DOI 10.29042/2019-4722-4731

Helix Vol. 9 (1): 4722- 4731

An Optimal Bio Inspired Genetic Algorithm for Load Balancing for Cloud Data Centre

[1]P.S.Latha Kalyampudi, [2]P.Venkata Krishna
[1]BVRIT HYDERABAD College of Engineering for Women, Hyderabad, India
[2]Professor, Sri Padmavathi Mahila University, Tirupati, Andhra Pradesh, India
Email: pslathakalyampudi@gmail.com, parimalavk@gmail.com

Received: 08th November 2018, Accepted: 24th November 2018, Published: 28th February 2019

Abstract
Demand for the high performance computing for industry, education and research is continuously motivating the application development industry to manoeuvre the existing and the upcoming applications towards the cloud. The deployment or the migration of the new or existing application on cloud data centres demands greater skills for development and maintenance of the applications. The biggest challenges for the data centre service providers are to balance the bottleneck of performance and cost. Many application or service owners have demanded for higher performance at a higher cost. Nevertheless, the traditional customers base have attracted towards cloud computing due to the low cost and higher performances. Thus for the data centre service providers it is the challenge to make the highest performance available towards the customers in the least cost. In order to make this challenge possible, the data centre service providers deploy various load balancing strategies for the most effective use. A number of research attempts are made towards achieving the best possible load balancing strategy by number of parallel research attempts. Many parallel research attempts have demonstrated the adaptive, strategic and just in time scheduling and load balancing algorithms with notable reduction in time for scheduling. Nonetheless, the previous works are always outperformed by the new algorithms proposed by other research works. In the recent time, the use of genetic algorithms and genetic optimization algorithms has demonstrated higher performances. However, the uses of bio-genetic algorithms have the chance to improve the performance further as always. Thus, this work proposes an additional optimization using proposed bio genetic optimization for effective balancing the data centre tasks. The proposed method demonstrates the 4% higher performance compared to the existing methods for a less loaded data centre and 16% improvements for a highly loaded data centre.

Keywords
Bio Genetic, Genetic Optimization, Pheromone, Opportunistic Balancing, Load Distribution Normality

Introduction

The rapid growth in the cloud computing domain is push forwarded by the industry to match the customer demands and propelled by the research outcomes. The application owners had constant demand for the high and scalable performance measures, which will not inflate the cost of the application deployments and maintenance. The paradigm of cloud computing fulfils these requirements from the client, data centre service providers and the application or service owners. The notable work by A. Weiss et al. [1] has proven the benefits of the cloud computing and defined it as the future of computing and services. The advantage from cloud computing is majorly achieved by the virtualization technique. The virtualization is a widely accepted method and demonstrated by various parallel research outcomes. The notable outcomes are by P. Barham et al. [2] on Xen as the VMM and by the J. Fisher et al. [3] on the hardware support. The basic principle of virtualization is the optimal utilization of the resources as proven by C. Fangzhe et al. [4]. Nevertheless, the allocation of the tasks on the virtual machine cannot be done manually, hence the multiple task allocation and assignment algorithms have evolved. The notable works by F. Chang et al. [5] and by J. Ren et al. [6] are a must to be considered for identifying the initial challenges of task allocation on cloud data centres. The complete automation of the task, called task scheduler and the optimization on cloud data centre is first introduced by H. Qiyi et al. [7]. Thereafter a number of research attempts are made to further optimize the task scheduling on cloud.

It is identified by multiple experiments that the optimization performance can be higher with the inclusion of bio-genetic dynamic programming methods as Ant-Colony-Optimization or ACO. This method was first introduced by M. Dorigo et al. [8] in the year of 2005 and the further improvements to make it applicable for cloud computing was produced by M. Birattari et al. [9] in the year of 2006. As stated by L.M. Gambardella et al. [10], the ACO is the best random search algorithm till date. Not limited to cloud computing, the ACO algorithm is also applicable to the problems for finite state like travelling salesman as demonstrated by L.M. Gambardella et al. [10], multiple solution

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

order problems like Graph Colouring as demonstrated by E. Salari et al. [11] and also for optimal solutions problems like Routing problems as demonstrated by Xiaoxia et al. [12].

The existing ACO algorithm work based on the real ant colony process driven by pheromone, evaporation of the pheromone and the number of ants present in the situations. Nevertheless, the optimum amount of initial pheromone, the rate of evaporation and the number of genetic is not calculated and none of the parallel research outcomes have demonstrated the strategy or the algorithm for deciding the optimality of the factors.

Hence, this work proposes a new optimization of parameters for genetic colony optimization algorithm to be applied for task scheduling to virtual machines on cloud based data centres.

The rest of the work is organized such that, in the Section – II, the parallel outcomes of the on-going researches on task scheduling is analysed. In the Section – III, the problem formulation and the mathematical model behind the optimization is furnished. In the Section – IV, the existing Genetic Optimization algorithm is analysed and the scopes for improves are analysed. In Section – V, the proposed optimization algorithm with the modified Genetic Optimization is elaborated. In order to establish the claim of performance improvements by the proposed algorithm, the comparative analysis is furnished in the Section – VI. The proposed algorithm is simulated and the obtained results are discussed in the Section – VII and finally this work presents the conclusion in Section – VIII.

Current State of Art

In this section of the work, the traditional approached for task scheduling are analysed in order to identify the problem formulate the optimization. The task scheduling algorithms are the NP-Complete problems and various parallel research attempts are made towards solving the completeness problems. For the analysis and testing of these algorithms, the simulation must be performed. The work by Rubinstein et al. [13] has demonstrated the fundamentals of simulation for completeness problems. The Ant Colony Optimization problems are widely studied and many of the research attempts have produced highly satisfactory results. One of the outcomes from the work of G. Ritchie et al. [14] has proven that the hybrid approach for ACO can provide better results. Also, considering the independent nature of the tasks, submitted to the cloud for analysis, the research attempt by M. Maheswaran et al. [15] is one of the most popular methods.

Further, the traditional approaches cannot be ignored as the entry point to this problem space can be identified by analysing those methods. Hence, this work also analyses the traditional approached in this section.

A. Opportunistic Balancing (OB) of Loads

Firstly, the opportunistic balancing method assigns the task to the virtual machines without considering the total estimated completion time but only considering the shortest schedule. The performance of the OB is significantly low as the total estimated time to complete the task can make huge difference on the performance indicators.

B. Minimum Execution Time (MET) for Loads

Secondly, the Minimum Execution Time strategy find the best matching of job and the virtual machine, where the allocation of the jobs or the tasks is done by only considering the fasted execution time. However, the already allocated jobs and priority of the jobs to be completed by that virtual machine makes very high difference on the final performance indicators.

C. Minimum Completion Time (MCT) for Loads

This third method, MCT is considerably better performing as during the assignment of the jobs or the tasks, the existing load and the execution time of the jobs are considered. Hence, this strategy is proven to better than many other heuristic strategies.

D. Hybrid Strategies for Loads

There are few strategies with the hybrid approach as minimum execution time and minimum completion time, maximum execution time and minimum completion time and minimum execution time and maximum completion time. The performances of these methods or strategies are highly subjected to the load conditions of the data centres and the configurations of the virtual machines.

E. Maximum Standard Deviation(Max-Std) for Loads

Finally, this scheduling strategy depends on the standard deviation of the execution time of the tasks. The highest standard deviation of the task execution time is always scheduled first and the tasks are having maximum variance of the execution time over multiple virtual machines will always be allocated last. This strategy is widely accepted for heterogeneous task scheduling environments as proven by E.U. Munir et al. [16].

Henceforth, this work draws the final conclusion of the literature reviews here [Table 1] and the strategies are ranked based on the advantages and demerits.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

Name of the Scheduling Strategy	Identified Issues	Ideal Condition for Acceptance	Acceptability Ranking (As High As Better)
Opportunistic Balancing	Time to complete the jobs are not considered, thus making the task completion time variable based on current loads on the VMs	Number of Virtual Machines are statistically equal to the number jobs per rotation and the length of the schedule is normally distributed	1
Minimum Execution Time	The length of the schedule for each jobs or tasks are not considered during the scheduling	The virtual machines are highly configured and the read / write operations by the task and significantly low.	2
Minimum Completion Time	The completion time may differ from one virtual machine to other	Best in the Heuristic situations	4
Hybrid	High dependency on the data centre configurations	The configurations of the virtual machines are not statistically different	3
Maximum Standard Derivation	Further optimization of the task or the jobs must be initiated	Heterogeneous nature of the tasks, which is valid for majority of the scheduling situations.	5

Table 1: Job/Task Scheduling Strategies with Identified Problems

This analysis is further analysed visually here [Fig 1].

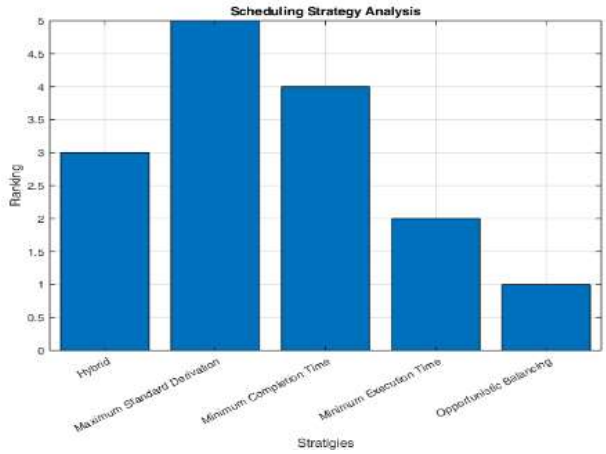

Fig. 1: Analysis of the Scheduling Strategies

Henceforth, in the next section of this work, the problem is been formulated based on the identified shortcomings from the existing works.

Problem Formulation

In this section of the work, the mathematical formulation is furnished. The proposed optimization on genetic algorithm can be achieved by optimizing the evaporation of the pheromone and the number of genetics in the system.

Thus, this work proposes two lemmas for proving the concept of proposed optimization in genetic optimization for Load Balancing strategies.

Lemma – 1: The number of genetic must be nearly equal to the number of jobs submitted to the cloud rather than equalling to the number of virtual machines for better optimization in the condition where the number of virtual machines are less than the number of jobs running.

Where,

L denotes the total number of Jobs

V denotes the total number of Virtual Machines

A_1 denotes the number Genetics in general

A_2 denotes the number of optimized Genetics

E denotes the number of possibilities of choosing any VM for Job allocation

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

Proof: In order to establish the thought of number of genetics depending on the number of jobs, this work elaborates the proof as following:
Firstly, considering the fact that, the numbers of virtual machines are less than the number of jobs submitted in the system.

$$V < L \quad \text{(Eq .1)}$$

Further, the number of possible ways to select any virtual machines based on the Genetic-Colony-Optimization strategy is as

$$A_1 = \frac{V(V-1)}{2} * L \quad \text{(Eq .2)}$$

Here the numbers of Genetics for this purpose are to be selected as, A_1.
Secondly, if the numbers of genetics are depending on the number of jobs to be submitted on the virtual machines, then it is equal to the number of jobs on the cloud system and can be deprecated as,

$$A_2 = L \quad \text{(Eq .3)}$$

It is natural to understand that, if the numbers of Genetics are less in the system, then the effectiveness of the algorithm decreases.
Also, it is natural to realize that in the general conditions of any data centres, the number of virtual machines will be less than the jobs to be processes and at the same time the numbers of Genetics based on virtual machine will be higher than the numbers of Genetics based on the number of jobs.

$$A_2 < A_1 \quad \text{(Eq .4)}$$

Thus, this lemma proves that the further optimization of the genetic Optimization can be achieved by deploying optimal number of genetics based on the number of jobs rather the number of virtual machines.

Lemma – 2: The less evaporation of the pheromone can increase the performance of Genetic Optimization.

Where,

A_1 denotes the number Genetics in general
A_2 denotes the number of optimized Genetics
E_1 denotes the evaporation factor in general
E_2 denotes the optimized evaporation factor
T_1 denotes the evaporation time in general
T_2 denotes the optimized evaporation time

Proof: In order to establish the thought of having less evaporation of the pheromone can improve the performance of Genetic Optimization; this work considers the following sequence of conversations:
Initially, the concept of pheromone in the Genetic Optimization helps the genetics to find the paths from the initial position to available virtual machines in case of job scheduling.
Thus, the number of Genetics visits the path will increase the amount of pheromone deposits on these paths and naturally it will take more time to evaporate by the standard rate of evaporation. Also, the more pheromone deposits on any paths will certainly make the process difficult to choose the correct path.
Hence, as considered,

$$A_2 \rightarrow E_2 \rightarrow T_2 \quad \text{(Eq .5)}$$

And

$$A_1 \rightarrow E_1 \rightarrow T_1 \quad \text{(Eq .6)}$$

It is already understood that, the more pheromone deposit and at the same point of time less pheromone deposit cannot optimize the Genetic Optimization strategy.
The time T_2 will be optimized if and only if the selection of Genetics are optimized, then the evaporation of the pheromone can also be optimized by less number of Genetics visit, which will result in less evaporation of pheromone. The optimization of Genetics is proved in the first lemma.

Existing Genetic Optimization for Load Balancing

This section of the work analyses the existing Optimization technique [17]. This analysis will help in identifying the proposed optimization scopes.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

Algorithm: Existing Optimization technique (ACO)

Step-1. Initialize pheromone for all virtual machines
Step-2. Initialize the Genetics at initial positions
Step-3. For all Genetics
  a. Start the tour
  b. Update the pheromone
  c. Calculate the makespan as makespan+1
  d. if(makespan< Max(makespan)
    i. Then Stop.
Step-4. End
Step-5. Report the optimal solution

Proposed Algorithm for Load Balancing

In this section of the work, the proposed algorithm is furnished.

Algorithm: Proposed Optimized Genetics and Optimized Pheromone based Optimization (OGOPO)

Step-1. Accept the Job list
Step-2. Accept the Virtual machine list
Step-3. For all Virtual machines
  a. Calculate the physical memory
  b. Calculate the random memory
  c. Calculate the network bandwidth
  d. Calculate the processing capabilities
  e. Calculate the total load of the virtual machine
Step-4. End
Step-5. Initialize pheromone for all virtual machines
Step-6. Calculate the initial Number of Genetics as Num_of_Genetics=Num_of_Jobs
Step-7. Initialize the Genetics at initial positions
Step-8. For all Genetics
  a. Start the tour
  b. Update the pheromone based on the optimal number of Genetics
  c. Calculate the makespan as makespan+1
  d. if(makespan< Max(makespan) and Load <Total_Load)
    i. Then Stop
Step-9. End
Step-10. Report the optimal solution

Henceforth, in the next section of the work, the analysis of improvements through comparative analysis is performed.

Comparative Analysis

In this section of the work, the proposed algorithm is compared with the existing current state of the arts algorithms based on the model time complexity and limitations.

A. Model Complexity Analysis and Ranking

Firstly, in this section of the work, the complexities of the existing models [18] are compared with the proposed algorithm model [Table 2].

Model Name	Complexity	Ranking (As Higher as Better)
Opportunistic Balancing	$O(N^N)$	1
Minimum Execution Time	$O(N^2 * \log N)$	2
Minimum Completion Time	$O(N^N)$	1
Hybrid	$O(N * \log N)$	3
Maximum Standard Derivation	$O(N / N * logN)$	4
OGOPO	$O(N^2 + M^2)$	5

Table 2: Model Complexity and Ranking Analysis

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

Here, it is natural to realize that the N is number of jobs and M denotes the number of Genetics in the system. As only the proposed algorithm derives the relation between the number of Genetics and the jobs, thus the derived complexity is as mentioned. Nonetheless, the proposed method defines the lowest complexity as M is already optimized. The comparative ranking is visualized graphically as well [Fig 2].

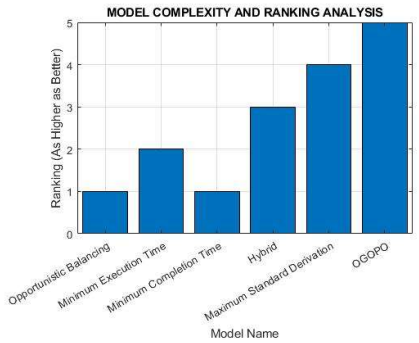

Fig. 2: Model Complexity and Ranking Analysis

Henceforth, it is clear to visualize and understand that the proposed method is having the best time or model complexity.

B. Algorithm Limitations Analysis and Ranking

Secondly, the limitations of the existing and the proposed algorithms are analysed and ranked here [Table 3].

Model Name	Limitations	Ranking (As Higher as Better)
Opportunistic Balancing	Completion Time variable depending on the VMs	1
Minimum Execution Time	The length of the schedule cannot be predicted	2
Minimum Completion Time	Completion Time variable depending on the VMs	1
Hybrid	Dependency on the VM configurations	3
Maximum Standard Derivation	Job optimization must be carried out	4
Genetic Optimization	Fixed configuration	5
OGOPO	All Limitations removed	6

Table 3: Algorithm Limitation and Ranking Analysis

Henceforth, looking at the number of limitations, the algorithms are ranks as the best algorithm is ranked high and the weakest algorithm is ranked low. The comparative ranking is visualized graphically as well [Fig 3].

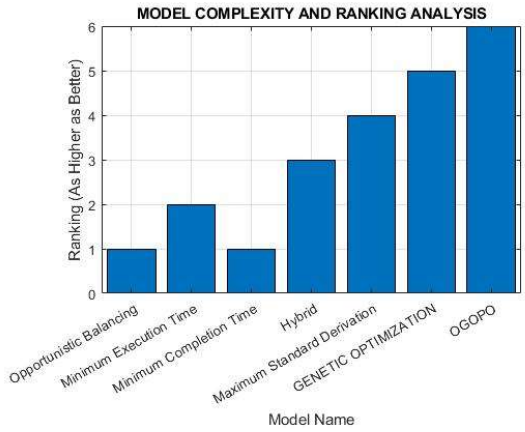

Fig. 3: Algorithm Limitations and Ranking Analysis

Henceforth, it is clear to visualize and understand that the proposed method is having the least number of limitations and thus ranked highest.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

Upon the confirmation as the proposed algorithm is strategically best among the parallel research outcomes, the results are now evaluated in the next section.

Results & Discussion

In this section of the work, the results obtained from this method is analysed and discussed. The results obtained from the proposed method or algorithm is also compared with the existing standard algorithms available as an outcome of the parallel researches. The analysis of the results are classified as response time, processing time, request servicing time from both the data centres, cost analysis and the normality pattern of the data centre loads.

Initially, this work presents the simulation scenario and the configuration is maintained same for all the algorithm simulations [Table 4].

Architecture Type	Number of Elements
Hadoop Architecture	Standalone with 2 Nodes
Number of Virtual Hosts	6
Number of Virtual Machines	6
Number of Initial Jobs	800
Duration of the Simulation	60 Days

Table 4: Simulation Configuration

A. Response Time Analysis

Firstly, the response time obtained from all the models are classified here [Table 5].

Algorithm Name	Response Time (Sec)
Round Robin	602.64
Execution Load	607.64
Throttled	607.64
Genetic Optimization	7533.51
Proposed Optimized Genetics and Optimized Pheromone based Optimization	602.61

Table 5: Response Time Comparisons

The results are visually graphically here [Fig 4].

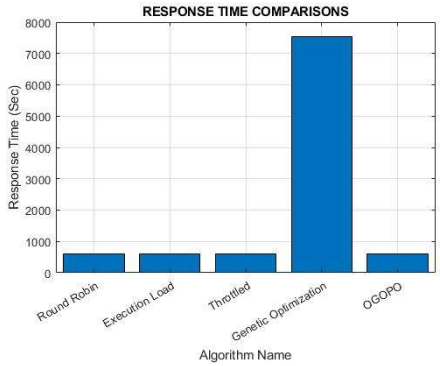

Fig. 4: Response Time Analysis

The proposed algorithm demonstrates the least response time.

B. Processing Time Analysis

Secondly, the processing time obtained from all the models are classified here [Table 6].

Algorithm Name	Processing Time (MSec)
Round Robin	1.12
Execution Load	1.12
Throttled	1.12
Genetic Optimization	7325.51
Proposed Optimized Genetics and Optimized Pheromone based Optimization	1.15

Table 6: Processing Time Comparisons

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

The results are visually graphically here [Fig 5].

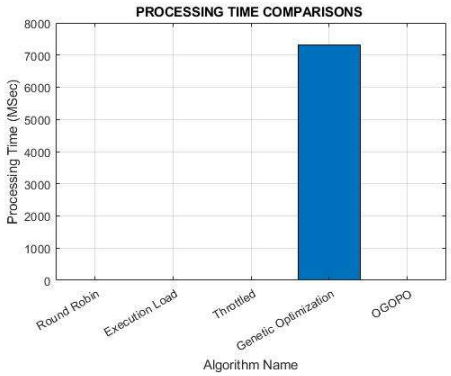

Fig. 5: Processing Time Analysis

The proposed algorithm demonstrates little higher processing time than the existing algorithms, but the time is less than the existing GENETIC OPTIMIZATION approach.

C. Request Servicing Time Analysis

Thirdly, the request servicing time obtained from all the models are classified here [Table 7].

Algorithm Name	Host 1 Servicing Time (MSec)	Host 2 Servicing Time (MSec)
Round Robin	1.10	1.12
Execution Load	1.10	1.12
Throttled	1.10	1.12
Genetic Optimization	1.10	7325.51
Proposed Optimized Genetics and Optimized Pheromone based Optimization	1.12	1.15

Table 7: Request Servicing Time Comparisons

The results are visually graphically here [Fig 6].

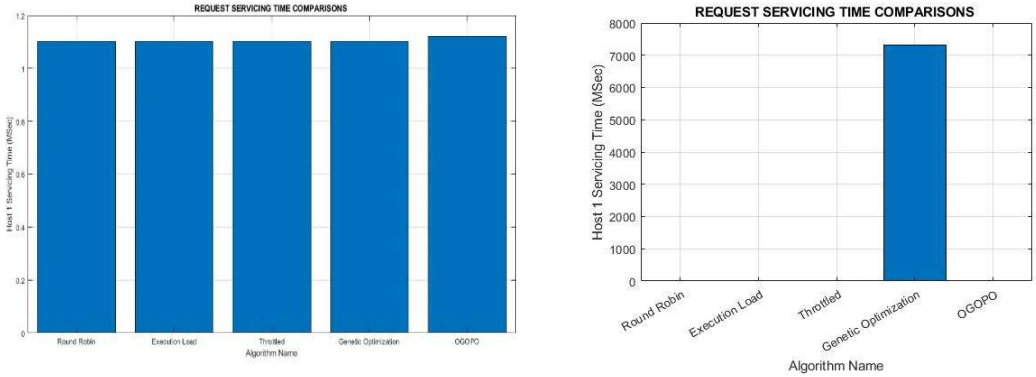

Fig. 6 (a) & (b) Request Servicing Time Analysis for Host 1 and Host 2

D. Virtual Machine Migration Cost Analysis

Fourthly, the migration cost obtained from all the models are classified here [Table 8].

Algorithm Name	Virtual Machine Cost (USD)
Round Robin	20.36
Execution Load	20.36
Throttled	20.36
Genetic Optimization	20.36
Proposed Optimized Genetics and Optimized Pheromone	20.36

Table 8: Migration Cost Comparisons

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

The results are visually graphically here [Fig 7].

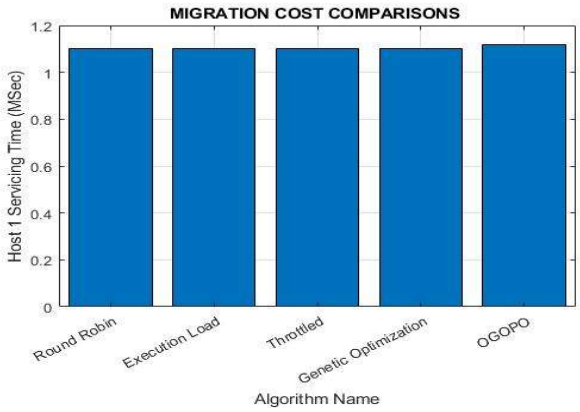

Fig. 7: Migration Cost Analysis

It is natural to understand that, the additional optimization has not increased the cost of the virtual machine migration in the proposed algorithm.

E. Data Centre Load Normality Pattern Analysis

Finally, the data centre load pattern normality analysis is performed here in order to identify the stability of the model [Table 9]. Considering that, the primary hypothesis defines normal distribution of the usage pattern for all the nodes under any algorithm. The alternative hypothesis defines that the utilization pattern is not normal.

Algorithm Name	**Utilization**						**Shapiro Test P Value**
	Node - 1	Node – 2	Node - 3	Node - 4	Node – 5	Node - 6	
Round Robin	60.82	240.53	375.12	600.14	602.64	242.11	0.3916
Execution Load	61.93	240.51	367.99	607.64	595.14	242.11	0.3954
Throttled	60.82	244.54	368	602.64	607.64	242.14	0.3928
Genetic optimization	60.85	240.51	368	600.12	607.64	7533.51	0.0001739
Proposed Optimized Genetics and Optimized Pheromone	61.87	240.53	367.99	602.61	600.12	242.14	0.3981

Table 9: NORMALITY PATTERN Comparisons

Henceforth, it is natural to understand that the abnormality in the usage pattern for the Genetic Optimization is completely resolved by the proposed OGOPO – Genetic Optimization algorithm.

The distributions are visualized here [Fig 8 to Fig 10].

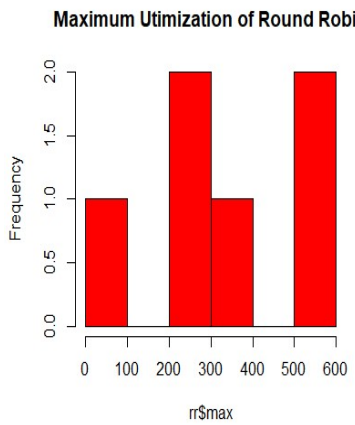

Fig. 8: Utilization Normality Analysis- Round Robin

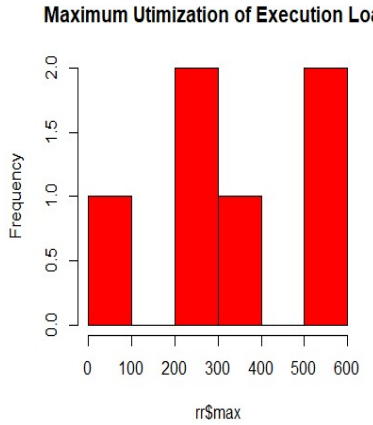

Fig. 9: Utilization Normality Analysis- Execution Load

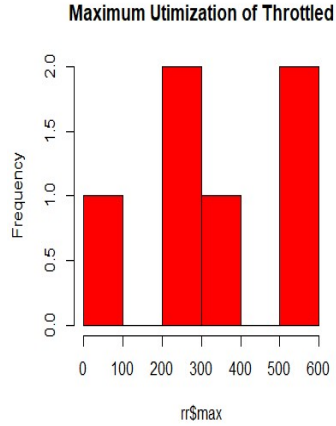

Fig. 10: Utilization Normality Analysis- Throttled

Henceforth, after the complete analysis and the discussions on the results, this work presents the final conclusion in the next section.

Conclusion

The modern application hosting and development industry demands higher performance and availability. The higher performance always cannot be reached with higher infrastructure. Thus the research demands better algorithms for load allocation and balancing. A number of research attempts have tried the optimization on load balancing and the recent trends demonstrated the use of genetic optimization techniques. Nevertheless, this work identified the shortcomings and proposed a new method for optimization using parametric Genetic-Colony-Optimization. The outcomes from the proposed algorithm have demonstrated a higher response time and more normal load distribution than the present methods. Thus, this work contributes towards an improvement and betterment of the load balancing strategies for better computing on cloud.

References

[1] A. Weiss, "Computing in the Clouds", netWorker on Cloud computing: PC functions move onto the web, vol. 11, pp. 16-25, Dec. 2007.
[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield, "Xen and the art of virtualization", Proc. of 19th ACM symposium on Operating systems principles, 2003.
[3] J. Fisher-Ogden, "Hardware support for efficient virtualization", April 2006.
[4] C. Fangzhe, R. Jennifer, V. Ramesh, "Optimal Resource Allocation in Clouds", 2009 IEEE International Conference on Software Testing Verification and Validation, pp. 91-100, 2009.
[5] F. Chang, J. Ren, R. Viswanathan, "Optimal Resource Allocation for batch testing", 2009 IEEE International Conference on Software Testing Verification and Validation, pp. 91-100, 2009.
[6] F. Chang, J. Ren, R. Viswanathan, "Optimal Resource Allocation in Clouds", 2010 IEEE 3rd International Conference on Cloud Computing, pp. 418-425, 2010.
[7] H. Qiyi, H. Tinglei, "An Optimistic Job Scheduling Strategy based on QoS for Cloud Computing", 2010 IEEE International Conference on Intelligent Computing and Integrated Systems (ICISS), pp. 673-675, 2010.
[8] M. Dorigo, C. Blum, "Ant colony optimization theory: A survey", Theoretical Computer Science, pp. 243-278, 2005.
[9] M. Dorigo, M. Birattari, T. Stutzel, "Ant colony optimization", IEEE Computational Intelligence Magazine, pp. 28-39, 2006.
[10] M. Dorigo, L.M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem", IEEE Transactions on Evolutionary Computation, pp. 53-66, 1997.
[11] E. Salari, K. Eshghi, "An ACO algorithm for graph colouring problem", Congress on Computational Intelligence Methods and Applications, pp. 5, 2005.
[12] Xiaoxia Zhang, Lixin Tang, "CT-ACO-hybridizing ant colony optimization with cycle transfer search for the vehicle routing problem", Congress on Computational Intelligence Methods and Applications 2005, pp. 6, 2005.
[13] Rubinstein, R. , Simulation and the Monte Carlo Method. John Wiley &Sons,(1981).
[14] G. Ritchie and J. Levine, A hybrid Ant algorithm for scheduling independent jobs in heterogeneous computing environments. University of Strathclyde.
[15] M.Maheswaran, Shoukat Ali, Dynamic Matching and Scheduling of a Class of independent Tasks onto Heterogeneous Computing Systems, 8th Heterogeneous Computing Workshop (HCW'99),(1999).
[16] E.U. Munir, MaxStd : A Task Scheduling Heuristic for Heterogeneous Computing Environment, Information Technology Journal, ISSN-1812-5638, 2008.
[17] Mustafa Muwafak Alobaedy, Ali A Khalaf and Ishola D. Muraina, Analysis of the Number of Ants in Ant Colony System Algorithm ,2017 Fifth International Conference on Information and Communication Technology (ICoICT)
[18] Jamshid Bagherzadeh*, MojtabaMadadyarAdeh, An Improved Ant Algorithm for Grid Scheduling Problem, Proceedings of the 14th International CSI Computer Conference (CSICC'18).

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495