Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2022.025137
Article

Tech Science Press

Evolutionary Algorithm Based Adaptive Load Balancing (EA-ALB) in Cloud Computing Framework

J. Noorul Ameen[1,*] and S. Jabeen Begum[2]

[1]Department of Computer Science and Engineering, E.G.S Pillay Engineering College, Nagapattinam, 611002, India
[2]Department of Computer Science and Engineering, Velalar College of Engineering and Technology, Erode, 638012, India
*Corresponding Author: J. Noorul Ameen. Email: noornilo@gmail.com
Received: 13 November 2021; Accepted: 25 January 2022

Abstract: In the present decade, the development of cloud computing framework is witnessed for providing computational resources by dynamic service providing methods. There are many problems in load balancing in cloud, when there is a huge demand for resources. The objective of load balancing is to equilibrate the cloud server computations for avoiding overloading problems. On addressing the issue, this paper develops a new model called Evolutionary Algorithm based Adaptive Load Balancing (EA-ALB) for enhancing the efficacy and user satisfaction of cloud services. Efficient Scheduling Scheme for the virtual machines using machine learning algorithm is proposed in this work. Initially, process of K-means clustering is used for computing optimal min-max rates and then, local search capability for solving the load balancing problems in cloud model is determined with the incorporation of Evolutionary Algorithm. The results show that the proposed model achieves better results in terms of load balancing factors, Virtual Machine (VM) migration, energy consumption and so on, when compared to the existing model.

Keywords: Evolutionary algorithm; load balancing; cloud computing; virtual machine (VM); clustering; load estimation

1 Introduction

In the present scenario, cloud computing is very much popular due to the capability to offer seamless computing services with the on-demand based model [1]. The cloud models provide resources to the physical machines in the form of VMs, based on the user requirements. Each virtual machine executes its own OS and obtains their resources from their host physical machine. Moreover, the services are provided based on the Service Level Agreement (SLA) with the consumers. In the process of resource provision, there may cause SLA violations and may decrease the model effectiveness [2].

For handling this problem, it is significant for the cloud providers to use the cloud resources effectively. For attaining that, load balancing operations are performed in cloud by migrating virtual machines from overloaded physical machines to idle machine. The load balancing models [3–5] combines the varied resource utilizations by selecting virtual machines for migrations and determines the appropriate host physical machines. In this process, each resource is allotted with some weight for determining the load rate of physical machines and their capacities, the process of VM migration is carried out accordingly. By allocating varied resources with predefined weight rates, the models ignore the distinctive cloud feature, may cause, time complexities and over resource consumption in the physical machines. Moreover, the cloud virtual machines use varied resources to provide different services with varied resource intensities. As the cloud jobs are varied from consumers to consumers, and also varied based on time, the over utilization problem in the physical machine also varied on time.

Fig. 1 clearly depicts the sample scenario, which has four physical machines (PM1, PM2, PM3 and PM4). PM4 is overloaded with 3 virtual machines. Since the CPU is overloaded, based on the resource intensity measures, virtual machine (VM1) is considered to be the best to migrate, since, it shows higher intensity. PM1 is considered as the best host for VM1. Moreover, the main objective of the proposed model is to effectively reduce the migration numbers in the process of load balancing. Additionally, cost effectiveness also considered the service time, bandwidth and so on [6,7]. Typically, load balancing is a method to provide balanced workload between the cloud servers. Moreover, load balancing technique involves in optimizing the resource usage, cost effectiveness, increase throughput and provide cost effectiveness. The main objective of the proposed work is to provide cost and time effectiveness with the load balancing model between virtual machines. Based on the features of green scheduling, this work analyzed the load balancing methods of cloud framework based on,

i) Server Workload Forecasting
ii) Selection of VMs
iii) Selection of Target Server

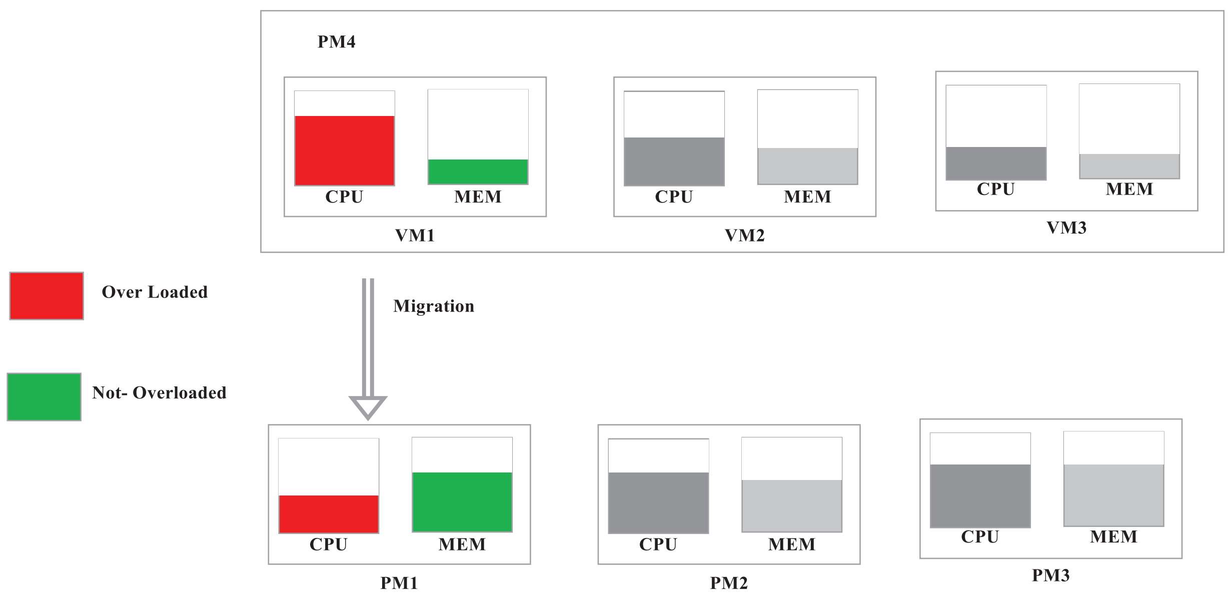

Figure 1: Virtual machine migration in cloud model

And the contributions of the proposed Evolutionary Algorithm based Adaptive Load Balancing (EA-ALB) are listed as follows**:**

i) The objective of the work is to determine the accurate CPU utilization of Cloud servers based on the aforementioned factors
ii) Incorporated K-Means Clustering (KMC) for finding the VMs with minimal migration, performance interferences and traffic.

iii) The Local Search Capability in enhanced with the Modified Evolutionary Algorithm (MEA). And the VM determined by KMC is migrated to the destination server to provide efficient load balancing.

The remainder of this work is organized as follows, Section 2 explains about the related works developed previously for solving load balancing problems in cloud. The complete work process with the flow diagram is explained in Section 3. The results and discussions with comparisons are provided in Section 4. Finally, the work is concluded in the Section 5 with some ideas for future work.

2 Related Works

Myriad works are proposed in recent times for handling cloud resources effectively by performing dynamic load balancing. The authors of [8] provided the basics of cloud computing, components, features, benefits and disadvantages. Additionally, the work discussed about the conceits of virtualization, cloud services and cloud security methods are also discussed in the work. In cloud framework, the load balancing techniques are performed as [9–11]

i) Centralized Load Balancing
ii) Distributed Load Balancing

In Centralized Load Balancing, the central node involves in the process of resource allocation and de-allocation. On the other hand, in distributed load balancing, multiple machines act as the coordinator and perform the process of resource allocation. Furthermore, several scheduling methodologies such as, First Come First Serve (FCFS), Round Robin and other load balancing techniques such as Ant Colony Optimization (ACO), Max-Min, etc., are involved in solving the problems on cloud resource management and provisioning [12,13].

Load balancing model provided in [14] determines to maximize the load balancing across the physical machines. Load memory rate is defined, when the CPU load is partitioned by their memory utilization to evaluate its resource utilization. And equal weights are assigned for resources, which may be inappropriate for varied time slots in each physical machine. In [15] the authors have discussed about the related works that are derived for managing the virtual machines in an efficient manner. An efficient virtual machine placement methods has been proposed in [16] for cost effectiveness in cloud data center. In [17] the performances of virtual machines are considered for taking the migration decision and resource provisioning.

Load balancing methodology for distributed cloud models is provided in [18] by moving the files to light servers. Moreover, PROTEUS is the technique developed in [19] for bandwidth allocation for cost effectiveness in cloud named FairCloud. Nevertheless, the model concentrated only on bandwidth**.** So, it is not efficient for processing load balancing in physical machines. The authors of [20] defined a migration process of virtual machine sample as a task set that execute at the PM of sender and receiver to evaluate the migration time and the resource utilization. Some models provided methodologies to deal load balancing on single resources as [21] for storage based resources and bandwidth based resource management in [22,23]. For enhancing the network security, traffic pattern based virtual machine migration model has been proposed in [24]. Further, in [25] AppAware model has been proposed for inter virtual machine dependencies and the primary factors of network topology to replace machines.

An algorithm for dynamic resource provisioning in data center for efficient resource utilization is proposed in [26]. In the model, the idle server is turned off for resource efficiency. In [27] for attaining throughput optimization, a cloud resource allocation model has been proposed. In the model, stochastic model of cloud cluster, where the tasks are allotted based on the virtual machine requests. But, the model is minimal in throughput, which used Best-Fit algorithm for scheduling. Cloudscale [28] model has been proposed for providing online based resource provisioning to accomplish dynamic resource allocation.

The review works in [29,30] provides valuable source of data on load balancing methods and materials in cloud computing environment.

3 Proposed Model

In the cloud model, the load balancing of servers is processed based on the number of Virtual Machines, VM Migration, memory, traffic flow and the server capacity. The server load condition is non-linear and periodic to certain level. In the proposed model, K-means clustering algorithm which is integrated with evolutionary algorithm for efficient load balancing between machines in cloud environment. In typical KMC algorithm, the following issues are noted.

i) The process of k-value selection using KMC is hard to determine
ii) In clustering, time and iterations are increased because of various reasons.
iii) When there is a huge dataset, time complexity may cause

Considering the issues, this paper derives an efficient KMC based on the cluster-center and the k-value for determining the appropriate min-max.

3.1 Implementation of KMC in Proposed Model

The operations of KMC in determining the Min-Max includes k-value selection and Cluster-Center (CC) selection.

3.1.1 K-Value Selection

In the proposed work, the VMs in the cloud model is categorized based on three factors, such as,

i) Load of the Server Machine
ii) VM migration cost
iii) Performance Interference

Moreover, the virtual machines clustering are provided as, three dimensional structure. Cost can be further considered as, low and high and further, the performance also considered as low and high, respectively. The server load is noted as, over load, under load and mediate load and their corresponding descriptions are given in Tab. 1.

Table 1: VM categorization

VM samples	Server load	Cost	Performance interference
1	Under	Low	Low
2	Mediate	Low	Low
3	Over	Low	Low
4	Under	Low	High
5	Mediate	Low	High
6	Over	Low	High
7	Under	High	Low
8	Mediate	High	Low
9	Over	High	Low
10	Under	High	High
11	Mediate	High	High
12	Over	High	High

3.1.2 Process of CC Selection

In KMC, initial cluster center (CC) is selected in random manner from the k-number of samples. When the selection process is instable, the derived solution is not optimal. Hence, the proposed model derives the optimal solution by CC selection to reduce the iteration numbers by increasing the distance between the initial CC. In the process of min-max selection of CC, initial CC is selected in random manner and given as CC_0. Following, the distance between CC_0 and each other sample point is calculated, in which the sample at shortest distance is taken as CC_1 and the sample at longer distance is noted as CC_2. The algorithm is presented in Tab. 2 for finding optimal solution with KMC.

Table 2: Algorithm for optimal solution using KMC in EA-ALB

Input: Samples $S = \{y_1, y_2, \ldots y_n\}$ and 'k' be the cluster num

Output: Initial $CC = \{cc_0, cc_1, \ldots cc_{k-1}\}$

1. Begin
2. Max_distance = 0
3. for i = 0, i < n, i++
4. for j = 0, i < n, i++
5. Calculate

$$\text{Distance}(i, j) = \|y_i - y_j\| \tag{1}$$

6. if $\text{Distance}(i, j) > \mathit{Max\_distance}$, then #n
7. Max_distance = Distance(i, j), $cc_0 = y_i$, $cc_1 = y_j$
8. end if
9. end for
10. end for
11. for $j = 1$, $j < k - 1$, j++
12. for i = 0, i < n, i++
13. $\text{Distance}_{(i,\,0)} = \|y_i - cc_0\|$, $\text{Distance}_{(i,\,1)} = \|y_i - cc_1\|$, …, $\text{Distance}_{(i,\,j)} = \|y_j - cc_j\|$
14. End for
15. $cc_{j+1} = \max[\min(\text{Distance}_{(i,\,0)},\ \ \text{Distance}_{(i,\,1)},\ \ \ldots \text{Distance}_{(i,\,j-1)})]$
16. End for
17. End

3.2 Adaptive Load Balancing Process

In the proposed model, the conventional differential evolution model is enhanced for improving the search ability. The process is employed to VM migration to make it more effective with respect to the aforementioned factors along with energy efficiency. For deriving optimal solutions, the fitness function is derived based on the following steps.

i) The traffic flow generated in the process of VM migration is based on the routing and memory. Here, the fitness function (FF) based on the traffic flow is given as,

$$\text{traffic flow}_{FF} = \sum_{i=1}^{n} dt_i \times len_{r(i)} \tag{2}$$

where, 'dt_i' denotes the size of data transmission of VM, during migration, '$len_{r(i)}$' denotes the length of routes between VMs based on their topology, between the source and target and the formula is given as,

$$len_{r(i)} = \left\{ \begin{cases} 3 \times 2, & \text{route link through core VM} \\ 2 \times 2, & \text{route link through aggregate VM} \\ 1 \times 2, & \text{route link through edge VM} \end{cases} \right\} \tag{3}$$

ii) Secondly, the FF based migration cost is derived based on the memory of the machine and the network bandwidth. The optimal solution is considered one which derives with minimal migration cost which is given as,

$$\text{Migration Cost (MC)} = \sum_{i=1}^{n} 0.1 \times \sum_{t_0}^{t_0 + t_{VM_i}} CPU_{VM_i} \tag{4}$$

where, 't_0' denotes the initiation time of VM migration, 't_{VM_i}' represents the total VM time and 'CPU_{VM_i}' denotes the CPU utilization of machines. And, the formula is given as,

$$t_{VM_i} = \frac{Mem_{VM_i}}{BW_{VM_i}} \tag{5}$$

where, 't_{VM_i}' denotes the time taken for migration, 'Mem_{VM_i}' represents the memory size and, the network bandwidth given as BW_{VM_i}.

iii) Further, the FF is computed based on the performance interference after the completion of VM migration. And the determined value is considered to be minimal, which is calculated as,

$$PI_{FF} = 1 - r^{\frac{T_i^{VM} - T_i}{T_i}} \tag{6}$$

From the above equation, 'T_i' denotes the running time of VM of 'i' th server.

iv) Energy Efficiency based FF derivation is processed in the fourth step, where optimal solution is considered as, $EE_{FF} = \text{Min} \ (EE_i)$ and the computation is presented below.

$$EE_i = \sum_{t(i)=1}^{t(n)} P(v_i(t_j)) \tag{7}$$

Here '$v_i(t_j)$' denotes the CPU utilization and the power consumption is given as, '$P(v_i(t_j))$' and the formula is given below.

$$P_i(v) = r_i \times P_i^{maxi} + (1 - r_i) \times P_i^{maxi} \times v_i \tag{8}$$

Hence, the energy efficient optimized solution is derived with the following formula, presented in (9).

$$OF = m_1 EE_{FF} + m_2 traffic\ \ flow_{FF} + m_3\ \ MC + m_4 PI_{FF} \tag{9}$$

And in (9) m_1, m_2, m_3 and m_4 are the balancing parameters on each derivation and their summation results unit value. Those factors can be adjusted to impact the other factors in determining the fitness function.

3.3 Incorporation of EA in Cloud Resource Allocation

In this process, the population_size is defined as, 'M' and the number of VM migrations is given as 'N' and the servers are given as 'S' and the inbetween links are in 'l' length. The placement of VM is provided as [1, S] with the maximal number of 'r' iterations. Further, the factor of mutation rate is given as, $\delta \in [0, 2]$ and the factor for crossover probability is provided as $C_p \in [0, 1]$. Hence, the 1st generation of ith VM is computed as,

$$x_r(0) = (x^r_{1j}(0),\ x^r_{2j}(0),\ \ldots,\ x^r_{ij}(0),\ \ldots,\ x^r_{mj}(0)) \tag{10}$$

In the above equation, $x_r(0)$, $(r = 1, 2, \ldots m)$, represented that k^{th} VM of 0^{th} generation is required to be migrated and $x^r_{1j}(0)$, $((i = 1,\ 2\ldots,\ m;\ j = 1,\ 2,\ \ldots r)$ denotes the migration of ith VM is to the j^{th} placement. And the computation is given as,

$$x = x_{j-min} + random(0,\ 1) \times (x_{j-max} - x_{j-min}) \tag{11}$$

From the above equation, the minimal and maximal vectors rates are provided with the mapping of [1, S]. Further, based on the evolutionary algorithm, the different VM patterns with randomly generated population 'e' are given as $x_{t_1}(e)$, $x_{t_2}(e)$ and $x_{t_3}(e)$. Here, the mutation operations are processed to create new population and the differential scaling factor is derived as,

$$scaling\ factor = \delta(x_{t_1}(e) - x_{t_2}(e)) \tag{12}$$

In the above equation, $(x_{t_1}(e) - x_{t_2}(e))$ denotes the differential scaling factor and considering the weights, the VM migration is stated as,

$$u_t(e+1) = x_{t_3} + scaling\ factor \tag{13}$$

Here, the newly produced individual is given as, '$u_t(e+1)$'. Further, the crossover operations are introduced to enhance the population diversity. Therefore, the new generations $v_t(e+1)$ and the older ones $x_{t_1}(e)$ are muted together to frame the new individuals as,

$$v_t(e+1) = (v^r_{1j}(e+1),\ v^r_{2j}(e+1),\ \ldots,\ v^r_{ij}(e+1),\ \ldots,\ v^r_{mj}(e+1)) \tag{14}$$

The formula for cross over is given as,

$$v^r_{ij}(e+1) = \begin{cases} u^r_{ij}(e+1), & random\ (i) \le C_p = random\ (r) \\ x^r_{ij}(e), & random\ (i) > C_p \ne random\ (r) \end{cases} \tag{15}$$

Here, *random* (i) is denoted as the random number between 0 to 1 and *random* $(r) \in [1, M]$. Then, new cycle is started for the selection function and the FF of the cross over results $v_t(e+1)$ is compared with the older individual $x_{t_1}(e)$ and the FF is given as,

$$x_t(e+1) = \begin{cases} v_t(e+1), & FF(v_t(e+1)) > FF(x_t(e)) \\ (x_t(e), & Others \end{cases} \tag{16}$$

In the above equation, *FF*(x) denotes the fitness function of the individual, which is newly generated. The computed fitness values are compared and the better solution is provided for the iteration process to find next generation. The algorithm is presented in Tab. 3. From the above experimentation, it is determined that the convergence speed of the proposed model is efficient and faster than the traditional models. Moreover, in the process of cross-over mutation, the algorithm is enhanced with adding a local search event and the algorithm is provided in Tab. 4. The above operation in local search determines the optimal situation, which may result in migrating, when overload happens. Based on the results of the algorithm, the resources are allotted to the tasks.

Table 3: Algorithm for EA in cloud resource allocation

Input: population_size is M, number of VM migrations is ‘N, server is, ‘S’ and maximal iterations ‘r’, factor of mutation rate $\delta \in [0, 2]$ and factor for crossover probability $C_p \in [0, 1]$
Output: Optimal Solution set for VM migration
1. Begin
2. No .of iterations, $e = 0$,
3. Declare Positive Integer $i = 1; j = 1$
4. for $i = 0; i < S$; i++
5. for $j = 0; j < N$;j++
6. $x = x_{j-min} + random(0, 1) \times (x_{j-max} - x_{j-min})$
7. End For
8. End For
9. while $(e \leq r)$
10. Selection of random items $x_{t_1}(e)$, $x_{t_2}(e)$ and $x_{t_3}(e)$ as weight ‘δ’
11. Generation of new individuals with Mutation
12. new individuals, $v_t(e+1) = (v^r_{1j}(e+1) \quad v^r_{2j}(e+1) \quad \ldots \quad v^r_{ij}(e+1) \quad \ldots \quad v^r_{mj}(e+1))$
13. if FF $(v_t(e+1)) > FF(x_{t_1}(e))$
14. $x_{t_1}(e+1) = v_t(e+1)$
15. else
16. $x_{t_1}(e+1) = x_{t_1}(e)$
17. End if
18. $e = e + 1$
19. Local Search Optimization is presented in Tab. 3
20. End While
21. Return Optimal Solution Set for VM migration
22. End

Table 4: Local search optimization in VM migration in cloud

Input: No. Of local iterations 'L', local search event 'α', population_size M, population factor $x_i(e)$, size_reduction parameter β
Output: better VM for next generation
Begin
While $(i < M)$
While $(i < L)$
Random Variable $\Delta x,\ \Delta x \epsilon [-\alpha_t,\ \alpha_t]$
$x_{new}(e) = x_i(e) + \Delta x$
If $(G(x_{new}(e)) = \text{true and } G(x_i(e)) = \text{False})$
$x_i(e) = x_{new}(e)$
End if
If $(G(x_{new}(e)) = \text{true and } G(x_i(e)) = \text{True})$
If $(FF\ (x_i(e)) < FF(x_{new}(e)))$
$x_i(e) = x_{new}(e)$
End if
End if
$\alpha_t = \alpha_t * \beta$
$r = r + 1$
end while
$i = i + 1$
end while
End

4 Results and Discussions

This section presents the results and discussions to prove the efficacy of the proposed model. The model is evaluated using the simulation software called CloudSim. And the results are compared with the existing models such as, First Come First Serve (FCFS), Ant Colony Optimization (ACO) and PROTEUS. The simulation parameters and the domain values are presented in the following Tab. 5. Moreover, the evaluations are carried out based on the factors such as, cost effectiveness, traffic flow and CPU utilization for measuring the overall model efficiency.

Fig. 2 presents the results for migration cost evaluations for cloud resource allocations, their corresponding values are given in Tab. 6, in which costs are denoted with units and the proposed model is cost effective than the compared works. The average migration cost of the proposed EA-ALB is 554.42, which is minimal than other models. The Fig. 3 depicts the performance interference based results on model evaluations. The results are carried out based on the execution time. From the results, it is evidenced that the model is efficient than the other models and the obtained results are presented in Tab. 7.

Table 5: Simulation parameters

Parameters	Values
Server based parameters	
Server MIPS	1.8 to 3.0 GHz
Memory size	4–16 GB
Bandwidth	1000 Mbit/s
Hard disk size	50–320 GB
VM based parameters	
VM MIPS	0.5 to 2.5 GHz
Memory size	613–1740 MB
Bandwidth	100 Mbit/s

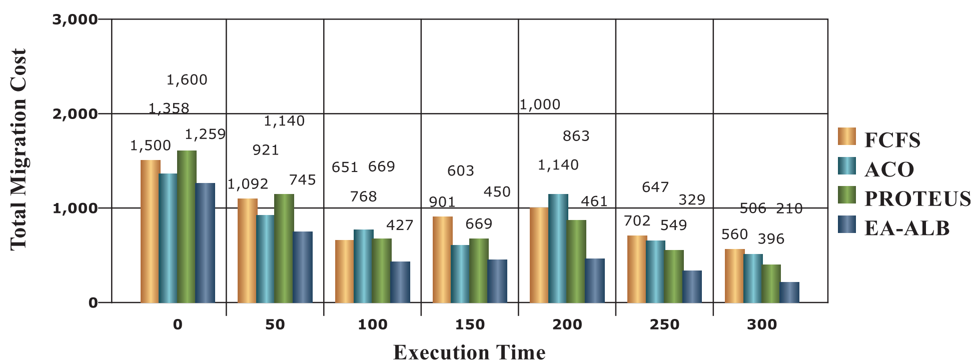

Figure 2: Migration cost *vs.* execution time

Table 6: Results for migration cost

Models	0	50	100	150	200	250	300
FCFS	1,500	1,092	651	901	1,000	702	560
ACO	1,358	921	768	603	1,140	647	506
PROTEUS	1,600	1,140	669	669	863	549	396
EA-ALB	1,259	745	427	450	461	329	210

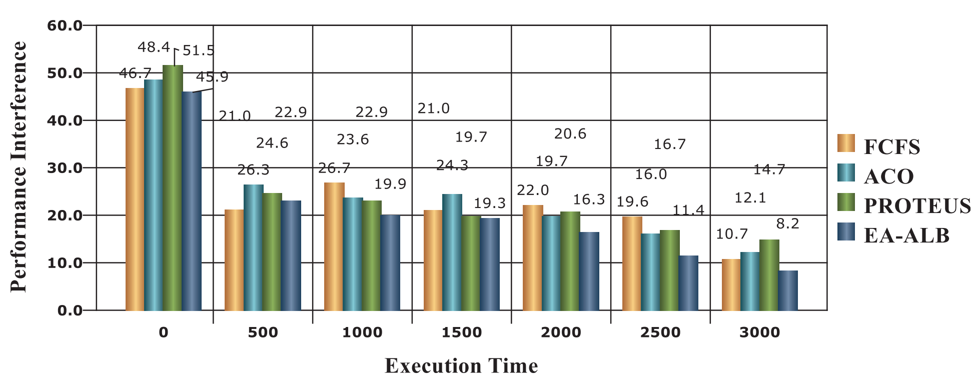

Figure 3: Execution time *vs.* performance interference

Table 7: Results for performance interference

Models	0	500	1000	1500	2000	2500	3000
FCFS	46.7	21.0	26.7	21.0	22.0	19.6	10.7
ACO	48.4	26.3	23.6	24.3	19.7	16.0	12.1
PROTEUS	51.5	25.0	23.0	19.7	21.0	16.7	15.0
EA-ALB	46.0	23.0	20.0	19.0	16.0	11.0	8.0

Traffic Flow based results are provided in Tab. 8 and their relevant comparison graph is given in Fig. 4. Further, Tab. 9 contains the values obtained for VM migration based analysis and their graph comparisons are given in Fig. 5. And the data center utilization based results are presented in Fig. 6 and the obtained values are given in Tab. 10. The evaluation for traffic flow is processed with respect to the execution time and the utilization based analysis is performed with number of tasks. In analyzing, VM migrations, it is the significant factor in evaluating the resource allocation model. The best solution is determined with the efficient incorporation of KMC with EA in the proposed work that effectively performs VM migration, by which load balancing is achieved.

Table 8: Traffic flow based results

Models	0	500	1000	1500	2000	2500	3000
FCFS	522	916	1,770	1,557	1,261	1,441	981
ACO	816	881	1,721	2,294	1,539	1,622	1,212
PROTEUS	522	1,031	1,999	2,361	1,721	1,409	1,490
EA-ALB	455	851	1,589	1,439	1,261	999	768

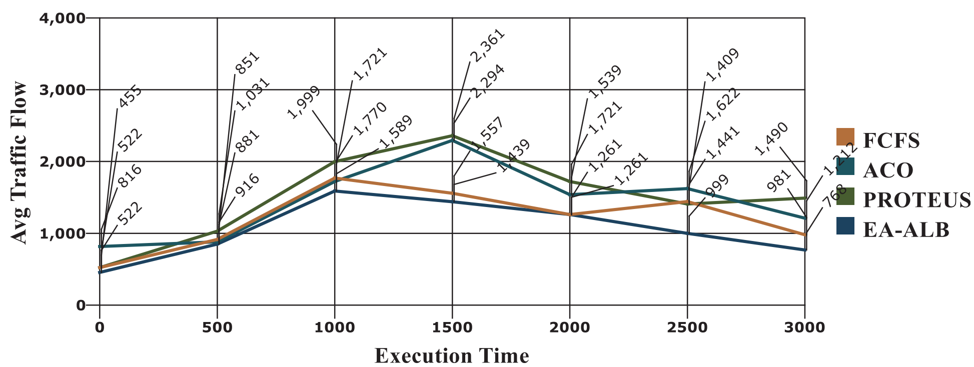

Figure 4: Average traffic data and execution time

Table 9: Average traffic flow based analysis

Models	0	50	100	150	200	250	300
FCFS	522	916	1,946	1,557	1,919	1,902	2,382
ACO	816	1,866	2,663	2,294	2,131	2,213	2,610
PROTEUS	522	1,031	2,459	2,796	2,358	2,461	3,001
EA-ALB	455	544	843	872	891	999	1,074

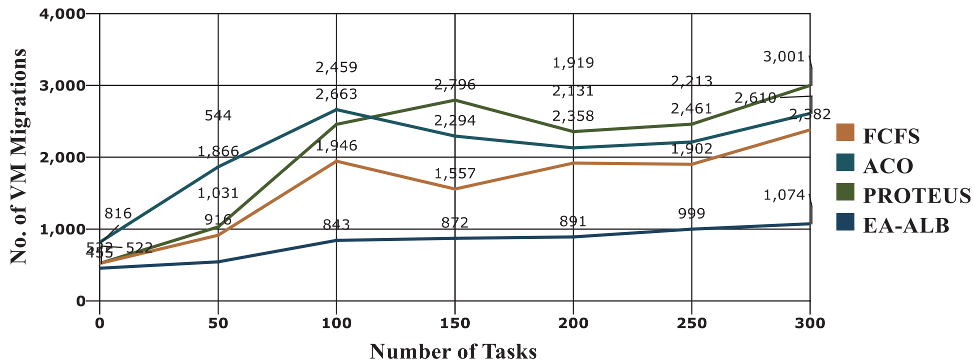

Figure 5: Comparisons for average time flow

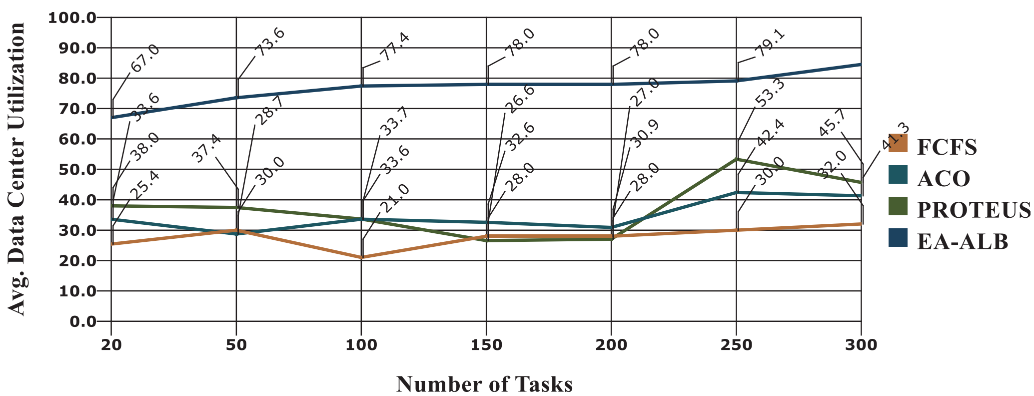

Figure 6: Average data center utilization based results

Table 10: CPU utilization

Models	20	50	100	150	200	250	300
FCFS	25.4	30.0	21.0	28.0	28.0	30.0	32.0
ACO	34.0	29.0	34.0	33.0	31.0	42.0	41.0
PROTEUS	38.0	37.0	34.0	27.0	27.0	53.0	46.0
EA-ALB	67.0	74.0	77.0	78.0	78.0	79.0	85.0

5 Conclusion and Future Work

For the process of efficient migration of virtual machines in Cloud computing process, this paper proposes a new model called EA-ALB. The model integrates the efficiency of KMC in determining best solution and the Evolutionary Algorithm for load balancing. The proposed model effectively predicts the resource utilization by machines, in which the min-max algorithm is used for finding the cluster centers. The model evaluation is carried out based on the factors such as cost effectiveness, CPU utilization, migration effectiveness and traffic flow. It is evidenced from the results that the proposed model achieves minimal cost, traffic flow and interference than other compared works. And the utilization is maximal, that is, the model effectively utilizes the machines about 95%, where load balancing is effectively achieved with the proposed model.

In Future, as the load balancing in cloud has a greater research scope, the potential applicability can be expanded for large scale cloud models. Methods can be developed to measure the algorithm's efficacy in applying it on a real life case to attain better results and routine.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] T. Wood, P. J. Shenoy, A. Venkataramani and M. S. Yousif, "Black box and gray box strategies for virtual machine migration," *Networked Systems Design and Implementation*, vol. 7, pp. 17–17, 2007.

[2] E. Arzuaga and D. R. Kaeli, "Quantifying load imbalance on virtualized enterprise servers," in *Int. Conf. on Performance Engineering -WOSP/SIPEW*, San Jose, California, USA, pp. 235–242, 2010.

[3] A. Singh, M. R. Korupolu and D. Mohapatra, "Server-storage virtualization: Integration and load balancing in data centers," in *The Int. Conf. for High Performance Computing Networking, Storage and Analysis-SC*, Austin, Texas, USA, pp. 53–59, 2008.

[4] G. Khanna, K. Beaty, G. Kar and A. Kochut, "Application performance management in virtualized server environments," in *Network Operations and Management Symp.*, Vancouver, Canada, pp. 373–381, 2006.

[5] M. Tarighi, S. A. Motamedi and S. Sharifian, "A new model for virtual machine migration in virtualized cluster server based on fuzzy decision making," *Journal of Telecommunications-arXiv:1002.3329*, vol. 1, no. 1, pp. 40–51, 2010.

[6] A. B. Nagarajan, F. Mueller, C. Engelmann and S. L. Scott, "Proactive fault tolerance for hpc with xen virtualization," in *Int. Conf. on Super Computing*, Seattle, Washington, USA, pp. 23–32, 2007.

[7] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul *et al.,* "Live migration of virtual machines," *Networked Systems Design and Implementation*, pp. 273–286, 2005.

[8] A. Nayyar, *Handbook of Cloud Computing: Basic to Advance Research on the Concepts and Design of Cloud Computing*, BPB Publications, India, 2019.

[9] R. Panwar and B. Mallick, "Load balancing in cloud computing using dynamic load management algorithm," in *Int. Conf. on Green Computing and Internet of Things*, Greater Noida, Delhi, India, pp. 773–778, 2015.

[10] G. Rastogi and R. Sushil, "Analytical literature survey on existing load balancing schemes in cloud computing," in *Int. Conf. on Green Computing and Internet of Things*, Greater Noida, Delhi, India, pp. 1506–1510, 2015.

[11] S. Aslam and M. A. Shah, "Load balancing algorithms in cloud computing: A survey of modern techniques," in *National Software Engineering Conf.*, Rawalpindi, pp. 30–35, 2015.

[12] F. Tang, L. T. Yang, C. Tang, J. Li and M. Guo, "A dynamical and load-balanced flow scheduling approach for big data centers in clouds," *IEEE Transactions on Cloud Computing*, vol. 6, no. 4, pp. 915–928, 2016.

[13] K. R. Babu and P. Samuel, "Enhanced bee colony algorithm for efficient load balancing and scheduling in cloud," in *Innovations in Bio-Inspired Computing and Applications*, Springer, Singapore, pp. 67–78, 2016.

[14] C. Tang, M. Steinder, M. Spreitzer and G. Pacifici, "A scalable application placement controller for enterprise data centers," in *Int. World Wide Web Conf. Committee*, pp. 331–340, 2007.

[15] F. Xu, F. Liu and H. Jin, "Managing performance overhead of virtual machines in cloud computing: A survey", state of art and future directions," in *Proc. of IEEE*, Banff, Alberta, Canada, pp. 11–31, 2014.

[16] X. Li, J. Wu, S. Tang and S. Lu, "Let's stay together: Towards traffic aware virtual machine placement in data centers," in *IEEE Int. Conf. on Computer Communications*, Toronto, Canada, pp. 1842–1850, 2014.

[17] F. Xu, F. Liu and H. Jin, "Heterogeneity and interference-aware virtual machine provisioning for predictable performance in the cloud," *IEEE Transactions on Computers*, vol. 118, pp. 1–14, 2016.

[18] D. Xie, N. Ding, Y. C. Hu and R. R. Kompella, "The only constant is change: Incorporating time-varying network reservations in data centers," *Special Interest Group on Data Communication*, vol. 42, no. 4, pp. 199–210, 2012.

[19] L. Popa, G. Kumar, M. Chowdhury, A. Krishnamurthy, S. Ratnasamy *et al.,* "Faircloud: Sharing the network in cloud computing," *Special Interest Group on Data Communication*, vol. 16, pp. 187–198, 2012.

[20] S. Lim, J. Huh, Y. Kim and C. R. Das, "Migration, assignment and scheduling of jobs in virtualized environment," in *HotCloud'11: Proc. of the 3rd USENIX Conf. on Hot Topics in Cloud Computing*, Texas, USA, vol. 10, pp. 1–5, 2011.

[21] H. Hsiao, H. Su, H. Shen and Y. Chao, "Load rebalancing for distributed file systems in clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 5, pp. 951–962, 2012.

[22] H. Ballani, P. Costa, T. Karagiannis and A. Rowstron, "Towards predictable data center networks.", *Special Interest Group on Data Communication*, vol. 41, no. 4, pp. 242–253, 2011.

[23] X. Meng, V. Pappas and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Int. Conf. on Computer Communications*, Shanghai, China, pp. 1154–1162, 2010.

[24] M. Lin, A. Wierman, L. L. H. Andrew and E. Thereska, "Dynamic right-sizing for power-proportional data centers," in *Int. Conf. on Computer Communications*, Shanghai, China, pp. 1098–1106, 2011.

[25] S. T. Maguluri, R. Srikant and L. Ying, "Stochastic models of load balancing and scheduling in cloud computing clusters," in *Int. Conf. on Computer Communications*, Shanghai, China, pp. 702–710, 2012.

[26] Z. Shen, S. Subbiah, X. Gu and J. Wilkes, "Cloudscale: Elastic resource scaling for multi-tenant cloud systems," in *ACM Symp. on Cloud Computing*, Cascais, Portugal, pp. 5–11, 2011.

[27] D. Abdul, K. Shafiq, N. Z. Jhanjhi and A. Abdullah, "Load balancing techniques in cloud computing environment: A review," *Journal of King Saud University-Computer and Information Sciences*, vol. 15, pp. 1319–1578, 2021.

[28] M. A. Shahid, N. Islam, M. Alam, M. M. Suud and S. Musa, "A comprehensive study of load balancing approaches in the cloud computing environment and a novel fault tolerance approach," *IEEE Access*, vol. 8, pp. 130500–130526, 2020.

[29] S. Afzal and G. Kavitha, "Load balancing in cloud computing–A hierarchical taxonomical classification," *Journal of Cloud Computing*, vol. 8, no. 15, pp. 5–24, 2019.

[30] A. Alkhatiba, A. Alsabbagh, R. Maraqa and S. Alzubi, "Load balancing techniques in cloud computing: Extensive review," *Advances in Science, Technology and Engineering Systems Journal*, vol. 6, no. 15, pp. 860–870, 2021.