Analysis of Cost-Aware Load Balancing Framework for Cloud Computing using Optimized Scheduling Algorithms

Navpreet Kaur Walia, Navdeep Kaur

Abstract*: With the increasing demand of the cloud, load on the cloud server increases and lead to high cost consumption for computations of the tasks. So, to deal with this challenge a lot of researchers develop load balancing algorithms, scheduling algorithms. The current existing algorithms handles the issues of load balancing but the cost of computation is not reduced till now. So, the main focus of this paper to propose a Cost-Aware load Balancing Framework for different existed optimized scheduling algorithms like GA, PBO, and ACO and proposed EPC Scheduling algorithm. EPC is enhanced PBO algorithm where combination of PBO and GA algorithm is used for enhancement. This combination helps to select the appropriate resources for the tasks for its execution and better allocation. In this framework tasks are divided into groups and accordingly resources will be provided to them which increases the utilization factor and reduces cost. For analysis of this proposed framework, homogeneous and heterogeneous cloud environments are generated using local cloud environment and analysis is done on the basis of load balancing factor and cost of computation where minimum 50 and maximum 500 jobs are allocated to 100 resources. Results of the proposed algorithm achieved better results along with the optimization algorithms.

***Index Terms*: Cloud Computing, Task Scheduling, Load Balancing, PBO, ACO, GA, Task Grouping, Cost.**

I. INTRODUCTION

The use of Cloud is increased day-by-day and this will increase load on the cloud servers and also increases cost so, for this an efficient cost-aware load mechanism is required which not only balance load on servers but also reduces cost by providing an efficient solution for jobs. Traditional load balancing mechanisms handles load with the help of schedulers but now-a-days there is a requirement that the input provided to the scheduler will also be optimized so that scheduler schedules that tasks as efficient as possible. Secondly, tasks are assigned dynamically to the servers as servers has different configuration which were not considered and some of the servers got overloaded. So, considering all these parameters this algorithm makes clusters of the tasks as per their requirement and also assigned tasks to the relevant resources that have configuration similar or nearly similar to the tasks which is going to execute. A number of algorithms were proposed to balance load on the basis of different mechanisms. For Instance, Volkova et al. [1] proposed a Throttled Load Balancing algorithm in which virtual machine indexes were maintained by Load Balancer along with the states that means whether the machine is busy or available. This will provide an efficient management of the jobs distributed to the cloud environment. Authors implemented this algorithm on 'Cloud Analyst' Simulator and set different environments for execution. Results achieved by this method were better than round robin and equally spread algorithm in terms of both Total Response Time and Data Center Processing Time. Different Soft Computing Technologies were also introduced in the field of cloud computing for Load Balancing like Fuzzy based Technology proposed by Velde and Rama [2]. As fuzzy is a rule based mechanism so different rules were generated with the objective to distribute equal load to achieve better performance. In their work, they combined round robin algorithm with fuzzy system and proposed Fuzzy based Round Robin. Simulation of this proposed technique was done using 30 machines. They compared performance of this proposed algorithm with the traditional Round Robin algorithm and proof that their proposed algorithm achieved better performance in terms of Data Center Processing Time and Overall Response Time. Another soft computing for the objective of load balancing was proposed by Mondal et al. [11] where they used stochastic hill climbing approach. This approach was used to for the allocation of resources as it maintains indexing of machines. Cloud analyst was used as a simulation tool for testing purpose and the performance of this were analyzed on the basis of response time. Results show the better performance of this proposed mechanism as compare to Round-Robin (RR) and First Come First Serve (FCFS). The other soft computing-based algorithms were proposed to efficiently balance load on cloud like artificial bee algorithm proposed by Yao and He [14], Bee-MMT by Ghafari et al. [15] and Ant Colony Optimization by Kumar et al. [16] and analyze performance of all these algorithms. Saeed et al. [3] proposed a novel architecture by integrating Cloud with Fog Environment. Fog computing is recently designed architecture for cloud which reduces load on cloud servers. With this integration, a new three layer architecture was formed where first layer contains clusters, second layer present fog and third layer is a connection between fog and cloud. In their work, they had taken the example of handling requests for the electricity and considering this request can be handled more efficiently through cloud computing. For the handling of request they proposed modified FCFS algorithm. They analyzed performance both at cluster and fog layer.

Revised Manuscript Received on April 09, 2019.

Navpreet Kaur Walia, Research Scholar, Department of Computer Science, Sri Guru Granth Sahib World University, Fatehgarh Sahib and Assistant Professor, Department of Computer Science, Chandigarh University, Gharaun (Punjab), India.

Navdeep Kaur, Professor Department of Computer Science, Sri Guru Granth Sahib World University, Fatehgarh Sahib (Punjab), India.

For cloud they compared three different algorithms and analyzed that the performance of FCFS is better than others in terms of response time. There are number of challenges which we are facing with cloud architecture and these challenges need to be cure because the use of cloud is increasing day-by-day. With this objective, Sharma and Nair [6] worked on two major issues where first issue is the distribution of the tasks and other is to resolve the complexities in the business services for cloud consumers. They added two layer priorities in their work, the first is technical layer priority (TP) and other is business layer priority (BP) and scheduled tasks accordingly. They analyzed performance in terms of waiting time and conclude that it was reduced.

Shi et al. [7] proposed a hybrid scheduling mechanism for load balancing in cloud environment. This hybrid mechanism works in two phases. In first phase static resource allocation were performed where as in second phase dynamic scheduling was implemented. This proposed hybrid load balancing algorithm was tested on live environment where it is applied to the land surface disaster and ecology model. This was implemented with 200 nodes and run on the IBM computer clusters having high performance. Experimental results show the better performance of this proposed mechanism. The other important factor which affects the performance of cloud is security which considered very less. But Gupta et al. [9] presented their approach for load balancing along the trust based approach to provide reliable data centers. In this trust factor was calculated on the basis of fault rate and is updated time to time. For analysis of this proposed mechanism, Qemo simulator was used and results were calculated. Sarood et al. [10] proposed a load balancing environment which is cloud friendly and were implemented this on High performance computing applications. In this approach, an object migration mechanism was used for parallel applications to achieve load balancing. The objective of this work not only reduces time but also to reduce energy consumption. They tested this proposed mechanism on different environments and achieved 50% improved performance. Similarly, an efficient approach for load balancing was proposed by Dhari and Arif [13] and named as LBDA (Load Balancing Decision Algorithm). In this approach, three different stages were there and these are calculation of the capacity of the virtual machine, time required for the execution of the task and third is distribution of the tasks. Results were calculated in terms of average make-span and average response time which were compared with the other existing algorithms like SJF, Round Robin and Max-Min. The performance of LBDA was better than these existing algorithms.

All the above defined algorithms and framework provides benefits to the cloud providers but still they required an efficient mechanism to reduce its cost of computation. To achieve this objective, a new cost-aware load balancing framework is proposed with efficient scheduling algorithms which not only balance load but also improve the performance of the scheduler and reduces the cost of computation. The next sections of this paper will discuss about this proposed framework and its analysis on homogeneous and heterogeneous cloud environment.

II. COST AWARE LOAD BALANCING FRAMEWORK

With the objective to reduce the cost of computation by balancing Load on Cloud, this work proposed an Efficient Algorithm for Cloud where load is balanced on the basis of job/task assigned to the Cloud by a user. This will help in assigning jobs to the particular VMs where these jobs can execute efficiently. For this objective, task grouping is performed in this work. It means tasks having similar requirement are clustered into one group and then mapping with the machines will take place. In this work, we divided jobs into three categories (a) Memory Intensive Jobs, (b) CPU Intensive Jobs and (c) Both CPU and Memory Intensive Jobs. Memory intensive Jobs are those jobs which need higher memory for execution. It means only that machines will be allocated to these jobs which have higher RAM or as per scheduler considering RAM parameter. CPU intensive jobs are those jobs which need higher processor for their execution and accordingly machine will be allocated to CPU intensive jobs. The third and last category is Both CPU and Memory Intensive where jobs are assigned to that machines that have high configuration because these jobs required both processor and memory.

In this framework when user assigned jobs to cloud portal then workload analyzer first analyze the type of job and their requirements. It means it checks whether the task is belongs to CPU-Intensive, Memory Intensive or both categories and accordingly analyze assign them to the cluster. Cloud is a large-scale network/server where more than one user can assign their tasks at the same time and all these tasks are handled by analyzer to identify the requirement of jobs and then all the similar jobs are grouped into one cluster and passed to scheduler for execution as shown in figure 1

In this, when user assign tasks to the cloud environment, then these tasks are further divided into different groups as per the requirement of the execution of the tasks. Some tasks need high memory (RAM) for their execution or a good processor or both. So, tasks are grouped together into these three groups as given in algorithm 1

Algorithm1: Job Grouping( )

Input: InputJob J_id, J_name, J_desc, R_inst,J_req; **Output:** VMs

JOB ← Set of Jobs

FOR (each job ϵ JOB) do

IF(J_req == CPUreq)

CPU-Intensive (Ji) ← J(i)

Order=3

ELSEIF(J_req == Memreq)

Mem-Intensive (Ji) ← J(i)

Order=1

ELSE

CM-Intensive (Ji) ←J(i)

Order=2

END OF IF

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

END OF IF
End of FOR

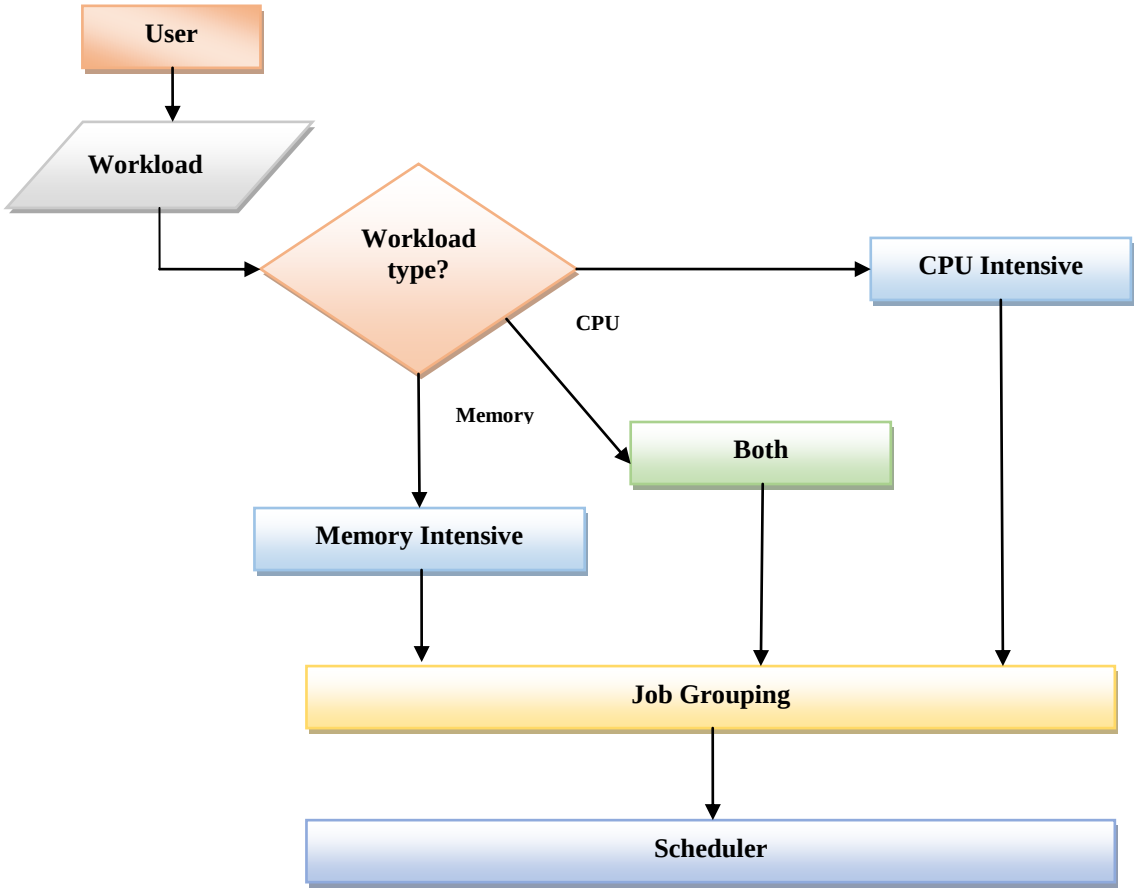


Figure 1: Task Grouping for Load Balancing

III. SIMULATION AND PERFORMANCE ANALYSIS

To analyze the performance of this Cost-Aware Load Balancing Framework, two types of Cloud Environments are simulated using .NET where web-based interface is used. These two types are: (a) Heterogeneous- where all the resources of the cloud have different configuration and, (b) Homogeneous- where same configuration resources are provided to Cloud Environment. This Cost Aware Load Balancing framework is assigned to both Homogeneous and Heterogeneous Environment with different scheduling mechanisms. The simulation setup for this framework is as shown in table 1.

Table 1: Simulation Parameters

Parameter	Value in Heterogeneous Environment	Value in Homogeneous Environment
Number of Resources	100	100
Temperature	Variable (Low, High, Very High)	Fixed (High)
RAM Size	Variable	Fixed
Number of Tasks	50, 100, 150, 200, 250, 300, 350, 400, 450, 500	50, 100, 150, 200, 250, 300, 350, 400, 450, 500
Machine Allocation	Preferable as requirement	Not Preferable
Optimized Scheduling Algorithm	GA, PBO, ACO, EPC	GA, PBO, ACO, EPC

In this work, experimentation is done using 50-500 jobs/tasks. As

per algorithm jobs have different requirement that are assigned randomly for experimentation. While implementing, number of jobs for all three categories which are randomly distributed is as given in table 2. These set of jobs/tasks are further tested on both Heterogeneous and Homogenous Environment.

Table 2: Number of Jobs in Different Categories

Total No. of Jobs	No. of Jobs		
	CPU-Intensive	Memory-Intensive	Both CPU & Memory-Intensive
50	17	17	16
100	31	36	33
150	48	59	43
200	59	59	82
250	86	83	81
300	106	87	107
350	120	98	132
400	139	116	145
450	152	132	166
500	173	145	182

A. Simulation in Heterogeneous Environment

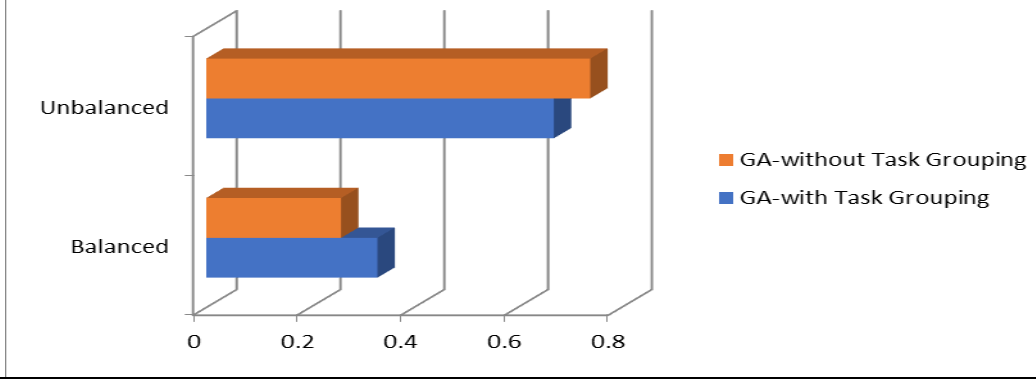
While implementing this proposed framework using above simulation parameters in Heterogeneous Environment on 100 resources then each resource generates different number of VMs on the basis of the configuration of the resources.

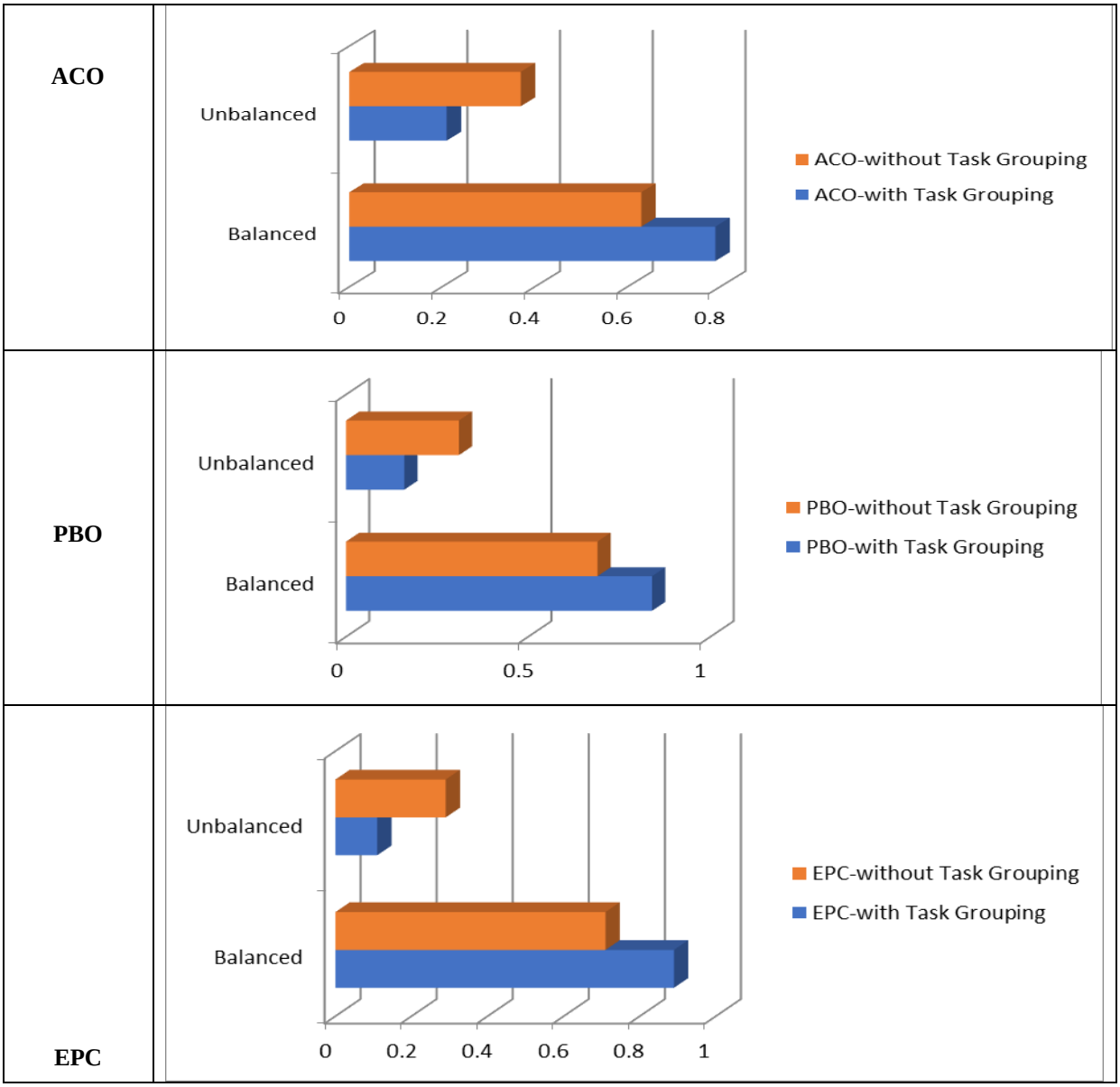
Performance Analysis:

To analyse performance of this proposed framework, different parameters are used. In this work, Load Balancing Factor and Cost of Computation is Calculated.

Load Balancing Factor is a very important factor to analyze the performance of the cloud. It also defines the load managed by the resources means number of jobs executed by the resources. While executing all the jobs the load is distributed to different resources and the Load Balancing factor of some existed scheduling algorithms like GA, PBO, ACO, and EPC with and without task grouping is as shown in Table 3.

Table 3: Load Balancing Factor

Scheduling Algorithm	Load Balancing Factor with and without Task Grouping
GA	



The above results show that there is more unbalanced load on cloud while task grouping based efficient mechanism was not applied in Cloud Framework with each GA, ACO PBO, and EPC based optimized scheduling algorithm. Here this performance factor is calculated when 500 jobs are executed.

The results show that with Task Grouping Mechanism the EPC Scheduling Algorithm performs better and provide more balanced environment as shown in Figure 2. It means the distribution of jobs is better with task grouping.

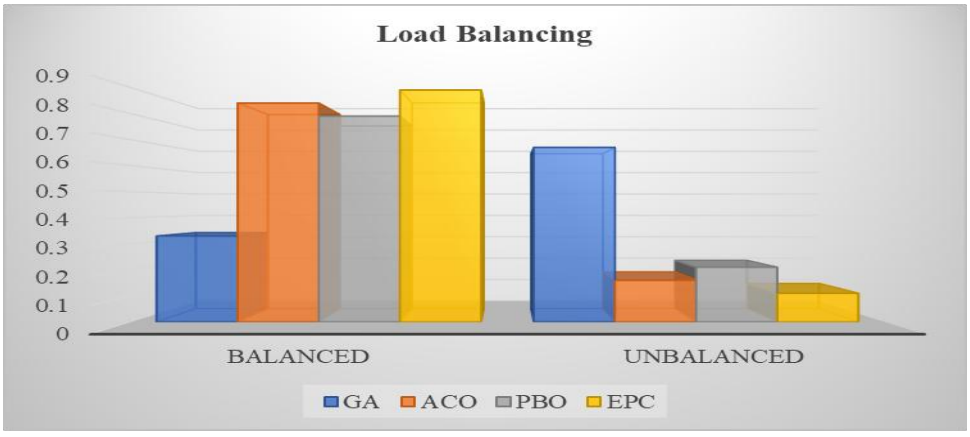


Figure 2: Load Balancing (Heterogeneous Environment)

Cost of Computation is another important performance metric to calculate or analyze the performance of cloud. This factor defines the use of Resources/VMs for execution of tasks and is calculated using following formula:

$$Cost\ of\ Computation = \left(Total\ No.of\ VMs\ used / Total\ No.of\ Resources\ used\right) * Total\ No.of\ Jobs$$

In this work, cost of computation is calculated for 50, 100, 150, 200, 250, 300, 350, 400, 450 and 500 jobs with different scheduling algorithms as shown in figure 3. The results show that the cost of computation in an efficient load balancing environment with EPC Algorithm are better as compare to other scheduling algorithms. This cost of computation is increased with the number of jobs. This cost of computation also affects on the performance of Cloud. It defines the use of resources for execution of the jobs and because here jobs are not directly assigned to resources it executed by VMs so this cost is depending on both resources and VMs as define in the formula given above:

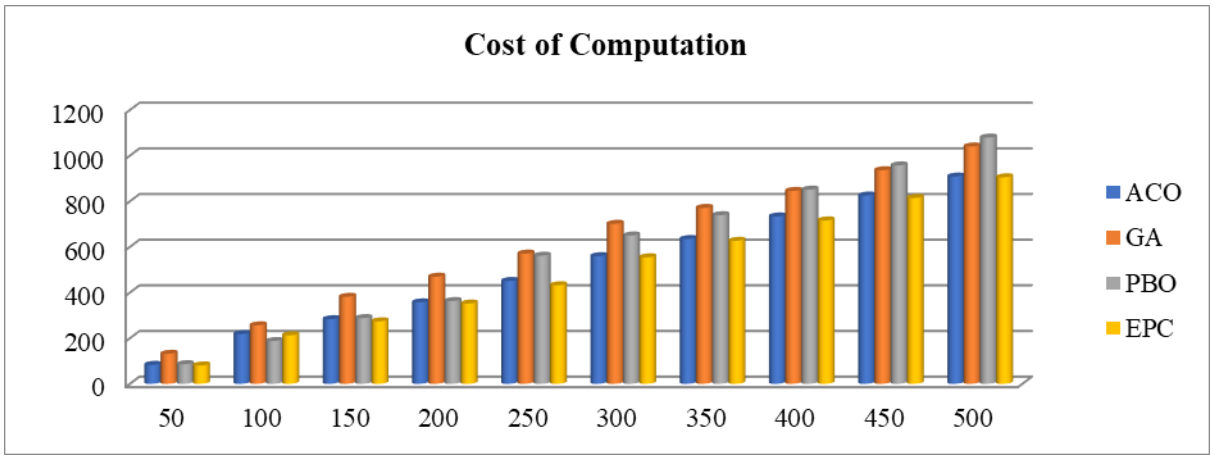


Figure 3: Cost of Computation (Heterogeneous Environment)

B. Simulation in Homogeneous Environment

In Homogeneous environment all the resources of clouds have same configuration so, same number of VMs will be generated in this. For experimentation, 100 resources are used and each resource contains 3VMs. The results of this proposed Cost-Aware Load Balancing Framework for Homogeneous Environment in terms of Load Balancing Factor are as given in figure 4. In this 50, 100, 150, 200, 250, 300, 350, 400, 450 and 500 jobs are simulated using GA, ACO, PBO and EPC based scheduling algorithm.

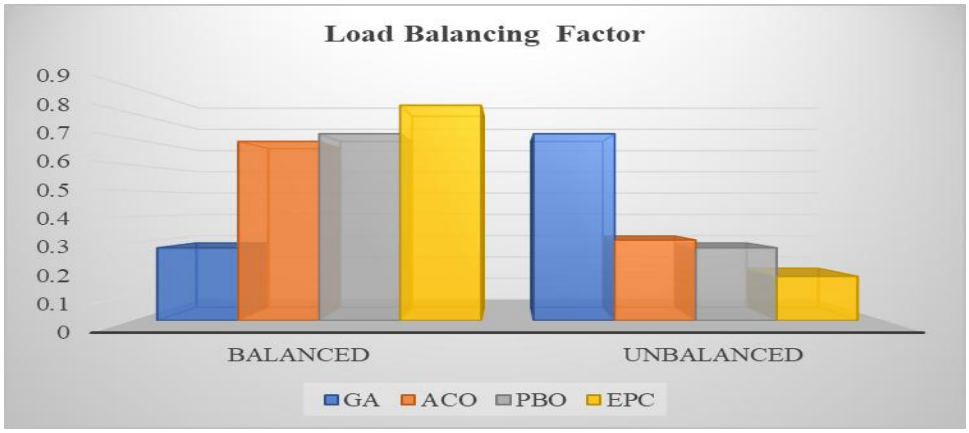


Figure 4: Load Balancing (Homogeneous Environment)

The above results show that there is more balanced load when EPC based scheduling algorithm is used for the execution of the jobs. This load balancing factor is calculated for 500 jobs for all GA, ACO, PBO and EPC algorithm. The cost factor calculated for Homogeneous factor is as shown in figure 5.

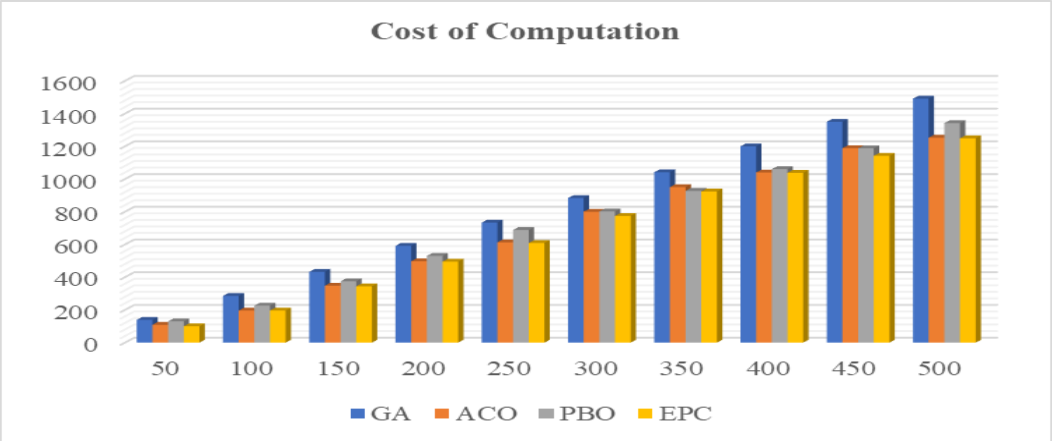


Figure 5: Cost of Computation (Homogeneous Environment)

IV. CONCLUSION

With the objective of Cost Reduction, this work proposed a cost-aware load balancing framework where tasks/jobs are divided into three different categories that are based on the requirement of the jobs for execution. Here in this work, tasks can be categorized into (a) CPU-Intensive, (b) Memory-Intensive, and (c) Both CPU-Memory Intensive. To analyze the performance of this proposed framework, two types of environment are generated (a) Heterogeneous, and (b) Homogeneous. For Experimentation and result analysis 100 resources are simulated on Local Cloud Environment where number of VMs is different in Heterogeneous but same in Homogeneous Environment. Different numbers of Jobs are assigned for testing purpose and the requirements of the jobs are randomly defined but same in both environments. The performance of this proposed framework is better in Heterogeneous environment but in homogeneous environment performance is degraded for both parameters as shown in Table 4 and Table 5. In terms of Scheduling Algorithm, Enhanced Pollination based Optimization (EPC) works better in heterogeneous environment and in most of the cases of homogeneous environment. With this experimentation it is concluded that this proposed EPC scheduler will help in cloud environment to provide efficient allocation of resources and its proper utilization which reduces cost factor. In future, the performance of this EPC scheduler can also be analyzed with different objectives like, energy efficiency.

REFERENCES

1. V. N. Volkova, L. V. Chemenkaya, E. N. Desyatirikova, M. Hajali, A. Khodar, and A. Osama, "Load balancing in cloud computing," 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), pp. 387–390, 2018.
2. V. Velde and B. Rama, "An advanced algorithm for load balancing in cloud computing using fuzzy technique," 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 1042–1047, 2017.
3. F. Saeed, N. Javaid, M. Zubair, M. Ismail, M. Zakria, M. H. Ashraf, and M. B. Kamal, "Load Balancing on Cloud Analyst Using First Come First Serve Scheduling Algorithm," Advances in Intelligent Networking and Collaborative Systems Lecture Notes on Data Engineering and Communications Technologies, pp. 463–472, 2018.
4. Y. Alexeev, A. Mahajan, S. Leyffer, G. Fletcher, and D. G. Fedorov, "Heuristic static load-balancing algorithm applied to the fragment molecular orbital method," 2012 International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–13, 2012.
5. S. Penmatsa and A. T. Chronopoulos, "Game-theoretic static load balancing for distributed systems," Journal of Parallel and Distributed Computing, vol. 71, no. 4, pp. 537–555, 2011.
6. V. Sharma and T. R. G. Nair, "An Optimum Scheduling Approach for Creating Optimal Priority of Jobs with Business Values in Cloud Computing," International Conference on Advances in Cloud Computing (ACC'12), pp. 1–8, 2012.
7. J. Shi, C. Meng, and L. Ma, "The Strategy of Distributed Load Balancing Based on Hybrid Scheduling," 2011 Fourth International Joint Conference on Computational Sciences and Optimization, pp. 268–271, 2011.
8. I. Riakiotakis, F. M. Ciorba, T. Andronikos, and G. Papakonstantinou, "Distributed dynamic load balancing for pipelined computations on heterogeneous systems," Parallel Computing, vol. 37, no. 10-11, pp. 713–729, 2011.
9. P. Gupta, M. K. Goyal, and P. Kumar, "Trust and reliability based load balancing algorithm for cloud IaaS," 2013 3rd IEEE International Advance Computing Conference (IACC), pp. 65–69, 2013.
10. O. Sarood, A. Gupta, and L. V. Kale, "Cloud Friendly Load Balancing for HPC Applications: Preliminary Work," 2012 41st International Conference on Parallel Processing Workshops, pp. 82–87, 2012.
11. B. Mondal, K. Dasgupta, and P. Dutta, "Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach," Procedia Technology, vol. 4, pp. 783–789, 2012.
12. G. Xu, J. Pang, and X. Fu, "A load balancing model based on cloud partitioning for the public cloud," Tsinghua Science and Technology, vol. 18, no. 1, pp. 34–39, 2013.
13. A. Dhari and K. I. Arif, "An Efficient Load Balancing Scheme for Cloud Computing," Indian Journal of Science and Technology, vol. 10, no. 11, pp. 1–8, 2017.
14. J. Yao and J.-H. He, "Load balancing strategy of cloud computing based on adaptive artificial bee colony algorithm," Journal of Computer Applications, vol. 32, no. 9, pp. 2448–2450, 2013.
15. S. M. Ghafari, M. Fazeli, A. Patooghy, and L. Rikhtechi, "Bee-MMT: A load balancing method for power consumption management in cloud computing," 2013 Sixth International Conference on Contemporary Computing (IC3), pp. 76–80, 2013.
16. K. Nishant, P. Sharma, V. Krishna, C. Gupta, K. P. Singh, Nitin, and R. Rastogi, "Load Balancing of Nodes in Cloud Using Ant Colony Optimization," 2012 UKSim 14th International Conference on Computer Modelling and Simulation, pp. 3–8, 2012

Table 4: Comparison of Homogeneous and Heterogeneous Load Balancing Framework for Cloud Computing

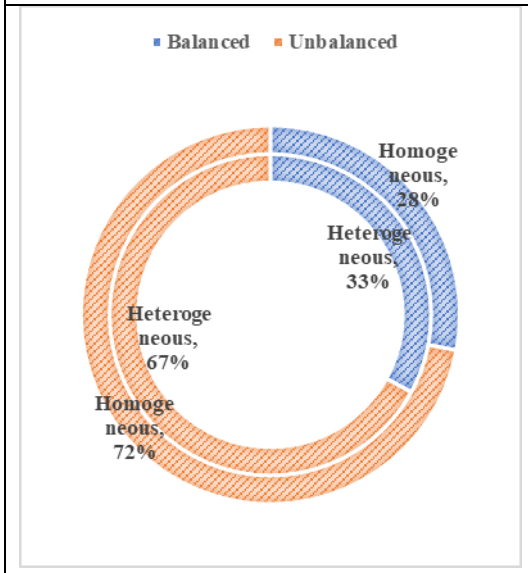
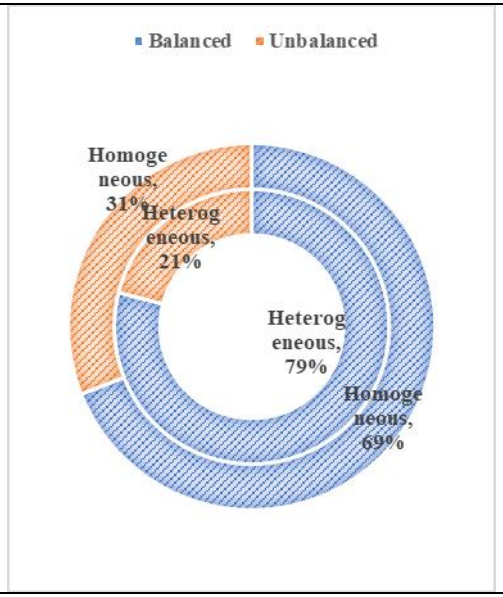
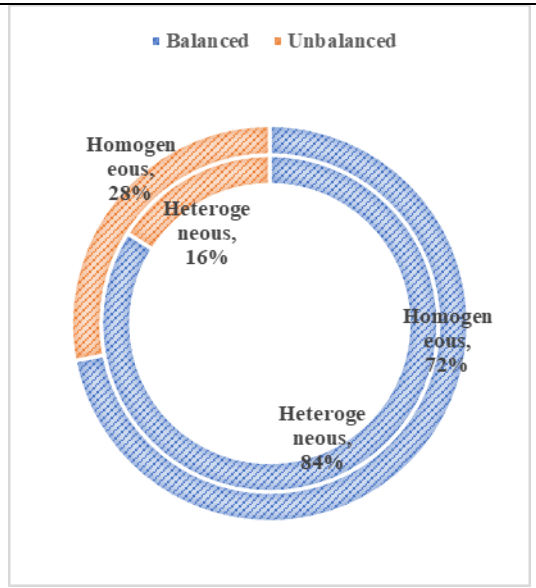
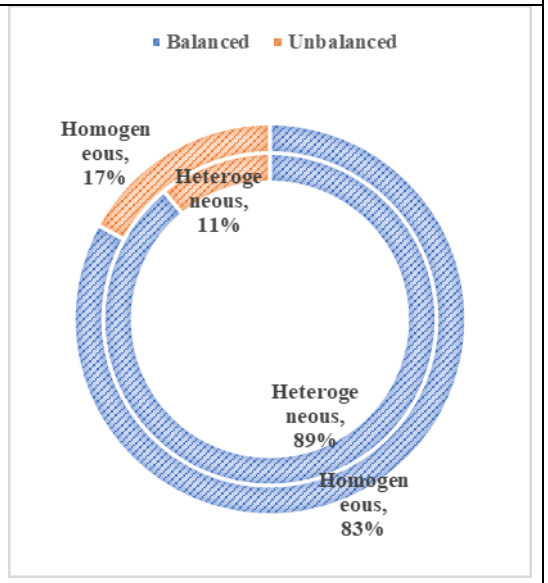
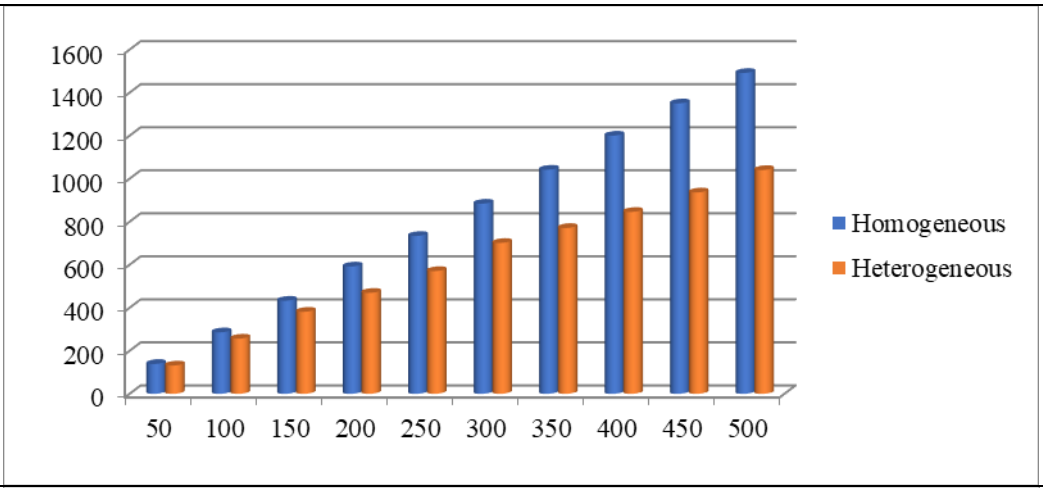
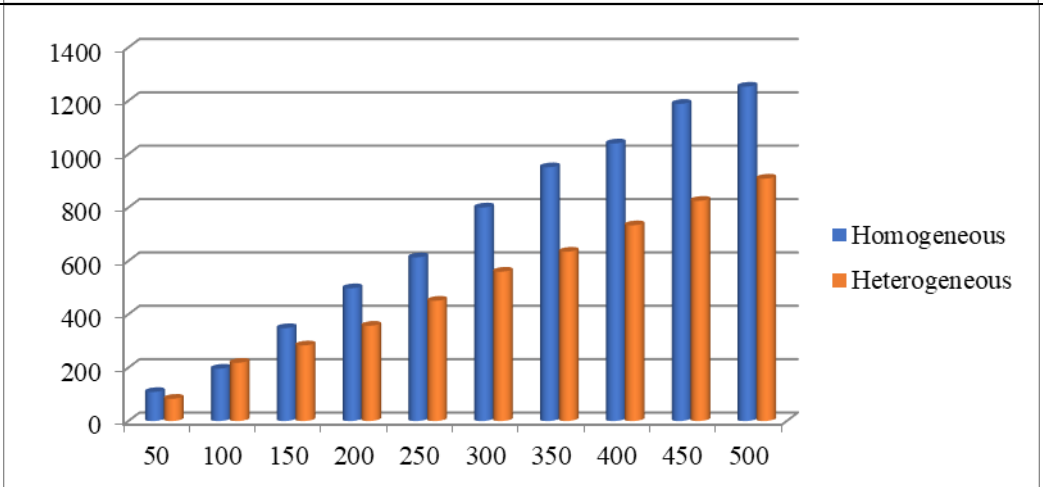
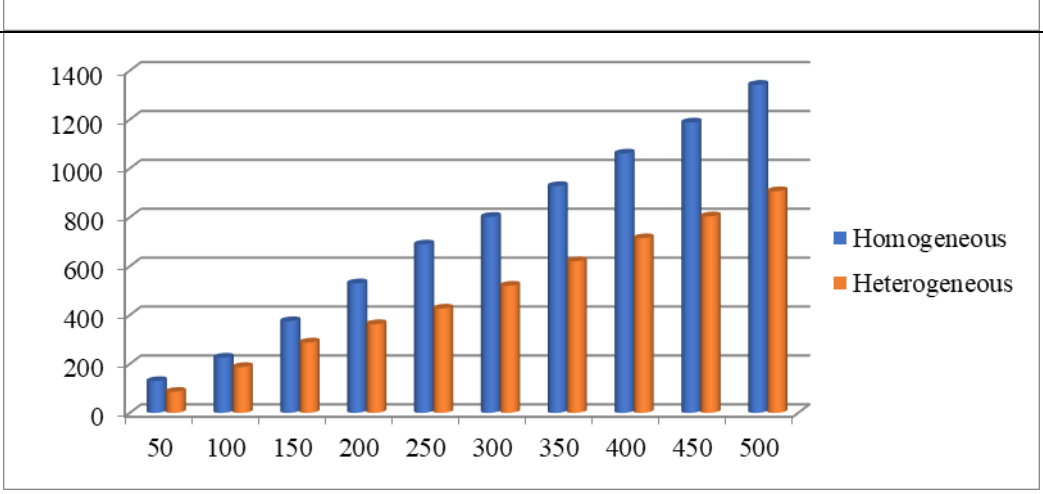
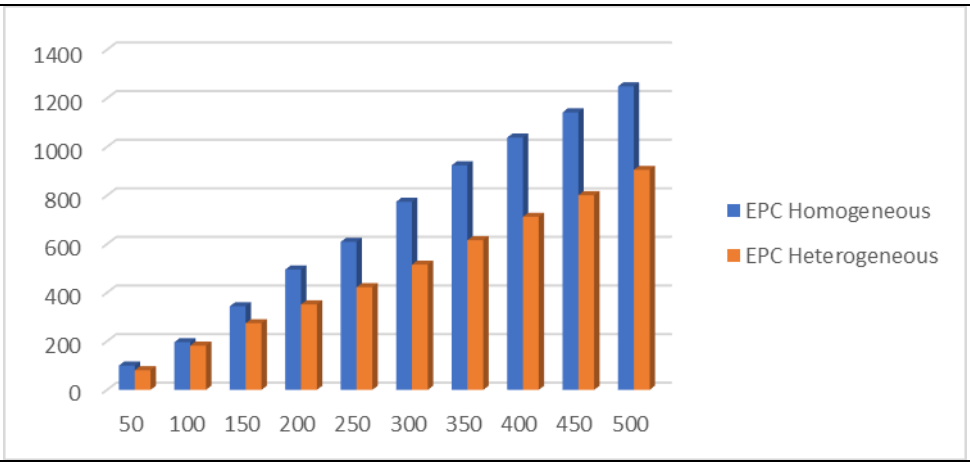
Load Balancing Factor (in %)			
GA	**ACO**	**PBO**	**EPC**
			

Table 5: Comparison of Homogeneous and Heterogeneous with Cost of Computation

Scheduling Algorithm	Cost of Computation for Efficient Load Balancing Framework
GA	
ACO	
PBO	

EPC	

1400
1200
1000
800
600
400
200
0
50
100
150
200
250
300
350
400
450
500
EPC Homogeneous
EPC Heterogeneous