Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Modified Genetic Algorithms (GA) for Load balancing in Cloud Computing

Yogasambhuta Dash[1], Rabindra Kumar Dalei[2], Kasturi Dhal[3]
[1]*Trident Academy of Technology*
[1]*Silicon University*
[2]*Silicon University*
[3]*Silicon University*

ARTICLE INFO

Received: 18 Dec 2024

Revised: 10 Feb 2025

Accepted: 28 Feb 2025

ABSTRACT

The efficiency of the upcoming cloud computing generation will be determined by how rapidly the infrastructure is constructed and how dynamically the resources are utilized. To prevent any one resource from being overworked or underutilised, load balancing divides the fluctuating workload among several nodes. This is one of the primary issues with cloud computing. A skilled load balancer should adjust its strategy in accordance with shifting tasks and conditions. This can be viewed as an optimisation challenge. This research uses a Genetic Algorithm (GA) to suggest a novel load balancing approach. The technique seeks to shorten the length of a specific operation by distributing the load on the cloud infrastructure. The Cloud Analyst simulator has been used to model the suggested load balancing approach. According to simulation findings for a typical example application, the suggested methodology performed better than the current approaches, such as Round Robing (RR), First Come First Serve (FCFS), and a local search algorithm called Stochastic Hill Climbing (SHC).

Keywords: Genetic algorithm, cloud data center,load balancing,response time,optimization

INTRODUCTION

The phrase "cloud" describes a new distributed computing concept. A class of advanced upon request cloud computer services provided by providers of cloud-based solutions like Microsoft, Google and Amazon is described in this article [1]. Businesses and individuals may access apps quickly from any location in the world because to this computer infrastructure. Processing, storage, and software are all provided to clients "as a service" by any cloud service provider. Any business can reduce its capital expenditures for software and hardware by utilizing cloud computing, which enables provisioning and de-provisioning on demand [2]. The industry has embraced cloud computing due to its exponential growth, which has resulted in a rapid expansion of Internet resource availability. Large demands must be handled by cloud computing service providers as the size of the cloud increases. The main difficulty then becomes maintaining or improving performance whenever such an outburst does place. Although the future of cloud computing is bright, many significant problems must be solved before it can be completely adopted [3]. Among these problems is load balancing.

To fully utilise the resources of distributed and parallel systems, it is regarded as one of the conditions. Application requests can be distributed among an infinite number of data centre-based application installations by Cloud Service Providers (CSP) using load balancing, which distributes the burden among a few nodes, data centres, hard drives, or other processing resources. In general, balancing of load systems can be classified as either periodic or non-periodic, or centralised or decentralised, or dynamic or static. The use of balancing of load strategies in cloud computing environments has not received much attention. Using Minimum Execution Time (MET), Armstrong et al. in [4] randomly assign each work to the nodes that are predicted to finish it the fastest, irrespective of the load that node is currently experiencing. The literature also discusses the use of several modern scheduling strategies for load balancing, such as First Cum First Serve [FCFS], Round Robin [RR], and Min-Min. Yang Xu and colleagues first proposed an intelligent load balancing technique [5]. It suggests an innovative approach for balancing data distribution in data-intensive applications like data mining in a distributed approach in order to enhance cloud

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

computing performance. Additionally, some soft computing techniques have been published in the literature, like Ant Colony [6].

The Genetic Algorithm (GA), a soft computing technique, employs the natural selection strategy mechanism in this study. Cloud Analyst: A visual modeller based on Cloud-Sim has been used to simulate and assess the algorithm. Two popular scheduling algorithms, First Cum First Serve and Round Robin, along with a local search mechanism called stochastic hill climbing, are used to compare the algorithm's performance [7]. In Section 2, the GA load balancing algorithm is suggested. The simulation findings and analysis in Section 3 are accompanied, for the sake of completeness, by an overview of Cloud Analyst in Section 3.1. Section 4 represents the final conclusion of this work.

CLOUD COMPUTING USING GA FOR LOAD BALANCING

The load balancing problem is referred to assigning R tasks submitted by Cloud clients to S cloud handling units, even though cloud computing is dynamic. The present status of utilization of each processing unit is shown by its processing vector unit (PVU). Included in this pattern are MIPS, which shows how many instructions per second the computer can process, β, the cost of executing instructions, and delay cost C. The projected delay cost is the sum that the cloud computer services provider will be required to reimburse the punter in the event that the development goes time-consuming than the service provider had anticipated.

$$PVU = f(MIPS, \beta, C) \quad (1)$$

Likewise, a job vector (JV) can be used to characterise each job that a cloud consumer submits. Therefore, equation 2 can be used to symbolise the attribute of several vocations.

$$JV = f(A_T, W_C, S_{IC}, T) \quad (2)$$

T is an acronym for Infrastructure as a Service (IAAS), Software as a Service (SAAS) and Platform-as-a-Service (PAAS), depending on the kind of service needed for the task. The processor determines the NIC, which is a count that shows how many instructions are in the job. The completion time for worst case (W_C) is the bare minimum of time needed for a processing unit to finish the task, whereas the arrival time of job (A_T) is the real time of the job's entry in the system.

Equation 3 illustrates that the cloud computer services provider must distribute these S tasks among R processing units in order to reduce the cost function γ.

$$\gamma = K_1 * \alpha (S_{IC} \div MIPS) + K_2 * L \quad (3)$$

Where the K1 and K2 weights are preset. A criterion that could be applied to determine or optimise the weights is that the bigger the weight, the more generic the component. Another illustration of logic is when users prioritise or favour one component over another. In this case, the second approach has been used, and the designated weight set has been optimised. Consideration is given to weights $K_1 = 0.7$ and $K_2 = 0.3$, so the sum is equals to 1.

Because of this, balancing the load is a challenging problem that might not be solvable computationally. Since linear programming cannot express such a problem, it is very challenging to find the globally the best approach using time algorithms for deterministic polynomial or principles. One of the most popular artificial intelligence methods, GAs [8] are mostly employed for efficient search and optimisation. It is a stochastic search technique that draws inspiration from genetics and natural selection. When looking for global optimum solutions, GAs have been shown to be incredibly effective and stable, especially in complex and/or large search spaces. GA is a load balancing technique that has been presented in this research to find the global ideal number of processors for a task in a cloud. Rearranging occupations is not taken into consideration because it will be a global optimum solution, and the arrival of jobs is seen as linear. The next section provides an explanation of the suggested method.

THE PROBABLE TECHNIQUE:

Genetic Algorithm consists of three operations: selection, genetic operation, and replacement. Among the benefits of this approach are its capacity to manage a wide search space, its suitability for intricate goal functions, and its ability to stay clear of local optimal solutions. Figure 2 illustrates how GA works for balancing of load in cloud computing; its specifics are explained below.

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

I. **The original population generation:** Each individual response's constant string of bits format is used by GA. Consequently, every potential solution in the outcome space is transformed into a string of binary. A random selection of several chromosomes is made from this initial population of ten (10) individuals.

II. **Crossover:** Choosing the most suitable pair of people for crossover is usually the aim of this phase. The fitness parameter provided in 3 is used to determine the fitness value of each chromosome. In this chromosome pool, a random single point crossover takes place, in which the section on one end of the bridging site is switched to the opposite side based on the crossover point. This creates a new pair of folks.

III. **Mutation:** An extremely low value (0.05) is now used to set the variation incidence. The bit pattern of the chromosomes alternate between 1 and 0 or 0 and 1, depending on the variation's incidence. A fresh mating pool that is prepared for crossover is the result of this.

Until the end point requirement is reached or the most suitable chromosome is discovered, this genetic algorithm approach is replayed.

The suggested technique is as follows:

1: Once a group of processing units has been encoded into strings of binary, launch them at random.

2: Utilizing equation 3, determine the level of fitness for every group.

3: Following the discovery of the best solution or the completion of a maximum amount of repetitions, take the following steps:

I: To create the mating pool which is Selection, the chromosome with the highest efficiency value is reserved for mating, and the chromosome with the lowest efficiency is taken twice.

II: To create a new offspring, single point bridging selects the bridging point, which is a Crossover at random.

III: The likelihood of a new offspring becoming mutated is 0.05.

IV: Put the additional offspring in the new population and utilise it for the subsequent iteration [Accepting].

V: Check for the final state [Test].

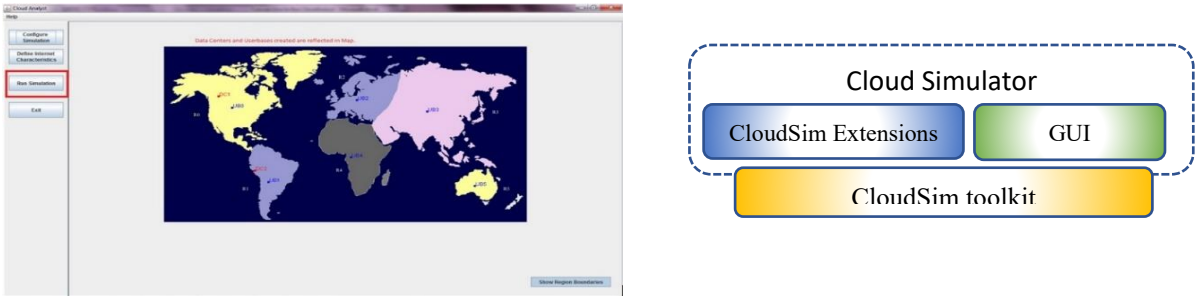
Step 4: End.

ANALYSIS AND OUTCOMES OF THE SIMULATION

An international bank's "Internet Banking" scenario is used to model the suggested GA algorithm using the simulation toolkit Cloud Analyst [9].

ANALYSIS USING CLOUD

Simulators of streaming services for resources are needed to meet the expectations of the approach to cloud computing at the framework and program levels. There aren't many simulators like Cloud Sim [10] and Cloud Analyst [9]. This research makes use of Cloud Analyst as a simulation tool. Figure 1(a) displays a screenshot of the Cloud Analyst simulation toolkit's graphical user interface, while Figure 1(b) shows its architecture.



(a) GUI of Cloud Analyst (b) Cloud Analyst's architecture is based on Cloud Sim.

Fig 1. Snapshot of Cloud Analyst

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

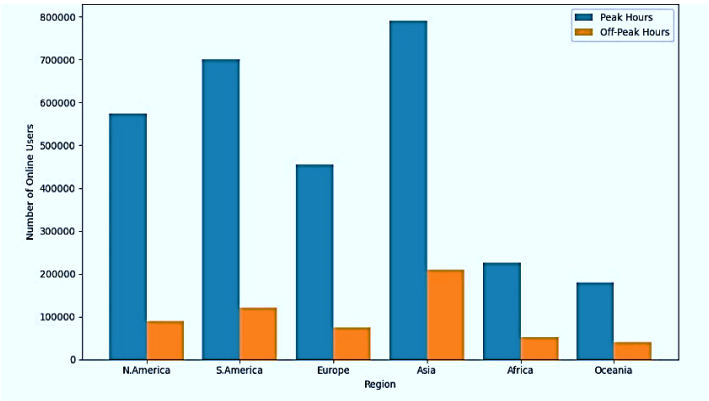
Cloud Analyst, an analysis tool, was created using Cloud Sim. Cloud Sim facilitates cloud exploration, including programmatic simulation and design. It allows any cloud research problem to be investigated by configuring the characteristics of a simulation environment. The parameters are also shown graphically in the simulation result according to the parameters that the program determines.
Cloud Analyst has been used to build a fictitious configuration. 6 regions, each representing one of the 6 major continents, make up the planet. Consideration is given to 6 "User bases," referring to groups of users. All client groups have been assigned to a specific time zone, and while only 20% of online registered individuals are online during times of low usage, it is expected that a wide variety of users will be present during high usage time. The information about the user groups utilized in testing are listed in Table 1. There are a specific number of virtual system (VS) reserved for the application on each of the fictional "data centre hosts." Four CPUs with 10,000 MIPS each, 1TB of storage, and 6 GB of RAM are included in every system.

SETUP FOR THE SIMULATION

Several situations are considered for testing, along with a single central data centre (CDC). Therefore, this one CDC, which has 25, 50, and 75 VSs of Cloud Specifications (CGs) assigned to the application, handles all user requests from all over the world. Together with the determined Average Response Time (ART) in millisecond for GA, SHC, RR, and FCFS, this modelling scenario is detailed in table 2. Figure 2 displays an assessment of performance graph of the same. Table 3 and figure 3 both display the outcome analysis of the next two CDCs, each of which has a combination of 25, 50, and 75 VS. Next, as shown in tables 4, 5, 6, and 7, we investigate three, four, five, and six CDCs utilizing a blend of 25, 50, and 75 VSs for each CG. Figures 4, 5, 6, and 7 show the appropriate assessment of performance graphs one after other.

Table 1: Simulation environment structure

Sl. No	User	Region	Users Online during high usage hrs.	Users Online during low usage hrs.
1.	U1	0-N.America	5,75,000	90,000
2.	U2	1-S.America	7,00,000	1,20,000
3.	U3	2-Europe	4,55,000	75,000
4.	U4	3-Asia	7,90,000	2,10,000
5.	U5	4-Africa	2,25,000	52,000
6.	U6	5-Oceania	1,80,000	40,600



STUDY OF COMPLEXITY

The evaluation of complexity includes the computing difficulties (sometimes called time complexity) and space complexity of any algorithm. Fitness calculation, selection, crossover, and mutation are the fundamental activities carried out by genetic algorithms. Since population initialisation is regarded as preprocessing in genetic algorithms, its complexity is not taken into account for analysis. The time complexity for programming into a string of bits is at most j1, and the cost function evaluation for verifying the cost c of k chromosomes is at most (c × k). The temporal complexity of the selection process is at most i; for single point traversing, it is at most i, where i is the length of the chromosome; and for mutation at any location, it is again i. The total time complexity 'CT' is obtained by repeatedly performing the 3 genetic algorithm procedures iteratively until the halting condition is satisfied.

$$CT = P\{j1 + (c \times k) + (j2 + 1)(i + i + i)\}$$

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 2: The simulation for Average Response Time (ART), expressed in milliseconds

S.No	Cloud Specification	CDC Configuration	ART using GA	ART using SHC	ART using RR	ART using FCFS
1.	CG1	Every single with 25 VM	330.01	330..02	330.03	330.13
2.	CG2	Every single with 50 VM	328.87	329.05	329.32	329.38
3.	CG3	Every single with 75 VM	246.00	328.34	328.67	328.54

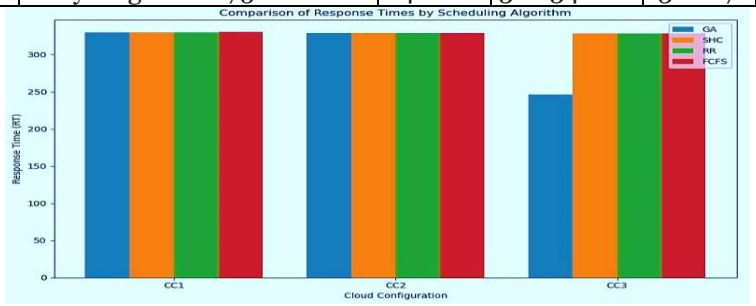


Figure 2: Evaluation of the suggested GA's performance using SHC, FCFS, and RR using 1 data centre

Table 3: The simulation for Average Response Time (ART), expressed in milliseconds

S.No	Cloud specification	CDC Configuration	ART using GA	ART using SHC	ART using RR	ART using FCFS
1.	CG1	Two CDCs every one has 25 VM	361.77	366.44	372.27	377.34
2.	CG2	Two CDCs every one has 50 VM	356.72	361.15	368.49	373.52
3.	CG3	Two CDCs every one has 75 VM	356.32	360.73	365.78	371.56
4.	CG4	Two CDCs every one has 25, 50 VM	351.58	357.72	363.91	369.87
5.	CG5	Two CDCs every one has 25, 75 VM	352.56	358.23	365.45	368.23
6.	CG6	Two CDCs every one has 75, 50 VM	353.01	358.04	362.61	362.01

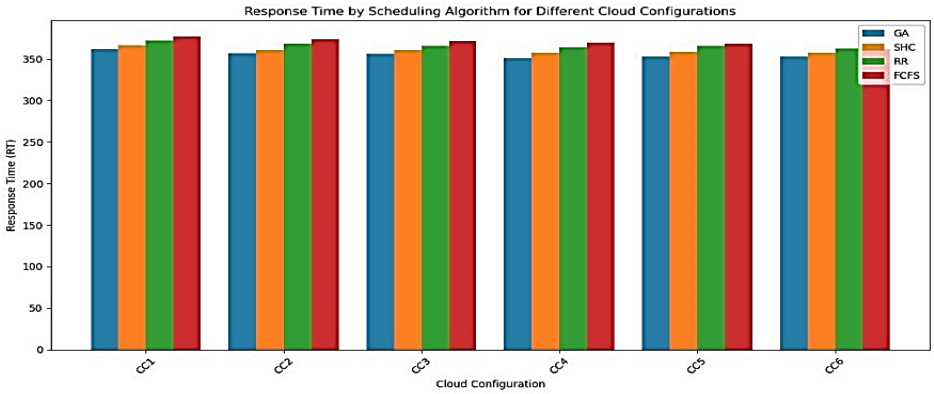


Figure 3: Evaluation of the suggested GA's performance using SHC, FCFS, and RR using 2 data centres

Table 4: The simulation for Average Response Time (ART), expressed in milliseconds (ms)

S.No	Cloud specification	CDC Configuration	ART using GA	ART using SHC	ART using RR	ART using FCFS
1.	CG1	Every single with 25 VM	351.32	357.82	362.17	364.34
2.	CG2	Every single with 50 VM	351.19	356.25	363.49	364.52
3.	CG3	Every single with 75 VM	347.01	351.73	357.18	362.56
4.	CG4	Every single with 25,50 and 75 VM	346.98	351.01	357.21	361.87

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

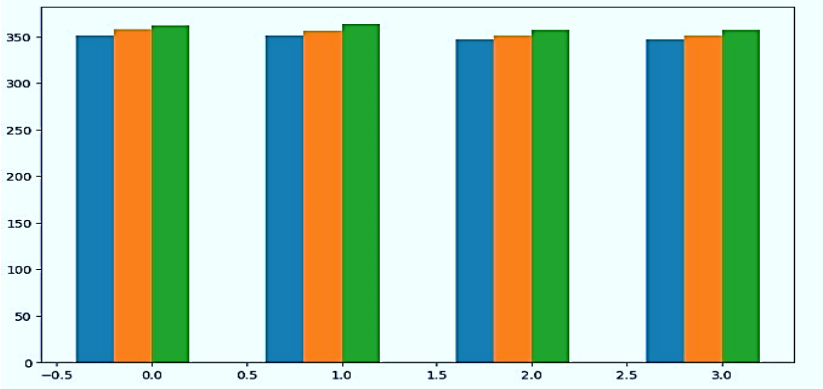


Figure 4: Evaluation of the suggested GA's performance using SHC, FCFS, and RR using 3 data centres

Table 5: The simulation for Average Response Time (ART), expressed in milliseconds (ms)

S.No	Cloud Specification	CDC Configuration	ART using GA	ART using SHC	ART using RR	ART using FCFS
1.	CG1	Every single with 25 VM	347.85	353.35	358.35	359.95
2.	CG2	Every single with 50 VM	344.54	349.71	355.93	358.97
3.	CG3	Every single with 75 VM	339.65	345.46	351.09	357.44
4.	CG4	Every single with 25, 50 and 75 VM	336.88	345.31	350	354.94

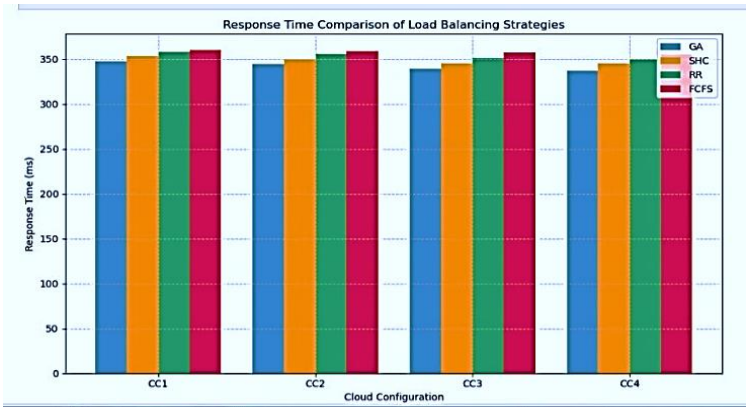


Figure 5: Evaluation of the suggested GA's performance using SHC, FCFS, and RR using 4 data centres

Table 6: The simulation for Average Response Time (ART), expressed in milliseconds

S.No	Cloud Specification	CDC Configuration	ART using GA	ART using SHC	ART using RR	ART using FCFS
1.	CG1	Every single with 25 VM	330.64	341.86	347.57	351.05
2.	CG2	Every single with 50 VM	325.02	330.84	338.76	344.44
3.	CG3	Every single with 75 VM	321.93	328.46	334.88	341.79
4.	CG4	Every single with 25, 50 and 75 VM	318.98	325.64	333.01	337.01

Journal of Information Systems Engineering and Management
2025, 10(54s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

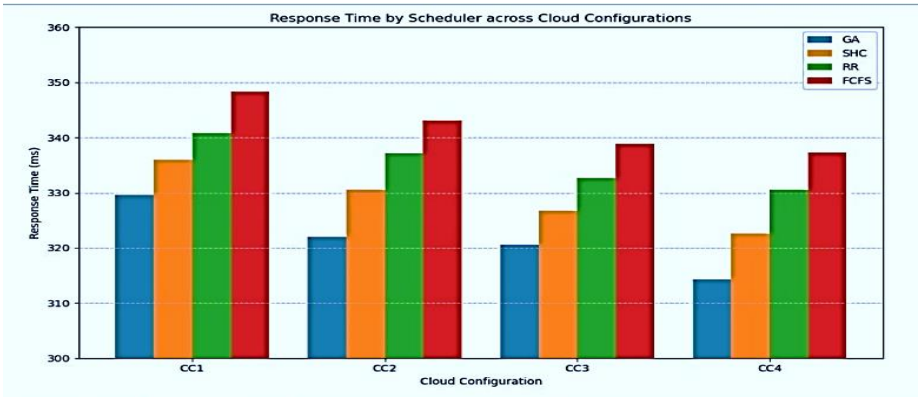


Figure 6: Evaluation of the suggested GA's performance using SHC, FCFS, and RR using 5 data centres

Table 7: The simulation for Average Response Time (ART), expressed in milliseconds

S.No	Cloud Specification	CDC Configuration	ART using GA	ART using SHC	ART using RR	ART using FCFS
1.	CG1	Every single with 25 VM	329.54	335.96	340.87	348.26
2.	CG2	Every single with 50 VM	322.01	330.56	337.14	343.04
3.	CG3	Every single with 75 VM	320.54	326.78	332.67	338.87
4.	CG4	Every single with 25, 50 and 75 VM	314.33	322.56	330.49	337.29

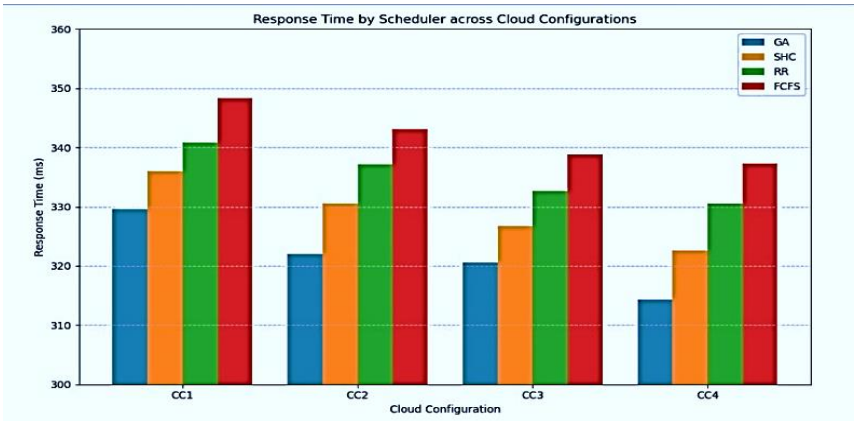


Figure 7: Evaluation of the suggested GA's performance using SHC, FCFS, and RR using 6 data centres

CONCLUSION

Here the study developed a balancing for load method for cloud computing based on evolutionary algorithms to facilitate efficient resource utilisation in a cloud setting. The results of the investigation show that the suggested balancing for load strategy not only surpasses some of the current methods but also ensures that the quality of service demands of the customer job are met. Despite the common misconception that all tasks are of equal importance, it can be handled in the JV and then addressed in the capability task. Although basic GA tactic already been utilised, future research could use variants of the bridging and assortment policies to get more accurate and useful results.

REFRENCES

[1] Çavdar, M. C., Korpeoglu, I., & Ulusoy, Ö. (2024). A utilization based genetic algorithm for virtual machine placement in cloud systems. *Computer Communications*, *214*, 136-148.
[2] Ghafir, S., Alam, M. A., Siddiqui, F., & Naaz, S. (2024). Load balancing in cloud computing via intelligent PSO-based feedback controller. *Sustainable computing: informatics and systems*, *41*, 100948.

Research Article

[3] Muneeswari, G., Madavarapu, J. B., Ramani, R., Rajeshkumar, C., & Singh, C. J. C. (2024). GEP optimization for load balancing of virtual machines (LBVM) in cloud computing. *Measurement: Sensors*, *33*, 101076.
[4] Sansanwal, S., & Jain, N. (2022). An improved approach for load balancing among virtual machines in cloud environment. *Procedia Computer Science*, *215*, 556-566.
[5] Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., & Dam, S. (2013). A genetic algorithm (ga) based load balancing strategy for cloud computing. *Procedia Technology*, *10*, 340-347.
[6] Buyya, R., Broberg, J., & Goscinski, A. (2011). Cloud computing. *Principles and Paradigms, Publisher*.
[7] Dikaiakos, M. D., Katsaros, D., Mehra, P., Pallis, G., & Vakali, A. (2009). Cloud computing: Distributed internet computing for IT and scientific research. *IEEE Internet computing*, *13*(5), 10-13.
[8] A Vouk, M. (2008). Cloud computing–issues, research and implementations. *Journal of computing and information technology*, *16*(4), 235-246.
[9] Armstrong, R., Hensgen, D., & Kidd, T. (1998, March). The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions. In *Proceedings Seventh Heterogeneous Computing Workshop (HCW'98)* (pp. 79-87). IEEE.
[10] Xu, Y., Wu, L., Guo, L., Chen, Z., Yang, L., & Shi, Z. (2011, August). An Intelligent Load Balancing Algorithm Towards Efficient Cloud Computing. In *AI for Data Center Management and Cloud Computing*.
[11] Mishra, R., & Jaiswal, A. (2012). Ant colony optimization: A solution of load balancing in cloud. *International Journal of Web & Semantic Technology*, *3*(2), 33.
[12] Mondal, B., Dasgupta, K., & Dutta, P. (2012). Load balancing in cloud computing using stochastic hill climbing-a soft computing approach. *Procedia Technology*, *4*, 783-789.
[13] Lagwal, M., & Bhardwaj, N. (2017, June). Load balancing in cloud computing using genetic algorithm. In *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 560-565). IEEE.
[14] Wickremasinghe, B., Calheiros, R. N., & Buyya, R. (2010, April). Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In *2010 24th IEEE international conference on advanced information networking and applications* (pp. 446-452). IEEE.
[15] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, *41*(1), 23-50.