International Journal of Engineering & Technology, 7 (3.28) (2018) 290-294

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper



A Genetic Algorithm for Optimal Job Scheduling and Load Balancing in Cloud Computing

Zarina Mohamad[1]*, Aminu Abdulkadir Mahmoud[1], Wan Nur Shuhadah Wan Nik[1], Mohamad Afendee Mohamed[1], Mustafa Mat Deris[2]

[1]Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Besut Campus, Terengganu, Malaysia
[2]Faculty of Computer Science and Information Technology, Universiti Tun Hussien Onn Malaysia, Johor, Malaysia
**Corresponding author E-mail: zarina@unisza.edu.my*

Abstract

Cloud Computing is a new concept for pool of virtualized computer resources. There are many approaches available to improve the job scheduling and load balancing in cloud environment. However, this research focused on the Job scheduling in cloud computing environment at Virtual Machines level by considering their bandwidth and RAM size. Three (3) traditional scheduling techniques are employed (min-min, max-min, and suffrage) to find the minimum completion time possible for a given job(s) for each Virtual Machine (VM). The Genetic Algorithm (GA) is applied after the job scheduling is completed for load balancing and to attained the Quality of Service (QoS) required by properly utilizing the resources available. A CloudSim simulator is used to test the efficiency of the proposed technique. We found that the proposed technique called Random Make Genetic Optimizer (RMGO) can minimize the makespan as compared to classical job scheduling techniques.

Keywords: *Job Scheduling; Load Balancing; Cloud Computing; Genetic Algorithm.*

1. Introduction

Cloud Computing is a model of providing resources and capabilities of Information Technology such as applications, storages, infrastructure, communication, and collaboration via services offered by cloud providers on demand [1-4]. In order to successfully achieve this target, cloud management plays an important role by providing dynamic resource scheduling, load balancing, secure data backup that is available real-time around the clock.

Numerous complicated factors need to be considered in order to perform job scheduling algorithm. The factors are changing dynamically according to the user request and resource availability. Improper resource scheduling management courses the system performance to be deteriorated.

Load balancing should also be considered in the scheduling algorithm, so that the load is equally spread on each node of the cloud. These situations optimize resource utilization, throughput and response time. Hence, dynamic scheduling in cloud computing environment deserves special attention by the researchers. Job scheduling can be done at datacenter level with available information of that particular datacenter from Cloud Information Service (CIS) or at either physical machine level in a particular datacenter or at virtual machine level within the physical machine. Our focused on this research will be job scheduling at virtual machine level because it is at this level that the jobs are executed. Nevertheless, to achieve QoS by proper scheduling of a job to an appropriate machine that has the shortest completion time.

In this paper a job scheduling and load balancing technique is proposed. The technique has two main phases i.e. job scheduling phase and load balancing phase. The three classical scheduling techniques are employed for job scheduling phase, while the Genetic Algorithm (GA) is used for load balancing phase. Simulation shows that the proposed technique improves of makespan.

The rest of the paper is organized as follows: Section 2 discusses on the related works. The proposed technique is discussed in Section 3. In Section 4 the simulation and the results discussion are presented, Section 5 contains conclusion.

2. Related Works

Cloud Computing is receiving more attention, both in publications and among users from individual organization to government [5]. There are several publications and research papers on this new field of modern day computing with a lot of expectations on its deliverance, especially where this research laid it emphases i.e. on job scheduling/ job allocation in cloud environment [6]. In [7] is a research paper in which the researcher proposed the used of Suffrage coupled with Genetic Algorithm, the objective of the research is to minimize the makespan of the job(s).

A research was conducted with the aim of addressing load balancing problems in Virtual Machine (VM) resource scheduling [8]. This research adopted genetic algorithm to address the problems of load balancing and high migration cost after VM is scheduled. In another research [9], an algorithm is proposed using two conventional task scheduling algorithms and this algorithm is based on min-min and max-min. The algorithm uses certain heuristics to select between the two mentioned conventional algorithms so that the overall makespan is minimized.

The tasks are scheduled on machines in either space shared or time shared manner and it was also compared with First Come First Serve (FCFS) scheduling. In addition, there was also a research based on scheduling in cloud computing environment [10] that proposed a generalized priority algorithm in order to achieve

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

efficient task execution. The algorithm was then compared with First Come First Serve (FCFS) and Round Robin Scheduling in terms of their performances. The test was conducted in CloudSim toolkit and the result shows that it gives better performance compared to other traditional scheduling algorithm. However, research in [11] proposed a resource dispatched mechanism using genetic algorithm based on Support Vector Regression (SVR) for re-scheduling of resources. The objective of the research is to design a sub-optimal resource scheduling system in cloud computing environment to accomplish tasks with lowest possible cost.

The proposed optimization system can estimate the number of resource utilization according to Service Level Agreement (SLA) more accurately. In another research [6], a strategy for effective load balancing using genetic algorithm is proposed with the aim of facilitating load balancing and reducing migration cost. Meanwhile in [12] a novel load balancing strategy based on genetic algorithm is proposed that will thrive in load balancing of cloud infrastructure and minimize makespan of a given task. When this algorithm was tested, the results show that it outperformed the existing approaches such as FCFS, Round Robin (RR) and Stochastic Hill Climbing (SHC). Research in [13] is a review of some algorithms suggested in some research works on Virtual Machine Management.

The paper reviewed the impact of all the stated algorithms with regards to various performance matrices and provides an overview of their impact on the latest approach in the field of VM management. The research focuses it review on renewable energy and parameters that include bandwidth and latency delay. In another research paper [14], the used of Artificial Bee Colony (ABC) for VM scheduling management in cloud computing environment was proposed. The research aimed to propose an algorithm that would enhance the performance of cloud tasks scheduling, increase the makespan and performance of resource utilization. The result of simulation using CloudSim tools shows that ABC algorithm performed significantly in changing the environment, load balancing and makespan of data processing. In addition, [15] proposed a meta-heuristic based scheduling algorithm by merging two existing techniques i.e. Shortest Cloudlet Fastest Processor (SCFP) and Longest Cloudlet Fastest Processor (LCFP) with GA. The aim of this research is to propose an algorithm that will minimize execution time and cost of a given task in cloud computing environment. The simulation results show that the proposed algorithm exhibit a good performance under heavy loads. Moreover, in [16], an algorithm is proposed that can find a fast mapping for tasks scheduling using genetic algorithm. The research aimed to propose an algorithm to improve system performance and quality of service. Simulation results show that the proposed algorithm increases the mapping time by using the throughput as objective function.

In a nutshell, almost all the papers that were reviewed have some similarities in aims and objectives with this research. Though some credit must be given to the researchers for their framework and the proposed algorithms and for their contributions to address the work on job scheduling and load balancing. However, there are still some needs for improvements in terms of management of resources because not only proper job scheduling is required to achieve the target, but also the load balancing. Load balancing alone cannot address the issue of enhancing resource management in a cloud computing environment.

In this research, the proposed algorithm coined as Random Make Genetic Optimizer (RMGO) which composes of three techniques i.e. Min-min, Max-min and Suffrage combine with genetic algorithm can improve the job scheduling and load balancing in the system. When the RMGO was tested on CloudSim tool-kit, the result shows that better performance has been accomplished. Having a harmonized algorithm that can inherit all the features of the reviewed algorithms and their functions, but in different and enhanced manner is already a considerable contribution.

3. Proposed Random Make Genetic Optimizer

3.1 Basic structure of Random Make Genetic Optimizer

The proposed technique focuses on job scheduling and load balancing on virtual machines. There are three scheduling approaches known as Min-min, Min-max and Suffrage being employed in the Random Make Genetic Optimizer (RMGO) as shown in Figure 1. Its goal is to achieve quality of service by proper scheduling of a job to an appropriate machine that has the shortest completion time. The primary objective is to choose the one that produce the shortest completion time when the jobs are scheduled. After which the result will be encoded as parameters of genetic algorithm.

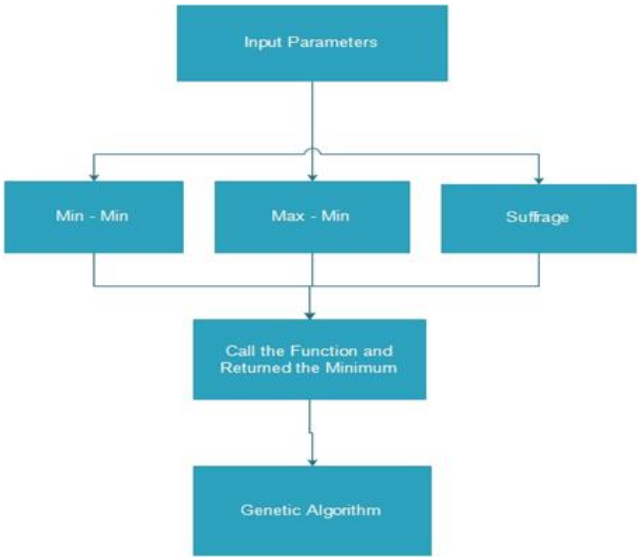


Fig. 1: Basic Structure of RMGO

3.2. RMGO Algorithm

1. Input all the parameters
   a. No. of Physical Machine (i)
   b. No. of Virtual Machines (j) in each Physical Machine (i)
   c. MIPS of each Virtual Machine (j)
   d. Cloudlet(s) size/File size of submitted jobs
   e. No. of Cloudlets (k)
2. **While** No. of physical machine (i) is not exceeded **do**
3. **For** all Virtual Machine (j)
4. **For** all Cloudlet (k)
5. $\text{Cloudlet Execution Time } (C_{kj}) = \frac{\text{Cloudlet size(k)}}{\text{MIPS (j)}}$
6. **End For**
7. **End For**
8.
9. **Begin Function** (Min – Min)
10. **For** all Cloudlets (k)
11. **For** all Virtual Machines (j)
12. $\text{Find } \text{Min}(C_{kj}) = T_k$
13. **End For**
14. $\text{Find } \text{Min}(T_k)$
15. $Assign\ C_k = V_j$
16. $Update\ C_{kj}$ for all values of k
17 $Remove\ C_{kj}$ from the list for all values of j
18. End For
19. $Return\ (Scheduled\ List, \max(C_k))$
20. **End Function** (Min – Min)
21. **Begin Function** (Max – Min)
22. **For** all Cloudlets (k)
23. **For** all Virtual Machines (j)
24. $\text{Find } \text{Min}(C_{kj}) = T_k$

25. **End For** $P_i = \frac{F_i}{\sum F_i}$
26. Find $\mathrm{Max}(T_k)$
27. $Assign\ C_k = V_j$
28. $Update\ C_{kj}$ for all values of k
29. $Remove\ C_{kj}$ from the list for all values of j
30. End For
31. $Return\ (Scheduled\ List, \max(C_k))$
32. **End Function** (Max – Min)
34. **Begin Function** (Suffrage)
35. **For** all Cloudlets (k)
36. **For** all Virtual Machines (j)
37. $T_k = 2^{nd}\mathrm{Min}(C_{kj}) - 1^{st}\mathrm{Min}(C_{kj})$ for all values of j
38. **End For**
39. Find $S_r = \mathrm{Max}(T_k)$
40. $Assign\ Min\ (S_r) = V_j$ for all values of j
41. $Update\ C_{kj}$ for all values of k
42. $Remove\ C_{kj}$ from the list for all values of j
43. End For
44. $Return\ (Scheduled\ List, \max(C_k))$
45. **End Function** (Suffrage)
46. Find Min (Function (Min-Min, Max-Min, Suffrage))
47. **Begin Main** (Genetic Algorithm)
48. [Initialization] Create population of n chromosomes using the returned min
49. [Fitness Function] Calculate the fitness function F_i using this formula
50. $F_i = \frac{C_i}{Makespan} \times MIPS_i$
51. [Selection] Using Roulette Wheel selection probability $P_i = \frac{F_i}{\sum F_i}$ to select the
52. Fittest individuals.
53. [Crossover] Generate new offspring using crossover probability
54. [Mutation] Mutate at some point(s) where necessary using mutation probability
55. [Accepting] New offspring is now part of the new generation
56. [Replacement] Replace the initial population with new generation
57. **End Main** (Genetic Algorithm)
58. Re- assign job(s) to virtual machines where necessary.

Fig. 2: The RMGO Algorithm

An algorithm called Random Make Genetic Optimizer (RMGO) is proposed in Figure 2 based on the structure as depicted in Figure 1. The following is the detail explanation of the algorithm:

Line 1, from a. to e., are all parameters needed to be inputted by the user.

Line 2 to Line 8 are nested loops that would take each physical machine and its corresponding virtual machines to find Cloudlet Execution Time (Ckj) of all the virtual machines within that physical machine until Cloudlet Execution Time (Ckj) of all the physical machines and their corresponding virtual machines are found.

Line 9 to Line 20 is a function Min – Min, which is used to schedule the cloudlet(s) using Min – Min technique.

Line 21 to Line 32 is a function Max – Min, which is used to schedule the cloudlet(s) using Max – Min technique.

Line 34 to Line 45 is a function Suffrage, which is used to schedule the cloudlet(s) using Suffrage technique.

Line 46 is a function call that would return the minimum among the three (3) called functions i.e. Min – Min, Max – Min, and Suffrage, which one has the smallest completion time possible.

Line 47 is a beginning of Main named Genetic Algorithm.

Line 48 Initialization, a population of n chromosomes would be created where n is the number of virtual machines from the returned minimum and the corresponding workload of each virtual machine.

Line 49 to Line 50 a fitness function is calculated using the relation

$$F_i = \frac{C_i}{Makespan} \times MIPS_i$$

where Ci is the completion time of workload(s) by machine (i), MIPSi is the speed of machine (i) and makespan is the maximum completion time of all the machines.

Line 51 to Line 52, fittest individuals would be selected using Roulette Wheel selection probability i.e.

$$P_i = \frac{F_i}{\sum F_i}$$

where Fi is the Fitness Function found in Line 50.

Line 53 is where the fittest individuals would be crossed over using crossover probability.

Line 54 is where mutation would take place at some point(s) to likely repair the damage that might be caused during crossover where necessary using mutation probability.

Line 55 Accepting and putting the individual chromosome(s) into a new population.

Line 56 Replacing initial population with new population.

Line 57 End of the Main Genetic Algorithm.

Line 58 Re-assigning job(s) to virtual machines where necessary.

4. Results and Discussion

The simulation is carried out with different parameters in CloudSim simulator. The components of parameters are different number of VMs, cloudlets, hosts and datacenters in order to check and balance between the techniques performance. However, the aim is to ensure that the RMGO performed better than all the individual techniques integrated in terms of completion time and resource load balancing. Meanwhile three different simulation tests are carried out using the same parameters on all the integrated techniques, RMGO and FCFS. The purpose of choosing to conduct the three different experiments (Tests) is to test and ensure that the RMGO can perform under different setups and scenarios. Furthermore, different setup and components are used in each test to ensure that the RMGO is fully evaluated under different environment with different setup in order to ascertain its performance. In addition, in the first instance experiment is carried out with 1 datacenter and 1 host while, the second and the third experiment were conducted with 2 datacenters and 10 hosts. The results of each test are presented and their components are considered as Test 1, and Test 2 parameters, respectively. Table 1 shows the Test 1 parameters setting for CloudSim.

Table 1: Test 1 Parameters Setting for Cloudsim Simulator

Component Type	Parameter	Value
Datacenter	Number of Datacenter Number of Host Scheduling Policy	1 1 Space-Shared
Datacenter Broker	Number of Datacenter Broker	1
VMs	Number of VMs Number of Processing Element (PE) MIPS of PE MIPS of VMS - VM 1 - VM 2 - VM 3 - VM 4 VM RAM size VM Scheduling Policy Bandwidth	4 1 2000 200 300 400 500 2048MB Space-Shared 10000
Cloudlet	Number of Cloudlets - CL 1 - CL 2	10 5200 3000

	- CL 3 - CL 4 - CL 5 - CL 6 - CL 7 - CL 8 - CL 9 - CL 10 Cloudlet Scheduler Policy CPU Utilisation Model RAM Utilisation Model Bandwidth Utilisation Model	2000 7000 6800 6600 5800 5400 5000 4800 Space-Shared Full Full Full

Figure 3 shows the overall execution time of each individual VM for Test 1. It shows that RMGO has the lowest completion time and it is having the least distance of completion time than others, hence it performed better in term of completion time and load balancing.

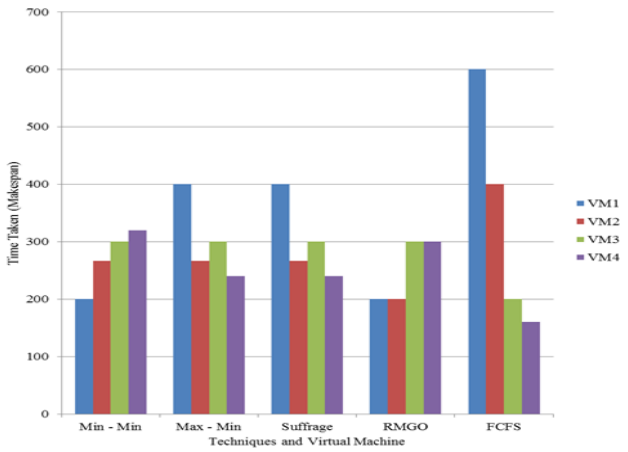


Fig. 3: Comparison of the three algorithms with FCFS and RMGO for Test 1

Figure 4 shows the execution time of each individual VM for Test 2. Based on this figure, RMGO has the lowest completion time and the least distance of completion time than other techniques being the default technique. Hence, RMGO performed better in terms of completion time and load balancing.

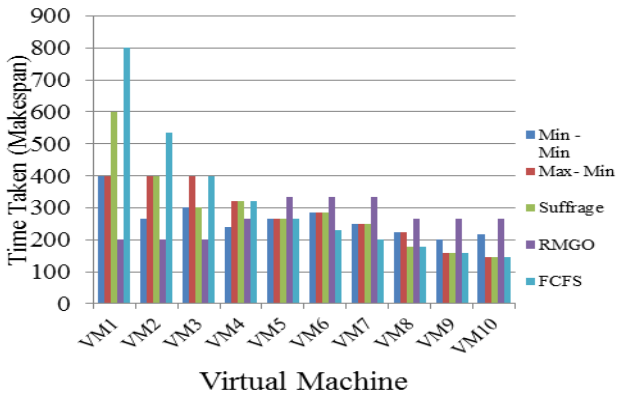


Fig. 4: Comparison of the three algorithms with FCFS and RMGO for Test 2

The simulation and testing of the algorithms in Test 1 and Test 2 parameters had been successfully completed. The results show that in all the techniques with exception of the RMGO, the VMs with smallest MIPS are over utilized and are having the highest execution time but in RMGO the reverse is the case and this will increase the response time. It shows that RMGO is a better alternative for implementation optimal job scheduling and load balancing in cloud environment.

5. Conclusion

There are many researchers conducted on jobs scheduling and load balancing but none of the research uses the integrated min-min, max-min, suffrage and GA. The reason for this integration is because each technique might likely outperform each other at a given period of time. Considering the uniqueness and capability of each technique depending on the nature and size of the jobs (tasks) submitted to the system hence, an integrated technique has been proposed.

The proposed algorithm can enhance job scheduling by producing the minimum execution time in the first phase of the algorithm. When the result is put in to genetic algorithm, it will balance the loads using a fitness function. The impact of this algorithm cannot be over emphasized considering the requirement of proper resource management in cloud computing environment in which this research has laid its priority. The fitness function used in the proposed algorithm is to enhance load balancing among the virtual machines.

There are a number of issues that need to be addressed in order to fully enhance and improve the performance of cloud services that are not covered in this paper. As a result, much work needs to be done for addressing the problems in the future. Meanwhile, our future research will intend to focus on the remaining issues of maintenance in cloud computing environment, especially availability of data storage, power consumption and green computing. Our future work will focus more on resource selection and data replication in the cloud computing environment to improve response time and availability of the system.

Acknowledgement

The authors gratefully acknowledge financial support from the Ministry of Higher Education, under the project FRGS-2-2013-ICT07-UniSZA-0201. The authors also would like to convey an appreciation to Research Management, Innovation and Commercialization Center (RMIC), UniSZA for providing full support towards this study.

References

[1] Google. www.google.com/trends.
[2] Erkoc M F, Kert S B. Cloud computing for distributed university campus: A prototype suggestion. Proceedings of the International Conference the Future of Education, 2014, pp. 1-5.
[3] Srichandana S, Kumar T A, Bibhudatta S. Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm. Future Computing and Informatics Journal, 2018, 1-21, 2018.
[4] Bittencourt L F, Goldman A, Madeira E R M, Fonseca N L S, Sakellariou R. Scheduling in distributed systems: A cloud computing perspective. Computer Science Review, 30, 31-54, 2018.
[5] Huth A, Cebula J. The basics of cloud computing. United States Computer, 2011.
[6] Keshanchia B, Souria A, Navimipour N J. An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: Formal verification, simulation, and statistical testing. Journal of Systems and Software, 124, 1-21, 2017.
[7] Goyal T, Agrawal A. Host scheduling algorithm using genetic algorithm in cloud environment. International Journal of Research in Engineering and Technology, 1(1), 7-12, 2013.
[8] Gu J, Hu J, Zhao T, Sun G. A new resource scheduling strategy based on genetic algorithm cloud computing environment. Journal of Computers, 7(1), 42–52, 2012.
[9] Katyal M, Mishra A. Application of selective algorithm for effective resource provisioning in cloud computing environment. International Journal on Cloud Computing: Services and Architecture, 4(1), 1-10, 2014.
[10] Agarwal A, Jain S. Efficient optimal algorithm of task scheduling in cloud computing environment. International Journal of Computer Trends and Technology, 9(7), 344-349, 2014.
[11] Huang C, Guan C, Chen H, Wang Y, Chang S, Li C, Wang C. An adaptive resource management scheme in cloud computing.

Engineering Applications of Artificial Intelligence, 26, 382-389, 2013.
[12] Rawat S S, Bindal U. Effective load balancing in cloud computing using genetic algorithm. International Journal of Computer Science, Engineering and Information Technology Research, 3(4), 91-98, 2013.
[13] Dasgupta K, Mandal B, Dutta P, Mondal J K, Dam S. A genetic algorithm (GA) based load balancing strategy for cloud computing. Procedia Technology, 10, 340-347, 2013.
[14] Khan D H, Kapgate D, Prasad P S. A review on virtual machine management techniques and scheduling in cloud computing. International Journal of Advanced Research in Computer Science and Software Engineering, 3(12), 838-845, 2013.
[15] Kruekaew B, Kimpan W. Virtual machine scheduling management on cloud computing using artificial bee colony. Proceeding of the International Multi-Conference of Engineers and Computer Scientists, 2014, pp. 12-14.
[16] Kaur S, Verma A. An efficient approach to genetic algorithm for task scheduling in cloud computing environment. International Journal of Information Technology and Computer Science, 10, 74-79, 2012.