A New Strategy to Optimize the Load Migration Process in Cloud Environment

Hamid Mirvaziri
Assistant professor of computer engineering,
Shahid Bahonar University of Kerman
Kerman, Iran

ZhilaTajrobekar
Student of Computer Engineering, Islamic Azad University of Kerman
Kerman, Iran

Abstract—**Cloud computing is a model of internet-based service that provides easy access to a set of changeable computational sources through internet for users based on their demand. Load balancing in cloud have to manage service provider resources appropriately. Load balancing in cloud computing is the process of load distribution between distributed computational nodes for optimal use of resources and have to decrease latency in order to prevent a situation in which some nodes overloaded and some others under-loaded or be in the idle mode. Load migration is a potential solution for most of critical conditions such as load imbalance. However, many load migration methods are only based on one purpose. Practically, considering just one objective for migration can be in contrary to the other objectives and may lose optimal solution to work in existing situation. Therefore, having a strategy to make load migration process purposeful is essential in cloud environment. The main idea of this research is to reduce cost and increase efficiency in order to be compatible with cloud different conditions. In the recommended method, it is tried to improve load migration process using several different criteria simultaneously and apply some changes in previous methods. The simulated annealing algorithm is employed to implement the recommended strategy in the present research. Obtained result show desired performance and efficiency in general. This algorithm is highly flexible by which several important criteria can be calculated simultaneously.**

Keywords—cloud computing; load balancing; migration; virtual machines; simulated annealing

I. Introduction

One of the modern developments in the internet is introduced by Cloud Computing (CC) technology. This technology becomes quickly popular due to its properties in which every kind of facilities offers to the users in the form of a service [13]. The CC is an internet-based service model as it provides easy access to a set of changeable computing resources through internet for users based on their demands. In such mode, users try to access based on their needs regardless of where the service is located or how it is delivered. Various types of computing services try to offer such services to the users. Some of these computing systems are cluster computing, grid computing and recently CC. There are some services provided by CC architecture based on IT customers' needs [4]. Naturally, any new changes and concepts in IT environment has its own specific problems and complexities; using CC is not an exception and put many challenges in front of experts in this field, such as load balancing, security, reliability, ownership, backing up data, data portability and supporting different platforms. Considering the importance of migration in load balancing of CC, it is going to improve this process in this paper [2]. This article organized as follows: In section two previous works will be reviewed. In section three our proposed method is stated. This method is evaluated in section four and finally there is a conclusion in the last section.

II. Literature Review

A. Cloud Computing

CC platform is a completely automatic and service providers let the users buy, remote creation, dynamic scalability and system management [9]. Operational and capital costs can be reduced by CC [5]. In addition, CC systems are elastic; the amount of resources available to a server has capable of increasing or decreasing. However, it covers all what really a cloud server is [9].

The CC is a new internet-based service becomes common for users to provide different services. Many different and wide sources can be used instead of local or remote servers in CC services. There is not any standard definition for CC. Generally, it is including of a set of known server, offering services and resources to different and demanding clients. Distributed computers provide those demand-based services. The main advantage of CC is quick reduction of hardware costs and computation and capacity increasing [5].

B. Details of Cloud system and its properties

A Cloud system is composed of three main parts: clients, data center and distributed servers. Each part has its own specific role and purpose defined in the following.

- **Clients:** end users interact with clients to manage cloud-related data.
- **Data center:** a data center can created to save data and applications.
- **Distributed servers**: parts of cloud hosting different programs all over the internet while a cloud user thinks that the programs run in his machine [5].

C. Cloud implementation models

There are four models for cloud implementation as follows [14].

- **Private Cloud:** It is an infrastructure of cloud computing only working for an organization and managed by the same organization or a third party organization.

- **Public Cloud:** It is also known as external cloud and describes CC as its main and traditional meaning. The services prepared dynamically through internet in the form of small units by a third party distributer who lends the resources in share to the users.

- **Community Cloud:** also known as group cloud in which cloud infrastructure shared and supported by several organizations with common goals in security mission or considerations. This cloud can be managed by the same or a third party organization. Since the costs divided between fewer users than in public clouds, this choice is more expensive than public cloud but has more privacy, security and compatibility with policies.

- **Hybrid Cloud:** a cloud in which its infrastructure composed of two or more types of clouds (private, community, public).

D. *Load balancing based on migration of virtual machines and efficiency*

- Load balancing between physical hosts is necessary for cloud environment in order to improve efficiency of data centers by increasing in throughput and decreasing in system latency. There are many physical hosts in a data centers therefore, based on migration of virtual machines between physical hosts, load balancing plays an important role to provide stable and highly efficient services. It is possible to have a situation in which physical hosts loaded excessively, that is, the number of virtual machines being run in physical hosts is more than the average. Therefore, running services are incapable of guarantee the needs. The efficiency of data centers servicing can be improved by migration of virtual machines from heavy loaded hosts to the light ones [16].

- Two-stage load balancing algorithms OLB+LBMM: A two-stage scheduling algorithm recommended to mix scheduling algorithms of Opportunistic Load Balancing (OLB) and Load Balance Min-Min (LBMM) for using better execution and maintaining load balancing of the system. The OLB scheduling algorithm keep any node in idle mode to reach the load balancing objective; the LBMM scheduling algorithm applied for reducing running time of any task on the node. Therefore, it decreases the total running time. This algorithm used in three-level CC networks in which efficiency and utilization criteria are considered. Such hybrid algorithm helps in effective use of resources, increases efficiency and offer better results than honey bee algorithm, random sampling and active clustering [4].

- Min-Min algorithm: It is started by a set of unassigned tasks. First of all, the minimum ending time of each task is found. Then, the least minimum is selected among these minimum times which is the minimum time between all tasks exist on each resource. After that, the task scheduled on the related machine within the minimum time. Now, the running time for all other tasks updated on the machine by adding up the assigned task running time to other tasks running time and the assigned task removed from the task list, assigned to the machine. This process repeated till all tasks assigned to the resources. However, there is a problem in this method which can be resulted in starvation [4]. In this algorithm, utilization of resources, overload, throughput, latency and efficiency are considered from load balancing criteria [13].

- A Lock-free multiprocessing solution for load balancing: this solution recommended because it is refused to use shared memory in comparison to other multiprocessing solutions of load balancing and keeps user session by locking. The memory this method obtained by applying Linux core and helps the improvement of general efficiency of load balancer in a multi core environment through running several load balancing processes in one load balancer [11].

- Ant Colony Optimization: A model presented in which unique ants act as very common insects. They have very limited memory and show individual behavior in order to have a big random performance. The ants are working together to find food sources and make use of food source for transferring food toward the colony simultaneously [10].

- Load Balancing Mechanism based on Ant colony and Complex Network Theory (ACCLB): It is presented in a federation of open CC with the purpose of overcoming complexity and problems of dynamic load balancing which use scale-free and small-world properties of complex networks to reach better load balancing. This method improved many aspects of the related ant colony algorithms and recommended to achieve load balancing in distributed systems. Moreover, it overcomes heterogeneity, compatible with dynamic environment, well in resisting against faults and has appropriate scalability since it helps the improvement of system efficiency [4]. Several studies showed that dynamic changes of criteria to calculate the possibility function for an ant in order to select a neighbor node, should have high efficiency for ant colony optimization algorithm, however such issues are not considered in this case [15]. In this algorithm in which complex network theory is used, utilization of resources, scalability and efficiency is considered from load balancing criteria [4].

- Join-Idle-Queue Algorithm: It recommends a load balancing algorithm for dynamic scalability of web services. This algorithm provides LB in large scale by distributed distributers. First, balance the load of idle processors in distributors is happened for any idle processor to access any distributor and then assign tasks to the processors in order to reduce the queue length of each processor. Removing load balancing task from vital route of demands processing, this algorithm reduces system load effectively, no connectional overload occurred at the time of tasks entrance and the real latency not increased. The environment this algorithm used in is cloud data centers in which latency and overload considered as efficiency criteria. This algorithm is capable of running relatively optimal when applied for web services. However, it cannot be used for web services of nowadays dynamic contents due to scalability and reliability [4].

E. *Simulated Annealing Algorithm*

- It is an algorithm for optimization issues inspired from nature. This algorithm introduced by works of Kirkpatrick and Cerny et al. in 1983 and 1985 respectively. It is used for solid substance reaches to the mode in which put arranged well with minimum energy. In this method, put the substance in high temperature and then it decline gradually in order to put the substance in a mode in which it is arranged and has minimum energy. In this algorithm, each point of searching space (s) regarded as a mode of substance (searching space for our algorithm is all possible modes for migration of load between virtual machines) and the E(s) function which is the energy function (fitness function) should be minimized. The purpose is to transfer a substance mode (problem answer) from starting point (initial population) toward optimal mode (optimal answer). Evolutionary algorithms such as this one begin with an initial solution which is the starting point for moving toward optimal solution produced randomly. This means, a simple solution can be regarded initially but it is not necessarily the optimal one and it is just produced for using the algorithm and moving toward more optimal solution [6].

III. Recommended Strategy

Suppose that *m* is a host in which *N* virtual machines exist. *n* tasks will be under service. In servicing, it is possible to some of virtual machines overloaded and therefore their energy consumption and latency increases while the efficiency decreases. In such cases, some methods required for migration of services from overloaded virtual machines to those without overloading; on the other hand, load migration occurs. The purpose of this research is to offer a strategy that solves the problem of load migration on virtual machines considering different criteria such as energy consumption, efficiency etc. the recommended strategy is as follows:

In this strategy, at first each criteria of load migration problem are modeled and relationship between them are expressed. Then, annealing algorithm and modeling the problem in form of a population, a population with optimal fitness obtained. The optimal population will determine the order of load migration on virtual machines.

A. *Criteria of the strategy*

The following criteria considered in this research and it is tried to optimize all of them simultaneously.

1) *load volume*

Since it is possible for a virtual machine to be overloaded during task running in different aspects such as processor, memory or network, a criterion set for load volume of virtual machine, this criterion is a combination of processor, memory and network loads as follows:

$$loadvolume = \frac{1}{1-CPUutilization} \cdot \frac{1}{1-MEMutilization} \cdot \frac{1}{1-NETutilization} \quad (1)$$

In equation (1), $CPUutilization$ ،$MEMutilization$ و $NETutilization$show the efficiency rate of CPU, memory and network [17].

2) *Energy consumption*

Excessive increase of CC networks result in increasing of energy consumption in data centers intensively which is a vital problem and global concern for industry and society. The following equation used to calculate the criterion of energy consumption [3].

$$EnergyConsumption = Static\ energy + Dynamic\ energy \quad (2)$$

In equation (2), *Static energy* is the static rate of energy consumption and *Dynamic energy* is the dynamic rate of energy consumed by the service on the virtual machine which is obtained by the following equation:

$$Dynamic\ energy = \alpha.time.Speed^2 \quad (3)$$

$$Speed = \frac{f_{\mathrm{cpu}}}{f_{\mathrm{maxcpu}}} \quad (4)$$

In equation (3), α is relative coefficient, *time* is the execution time of application and *Speed* is the processor speed. In equation (4), f_{cpu} is normal frequency and f_{maxcpu} is maximum frequency of the CPU [3].

3) *Resource utilization*

Resource utilization depends on considering balance in using resources. The following equation used to calculate it; by simplifying the equation into CPU, memory and network we have:

$$Resource\ Utility = (Mem\ Utility - CPU\ Utility) + (Mem\ Utility - Net\ Utility) \quad (5)$$

In equation (5), $CPU\ utility$ ،$Mem\ utility$ and $Net\ utility$are efficiency rate of CPU, memory and network respectively [18].

4) *Migration cost*

Migration cost of different tasks may be different significantly regarding various settings of virtual machine and feature of the task volume. Most of the load balancing methods is highly efficient but unfortunately load migration cost, is ignored during designing the method which resulted in overloading and recommended methods become useless for cloud environment. Therefore, this criterion considered in the present research. The following equation used for calculating migration cost [1, 14].

$$Migration\ Cost = a.MigTraffic + b.MigTime + c.MigEnergy + d.MigDowntime \quad (6)$$

In equation (6), a, b, c, d are weights of the cost criteria which their sum should be equal to one. $MigTraffic$ is total network traffic of migration process, $MigEnergy$ is total energy consumed by migration process, $MigTime$ is the migration time and $MigDowntime$is idle time emerged by migration process.

$$MigTime = \frac{V_{mem}}{M_{TR}} \quad (7)$$

$$MigEnergy = E_{sour} + E_{dest} + E_{net} \quad (8)$$

$$E_{sour} = V_{mem} \cdot P_{sour} \quad (9)$$

Simply suppose that in offline migration, $MigTraffic$ is the very V_{mem} which is equal to migrated memory in virtual machines in equation (4-7), M_{TR} is the memory transfer rate. In equation (8), E_{sour} ‘E_{dest} ‘E_{net} are amount of extra energy consumed by the source, destination and network interfaces. In equation (9), P_{sour} is the amount of energy needed for transferring one megabit to the resource node.

5) Fault Tolerance

In the present research it is tried to solve the load migration problem without violating fault tolerance required for services. Expressing this criterion, it is supposed that if amount of fault tolerance level of each service is k_i then the following equation guarantee fault tolerance of each service not to be violated.

$$\sum_{j=1}^{m} placement[i][j] - \sum_{j=1}^{k_i} placement[i][j] \geq N_i\,,\ i = 1, \dots, n \quad (10)$$

Where N_i is base number of virtual machines used by *i'th* service. Moreover, in $placement[i][j] = a$ which denotes the number of virtual machines put on *j*'th host by *i*'th service [16].

6) Efficiency

This criterion calculated by both hops time and waiting time parameters. Hop time is the time spent for transferring load from overloaded virtual machines to the ones without overloaded. Waiting time is the time in that virtual machines is preparing to receive services 12].

7) Implementation

The above mentioned criteria is combined linearly and considered as fitness function. Weight of each criterion determined regarding the importance of each one has. In addition, upper and lower limit of load can be considered for each virtual machine by which if the existing load in a virtual machine is more than the upper limit, the load must transfer to the machines with loads less than the lower limit. Efficiency rate of CPU in all virtual machines calculated by equation as follows (4-11) [4].

$$VMutility = \frac{totalRequestedMips}{totalMips\ for\ that\ VM} \quad (11)$$

$$HostBw = \sum current\ allocated\ bandwidth\ for\ VMs\ for\ host \quad (12)$$

$$HostRam = \sum current\ allocated\ Ram\ for\ VMs\ for\ host \quad (13)$$

$$Sum = \sum Uvm \quad (14)$$

Equation (12) and equation (13) expresses total bandwidth and total memory assigned to virtual machines on each host respectively. Equation (14) delineated the efficiency rate of all virtual machines. The virtual machine which its CPU efficiency is more than the upper limit of load means excessively loaded and needs migration of load to another virtual machine with lower load limit. Following equations is used to calculate upper limit of the load [18].

$$temp = Sum + \left(\frac{HostBw}{\sum Bw\ for\ all\ hosts}\right) + \left(\frac{HostRam}{\sum Ram\ for\ all\ hosts}\right) \quad (15)$$

$$T_{Upper} = 1 - \Big(\big((Puu * temp) + Sum\big) - \big((Pul * temp) + Sum\big)\Big) \quad (16)$$

In equation (15), the $temp$ variation is the sum of considered criteria. In equation (16), T_{Upper} is the upper limit of the load. Spare rate of CPU with Puu as upper probability and Pul as lower probability maintained for each host. Moreover, P_l is lower limit for spare capacity of the CPU. Spare capacity of CPU means that number of running services in CPU is lower than its capacity. The virtual machine that its CPU efficiency is less than the lower limit of the load becomes under-loaded and the overloaded machines are appropriate for migration of load to it. If CPU efficiency is less than 30%, the lower limit of load is always 0.3 [18].

$$if\ CPU\ utilization\ is < 30\% => T_{Lower} = 0.3 \quad (17)$$

$$if\ CPU\ utilization\ is \geq 30\% => T_{Lower} = 1 - \big((Pl * temp) + Sum\big) \quad (18)$$

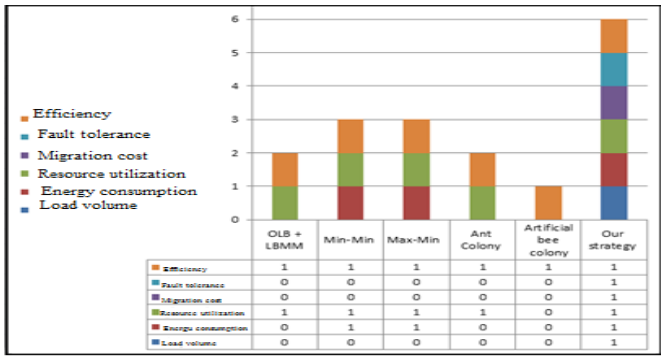
Where T_{Lower} is the lower limit of the load [18].

Fitness function calculated through equation (18) in which there is $c1 + c2 + c3 + c4 + c5 + c6 = 1$ where $c1, c2, c3, c4, c5, c6$ are the weights of fitness calculation criteria, each one considered $\frac{1}{6}$ by default. However, as mentioned previously, weight of each criterion can be determined regarding the importance of each one.

$$a)\ fitness = c1 * loadvolume + c2 * PowerConsumption + c3 * Resource\ Utility + c4 * Migration\ Cost + c5 * fault\ tolerance + c6 * performance.$$

IV. Evaluation of Proposed Algorithm and Several Studied Algorithm in the Related Works Chapter

Our intended strategy consist of six criteria “load volume, energy consumption, resource utilization, migration cost, fault tolerance and efficiency” to calculate fitness and obtained by equation (18). Proposed strategy will be compared with several algorithms of related works and equation (18) will be evaluated for them which show in the form of a chart. Fitness function of the recommended strategy considered for all these algorithms: OLB + LBMM, Min-Min, Max-Min, Ant Colony and Artificial Bee Colony and the comparison is demonstrated a table. As shown in the table, just the recommended strategy is based on SA algorithm which is including all criteria and can present an appropriate strategy for making load migration process purposeful in cloud environment. The following chart is the comparison between the recommended strategy and other algorithms.



	OLB + LBMM	Min-Min	Max-Min	Ant Colony	Artificial bee colony	Our strategy
🟧 Efficiency	1	1	1	1	1	1
🟦 Fault tolerance	0	0	0	0	0	1
🟪 Migration cost	0	0	0	0	0	1
🟩 Resource utilization	1	1	1	1	0	1
🟥 Energy consumption	0	1	1	0	0	1
🟦 Load volume	0	0	0	0	0	1

Fig. 1. comparing the recommended strategy with other algorithms based on the number of important criteria for making migration process purposeful in form of bar chart

1) Operation of recommended strategy (pseudo code)

At first, the services is assigned to the virtual machines in different hosts randomly and then the upper and lower limit of the load calculated in each virtual machine on each host and if there is at least one overloaded virtual machine, the load migration strategy activated and the load migration is done. In addition to finding a load-less combination, the algorithm calculates the above-mentioned determined criteria using fitness function produces fitness of each solutions of the population. Our purpose is to return a combination without overload and a better fitness. Then, the n number of better solutions (with higher fitness) is kept and the new population is produced by means of Crossover. After that, the amount of fitness of each solution in the new population is calculated and again the n number of better solution is kept; this process continued until the end condition of the algorithm occurred. Pseudo code for the recommended strategy is as follows:

Load Migration Algorithm

```
1: Calculate upper and lower bound load per every virtual
   machine for every host.
2: if there is at least one virtual machine with over load then
3:     Initialize initial population by generating a random
       migration load from any virtual machine with over
       load to any virtual machine with minimum load.
4:     repeat
5:         by Crossover generate new population by
           migration load from any virtual machine with
           over load to any virtual machine with minimum
           load
6:         Mutate new population
7:         calculate fitness function for every member of new
           population
8:     until termination conditions meet.
9: end if
```

a) Temperature and fitness function in the recommended strategy based on SA algorithm

As previously explained in SA algorithm, temperature is also one of the main parameters of this algorithm. Therefore, in this section, testing temperature in different values and calculating fitness function based on the related temperature shows that the more temperature decreased the more appropriate fitness value is obtained. Thus, the major task now is to determine the temperature situation. This main part is temperature reduction schedule which start at (1000˚C), and finish at (0-1˚C) and it is a linear function ($T_{k+1} = T_k * a$) [19, 20]. According to the calculation of temperature reduction rate, temperature variations chart and its effect on optimization is based on the points selected from the above table shown in the following figure.

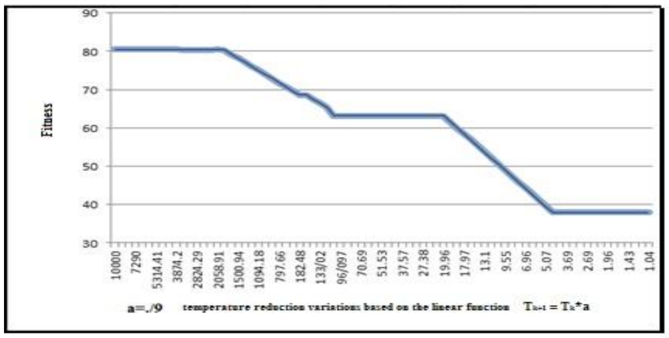


Fig. 2. Fitness and temperature reduction variations based on the linear function

2) Implementation and the strategy analysis

The algorithm simulated in MATLAB software by 5 hosts, 3 virtual machines and 60 services. A population consist of 10 solution is considered in the simulation. The recommended strategy algorithm starts up by random assignment of services to virtual machines in different hosts. Now, two modes may occur in this state for the algorithm:

1st mode: There is not any overloaded machine in the hosts after random assignment as shown in figure 3 in which each square denotes a host and it is clear, there are 5 hosts including 3 virtual machines, each one has different services.

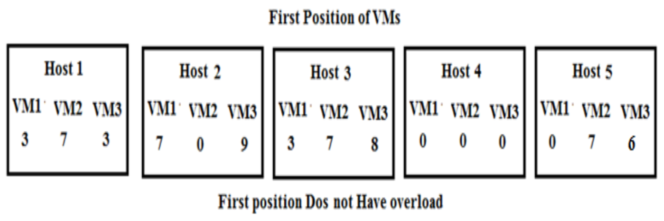
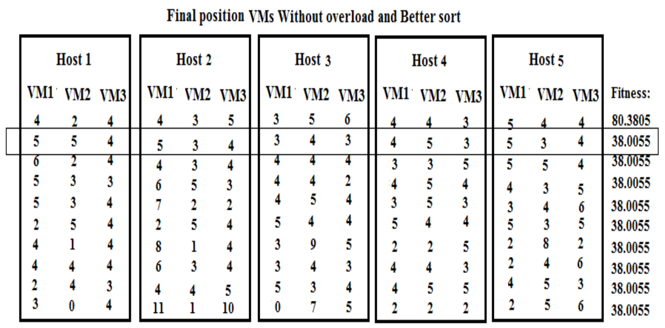


Fig. 3. primary random assignment without overload

In this mode, although there is not any overload, an appropriate load balancing algorithm guarantees,using of that equal amount of all existing resources at any moment. Therefore, since the primary random assignment without overload may not use all virtual machines for services assignment, the algorithm run in this mode and returned a population with appropriate order including 10 solution shown in figure 4 in which number of chromosomes is more than one row that is, there is a population includes 10 chromosomes each one has to return a fitness value. Therefore, there are 10 solutions and each solution is related to fitness of each population. Among these 10 answers, one with optimal fitness function can be selected as order of assigning the services to the virtual machines. In fact, selection process of optimal fitness is that the first and the least values of fitness selected as the optimal solutions.

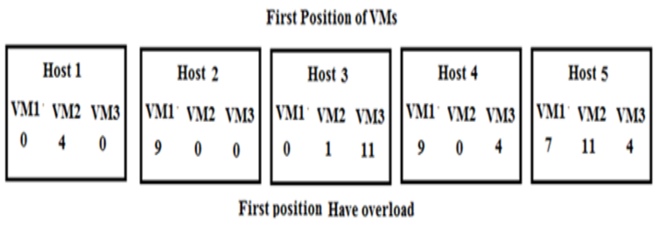


Final position VMs Without overload and Better sort

Host 1			Host 2			Host 3			Host 4			Host 5			
VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	Fitness:
4	2	4	4	3	5	3	5	6	4	4	3	5	4	4	80.3805
5	5	4	5	3	4	3	4	3	4	5	3	5	3	4	38.0055
6	2	4	4	3	4	4	4	4	3	3	5	5	5	4	38.0055
5	3	3	6	5	3	4	4	2	4	5	4	4	3	5	38.0055
5	3	4	7	2	2	4	5	4	3	5	3	3	4	6	38.0055
2	5	4	2	5	4	5	4	4	5	4	4	5	3	5	38.0055
4	1	4	8	1	4	3	9	5	2	2	5	2	8	2	38.0055
4	4	4	6	3	4	3	4	3	4	4	3	2	4	6	38.0055
2	4	3	4	4	5	5	3	4	4	5	5	4	5	3	38.0055
3	0	4	11	1	10	0	7	5	2	2	2	2	5	6	38.0055

Fig. 4. population without overload with better order

2nd mode: it is assessment of the pattern for overloaded virtual machines. That is, one or more of the virtual machines in the hosts include overload after running of the algorithm and production of primary chromosome.



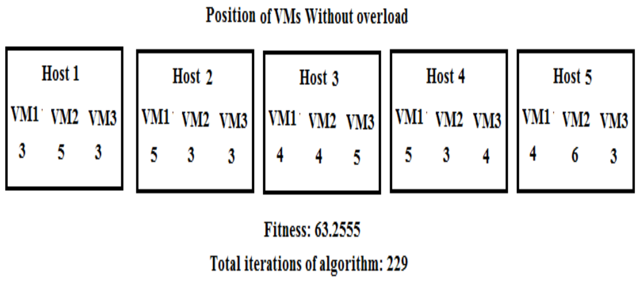
First Position of VMs

Host 1			Host 2			Host 3			Host 4			Host 5		
VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3
0	4	0	9	0	0	0	1	11	9	0	4	7	11	4

First position Have overload

Fig. 5. primary random assignment including overload

The above figure shows that the algorithm performs until finding a situation without overload in this mode. Thesprocess is continued until finding a chromosome in which there is not any overloaded virtual machine. After finding a mode without overload for all virtual machines the algorithm ended and the output, is containing primary chromosome without overload obtained from the algorithm. The number of algorithm repetitions to find such chromosome and the output is shown in figure 6.



Position of VMs Without overload

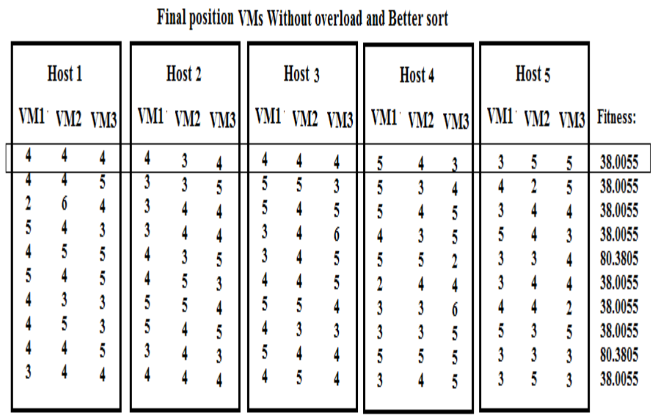
Host 1			Host 2			Host 3			Host 4			Host 5		
VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3
3	5	3	5	3	3	4	4	5	5	3	4	4	6	3

Fitness: 63.2555

Total iterations of algorithm: 229

Fig. 6. population without overload with better order

The algorithm returns the first chromosome without overload as the solution. This solution may not have an appropriate fitness function therefore the process continues again and returns other chromosome without overload along with their fitness function (figure 7). The same as previous, a chromosome with optimal fitness function can be selected as the order of assigning services to the virtual machines.



Final position VMs Without overload and Better sort

Host 1			Host 2			Host 3			Host 4			Host 5			
VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3	Fitness:
4	4	4	4	3	4	4	4	4	5	4	3	3	5	5	38.0055
4	4	5	3	3	5	5	5	3	5	3	4	4	2	5	38.0055
2	6	4	3	4	4	5	4	5	5	4	5	3	4	4	38.0055
5	4	3	3	4	4	3	4	6	4	3	5	5	4	3	38.0055
4	5	5	4	3	5	3	4	5	5	5	2	3	3	4	80.3805
5	4	5	4	5	3	4	4	5	2	4	4	3	4	4	38.0055
4	3	3	5	5	4	5	5	4	3	3	6	4	4	2	38.0055
4	5	3	5	4	5	4	3	3	3	3	5	5	3	5	38.0055
4	4	5	3	4	3	5	4	4	5	5	5	3	3	3	80.3805
3	4	4	4	4	4	4	5	4	3	4	5	3	5	3	38.0055

Fig. 7. population without overload with better fitness funtion and order

Moreover, the maximum number of genetic algorithm to be run is set to 1000 repetitions in order to prevent endless running. Generally, for determining whether virtual machines is overloaded or not, first CPU efficiency is calculated for each machine and the upper and lower limits of load is calculated for each host beside. The lower limit of load is calculated by the equation mentioned in previous parts if the CPU efficiency is more than 30% (the lower limit of load determined as 0.3 for each host in this research). Now, we have efficiency and the lower and upper limits of virtual machine. Therefore, the virtual machine with CPU efficiency more than upper limit of load is overloaded and needs migration of load to the other virtual machine with lower limit of load while the virtual machine with CPU efficiency less than lower limit of load is under-loaded and appropriate for migration of load from overloaded virtual machines.

V. Conclusion

In this paper, it is tried to study the differences resulted from applying different criteria in load balancing process in cloud environment. As mentioned before, load balancing process in cloud environment is very important which has high effective in applying cloud services. In the present research, a strategy is introduced in order to optimize migration process in load balancing. Considering studies done in this research, a criterion as the purpose of migration in load balancing process can be in contrary to other purposes, and the optimal solution for dealing with current situation may be lost and such algorithm neither used in all cloud conditions nor lead to the best results. Considering this result, using an algorithm capable of considering several load-balancing criteria in load migration process and optimize them simultaneously may overcome such defect to somehow. Thus, after studying load-balancing challenge, the present research uses simulated annealing method, which is an algorithm for optimization issues and inspired from nature, for utilizing the criteria simultaneously.

After evaluating of the recommended method, the method could prove its efficiency. In addition, the recommended method is highly flexible and developed in a way that several load-balancing criteria used easily in this method, the number of criteria can be increase or decrease. The most important innovation of this research is consideration of several criteria at the same time as the purpose of migration and using simulated annealing for this reason. This makes the using of recommended algorithm possible in cloud environments with different conditions. The obtained results shows, the importance of using load-balancing process in cloud environment. According to previous studies, an introduced algorithm for load balancing does not have always the best result and it is depending on various factors but making migration purposeful can improve load-balancing process significantly. It is recommend to use dynamic programming algorithms in the future works to improve migration process in load balancing algorithms.

REFERENCES

[1] Aikebaier A, Enokido T, Takizawa M. "Trustworthy Group Making Algorithm in Distributed Systems".Human-centric computing and information sciences 1, No. 1, pp. 1-15, 2011.

[2] Begum S, Prashanth C.S.R., "Review of Load Balancing in Cloud Computing". IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 1, No. 2, pp. 343-352, 2013.

[3] Beloglazov A, Buyya R., "Energy Efficient Resource Management in virtualized Cloud Data Centers". 10th IEEE/ACM Int Conf. Cluster, Cloud and GridComputing , pp. 826-831, 2010.

[4] Kansal N.J, Chana I., "Cloud Load Balancing Techniques : A Step Towards Green Computing".IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No. 1, pp. 238-246, 2012.

[5] Padhy R. P, Rao G. P., "Load Balancing in cloud computing Systems". Thesis submitted in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering Bachelor Thesis, Orissa, India, pp. 1-46. 2011.

[6] Kirkpatrick S, Gelatt C.D, Vecchi M.P., "Optimization by Simulated Annealing". Science, New Series, Vol. 220, No. 4598, May 13, pp. 671-680, 1983.

[7] Kokilavani T, George Amalarethinam D.I., "Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing". International Journal of Computer Applications, Vol. 20, No. 2, pp. 42-48, 2011.

[8] Beloglazov A, Buyya R., "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers", Concurrency and Computation: Practice & Experience, Volume 24 Issue 13, pp. 1397-1420, 2012.

[9] Membrey P, Hows D, Plugge E., "Load Balancing in the Cloud", Chapter 13, pp.211-224, 2012.

[10] Nishant K, Sharma P, Krishna V, Rastogi N, Rastogi R., "Load Balancing of Nodes in Cloud Using Ant Colony Optimization", 14th International Conference on Modelling and Simulation, pp.3-8, 2012.

[11] Pathak K.K, Yadav P.S, Tiwari R, Gupta T.K., "A Modified Approach for Load Balancing in Cloud Computing Using Extended Honey Bee Algorithm", IJRREST: International Journal of Research Review in Engineering Science and Technology, Vol. 1, Issue 3, pp. 1-8, 2010.

[12] Rashmi K.S, Suma V, Vaidehi M., "Enhanced Load Balancing Approach to Avoid Deadlocks in Cloud". Special Issue of International Journal of Computer Applications on Advanced Computing and Communication Technologies for HPC Applications (ACCTHPCA), pp. 31–35, June 2012.

[13] Sran N, Kaur N., "Comparative Analysis of Existing Load Balancing Techniques in Cloud Computing". International Journal of Engineering Science Invention, Vol. 2, Issue 1, pp. 60-63, 2013.

[14] Wang S.C,Yan K.Q, Liao W.P, Wang S.S., "Towards a Load Balancing in a Three-level Cloud Computing Network". Chaoyang University of Technology Taiwan, R.O.C. 978-1-4244-5540-9/10/$26.00 ©2010 IEEE, pp. 1-6, 2010.

[15] Zhang S,Yan H,Chen X., "Research on Key Technologies of Cloud Computing". International Conference on Medical Physics and Biomedical Engineering, Hebei Province, China, pp. 1791–1797, 2012.

[16] Yao L, Wu G, Ren J, Zhu Y , Li V., "Guaranteeing Fault-Tolerant Load Requirement Balancing Scheme" Published by Oxford University Press on behalf of The British Computer Society, pp. 1-8. 2013.

[17] Wood T, Shenoy P, Venkataramani A, Yousif M., "Black-box and Gray-box Strategies for Virtual Machine Migration". 4th USENIX Symposium on Networked Systems Design, Implementation, Cambridge, April 11–13, pp. 229–242, 2007.

[18] Xu J, Fortes J.A., "Multi-objective virtual machine placement in virtualized data center environments". In Green Computing and Communications (GreenCom), IEEE/ACM Int'l Conference on & Int'l Conference on Cyber, Physical and Social Computing (CPSCom), pp. 179-188, 2010.

[19] Jonathan R, Wolfgan K, and Jurgen W., "Temperature Measurement and Equilibrium Dynamics of Simulated Annealing Placements";IEEE Transactions On Computer-Aided Design.VOL. 9. NO. 3, March, 1990.

[20] Sakamoto Sh, Tetsuya Oda, Elis K, Makoto I, Leonard B and FatosXh. (2013)." Performance Analysis of WMNs Using Simulated Annealing Algorithm for Different Temperature Values"; Seventh International Conference on Complex, Intelligent, and Software Intensive Systems.