IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 11, November 2021

DOI: 10.17148/IJARCCE.2021.101109

# A Genetic Algorithm (Ga) Based Load Balancing Strategy For Cloud Computing

**Nita J Goswami[1], Asst. Prof Jinal Patel[2]**

[1]Student of M.E, C.E Department, Gujarat Technology University, Himatnagar, Gujarat

[2]Asst Prof. Grow More Faculty of Engineering, Himatnagar, Gujarat

**Abstract:** The next-generation of cloud computing will thrive on how effectively the infrastructure are instantiated and available resources utilized dynamically. Load balancing which is one of the main challenges in Cloud computing, distributes the dynamic workload across multiple nodes to ensure that no single resource is either overwhelmed or underutilized. This can be considered as an optimization problem and a good load balancer should adapt its strategy to the changing environment and the types of tasks. This paper proposes a novel load balancing strategy using Genetic Algorithm (GA). The algorithm thrives to balance the load of the cloud infrastructure while trying minimizing the make span of a given tasks set. The proposed load balancing strategy has been simulated using the CloudAnalyst simulator. Simulation results for a typical sample application shows that the proposed algorithm outperformed the existing approaches like First Come First Serve (FCFS), Round Robing (RR) and a local search algorithm Stochastic Hill Climbing (SHC).

**Keywords:** Cloud Computing; Load balancing; Genetic Algorithm.

## 1. INTRODUCTION

A new paradigm of large scale distributed computing is "Cloud". It describes a category of sophisticated on- demand computing services offered by cloud service providers, such as Amazon, Google, and Microsoft [1]. This computing infrastructure is used by businesses and individuals to access applications from anywhere in the world on demand. Any cloud service provider offers computing, storage, and software "as a service". Cloud computing accommodates provisioning and de-provisioning on demand and helps any organization in avoiding the capital costs of software and hardware [2]. Due to the exponential growth of cloud computing, it has been widely adopted by the industry and thus making a rapid expansion in availability of resource in the Internet. As the size of cloud scales up cloud computing service providers requires handling of massive requests. The primary challenge then becomes to keep the performance same or better whenever such an outburst occurs. Thus in spite of glorious future of Cloud Computing, many critical problems still need to be explored for its perfect realization [3]. One of these issues is **Load balancing**.

It is considered as one of the prerequisites to utilize the full resources of parallel and distributed systems. Load Balancing allows distribution of workload across one or more servers, datacentres, hard drives, or other computing resources, thereby providing Cloud Service Providers (CSP) a mechanism to distribute application requests across any number of application deployments located in data centres. Load balancing mechanisms can be broadly categorized as centralized or decentralized, dynamic or static, and periodic or non-periodic. There has been few research on load balancing techniques in cloud computing environment. Armstrong et. al. in [4] uses Minimum Execution Time (MET) to assign order to each job in arbitrary manner to the nodes on which it is expected to be executed fastest, regardless of the current load on that node. Use of some existing scheduling techniques like Min-Min, Round Robin and FCFS for load balancing also exist in literature. An intelligent method for load balancing has been proposed by Yang Xu et. al. [5]. It proposes a novel model to balance data distribution to improve cloud computing performance in data-intensive applications, such as distributed data mining. A few soft computing techniques like Ant Colony [6] is also reported in literature.

In this paper Genetic Algorithm (GA) has been used as a soft computing approach, which uses the mechanism of natural selection strategy. CloudAnalyst - A CloudSim based Visual Modeler has been used for simulation and anal- ysis of the algorithm. The performance of the algorithm is compared with two commonly used scheduling algorithm FCFS and RR and a local search algorithm Stochastic hill climbing [7]. The rest of paper is organized as follows. Section 2 proposes the GA algorithm for load balancing. Section 3 presents the simulation results and its analysis with an overview of CloudAnalyst in Section 3.1 for the sake of completeness. Finally, Section 4 concludes this paper.

## 2. GA FOR LOAD BALANCING IN CLOUD COMPUTING

Though Cloud computing is dynamic but at any particular instance the said problem of load balancing can be formulated as allocating N number of jobs submitted by cloud users to M number of processing units in the Cloud. Each of the processing unit will have a processing unit vector (PUV) indicating current status of processing unit utilization. This

  

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 11, November 2021

DOI: 10.17148/IJARCCE.2021.101109

vector consists of MIPS, indicating how many million instructions can be executed by that machine per second, α, cost of execution of instruction and delay cost L. The delay cost is an estimate of penalty, which Cloud service provider needs to pay to customer in the event of job finishing actual time being more than the deadline advertised by the service provider.

$PUV = f(MIPS, \alpha, L)$ (1)

Similarly each job submitted by cloud user can be represented by a job unit vector (JUV). Thus the attribute of different jobs can can be represented by 2.

$JUV = f(t, NIC, AT, wc)$ (2)

where, t represents the type of service required by the job, Software as a Service (SAAS),Infrastructure as a Service (IAAS) and Platform-as-a-Service (PAAS). NIC represents the number of instructions present in the job, this is count of instruction in the job determined by the processor. Job arrival time (AT ) indicate wall clock time of arrival of job in the system and worst case completion time (wc) is the minimum time required to complete the job by a processing unit.

The Cloud service provider needs to allocate these N jobs among M number of processors such that cost function $\zeta$ as indicated in equation 3, is minimized.

$\zeta = w_1 * \alpha(NIC \div MIPS) + w_2 * L$ (3)

where $w_1$ and $w_2$ are predefined weights. It is very difficult to decide/optimize the weights, one criterion could be that more general the factor is, larger is the weight. Another logic is users preference or importance given to a particular factor over the other. Here the later approach has been used and the optimization in then performed on the given set of weights. The weights are considered as w1 = 0.8 and w2 = 0.2 such that their summation is 1.

Thus the load balancing problem are complex and can be considered as computationally intractable problem.

Such a problem cannot be formulated by linear programming hence it is quite difficult to find the globally optimal solution by using deterministic polynomial time algorithms or rules. GAs [8] are considered as one of the most widely used artificial intelligent techniques used primarily for effective search and optimization. It is a stochastic searching algorithm based on the mechanisms of natural selection and genetics. GAs has been proven to be very efficient and stable in searching out global optimum solutions, specially in complex and/ or vast search space. In this paper, GA has been proposed as a load balancing technique for cloud computing to find a global optimum processors for job in a cloud. The arrival of job is considered as linear and rescheduling of jobs is not considered as the solution will be global optimum in nature. The proposed technique is explained in the next section.

### 2.1. Proposed Algorithm

A simple GA is composed of three operations: selection, genetic operation, and replacement. The advantage of this technique is that it can handle a vast search space, applicable to complex objective function and can avoid being trapping into local optimal solution. The working principle of GA used for the load balancing in Cloud computing is depicted in figure 2 and details of GA are described as follows.

1. **Initial population generation:** GA works on fixed bit string representation of individual solution. So, all the possible solutions in the solution space are encoded into binary strings. From this an initial population of ten (10) many chromosomes are selected randomly.
2. **Crossover:** The objective of this step is to select most of the times the best fitted pair of individuals for crossover. The fitness value of each individual chromosome is calculated using the fitness function as given in 3. This pool of chromosomes undergoes a random single point crossover, where depending upon the crossover point, the portion lying on one side of crossover site is exchanged with the other side. Thus it generates a new pair of individuals.
3. **Mutation:** Now a very small value (0.05) is picked up as mutation probability. Depending upon the mutation value the bits of the chromosomes, are toggled from 1 to 0 or 0 to 1. The output of this is a new mating pool ready for crossover.

This GA process is repeated till either the fittest chromosome (optimal solution) is found or the termination condition (maximum number of iteration) is exceeded.

**The proposed algorithm is as given below:**

**Step 1:** Randomly initialize a population of processing unit after encoding them into binary strings [Start].

**Step 2:** Evaluate the fitness value of each population using equation 3 [Fitness].

**Step 3: While** either maximum number of iteration are exceeded or optimum solution is found **Do**:

**Step 3(a):** Consider chromosome with lowest fitness twice and eliminate the chromosome with highest fitness value to construct the mating pool [Selection].

**Step 3(b):** Perform single point crossover by randomly selecting the crossover point to form new offspring [Crossover].

**Step 3(c):** Mutate new offspring with a mutation probability of (0.05) [Mutation].

**Step 3(d):** Place new offspring as new population and use this population for next round of iteration [Ac- cepting].

**Step 3(e):** Test for the end condition [Test].

**Step 4:** End.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 11, November 2021

DOI: 10.17148/IJARCCE.2021.101109

## 3. SIMULATION RESULTS AND ANALYSIS

The proposed GA algorithm is simulated by considering a scenario of "Internet Banking" of an international bank in a simulation toolkit CloudAnalyst [9].

### 3.1. Cloud Analyst

To support the infrastructure and application-level requirements arising from Cloud computing paradigm, such as modelling of on demand virtualization enabled resource simulators are required. Few simulators like CloudSim [10] and CloudAnalyst [9] are available. CloudAnalyst has been used in this paper as a simulation tool. A snapshot of the GUI of CloudAnalyst simulation toolkit is shown in figure 1(a) and its architecture in depicted figure 1(b).

**Fig. 1: Snapshot of CloudAnalyst (a)GUI of CloudAnalyst (b) Architecture of CloudAnalyst build on CloudSim**

CloudAnalyst developed on CloudSim is a GUI based simulation tool. CloudSim facilitates modelling, simulation and other experimentation on cloud programmatically. CloudAnalyst uses the functionalities of CloudSim and does a GUI based simulation. It allows setting of parameters for setting a simulation environment to study any research problem of cloud. Based on the parameters the tool computes, the simulation result also shows them in graphical form.
A hypothetical configuration has been generated using CloudAnalyst. Where, the world is divided into 6 ''Regions" that coincide with the 6 main continents in the World. Six "User bases" modeling a group of users representing the six major continents of the world is considered. A single time zone has been considered for the all the user bases and it is assumed that there are varied number of online registered users during peak hours, out of which only one twentieth is online during the off-peak hours. Table 1 lists the details of user bases used for experimentation. Each simulated "data centre hosts" has a particular amount of virtual machines (VMs) dedicated for the application. Each of the Machines has 4 GB of RAM and 100GB of storage and each machine has 4 CPUs, and each CPU has a capacity power of 10000 MIPS.

### 3.2. Simulation setup

Several scenarios are considered for experimentation starting with only a single centralized cloud Data Center (DC). Thus all user requests around the world are processed by this single DC having 25, 50 and 75 VMs of Cloud Configuration (CCs) allocated to the application. This simulation setup is described in Table 2 with calculated overall average Response Time (RT ) in ms for GA, SHC, RR and FCFS. A performance analysis graph of the same is depicted in figure 2. Next two DCs are considered each having a combination of 25, 50 and 75 VMs as given in Table 3 and performance analysis is reported in figure 3. Subsequently three, four, five and six DCs are considered with combination 25, 50 and 75 VMs for each CCs as given in Tables 4, 5 6 and 7. The corresponding performance analysis graphs are displayed beside them in figures 4, 5, 6 and 7.

**Table 1: Configuration of simulation environment**

| S.No | User Base | Region | Online users during peak hrs. | Online users during off-peak hrs. |
|---|---|---|---|---|
| 1. | UB1 | 0-N.America | 4,70,000 | 80,000 |
| 2. | UB2 | 1-S.America | 6,00,000 | 1,10,000 |
| 3. | UB3 | 2-Europe | 3,50,000 | 65,000 |
| 4. | UB4 | 3-Asia | 8,00,000 | 1,25,000 |
| 5. | UB5 | 4-Africa | 1,25,000 | 12,000 |
| 6. | UB6 | 5-Oceania | 1,50,000 | 30,500 |

  

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 11, November 2021

DOI: 10.17148/IJARCCE.2021.101109

**Complexity analysis**

The complexity analysis of any algorithm includes computation complexity (time complexity) analysis and space complexity analysis. The basic operations performed in genetic algorithm are fitness calculation and selection oper- ation, crossover operation and mutation operation. In genetic algorithm, the population initialization is considered to be the preprocessing hence its complexity is not considered for analysis. For encoding into binary string a time complexity of at most $n_1$, for evaluation of cost function 3 it is at most (c k) for checking cost c of k number of chromosomes. Selection process has a time complexity of at most m, for single point crossover the time complexity is at most m, where m is chromosome length and for mutation at any place it is again m. The three operation of GA are repeated iteratively till the stopping criteria is met so the total time complexity G is given by,

$$G = O\{n_1 + (c \times k) + (n_2 + 1)(m + m + m)\} \quad (4)$$

**Table 2: Simulation scenario and calculated overall average response time (RT ) in (ms)**

| S.No | Cloud con- figuration | DC specification | RT using GA | RT using SHC | RT using RR | RT using FCFS |
|---|---|---|---|---|---|---|
| 1. | CC1 | Each with 25 VMs | 329.01 | 329.02 | 330 | 330.11 |
| 2. | CC2 | Each with 50 VMs | 328.97 | 329.01 | 329.42 | 329.42 |
| 3. | CC3 | Each with 75 VMs | 244.00 | 329.34 | 329.67 | 329.44 |

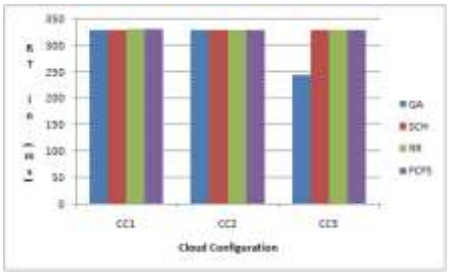


**Fig. 2: Performance analysis of proposed GA with SHC, FCFS and RR**

**Results using one data center**

**Table 3: Simulation scenario and calculated overall average response time (RT ) in (ms)**

| S.No | Cloud con- figuration | DC specification | RT using GA | RT using SHC | RT using RR | RT using FCFS |
|---|---|---|---|---|---|---|
| 1. | CC1 | Two DCs with 25 VMs each | 360.77 | 365.44 | 371.27 | 376.34 |
| 2. | CC2 | Two DCs with 50 VMs each | 355.72 | 360.15 | 367.49 | 372.52 |
| 3. | CC3 | Two DCs with 75 VMs each | 355.32 | 359.73 | 364.78 | 370.56 |
| 4. | CC4 | Two DCs with 25, 50 VMs each | 350.58 | 356.72 | 362.91 | 368.87 |
| 5. | CC5 | Two DCs with 25, 75 VMs each | 351.56 | 357.23 | 364.45 | 367.23 |
| 6. | CC6 | Two DCs with 75, 50 VMs each | 352.01 | 357.04 | 361.61 | 361.01 |

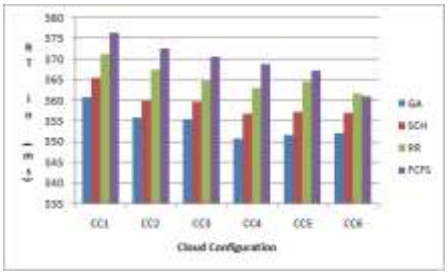


**Fig. 3: Performance analysis of proposed GA with SHC, FCFS and RR**

**Results using two data centers**

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 11, November 2021

DOI: 10.17148/IJARCCE.2021.101109

**Table 4: Simulation scenario and calculated overall average response time (RT ) in (ms)**

| S.No | Cloud configuration | DC specification | RT using GA | RT using SHC | RT using RR | RT using FCFS |
|---|---|---|---|---|---|---|
| 1. | CC1 | Each with 25 VMs | 350.32 | 356.82 | 361.17 | 363.34 |
| 2. | CC2 | Each with 50 VMs | 350.19 | 355.25 | 362.49 | 363.52 |
| 3. | CC3 | Each with 75 VMs | 346.01 | 350.73 | 356.18 | 361.56 |
| 4. | CC4 | Each with 25,50 and 75 VMs | 345.98 | 350.01 | 356.21 | 360.87 |

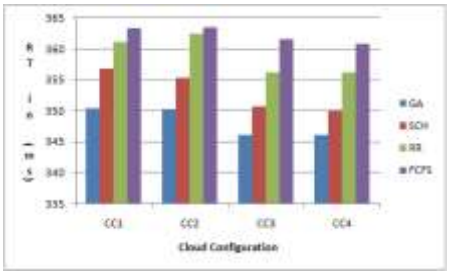


**Fig. 4: Performance analysis of proposed GA with SHC, FCFS and RR**

Results using **three data centers**

**Table 5: Simulation scenario and calculated overall average response time (RT ) in (ms)**

| S.No | Cloud configuration | DC specification | RT using GA | RT using SHC | RT using RR | RT using FCFS |
|---|---|---|---|---|---|---|
| 1. | CC1 | Each with 25 VMs | 348.85 | 354.35 | 359.35 | 360.95 |
| 2. | CC2 | Each with 50 VMs | 345.54 | 350.71 | 356.93 | 359.97 |
| 3. | CC3 | Each with 75 VMs | 340.65 | 346.46 | 352.09 | 358.44 |
| 4. | CC4 | Each with 25, 50 and 75 VMs | 337.88 | 344.31 | 351 | 355.94 |

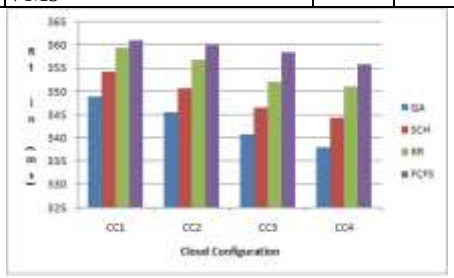


**Fig. 5: Performance analysis of proposed GA with SHC, FCFS and RR**

Results using **four data centres**

**Table 6: Simulation scenario and calculated overall average response time (RT ) in (ms)**

| S.No | Cloud configuration | DC specification | RT using GA | RT using SHC | RT using RR | RT using FCFS |
|---|---|---|---|---|---|---|
| 1. | CC1 | Each with 25 VMs | 335.64 | 342.86 | 348.57 | 352.05 |
| 2. | CC2 | Each with 50 VMs | 326.02 | 332.84 | 339.76 | 345.44 |
| 3. | CC3 | Each with 75 VMs | 322.93 | 329.46 | 335.88 | 342.79 |
| 4. | CC4 | Each with 25, 50 and 75 VMs | 319.98 | 326.64 | 334.01 | 338.01 |

  

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 11, November 2021

DOI: 10.17148/IJARCCE.2021.101109

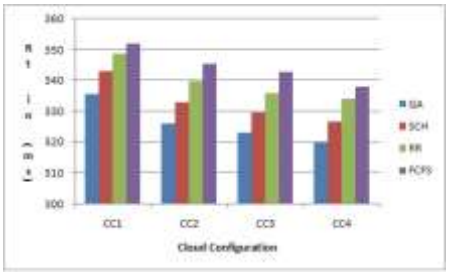


**Fig. 6: Performance analysis of proposed GA with SHC, FCFS and RR**

Results using **five data centers**

| S.No | Cloud con-figuration | DC specification | RT using GA | RT using SHC | RT using RR | RT using FCFS |
|---|---|---|---|---|---|---|
| 1. | CC1 | Each with 25 VMs | 330.54 | 336.96 | 341.87 | 349.26 |
| 2. | CC2 | Each with 50 VMs | 323.01 | 331.56 | 338.14 | 344.04 |
| 3. | CC3 | Each with 75 VMs | 321.54 | 327.78 | 333.67 | 339.87 |
| 4. | CC4 | Each with 25, 50 and 75 VMs | 315.33 | 323.56 | 331.49 | 338.29 |

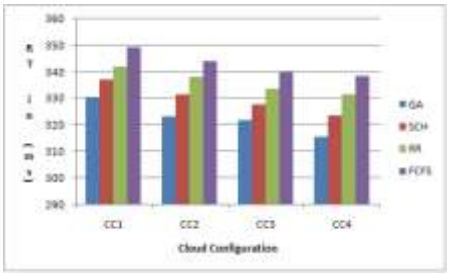


**Fig. 7: Performance analysis of proposed GA with SHC, FCFS and RR**

Results using **six data centers**

## 4. CONCLUSION

In this paper, a genetic algorithm based load balancing strategy for Cloud Computing has been developed to provide an efficient utilization of resource in cloud environment. Analysis of the results, indicates that the proposed strategy for load balancing not only outperforms a few existing techniques but also guarantees the QoS requirement of customer job. Though it has been assumed that all the jobs are of the same priority which may not be the actual case, this can be accommodated in the JUV and subsequently taken care in fitness function. Also a very simple approach of GA has been used however variation of the crossover and selection strategies could be applied as a future work for getting more efficient and tuned results.

## REFERENCES

[1] Rajkumar Buyya, James Broberg and Andrzej Goscinski CLOUD COMPUTING Principles and Paradigms, Jhon Wiley & Sons, 2011.
[2] M. D. Dikaiakos, G. Pallis, D. Katsa, P. Mehra, and A. Vakali, "Cloud Computing: Distributed Internet Computing for IT and Scientific Research", in Proc. of IEEE Journal of Internet Computing, Vol. 13, No. 5, pp. 10-13, 2009.
[3] A. Vouk, "Cloud computing- issues, research and implementations", in Proc. of Information Technology Interfaces, pp. 31-40, 2008.
[4] T. R. Armstrong, D. Hensgen, "The relative performance of various mapping algorithms is independent of sizable variances in runtime predic- tions", in Proc. of 7th IEEE Heterogeneous Computing Workshop (HCW 98), pp. 79-87,1998.
[5] Yang Xu, Lei Wu, Liying Guo, Zheng Chen,Lai Yang, Zhongzhi Shi, "An Intelligent Load Balancing Algorithm Towards Efficient Cloud Computing", in Proc. of AI for Data Center Management and Cloud Computing: Papers, from the 2011 AAAI Workshop (WS-11-08), pp. 27–32, 2008.
[6] Ratan Mishra and Anant Jaiswal, "Ant colony Optimization: A Solution of Load balancing in Cloud",in International Journal of Web & Semantic Technology (IJWesT), Vol.3, No.2, pp. 33–50, 2012.
[7] Brototi Mondal,Kousik Dasgupta and Paramartha Dutta, "Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Com- puting Approach", in Proc. of C3IT-2012, Elsevier, Procedia Technology 4(2012), pp.783-789, 2012.
[8] D. E. Goldberg, Genetic algorithms in search, optimization, and machine learning,Addison-Wesley, 1989.
[9] B. Wickremasinghe, R. N. Calheiros and R. Buyya, "Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environ- ments and applications", in Proc. of Proceedings of the 24th International Conference on Advanced Information Networking and Applications (AINA 2010), Perth, Australia, pp.446-452, 2010.
[10] R.N.Calheiros,R.Ranjan,A.Beloglazov,C.Rose,R.Buyya,"Cloudsim:A toolkit for modeling and simulation of cloud computing environ- ments and evaluation of resource provisioning algorithms",in Software:Practice and Experience(SPE),Vol:41,No:1,ISSN:0038-0644,Wiley Press,USA,pp:23-50,2011.