Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 8, No. 2, May 2017), Pages: 79-93
www.jacr.iausari.ac.ir

GASA: Presentation of an Initiative Method Based on Genetic Algorithm for Task Scheduling in the Cloud Environment

Somayeh Taherian Dehkordi ✉[1] and Vahid Khatibi Bardsiri[2]
1) Department of Computer Engineering, Kerman Branch, Islamic Azad University, Kerman, Iran
2) Department of Computer Engineering, Bardsir Branch, Islamic Azad University, Kerman, Iran
s.taherian2000@gmail.com; kvahid2@live.utm.my

Received: 2016/06/29; Accepted: 2016/08/17

Abstract

The need for calculating actions has been emerged everywhere and in any time, by advancing of information technology. Cloud computing is the latest response to such needs. Prominent popularity has recently been created for Cloud computing systems. Increasing cloud efficiency is an important subject of consideration. Heterogeneity and diversity among different resources and requests of users in the Cloud computing environment creates complexities and problems in task scheduling in the cloud environment. Scheduling consists of selecting the most appropriate resource with the aim to distribute load in resources, and maximum productivity from them, while it should minimize the response time and the time of completion of each task, as well as minimizing the service costs. In addition to analyzing the Cloud computing system and scheduling aspects in it, it has been tried in this article to provide a combined algorithm for appropriate mapping of tasks to the existing virtual machines for reducing the completing times and increasing the productivity of virtual machines. According to the scheduling parameters, the presented method improves the load balancing according to the Sufferage and genetic algorithm as compared to previous algorithms, while it also reduces the total time of requests. The results of simulating the proposed algorithm in CloudSim environment and comparing it with the studied methods show that the proposed algorithm has reached a more optimized response, both for the load balancing and also for the total completion time.

Keywords: *Cloud Computing, Task Scheduling, Genetic, Sufferage*

1. Introduction

Cloud computing includes a set of resources and services provided on internet for the users to access to their required resources through following the rule of payment for usage. Cloud computing shares the data to each other and provides services, clearly, via internet. By increasing the no. of users, the tasks to be schedules are also increased.

Scheduling in cloud is a mechanism allocating the users' tasks to appropriate resources for execution, and directly affects the cloud performance [1]. One of the important problems in Cloud computing, which has great effects on efficiency, is the task scheduling. Task scheduling is used in cloud environment for the users' tasks in the cloud environment, relative to suitable virtual machine.

Traditional task scheduling algorithms cannot provide scheduling in cloud environment, since they have overhead costs. Thus, the providers tend to use

metaheuristic or combined algorithms [2]. In addition to studying cloud computing and the required services in it, this article has analyzed and compared the tasks scheduling algorithms and techniques. The objective of this article is identifying the weak points of scheduling algorithms in cloud environment and trying to eliminate or rectify them. In fact, by analyzing Cloud computing and scheduling techniques, it is dealt with presenting and proposing a task scheduling algorithm for improving the total completion time. According to relevant considerations, one of the best implementable scheduling algorithms in the cloud environment is Sufferage algorithm, which is an algorithm with the least delay. Although the mentioned algorithm has shown an acceptable performance, bur diversity of requests in the cloud environment has limited the range of applications of this algorithm. Presenting a combined model based on using Sufferage algorithm and genetics metaheuristic algorithm in this domain can lead to improving operational parameters. Studying the research literature shows that the considered combination with the exclusive target function guaranteeing appropriate fitness has no previous records, and can be considered as a new method. In fact combining Sufferage algorithm with metaheuristic algorithms can guarantee the reduction in completion times. The proposed combined algorithm is a new algorithm based on metaheuristic algorithms that tries, while generating less completion time, to guarantee the balanced distribution of tasks between cloud computing resources, in addition to considering the previous works and the advantages/disadvantages of promoted scheduling techniques. To follow, the article will be organized in five sections. Scheduling and related works in this domain are considered in section 2. The procedures for performing the proposed algorithm are stated in section three. Simulation and comparing the proposed algorithm with other algorithms are shown in section four, and finally, conclusion shall be considered in section 5.

2. Related Work

Cloud computing is a computer model that tries to facilitate users' access in accordance with the type of demands they have from the information and calculating resources. With minimum needs to human resources, reducing costs and speeding up the access to the information, this model tries to be responsive to the users. The phrase "cloud" is a metaphor that points to internet. The reason to compare internet to cloud is that like clouds, internet hide its technical details from the users, creating an abstractive layers between technical details and the users [3]. Scheduling in cloud computing environment has nowadays been transformed into one of the most important challenges in distributing systems. Calculating resources in cloud computing systems are presented as virtual machine s. Scheduling algorithms play an important role in such scenarios, since the aim is effective task scheduling, in order to reduce the time and improve operation of resources. The scheduling system controls various tasks in the cloud system for increasing the rate of completion of the task and increasing the productivity of resources, hence increasing the calculating potentials. Therefore, scheduling the tasks in all the physical machines is a critical duty. Task Scheduling is used in cloud environments, for the users' tasks to be properly scheduled in the cloud environment, as compared to virtual machines. Task Scheduling indicates allocation of a limited number of virtual machines to a definite set of tasks, for optimizing some effective criteria.

The aim in scheduling in cloud computing environments is balancing between the time of executing the tasks and managing virtual machines, for establishing the load balance [4].

2.1 Scheduling Algorithm in Cloud Computing

Numerous scheduling algorithms are provided in a cloud computing environment for giving service to the request of users. The aim of scheduling is providing the appropriate turns to the tasks that are in their ready state, and one of them is about to be selected for the required execution. The methods used for selecting the tasks create scheduling algorithms, and each algorithm has its specific criterion to be selected. There are different scheduling algorithms proposed for the cloud computing environment, that their main aim is obtaining high calculating performance, reducing completion time and guaranteeing the load balance [5]. The purpose of most scheduling algorithms is to decrease completion time[1], i.e. implementing tasks is supposed to end as soon as possible. To follow, it will be dealt with analyzing some task scheduling algorithms and some works done in scheduling in clouds.

- **FCFS**

The First Come First Service algorithm is one of the common scheduling algorithms. Schedule keeps a queued new task in accordance with its entrance time. The task that is placed at the beginning of the waiting list is selected for execution. Since this schedule has minimum load, its implementation shall be easy. The aim of this algorithm is establishing just affairs [6].

- **OLB**

The Opportunistic load balancing algorithm is one of the well-known algorithms for guaranteeing the load balance, which allocates a task to a special resource that is available in future without considering the executing time in the machine [7].

- **Max-min**

In this algorithm, a set of minimum times are primarily calculated for each task on the resource. Then, the task with more required time will be selected for allocating to its considered machine. The selected task is eliminated from the set of tasks and by adding the execution time of it to other tasks on the selected resource, the executing time of other tasks will be updated. But, this method has a main problem that can lead to hunger. The criterion of exploiting from resources, overloads, operating capability, response time and efficacy are considered among the scheduling criteria, in this algorithm [8].

- **Min-Min**

This algorithm starts with a set of unallocated tasks. The minimum completion time is calculated for all tasks. Then, the shortest time is selected among the related times, which is the minimum time among all the tasks on each existing resource. Then, according to the minimum time, the task will be scheduled on the related machine. After that, the executing times for all the other tasks on the machine are updated by adding the allocated executing times to the time of executing other tasks on the machine, and the

[1] Makespan

allocated task will be eliminated from the tasking list allocated to the machine. This trend is followed until the time that all the tasks are allocated to the resources [9].

- **Load balance based on servers**

A load balance policy based on new service for web-servers distributed all over the world is proposed. The policy helps reduce the time of service by limiting the number of changing routes of a request to the nearest faraway server, without overloading them. An intermediate tool is described for implementing this protocol. It also uses an exploring method for helping web-servers to tolerate the load [10].

- **Dynamic load balance of justified scheduling**

The main aim in this algorithm is justified and balanced distribution and rapid response to the requesting applicant. The tasks requiring less time in this algorithm are prior to the other tasks. Schedulewise, it indicates that the tasks are queued for justified planning. Then, the completion time of tasks are calculated using justified scheduling algorithm, and afterwards, they are categorized according to the shortest to longest times. Finally, the tasks are allocated to the processors with regards to the completion time of each task. Dynamic load balance is used due to the problem of hunger and eliminating this problem in the algorithm. At the first stage, the tasks are entered and queued, for allocation to a processor after determining their completion times. Then, the scheduler allocates the tasks to the processors, and in case the tasks are not distributed evenly, the scheduler reprograms and reallocates the tasks. It allocates the entered tasks to the idle processors. Finally, after finishing the task, the processors collect the tasks to provide a general response to the requesting body [11].

- **Load balance according to fuzzy logic**

It has been tried in this method to implement the load balance technique in accordance with fuzzy logic. In fact, this algorithm considers the processor speed and virtual machine load as two input parameters, for better load balance in the cloud and using the fuzzy logic. These parameters are given to the fuzzilier as inputs, which are used as the output for measuring the load balance. The two parameters of processor speed and allocated load are used together for evaluating the balanced load on data centers of cloud computing environments, via fuzzy logic. The obtained results with efficacy values can lead to loads balance by reducing time or improving the response time, ending in maximum use of the resources. The processor time and the allocated load to the virtual machine are applied via fuzzy logic, for the load balance in cloud computing [12].

In this section, the concept of cloud computing and the main concepts of scheduling and load balance were first analyzed. Then, regarding the nature of scheduling and its importance in efficiency of could computing, the points about load balancing and different examples of scheduling algorithms as well as the load balance cases that are so far proposed were briefly described. Analyzing task scheduling algorithms in cloud computing environment, it can be concluded that by considering secure service management, increasing hardware potentials and completion of cloud computing infrastructures and supporting complex applied models, resource data and principles of changing the tasks, an opportunity shall be provided for more complex and more complete scheduling algorithms to be executed in this environment. As it was observed, none of the scheduling methods is not perfect and complete, and each one try for

improving a parameter. In fact, these algorithms do not involve all the considered elements by the users. Thus, according to investigations and analyzing the aspects of scheduling algorithms and considering the users' satisfaction and quality service, it seems that combining metaheuristic algorithms and common scheduling algorithms is essential in cloud computing. The metaheuristic methods and complementary algorithms can be used along with common scheduling algorithms in cloud computing environments to obtain a rather optimized response in a period of time and for obtaining appropriate result. Genetic algorithm among the complementary algorithms provides the more accurate response close to the optimized response, in parallel search cases. The most important problem in scheduling matter based on genetic algorithm is rapid response to the existing requests in the queue. The faster the response, the users shall be more satisfied. Genetic algorithm cycle is repeated up to a definite amount to reach the best schedule with the aim to reduce the completion time.

3. Proposed Algorithm

Task scheduling mechanism is related to optimized comparison of tasks and resources and dependence of scheduling algorithm to optimized distribution of tasks has transformed load balance to an important criterion in the cloud [13]. It has been tried in the proposed algorithm to present a combined exploring algorithm for proper mapping of the tasks to the existing virtual machines and available virtual machines for the reduction completion time. The optimum characteristics of Sufferage and genetic combined algorithm are used in the proposed algorithm of GASA[1] in combined double-phase form, with the aim to combine the two algorithms to reduction completion time. In the first phase of the proposed algorithm, a non-random initial population is established with intentional purposes, for reducing the possibility of getting involved in the local optimum, according to Sufferage algorithm. Genetic algorithm is metaheuristic, having the potential for overall search and solving complex problems. It also has diverse operators for producing diverse population. Thus, this algorithm is used in the second phase, for optimizing the first phase output. The applied algorithms in the proposed algorithm are described, as follows.

3.1 Sufferage Algorithm

In the Sufferage algorithm, the least and the one but the least time of task completion is found for each task. Then, the difference between these two values is defined as the sufferage value. The second step includes allocating the task with maximum sufferage value to the machine associated with the least time. The sufferage method is based on the idea that solutions can be generated by allocation of a machine to the task which otherwise would have to wait to take over a machine (sufferage value = second earliest completion time - earliest completion time) [14].

3.2 Genetic Algorithm

The main aim cloud computing is satisfying the cloud users, with regard to the agreed service quality, and providing the benefits for cloud providers. A proper model for tasks scheduling is essential for reaching this aim, for establishing the load balance and reducing the completion time. Scheduling system follows different aims, such as

[1] Genetic Algorithm-Sufferage Scheduling (GASA)

increasing the productivity of resources, balancing the load and reducing the completion time.

The main aim of task scheduling is proper execution of tasks on the resources and providing service to the users at the shortest time. It is indeed finding a suitable follow up of tasks for execution by considering the applied limits, in order to guarantee the load balance and reduce the completion time [15]. As previously stated, extensiveness and dynamicity of cloud space have caused the certain algorithms not to have appropriate efficiency for establishing the load balance. This has made the researchers to experience complementary and metaheuristic algorithms in that regard. Genetic algorithm among others is considered the best heuristic method, since it can search the problem situation in whole and different directions, in a moment. Thus, it is used for solving many optimization problems.

Genetic algorithm is made by inspiring from biology concepts, such as inheritance, natural selection, crossover and mutation, based on a random search. Genetic algorithm adopts from the existing laws of genetics for the children in a generation (set of problem solutions in one stage), and by using them, it generates children with better characteristics (closer responses to the problem target), obtaining a better approximation for the final response in each generation, by the aid of a selected process, which is according to the values of responses [16]. The use of genetic algorithm reduces the time for reaching a response as compared to other methods, to an acceptable extent.

3.3 Describing the Proposed Algorithm

The aim of presenting the proposed algorithm is reaching the improvement of independent and static tasks scheduling, based on genetic algorithm in the cloud environment. It has been tried in the proposed algorithm to present a combined exploring algorithm for appropriate mapping of tasks to virtual machines and reducing the completion time of the tasks, by combining genetic algorithm and the Sufferage algorithm. Since Sufferage algorithm had better performance than other algorithm in previous studies [3], [6], [14] in the proposed method tries to present a proper combination of Sufferage and Genetic Algorithms. The proposed algorithm shall be hereby described.

3.3.1 Mechanism of the Proposed Algorithm

Scheduling in distributed environments, such as cloud computing environment, depends on a great deal of parameters, since inhomogeneous resources exist in these systems with dynamic characteristics. Moreover, the workload in the system is more than local and homogeneous networks. Some parameters in these environments are more important than others. The completion time parameter reduces in the proposed algorithm by suitable allocation of resources to the tasks. This algorithm is based on the genetic metaheuristic algorithm, since it has higher control in searching aspects, according to previous studies [16], [17], [18] and in addition to simplicity of the calculations. Also, the crossover and mutation operators in this algorithm increase the diversity of the population, helping in solving scheduling problems in the distributed cloud environment and reducing the complexities. The proposed algorithm pseudocode is shown in figure 1.

1. **Begin Main**
2. Get new tasks to be scheduled
3. Get Assignable VMs from Clouds.
4. Requirements: **ET**
5. **[Initialize]** Use **Sufferage** for produce population initialization.
6. **[Fitness]** Apply Fitness Function to evaluate all the individuals.
7. While the exit conditions are not satisfied
8. **[Selection]** According to the fitness value, select two parent individuals from the population.
9. **[Crossover]** Generation of the new offspring by reforming the parents using crossover probability.
10. **[Mutation]** Mutate the new child at some positions with the probability of mutation.
11. **[Accepting]** now the new offspring is a part of next generation of population.
12. **[Replace]** use the new generation as the current generation.
13. End while
14. Go to 6
15. **End Main**

Figure 1. The GASA Algorithm

3.3.2 Utilization Steps of Genetic Algorithm in Proposed Algorithm

The procedures of executing genetic algorithm for tasks scheduling in cloud environment will be describe according to the proposed algorithm. The aim in the algorithm is reducing completion time in cloud computing environment. One of the important features of this algorithm is aimed establishment of the initial population and providing an efficient cost function for reducing completion time, simultaneously.

- **Chromosome structures in the proposed algorithm**

Each solution in the proposed algorithm is shown by one chromosome. One-dimensional arrays with length of the total existing tasks for processing are used, for exhibiting the chromosomes.

$$T_x = \{(p_i, vm_j)\} \qquad i=1, 2 \ldots I \qquad j=1, 2 \ldots J \qquad (1)$$

Each block of a chromosome that is called gene is formed by a field, shown as "vm_j". T_k is a vector that shows the type of scheduling, and its number in each genetic algorithm generation is equal to the number of responses, p_i is the ith task, and vm_j is the jth virtual machine. vm_j shows that ith task is done by jth virtual machine. Genetic algorithm choromosomes that each indicates a combination of doing task by virtual machine are shown by the vector T_k. Consider a chromosome sample as follows.

$$T = \{(p_3, vm_2), (p_2, vm_1), (p_4, vm_2), (p_1, vm_1)\}$$

Figure 2. A chromosome sample in proposed algorithm

In this sample chromosome, we assume that we have 4 tasks and 2 virtual machines. According to this layout, task 3 is done by virtual machine 2, task 2 is done by virtual machine 1 and task 4 is done by virtual machine 2 and task 1 is done by virtual machine 1. Task priorities are tasks 3, 2, 4 and 1, respectively, and each one has a weight and the task priorities are determined by the weights. As it can be observed,

placement of chromosomes indicates their priorities. Thus, each chromosome includes information about performing the tasks, determining how the prioriries for tasks are, and which virtual machine is doing the tasks. Thus, each produced response can be demonstated as a chromosome, as follows.

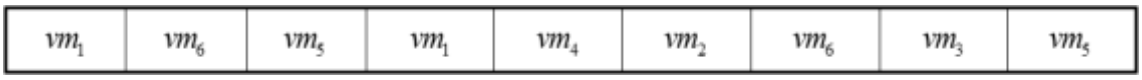

Figure 3. A produced response in proposed algorithm

In which we have 9 tasks, and each task is done by a definite virtual machine. The problem parameters, which are the numbers of responses to each generation and the number of repetition of generations up to reaching to algorithm stappage, and the possibility of doing the crossover and mutation operators for each chromosome, will be determined after organizing the parameters. For generating new responses, we use two operators, crossover and mutation, in the proposed algorithm.

- **Generating the Initial Population in the Proposed Algorithm**

Different methods are proposed for creating the initial population and initial solutions [19], each of which have advantages and disadvantages. These methods can be categorized into "random" and "non-random" methods. One of the disadvantages of non-random (heuristic) methods is the problem of limiting the searching space, or in other words, increasing the probability of losing the optimized solutions and intensifying the local optimized status. On the other hand, generating the solutions that may be far away from the target is also considered a problem in this method, which will lead to elongation of the searching time or even lack of obtaining the optimum response. Thus, to confront with the problems, the heuristic and aimed method is used in the proposed algorithm for generating the initial population. The first phase is the proposed approach for generating the initial population, and according to the hypotheses, the initial population of the proposed algorithm will be generated non-random and by using Sufferage algorithm. For generating non-random solutions, the Sufferage algorithm is used.

A cost function is consdidered in the proposed algorithm, according to the amounts of tasks and precessing speed of virtual machines, with the aim to reduce the completion time as the fitness function, to evaluate the produced chromosomes. Afte evaluating the produced chromosomes, the chromosomes should be selected from this population dor combining and reproduction, which have more optimization for their fitness. These chromosomes have higher chance of being selected. After the solutions and their evaluations, the next generation is produced by applying the selection operator, crossover and mutation on a population. The descriptions of fitness function and the used operators are given in the proposed algorithm, as follows.

- **Evaluation of chromosomes using fitness function**

Fitness function is quite important in the evolving algorithms, since these algorithms are roperly operating only when the function is defined properly. This function is a criterion for etermining the appropriateness or competency of a response relative to other responses. In fact, to determine how much a solution can provide a suitable response, its values should be measured by a fitness function. A fitness function is used

in the proposed algorith, with the aim to reduce the completion times of all the existing tasks for processing in a scheduling system in cloud environment as well as optimized productivity of cloud resources, which will be described as follows.

$$FitFun = \sum_{i=1}^{m} \left(\frac{\sum_{j=1}^{n} V_j}{mips_{vm_i}} + \frac{T_{vm_i}}{T_{total}} \right) \quad (2)$$

In the above relation, v_j indicates the rate of j[th] task, and $\sum_{j=1}^{n} v_j$ is the total rate of the tasks to be processed by the i[th] virtual machine. $mips_{vm_j}$ Shows the speed of processing by the virtual machine. T_{vm_j} Is the total time that the j[th] virtual machine is used in processing, and T_{total} is the completion time for all tasks.

- **used Operators in proposed algorithm**

After generating the solutions and evaluating them, the next generation is produced by applying the operators on the population. The method of applying the operators in the proposed algorithm is as follows:

- **Selection operator**

After evaluating the produced chromosomes, the chromosomes should be selected out of this population that have more optimization state in accordance with fitness function. Such chromosomes have better chance of being selected. The selected chromosomes for relating with other chromosomes to produce the new generation are collected in another place. There are different methods for selection, such as competitive selection, roulette wheel selection, or random selection [20], [21].

Classification technic is used in the prposed algorithm for selecting the parents. Selection in classification style is such that all the chromosomes are classified according to fitness function and regarding their competencies. The best chromosome receives the greatest competency grade and the worst chromosome receives grade 1. Thus, in this method for selecting parents, all the chromosomes have the chance of being selected. Selecting chromosomes for the next generation is by a combined method for the proposed algorithm. In this selection style, the chromosomes are arranged according to their competencies, and the repeating chromosomes are eliminated. Then, a percentage of chromosomes is selected for their better competencies and the rest are selected randomly, for the next generation population. Dispersion of chromosomes is maintained in this method, and chromosomes are dispersed in the required space, and the early local optimization is also eliminated.

- **Crossover operator**

After selecting two parent chromosomes according to the assumptions in the previous section, some parts of parent chromosomes are swapped or changed according to the existing methods, for producing new children. The aim of crossover operator is creating new children, with the hope that the proper charateristics of the parents are collected in the children, and reproduce better children (solutions). Single point crossover operator is used in the proposed algorithm. In this crossover method and after selecting the

parents for the crossover operation, a random number is selected in the range "0" to "n-1" (n= number of tasks). The genes located in the random limit of "0" to the selected random number are transferred from the first parent to the child, and then the similar genes in the second parent are eliminated. The remaining genes in the second parent are respectively placed in the children empty blocks. The aim of this method is exchanging similar children, so that better children can be produced from parents with higher compentence. Empirical results show that using the exchanges, the chromosomes are dispersed in a more suitable space, causing better response in better period. The following figure shows the application of crossover operator for the two selected chromosomes.

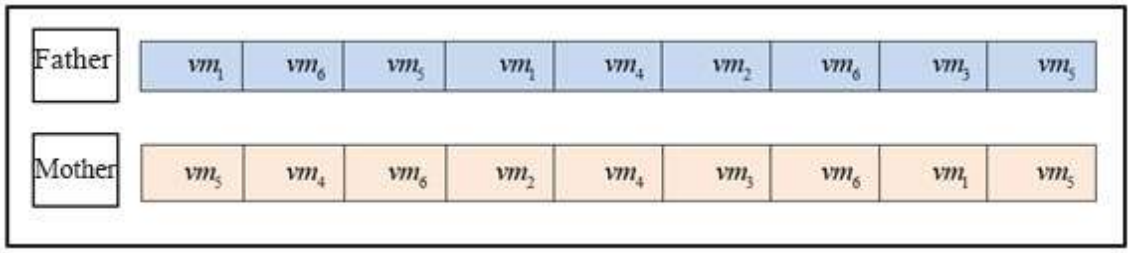

Figure 4. Parents in the proposed algorithm

Hence, the new responses will be as follows.

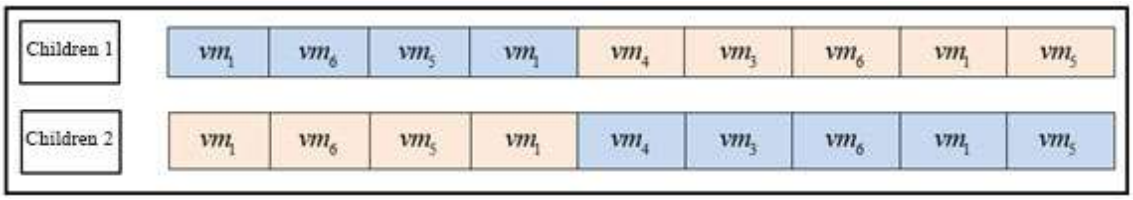

Figure 5. Applying single point crossover operator for the selected chromosomes

As it be observed, the first produced responses for the first 4 tasks of the crossover of virtual machines are taken from the first part of father chromosome and the text 5 tasks from the the second part of mother chromosome. Moreover, the second response takes the first part from mother chromosome and also takes the second part from the father chromosome.

- **Mutation operator**

Mutation operator creates unwanted changes in the population, producing new children. The optimized response may be seen if only the mutation is used, but using the crossover does not guarantee the optimized response on its own. There are various methods for the mutation operation, including simple inverse mutation, swapping mutation, changing mutation,and random mutation. Since the aim for mutation is better chromosomes or finding a new solution, therefore, instead of random changing of chromosomes, changing them can be purposive. To do the operation, and depending on the type of the case, one of the classical methods of solving the problem is applied on the genes of a chromosone, as selected, and the result is is to be replaced as the new chromoisome. Mutation leads to searching in the original spaces of the case. The swapping mutations are used in the proposed algorithm, such that after selecting a parent chromosome, one gene is selected randomly and its provessing field is changed.

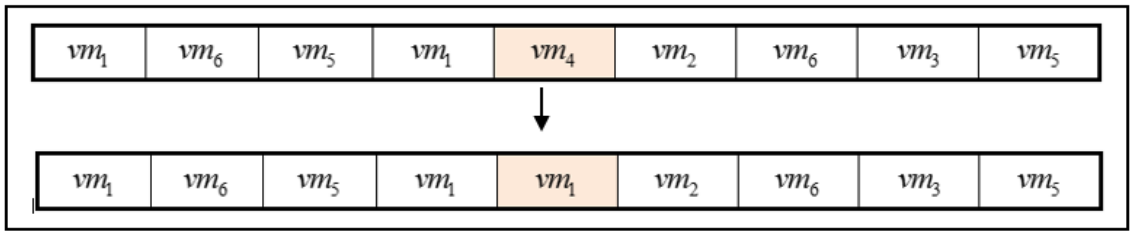

Figure 6. Applying the swapping mutation operator on the selected chromosomes

The defined task is the fifth task, done by virtual machine no.4. Now, the response will be changed by applying the swapping mutation operator. All the tasks are done in this response as in the previous response, and it is only the fifth task that is done by for example virtual machine 1 instead of virtual machine 4. Thus, better responses are selected for producing new responses for each generation, and new generations are produced by the mentioned operators.

- **Termination condition**

The termination condition in the proposed algorithm is limiting the no. of generations. i.e. if the algorithm reaches the maximum no. of the considered generations, the best algorithm will be exhibited and the algorithm will be ended. Otherwise, the algorthm shall return to the production stage.

4. Simulation of the Proposed Algorithm

CloudSim is a simulation software application in cloud environment which has been implemented by Gridbus project team and Grid Lab of Melbourne University. CloudSim can run on Windows and Linux systems and therefore is a Cross-platform tool. This simulator is in fact a Java Library in which there are useful functions for simulation. The newest open-source platform simulation version of CloudSim is Cloudism3.0. The proposed algorithm is implemented according to the following hypotheses in the cloud computing environment, and compared with FCFS, OLB, Sufferage and genetic algorithms for evaluations.

4.1 Analysis of the Result

Resource planning, providing resources and tasks scheduling and resources are among the main problems of cloud computing environments. By studying related works, it can be concluded that planning and scheduling strategies in clouds are according to the developing technics in the distributed systems and the cloud environment. A number of algorithms such as OLB, max-min, min-min and some metaheuristic algorithms, such as genetics algorithm, particle swarm optimization, ant colony optimization and Simulated annealing are existing at present, for planning and tasks scheduling and resources. Although a number of algorithms are provided by researchers for Cloud computing, but none of the above mentioned algorithms has properly done the scheduling. The proposed algorithm focuses on the optimization of tasks based on metaheuristic algorithms, done by the combination of two genetic algortith and Sufferage algorithm in a private cloud environment.

Regarding the speed of OLB, FCFS and Sufferage algorithms, we compare the results from the proposed algorithm with these three standard algorithms. Genetic algorithm is

also selected, since the proposed algorithm is a improvement of that, and since we we intented improve the genetic alkgorithm by combining it with Sufferage algorithm. For the calculation of the algorithm performance, we compare the related results in five states, as examples by the coutesy of cloudSim Library, with FCFS, Sufferage, OLB anf genetic algorithms. Then, we compare the average of the obtained results from 30 times execution of the proposed algorithm and the genetic algorith for five cases, with FCFS, Sufferage and OLB algorithms. The information about tasks and virtual machines can be observed in 5 considered cases in the following table. The proposed algorithm and the mentioned algorithms are simulated according to the following examples and changinging the number of tasks and virtual machines.

Table 1. Total number of tasks and virtual machines

State	Number of Virtual machine	Number of Task
1	4	6
2	8	12
3	10	50
4	25	120
5	30	350

The following graphs show the simulation results of the proposed algorithm and the comparing algorithms, according to the two parameters of total completion time and load balancing .

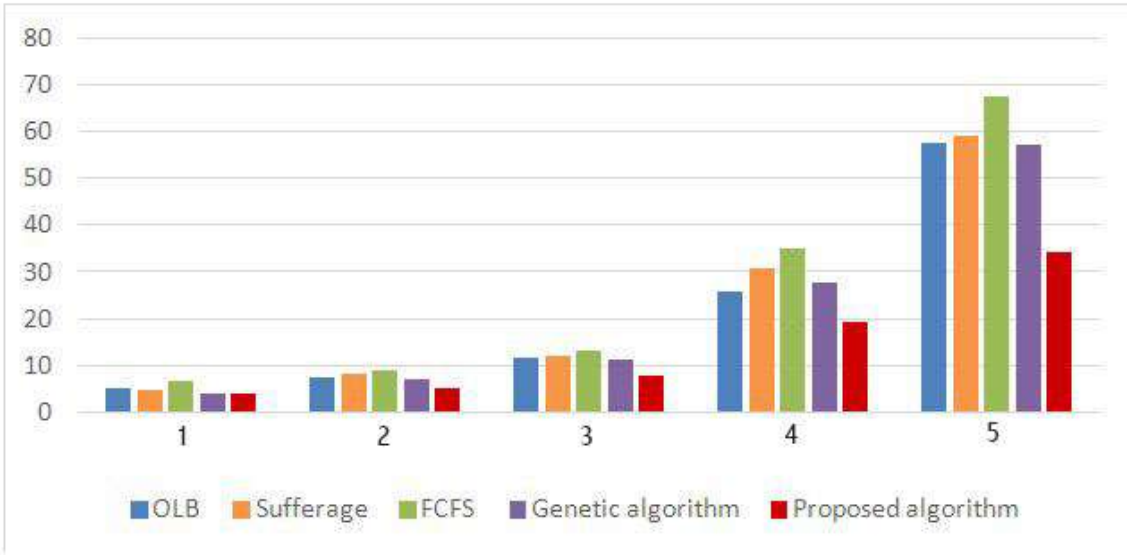

Figure 7. Results of simulation of the proposed algorithm according to completion time parameter

As stated for fitness function, completion time is the completion time of the last task, and load balancing is the time percentage that all the virtual machines are busy. Hence, the more the percentage rate, the rate of optimization will be more, since no idle virtual machine can exist, and load balancing will be established.

As it be observed, the completion time for the proposed algorithm has no significant difference for the first state with ordinary genetic algorithm, but the results are better than the other three algorithms. Equivalence of the results of the proposed and genetic algorithms is due to few number of tasks, and the proposed algorithm could not create

any considerable changes in the responses. According to the results in the other states, the completion time has a significant improvement in the proposed algorithm. The improvement is because the number of tasks have increased. Thus, in each produced generation by the genetic algorithm, the proposed algorithm could change and improve more responses.

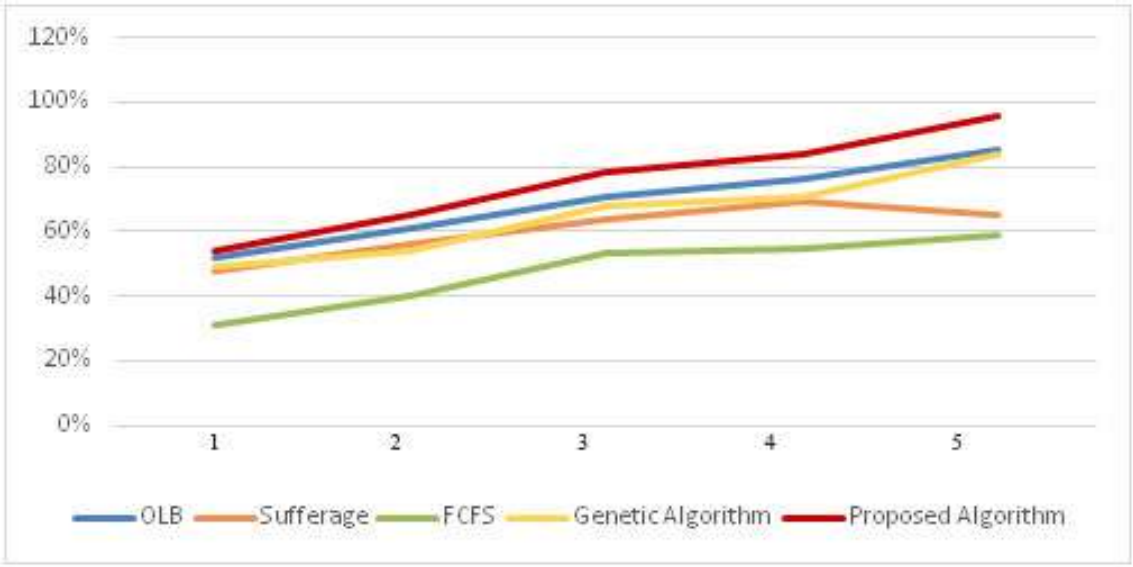

Figure 8. Results of the proposed algorithm simulation according to load balance parameter

Since the algorithms are simulated according to the rate of load balance, the guaranteed balance of the load will be more by the increased load balance percentage. According to figure 8, the proposed algorithm has observed reduction in completion time as well as the load balane. It isto note that in the first state, no significant improvement as compared to OLB algorithm that guarantees the load balance due to few number of tasks and virtual machines. But, the proposed algorithm has properly guranteed the load balance by increasing the number of tasks and the machineries. Thus, it seems that by increasing the no. of tasks and the machineries that leads to more existing states for making the allocated combinations and responses in each genetic algorithm gereration, the rates in load balance and completion time parameters in the proposed algorithm will be improved as compared to other algorithms.

5. Conclusion

It was tried in this article to provide an algorithm for improving the common scheduling methods. In fact, a new method for task scheduling is proposed in cloud environment, by the discussions existing in the subject of cloud computing for task scheduling. This new method is based on genetic and Sufferage algorithms. The strong ponts of genetic and Sufferage algorithms were combined in the proposed method to achieve a better algorithm.

The mentioned algorithms were combined in such a way to cover the weak disadvantages, and become a completely integrated and optimum algorithm. Combining the two algorithms create a schedule for executing some input tasks to the cloud environment. By the definition of appropriate fitness function, the obtained schedule optimizes the completion time of the tasks and productivity from resources. In fact, by

simulation of the proposed algorithm and the other considered algorithms in cloudSim environment and analyzing the results of the proposed algorithm with respect to the other methods, the proposed algorithm has shown a better efficiency.

References

[1] N.A. Bahnasawy, F .Omara, M.A. Koutb, and M. Mervat, "Optimization procedure for algorithms of task scheduling in high performance heterogeneous distributed computing systems" , Egyptian Informatics Journal, 2014, 219-229.

[2] N. Bansal, A. Awasthi, and Sh. Bansal, "Task Scheduling Algorithms with Multiple Factor in Cloud Computing Environment", Information Systems Design and Intelligent Applications, Proceedings of Third International Conference INDIA, 2016, 619-627.

[3] F. Durao, S.F.J. Carvalho, A. Fonseka, and C.V. Garcia. "A systematic review on cloud computing", The Journal of Supercomputing, Springer US, 2014, 1321-1346.

[4] T. Mathew, K.C. Sekaran, and J. Jose, "Study and analysis of various task scheduling algorithms in the cloud computing environment", International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2014, 658- 664.

[5] N. Patel, "A Survey on Load Balancing and Scheduling in cloud computing", International Journal for Innovative Research in Science and Technology (IJIRST), 2015, 185-189.

[6] L. Tripathy, and R.R. Patra, "Scheduling in Cloud Computing", International Journal on Cloud Computing: Services and Architecture (IJCCSA), 2014, 21-27.

[7] C.W. Tsai, and. J.P.C. Rodrigues, "Metaheuristic Scheduling for Cloud: A Survey", Systems Journal, IEEE, 2013, 279-291.

[8] Y. Mao, X. Chen, and X. Li, "Max–Min Task Scheduling Algorithm for Load Balance in Cloud Computing", Proceedings of International Conference on Computer Science and Information Technology of the series Advances in Intelligent Systems and Computing, 2013, 457-465.

[9] H. Chen, F. Wang, N. Helian, and G. Akanmu, "User-priority guided Min-Min scheduling algorithm for load balancing in cloud computing", National Conference on Parallel Computing Technologies (PARCOMPTECH), 2013, 1-8.

[10] K, K. Pathak, P, S. Yadav, R. Tiwari, and T, K. Gupta, "A Modified Approach for Load Balancing in Cloud Computing Using Extended Honey Bee Algorithm", International Journal of Research Review in Engineering Science and Technology: IJRREST, 2010.

[11] k. Karthick, "A Dynamic Load Balancing Algorithm in Computational Grid Using Fair Scheduling", International Journal of Computer Science Issues (IJCSI), 2011.

[12] S. Sethi, and A. Sahu, S.K. Jena, "Efficient load balancing in Cloud Computing using Fuzzy Logic", IOSR Journal of Engineering (IOSRJEN), 2012, 65-71.

[13] M. Mesbahi, and AM. Rahmani, "Load Balancing in Cloud Computing: A State of the Art Survey", I.J. Modern Education and Computer Science, 2016, 64-78.

[14] H. Suo, And H. Yan, "Research on Resource Scheduling in Cloud Computing: Issues and Solutions", Applied Mechanics and Materials, Numerical Methods, Computation Methods and Algorithms for Modeling, Simulation and Optimization, Data Mining and Data Processing, 2014, 1801-1804.

[15] W. Mingxin, "Research on Improvement of Task Scheduling Algorithm in Cloud Computing", Applied Mathematics & Information Sciences An International Journal, 2015, 507-516.

[16] Kaur, S. and Verma, A. 2012. An Efficient Approach to Genetic Algorithm for Task Scheduling in Cloud Computing Environment. International Journal of Information Technology and Computer Science (IJITCS). Vol.4. pp: 74-79.

[17] Goyal, T. Agrawal, A. 2013. Host Scheduling Algorithm Using Genetic Algorithm in Cloud Computing Environment. International Journal of Research in Engineering & Technology (IJRET). Vol.1. pp: 7-12.

[18] G. Guo-ning, H. Ting-lei, and G. Shuai, "Genetic simulated annealing algorithm for task scheduling based on cloud computing environment", International Conference on Intelligent Computing and Integrated Systems (ICISS), 2010, 60-63.

[19] S. Taherian Dehkordi, and V. Khatibi Bardsiri, "TASA: A New Task Scheduling Algorithm in Cloud Computing", Journal of Advances in Computer Engineering and Technology (JACET), 2015, 25-32.

[20] M. Juntao, L. Weitao, Fu. Tian, Y. Lili, and H. Guojie, "A Novel Dynamic Task Scheduling Algorithm Based on Improved Genetic Algorithm in Cloud Computing", Wireless Communications, Networking and Applications, 2015, 829-835.

[21] Zh. Zhan, X. Liu, Y. Gong, J. Zhang, H. Chung, and y. Li, "Cloud Computing Resource Scheduling and a Survey of Its Evolutionary Approaches", International Journal of ACM