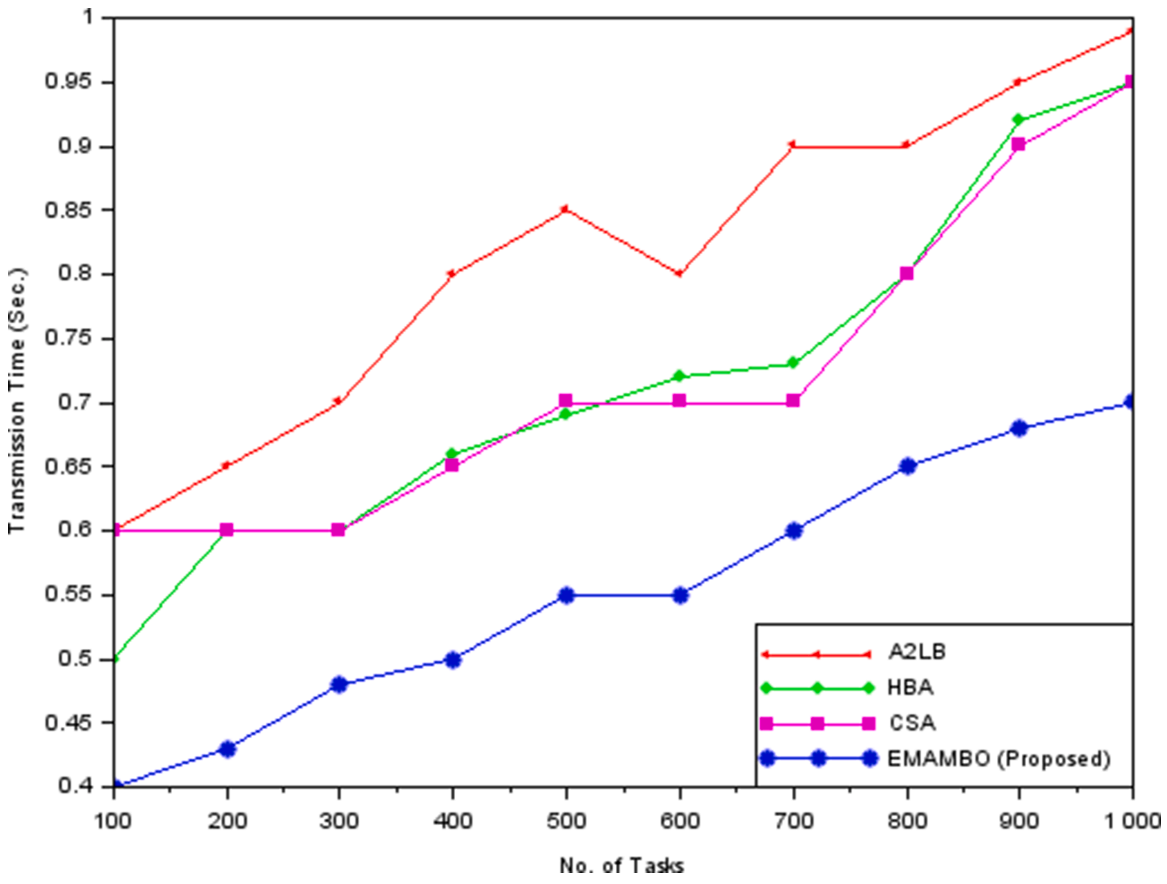

Fig. 14. EMAMBO transmission time vs no. of task.

and is handed out to different systems in a network. Some of the issues that had arisen in distributed computing were resolved in cloud computing. Cloud computing has several issues out of which load balancing is considered to be an important one. Load balancing deals about maintaining equal workload for all nodes in cloud. If a node is found to have tremendous workload then some of its tasks is shifted to a node with much lesser workload. Based on the simulation results it is quite evident that the proposed EMAMBO fairs quite better when compared to benchmarked systems like CSA, HBA and A2LB. EMAMBO has maximized fault tolerance rate by 4%, 6% and 9% respectively when compared to its bench mark chosen. The major reason behind the improvisation in the fault tolerance of the system is due to effective handling of the task migration in case of system failure. In EMAMBO, the efficiency is improved by 12%, 13% and 17% respectively when compared to CSA, HBA and A2LB. As a future work, this proposed system can be integrated with several other meta-heuristics approaches.

Declaration of Competing Interest

The authors have no conflict of interest with any reviewers and also with the journal and no funding agencies are involved in the creation of the manuscript.

References

[1] Xu Yinan, Liu Hui. Zhihao Long "A distributed computing framework for wind speed big data forecasting on Apache Spark. Sustain Energ Tech Assess 2020;37. Feb.
[2] Bandopadhaya Shuvabrata, Dey Rajiv, Suhag Ashok. Integrated healthcare monitoring solutions for soldier using the internet of things with distributed computing. Sustainable Computing: Informatics and Systems, 26; June 2020. Vol.
[3] Dariusz Mrozek "A review of Cloud computing technologies for comprehensive microRNA analyses. Comput Biol Chem Oct. 2020;88.
[4] Gireesha Obulaporam, Somu Nivethitha, Krithivasan Kannan, Shankar Sriram VS. IIVIFS-WASPAS: An integrated Multi-Criteria Decision-Making perspective for cloud service provider selection. Fut Gen Comp Syst Feb. 2020;103:91–110.
[5] Lang Michael, Wiesche Manuel, Krcmar Helmut. Criteria for selecting cloud service providers: a delphi study of quality-of-service attributes. Information & Management Sep. 2018;55(6):746–58.
[6] Harikrishna P, Amuthan A. A survey of testing as a service in cloud computing. In: International Conference on Computer Communication and Informatics (ICCCI); 2016. p. 1–5. Jan. 2016.
[7] Hamad RMH, Al Fayoumi M. Modernization of a Classical Data Center (CDC) vs. adoption in cloud computing calculate total cost of ownership for both cloud and CDC - Jordanian case study. In: 2018 International Arab Conference on Information Technology (ACIT); Mar. 2018.
[8] Mahmood Z. Cloud computing: characteristics and deployment approaches. In: 2011 IEEE 11th International Conference on Computer and Information Technology, Pafos; 2011.
[9] Song Chi-hoon, Kim Sang Woo, Sohn Young-woo. Acceptance of public cloud storage services in South Korea: A multi-group analysis. Int J Inf Manage Apr. 2020;51.
[10] Farrukh Nadeem, Rizwan Qaiser, "An early evaluation and comparison of three private cloud computing software platforms," Vol. 30, Pp. 639-654, May 2015.
[11] Park Joonseok, Kim Ungsoo, Yun Donggyu, Yeom Keunhyuk. Approach for selecting and integrating cloud services to construct hybrid cloud. Grid Computing. May 2020.
[12] Asaka RA, Mendes GHS, Ganga GMD. Factors influencing customer satisfaction in software as a service (SaaS): proposal of a system of performance indicators. IEEE Lat Am Trans Jul. 2017;15(8):1536–41.
[13] Linthicum DS. PaaS death watch? IEEE Cloud Comput Jan.-Feb. 2017;4(1):6–9.
[14] Tsai Wen-Lung. Constructing assessment indicators for enterprises employing cloud IaaS. Asia Pacif Manage Rev Aug. 2020.
[15] Pillutla Harikrishna, A Amuthan,"SDN-based DDoS attack mitigation scheme using convolution recursively enhanced self organizing maps," Vol. 45, May 2020.
[16] Pillutla H, Arjunan A. Fuzzy self organizing maps-based DDoS mitigation mechanism for software defined networking in cloud computing. J Ambient Intell Hum Comput 1559 Apr. 2019;10(4):1547.
[17] Krancher, O., & Luther, P. (2015). Software development in the cloud: exploring the affordances of platform-as-a-service.
[18] Kolb L, Thor A, Rahm E. Block-based load balancing for entity resolution with MapReduce. In: International Conference on Information and Knowledge Management (CIKM); 2011. p. 2397–400.

[19] Bok K, Hwang J, Jongtae Lim J, Kim Y, Yoo J. An efficient MapReduce scheduling scheme for processing large multimedia data. Multimed Tools Appl 2016:1–24.
[20] Ghoneem M, Kulkarni L. An adaptive MapReduce scheduler for scalable heterogeneous systems. In: Proceeding of the International Conference on Data Engineering and Communication Technology; 2016. p. 603–6011.
[21] Yakhchi M, Ghafari SM, Yakhchi S, Fazeliy M, Patooghi A. Proposing a LoadBalancing method based on cuckoo optimization algorithm for energy management in cloud computing infrastructures. In: Published In: Proceedings of the 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO); 2015.
[22] Babu LDD, Krishna PV. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Appl Soft Comput 2013;13(5):2292–303.
[23] Nishant K, Sharma P, Krishna V, Gupta C, Singh KP, Nitin N, Rastogi R. Load balancing of nodes in cloud using ant colony optimization. In: Proceedings of the 14th International Conference on Modelling and Simulation; 2012. p. 3–8.
[24] Singh P, Baaga P, Gupta S. Assorted load-balancing algorithms in cloud computing: a survey. Int J Comput Appl 2016;(7):143.
[25] Tasquier L. Agent based load-balancer for multi-cloud environments. Columbia Int Publ J Cloud Comput Res 2015;1(1):35–49.
[26] Chien NK, Son NH. Load-balancing algorithm based on estimating finish time of services in cloud computing. Int Conf Adv Commut Tech (ICACT) 2016:228–33.
[27] Sarood O, Gupta A, Kale LV. Cloud friendly load balancing for HPC applications. In: Preliminary Work. International Conference on Parallel Processing Workshops; 2012. p. 200–5.
[28] Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, &Buyya R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw Pract Exp 2011;41(1):23–50.
[29] Buyya R, Ranjan R, &Calheiros RN. Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities. In: 2009 international conference on high performance computing & simulation. IEEE; 2009. p. 1–11.
[30] Kumar R, &Sahoo G. Cloud computing simulation using CloudSim. arXiv preprint 2014. *arXiv:1403.3253*.
[31] Wang GG, Deb S, Cui Z. Monarch butterfly optimization. Neur Comput Appl 2019; 31(7):1995–2014.
[32] Feng Y, Deb S, Wang GG, Alavi AH. Monarch butterfly optimization: A comprehensive review. Expert Syst Appl 2020:114418.
[33] Faris H, Aljarah I, Mirjalili S. Improved monarch butterfly optimization for unconstrained global search and neural network training. Appl Intellig 2018;48(2): 445–64.
[34] Ghetas M. Learning-based monarch butterfly optimization algorithm for solving numerical optimization problems. Neur Comput Appl 2022;34(5):3939–57. https://doi.org/10.1007/s00521-021-06654-8.
[35] Yazdani S, Hadavandi E, Mirzaei M. CCMBO: A covariance-based clustered monarch butterfly algorithm for optimization problems. Memet Comput 2022. https://doi.org/10.1007/s12293-022-00359-8.
[36] Mohanty S, Patra PK, Ray M, Mohapatra S. A novel meta-heuristic approach for load balancing in cloud computing. Research Anthology on Architectures, Frameworks, and Integration Strategies for Distributed and Cloud Computing. 2021. p. 504–26. https://doi.org/10.4018/978-1-7998-5339-8.ch023.
[37] Dam S, Mandal G, Dasgupta K, Dutta P. An ant-colony-Based meta-heuristic approach for load balancing in cloud computing. Research Anthology on Architectures, Frameworks, and Integration Strategies for Distributed and Cloud Computing. 2021. p. 873–903. https://doi.org/10.4018/978-1-7998-5339-8.ch041.
[38] Annie Poornima Princess G, Radhamani AS. A hybrid meta-heuristic for optimal load balancing in cloud computing. J Grid Comput 2021;19(2). https://doi.org/10.1007/s10723-021-09560-4.