IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 6, No. 3, September 2017, pp. 124~138
ISSN: 2252-8938, DOI: 10.11591/ijai.v6.i3.pp124-138

An Approch based on Genetic Algorithm for Multi-tenant Resource Allocation in SaaS Applications

Elaheh kheiri[1], Mostafa Ghobaei Arani[2], Alireza Taghizadeh[3]
[1]Department of Computer Engineering, Islamic Azad University Mahallat, Mahallat Branch, Iran
[2, 3]Department of Computer Engineering, Islamic Azad University Tehran, Parand Branch, Iran

Article Info

Article history:

Received Jun 10, 2017
Revised Aug 12, 2017
Accepted Aug 26, 2017

Keyword:

Cloud computing
Genetic algorithm
Multi tenant
Resource allocation
Utilization

ABSTRACT

In recent years, the use of cloud services has been significantly expanded. The providers of software as a service employ multi-tenant architectures to deliver services to their users. In these multi-tenant applications the resource allocation would suffer from over-utilization or under-utilization issues. Considering the significant effects of resource allocation on the service performance and cost, in this paper we have proposed an approach based on genetic algorithm for resource allocation which guarantees service quality through providing adequate resources. The proposed approach also improves system performance, meets the requirements of users and provides maximum resource efficiency. Simulation results show that the proposed approach has better response rate and availability comparing to other approaches, while provides an efficient resource usage.

Corresponding Author:

Elaheh kheiri,
Department of Computer Engineering,
Islamic Azad University Mahallat,
Mahallat Branch, Iran.
Email: elaheh_kheiri2002@yahoo.com

1. INTRODUCTION

Cloud computing as a new model for hosting and delivery services is attractive for company owners. It eliminates the need for users to design logistics and allows companies to start with small amount of resources and increase the resources according to thedemand [1]. Price model in cloud computing is based on consumption rate. A cloud provides its online business applications through a web browser or another software. Applications and data are stored on the servers and are available to users on demand. Details are kept hidden from the users who do not need proficiency or control over the cloud infrastructure that they use [2].

Cloud applications are provided to users as a service in the form of software or SaaS. This enables organizations to receive software services through the Internet, without having to pay the high cost of server, license, installation and commissioning of hardware and software. In this model, cloud providers install and run application software in the cloud, and cloud users access to the software by cloud clients. This eliminates the need to install and run the applications on the user's computer, and simplifies its maintenance and support [3].

Cloud computing architecture is primarily a multi-tenant service-oriented architecture. Multi-tenancy in cloud computing refers to the manner in architectural design of system that offer software as a service. This implies the fact that companies must not purchase and maintain IT infrastructure and their communications. A multi-tenant system shares a running application between a group of participants or tenants (service customer). Multi-tenancy is an organizational approach for SaaS applications that was introduced around 2005 [4]. In multi-tenancy the program distribution among servers is easier as only a

sample of the program should be distributed. On the other hand, hardware utilization rate would be improved. These two factors reduce the costs of all programs [5].

Since the multi-tenancy is a software architecture in the business model of SaaS, the price and quality of service are important issues in this area. Multi-tenant SaaS applications run on multiple servers in a distributed method. Each server has a portion of system resources such as processing capability of CPU, network bandwidth, storage capacity, etc. In multi-tenant SaaS applications, each tenant is used for a group of users with similar needs. As a result, the users of various tenant may need to have different kind and amount of resources to run the application. Therefore, each request that is sent by the users, obtain a part of providers' resources. The provider should ensure that the customers' requests will be met completely. The issue of resource allocation for multi-tenant applications has special significance due to its impact on the performance. There are various model for the allocation of resources in the field of cloud computing, each one uses specific techniques and algorithms [6].

Despite that in the cloud, one can automatically receive the resources on-demand, the other can still be faced with the problems related to insufficient resources. These creates under-utilization or over-utilization situation due to the use of pay-per-use model. Under-utilization or over-utilization are critical and unresolved challenges in the field of resource allocation [7].

Over-utilization happens once provider can not meet the requested service level. The profit from customers is lost due to poor performance and customers stop using programs after experiencing weak service that leads to permanent damage and loss of customers [8].

Under-utilization happens once the additional resources are kept even if they are not required (resources at non-peak time remain unused). As a result without an effective model, costly resources are lost when the load is not maximum [9]. From an economic viewpoint using less than capacity is a waste of money. This means that we pay money for something that we do not need or will not use. Using too little, describes the situation that some cloud resources are not used by the Virtual Machine (VM) and a program is run [8].

According to the description mentioned above, tenant based resource allocation is one of the ways to deal with non-optimum use of resources and has a considerable impact on cost-effectiveness of the SaaS system. So in order to use of resources optimally, the application of genetic algorithms in this field were investigated.

The rest of this paper is organized as follows: Section II is devoted to the related work and the proposed approach is presented in section III. Evaluation and simulation results of the proposed approach are presented in the section IV and finally in section V, conclusions and recommendations are provided.

2. RELATED WORKS

In this section various studies have been introduced about the multi-tenant resource allocation. The section also presents different categories and compares the presented approaches in term of important parameters

2.1. Tenant Based Resource Allocation Approaches

1. H.Cai et al. [10] suggest a Toolkit based on Java mechanism that supports multi-tenant granular mechanism. The authors used Context of Tenants, Context run-time elements that contain information of tenants.
2. J. Hua et al. [11], provide IVIC that is a platform for academic researchers to create custom virtual computing environments, dynamically for different scientific calculation, simulation and analysis through technology VM. They used Eucalyptus cloud computing platform, which offers SOAP interfaces and provides the ability to use according to VM instances request.
3. Y. Jie et al. [12] concluded that with current resource allocation models, SaaS providers will pay consecutive costs to use the world's resources without regard to resources used by each tenant. As a result, there is a need to create a flexible and correct architecture on the part that SaaS providers pay for actual use of resources.
4. Javier Espadas et al. [8], to achieve cost-effective scalability of SaaS and solving over-utilization and under-utilization problems, introduce a tenant based resource allocation model (TBRAM) for SaaS applications on cloud computing infrastructure. This model is comprised of three complementary approaches. The first is separation based on the tenant that separates context for different tenants. The second method is tenant based VM allocation. With this approach we will be able to calculate the actual number of required VM of each tenant at any given moment. But the last one is tenant based load balancing. This allows load balancing of virtual machines to be done about a certain tenant.

5. Marek Woda et al. [7], offered cost-effectiveness of model (TBRAM) for a SaaS system. Tenant based resource allocation model based is one of the ways to deal with non-optimum use of resources. When compared with traditional source scale, this method can reduce the cost of implementing the SaaS systems in cloud environments.

2.2. Resources utilization control in multitenant applications approaches

1. W. Wang et al. [13], in the field of separation and control of resources, use a Kalman filter to estimate CPU usage of the tenants who send different types of requests. These request estimates apply only to identify the malicious workload, not to identify individual subscriptions.
2. Previous valuation methods have been limited to a smaller number of requests to estimate the resource demands. There is no previous work by taking 20 or more requests. If Q. Zhang et al. [14], had studies by linear regression or Kalman filters using the application basis TPC-W that has 14 different types of requests. Kraft et al. [15], have also evaluated the impact of the number of requests (between one and five) on different linear regression and maximum likelihood of resource estimation methods.
3. Simon Spinner et al. [16], provide control of resource utilization in multi-tenant applications. They argue a way to support multi-tenant applications in order to guarantee the performance for each tenant. They explain their way to control the use of resources in MTAs. The general idea is that the resource demand of every tenant is checked in the first step. In the second step, these resource demands are used to control the tenants' resource use individually. They evaluated three different approaches of resource demand estimation based on Kalman filtering, linear regression and the Service Demand Law (SDL).

2.3. Tenant based resources allocation optimization approaches

1. System resources allocation optimization approach is an effective way to improve system performance, ensure QoS and meet the requirements of the user. As a classic type of optimization issue, system resources allocation optimization issue is a NP-hard problem without a resource allocation optimization algorithm with polynomial time complexity. Therefore, an effective algorithm must agree on the accuracy of performance. SaaS applications consume resources in a manner similar to web-based applications. In the research optimized system resource allocation in web-based application systems, there is a number of well-known algorithms such as LPT, BoundFit and MULTIFIT. These algorithms are easily implemented and under some conditions they can obtain solutions close to the best solution. But they also have disadvantages such as high cost and quality of unsustainable results. And some algorithms have been restricted to solving problems with a source or to homogeneous environment of resources. So they are not adequate and suitable for optimization problems with some type of multi-tenant SaaS applications with heterogeneous resources. Rajkumar R et al. [17], have studied QoS based on multi-dimensional resource allocation problem, based on the above-mentioned algorithms. But results are informal and theoretical and not applied.
2. LIU Anfeng et al. [18], proposed a two-phase optimization heuristic algorithm based on LPT and MMKP algorithms for cluster web. These two algorithms have been successful for solving problems related to QoS resources. But the purpose of these two algorithms is maximizing the use of resources that is different with the purpose system of resource allocation of SaaS applications. In addition, in two above algorithms, each task is limited to being in just one server but in SaaS application, each tenant can be spread to multiple servers with a limited number of users.
3. Deshuai Wang et al. [19], considering the weakness of the previous algorithm, offered a system resource allocation method for multi-tenant SaaS applications to ensure QoS, of SaaS applications while overall performance is optimized. System resource allocation procedures are important for multi-tenant SaaS application to provide services with acceptable performance and quality. They raised optimization problem of system resource allocation in multi-tenant SaaS applications, and then they have submitted two tenant QoS oriented system resource allocation algorithm based on resource efficiency (TQOSRAA-RE) and Tenant QoS oriented system resource allocation algorithm based on genetic algorithm (TQOSRAA-GA).

Finally Table 1 shows the comparison of above techniques.

Table 1. Comparison of the related works

Context	Reference	Technique	Advantage	Disadvantage
Tenant-based resource allocation	H. Cai et al. [11]	Providing a Java mechanism-based toolkit	-Consideration of multi-tenant mechanism -separation of tenant's data	-Lack of consideration of applications' nature -Lack of consideration of performance
	J. Huai et al. [11]	providing an IVIC as a platform to create customized virtual computing environments	-Various scientific calculation, simulation and analysis by VM technology -Monitoring and measuring virtual resources	-Lack of consideration of applications' nature -Lack of consideration of performance
	Y. Jie et al. [12]	Providing scalable applications, virtualization, cloud mechanisms, resource allocation models	-Achievement of scalability based on number of users	-Lack of access to affordable scalability of SaaS -inefficiency
	Javier Espadas et al. [8]	A tenant-based resource allocation model (TBM)	-Access to affordable scalability of SaaS	-Ignoring all parameters affecting the performance, guarantee quality of service and improve system performance and etc. simultaneously
	Marek Woda et al. [7]	Suggest of cost-effectiveness of tenant-based resource allocation model for a SaaS system and comparison of sources scale	-Reduced costs compared to traditional source scale -Cost-effectiveness of TBRAM	-Ignoring all parameters affecting the performance, guarantee quality of service and improve system performance and etc. simultaneously
Control the use of resources in multi-tenant applications	W.Wang et al. [13]	Controlling use of resources (Kalman filter method)	-CPU usage estimate of tenants that send different types of demands -Demand estimation are used only to identify malicious workload -Identifying the tenant and the type of demand that leads to the maximum usage	-Ignoring malicious requests -lack of sharing resources -Demand estimation are used only to identify malicious workload
	Q.Zhang et al. and Kraft et al. [14]	Estimation of resource demand (Kalman filter method, linear regression)	Resource demand estimation	Resource demand estimation with a fewer variety of source requests (less than 20 requests)
	Simon Spinner et al. [16]	-Controlling the use of resources in multi-tenant applications and request resource estimation (Kalman filter, linear regression and the service demand law) -Combining admission control mechanism by request resource estimation technique	-Different resources request support -Able to share resources -Managing malicious requests -Estimation of the resource demands for a large number of resource demands -Performance guarantee	-Ignoring all parameters affecting the performance, guarantee quality of service and improve system performance and etc. simultaneously
Tenant-based resource allocation optimization	Rajumar R et al. [17]	-Evaluates some well-known algorithms such as LPT, MULTI FIT and Bound Fit	-Easy implementation -Under some circumstances, they can get the solutions near the best solution -These algorithms to respond to issues of one type source or in homogeneous resource circumstance	-The disadvantages of high cost -Unstable quality of results -These algorithms are not appropriate to respond problems from some kind of sources or in a heterogeneous sources environment
	LIU Anfeng et al. [18]	Providing Two-phase heuristic optimization algorithm for of web-based cluster on development of LPT algorithm and heuristic algorithm MMKP	-These two algorithms are able to respond to resource allocation problems related to quality of service -Maximizing the use of resources for the purpose of allocating system resources to different SaaS applications	-The use of mentioned algorithms to respond to the issue of system resources allocation of SaaS application that will not get optimized answer and performance -Inefficiencies and optimized performance
	Deshuai Wang et al. [19]	system resource allocation methods for multi-tenant SaaS application (algorithms of TQOSRAA-RE and TQOSRAA-RA)	-Obtaining near best results with polynomial time complexity -Optimized Performance -Responding to issues that call and receive optimized performance	-Ignoring all parameters affecting the performance, guarantee quality of service and improve system performance and etc. simultaneously

3. PROPOSED SOLUTION

In this section, to overcome the issues of over-utilization and under-utilization in resource allocation for SaaS applications, we propose tenant based resource allocation model algorithm using genetic algorithm (TBRAMA-GA) and tenant based resource allocation model algorithm using heuristic algorithm (TBRAMA–HE):

3.1. Tenant-based VM allocation algorithm based on genetic algorithm of TBRAMA-GA

Based on the idea of genetic algorithm, we have proposed the TBRAMA-GA algorithm, TBRAMA-GA steps have been described below:

Step 1. Initial population

In TBRAMA-GA the population is made up of chromosomes that, each chromosome represents the weight of the tenant. And the initial population is equal to the weight vector.

Weight Vector= {w1, w2, w3,…}

So, the first generation of population including n chromosomes is generated of the population randomly.

Step 2. Fitness Function

As we said, our goal is to determine the minimum number of VM samples required to allocate weight vector with respect to VM capacity. Fitness function measures the quality of solutions offered. So based on our goal, fitness function can be expressed as follows (Equation 1).

$$F(x)=\sum \left(\max \sum_{j=1}^{w.lenght} w_j \; where \; w_j \; \leq VM_{capacity} \right) \quad (1)$$

Then, for each chromosome of the population, the fitness function is calculated.

Step 3. Selection

The roulette wheel selection operator is chosen as the selection operator for TBRAMA-GA. In roulette wheel, the wheel area is divided into sections that their number is equal to the number of members and area of each section is proportional to the level of fitness of each chromosome. Then, the wheel is turned until it stops at a point accidentally. This point highlights the selected chromosome. So the chromosomes that have high fitness value may be selected several times in the production process, while the chromosomes that have low fitness may not be chosen ever. First, eliminates all chromosomes with negative compatibility. Then calculates the compatibility probability for the remaining chromosomes. During each successful production, roulette selection operator keeps the quality and diversity for the next generation by choosing the best solution randomly.

Additional weights are identified and the number of VM prototype for weight vector is calculated and as far as possible assigns weights with a primary VM.

Step 4. Reproduction

The next step is the production of second-generation population from solutions from those selected, that genetic operators is one of them: crossover and mutation.

1. Crossover

We adopted uniform crossover method and genes were adapted as the main unit for 2 parents to produce child. Each gene is selected randomly from one parent. At this step, crossover operator acts with the Pc possibility on parent chromosome and produces a new chromosome by combining them.

2. Mutation

We changed each gene to a random value in a constant possibility. Possibility is usually set on 10%. At this step, mutation action is done with Pm probability on chromosome resulted of movement, and new information is obtained by changing the bits of chromosomes.

At this point a new population is selected to enter the next stage of the algorithm. This is done by comparing the fitness of chromosomes.

At this step, removing excess weights for weight vector processing is done. The remaining weights that have not been allocated in the first generation, have been elected for the next generation.

Step 5. End

TBRAMA-GA end condition is:

If the remaining weight vector has zero length algorithm will be ended and number of VM samples that should be implemented, so all tenant workload would be allocated, is achieved. Otherwise, it returns to step 2.

3.2. Tenant-based VM allocation algorithm based on heuristic algorithm TBRAMA-HE

In literature of system resources studies, greedy algorithm-based method is popular because of efficiency and simplicity. By adopting the idea of greedy algorithm, we have proposed algorithm, TBRAMA-HE steps have been described.

Input: weight vector set {w1, w2, w3, ...} and VM capacity

Output: The number of VM samples

Goal:

$$F(x)=\sum (\max \sum_{j=1}^{w.lenght} w_j \; where \; w_j \; \leq VM_{capacity}) \quad (2)$$

Algorithm1: TBRAMA -HE

```
TBRAMA -HE (Capacity, Tenant Weight)
BEGIN
   VMs=0;
   While (Tenant Weight !=0)
      X=Select (Tenant Weight);
      Tenant Weight=Tenant Weight-{X};
      IF (X>Capacity)
         VMs=X/Capacity +VMs;
      ELSE
         VMs++;
      END IF
   End While
   Return VMs;
END
```

According to the idea of greedy algorithm during each stage, an element is added to the solution set, restriction condition has been examined, and if there is no problem, element and the next elements are added to the solution set accordingly. During these steps if you reach a certain final condition, or there is no possibility to select another element to add to the solution set, the algorithm has been ended and achieved solution set is provided as optimum solution.

4. PERFORMANCE EVALUATION

In this section, the results of the implementation of the proposed method is examined for allocating resources in multi-tenant cloud architecture of SaaS applications. Simulation was performed in Netbeans environment by using the Cloudsim tools. Configuration of simulated parameters has been shown in Tables 2 and 3.

Table 2. Characteristics of the data center

Data Center	Architecture	X86
	Operating System	Linux
	Virtual Machine Manager	Xen

Table 3. Characteristics of host

Server name	HP ProLiant G5	Cores	Frequency (MIPS)	RAM (GB)	BW	Cost
HP ProLiant G4	Intel Xeon 3040	2	5580	4	1 Gbit/s	1200
HP ProLiant G5	Intel Xeon 3075	2	7980	4	1 Gbit/s	1800

A user records several tasks where each task is heterogeneous with different facilities of computational parameters such as different processor speed, disk memory, RAM and network parameters, including latency and different bandwidth for combining concepts.

For the table of weights associated with each tenant each, file is used that, on each row, weight associated with considered tenant can be harvested. Request rate of the servers is considered 10 MIPS by default. The source used in this simulation is CPU, so that the data sent from the server is processed for a specific time and after processing, the response is sent back to the server again. In the simulation, a queue has been considered for Data Center as the first control structure, that the queue size will vary based on the resulting VM of the proposed method. Time of simulation can vary based on the number of servers. Thus, at

the beginning of simulation, each server sends a request and simulation will continue to get answers to all servers. Figure 1 shows a view of a simulated environment.

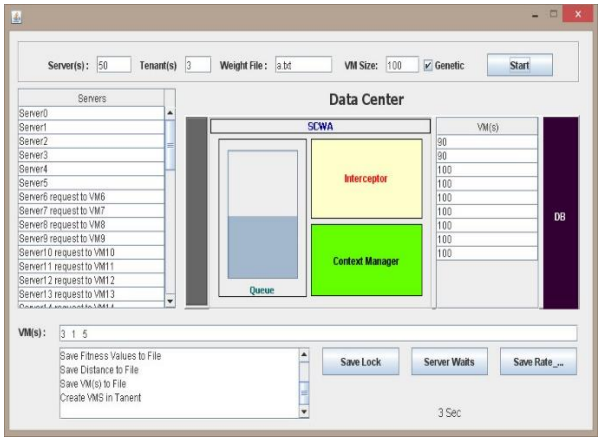

Figure 1. A view of simulated environment

4.1. Performance Metrics

To evaluate the proposed approach, three different scenarios have been defined according to Table 8. In each scenario, the number of tenants is constant and the number of servers is varied. In each simulation the three criteria of response time, waiting time of servers and availability of Data Center in TBRAMA-GA algorithm and TBRAMA-HE are examined.

Response time: Response time is the exact time difference between the time of request and time of delivered task.

Server waiting time: The duration that each server is waiting for finding source of request transfer.

Availability: Assume that $R_1, R_2, \dots, R_n$ are cloud sources, for$= 1,2, \dots, n$, N_k has been adapted as the number of recorded tasks and A_k is adapted number of tasks for the cloud source R_k in the time range of T; So that, the availability is obtained according to the Equation 3.

$$Availability\ (AV)\ R_k = \frac{A_k}{N_k} \quad (3)$$

There has been defined three scenario for evaluation proposed approach, in Table 4:

Table 4. Evaluated Scenarios

Objective	Number of servers	Number of tenants	scenario
Calculation of the response time, server waiting time and availability of Data Center in TBRAMA-GA and TBRAMA-HE algorithms	200,500,1000	3	1st. scenario
	200,500,1000	5	2nd. scenario
	200,500,1000	10	3rd. scenario

4.2. Evaluation of the First Senario

Resource management for multi-tenant structure is the main problem in this research. The number of tenants in this simulation is considered 3. Number of servers is variable and have been considered 200, 500 and 1000. Weight allocated to each tenant has been specified in Table 5.

Table 5. The weight assigned to each tenant

Tenant 1	Tenant 2	Tenant 3
254	21	434

The simulation is done using the proposed method. Considering the use of genetic algorithm, the number of VMs is determined. Number of generations and the initial population has been considered 50, and

the mutation operator value has been considered 0.7 and 0.01. Figure 2 shows graphs of the mean fitness function during 50 generations.

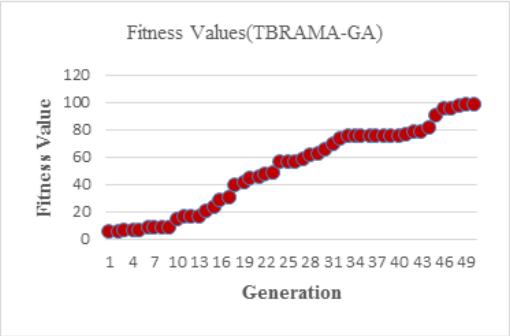

Figure 2. The graph of the fit function values during 50 generations

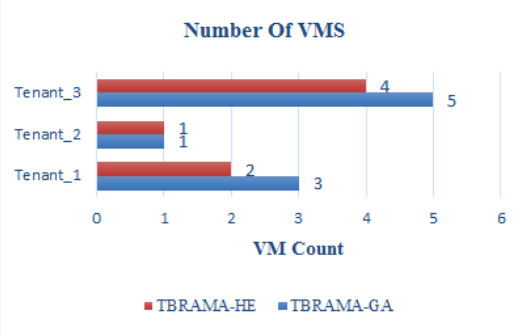

Figure 3. VM number per each tenant

The number of VMs is determined based on the weight assigned to each tenant. Figure 3 shows VM numbers per tenant in two TBRAMA-GA and TBRAMA-HE algorithms.

One of the most important indicators of the service quality is request time. Figure 4 has investigated response time in TBRAMA-HE and TBRAMA-GA algorithms.

Figure 4 indicates that the proposed algorithm or TBRAMA-GA has the highest speed in responding to requests by servers.

Figure 5 shows the average waiting time of servers using graph algorithms TBRAMA-GA and TBRAMA-HE. It is clear that the longer waiting time to find resource, the longer response time will be.

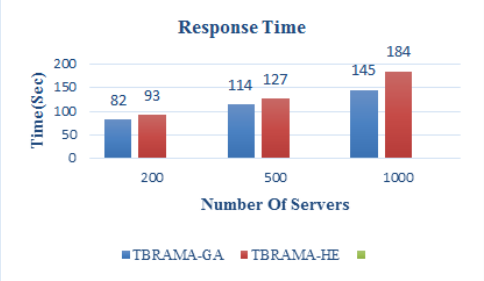

Figure 4. Response time in both algorithms

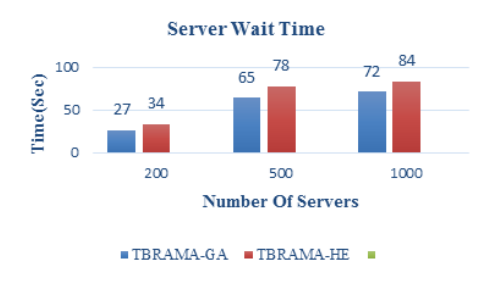

Figure 5. Servers waiting time for finding source in three algorithms

As Figure 5 shows, waiting time of servers in TBRAMA-GA method is less than another algorithm, which improves the efficiency of multi-tenant cloud computing system. One of the most important factors in the selection of optimal algorithm is availability of resources and Figure 6 shows this factor in two algorithms. The higher system availability is, the higher service quality we have. Given the number of requests and response rates per unit of time we can calculate availability of any source, and finally the availability of the entire system.

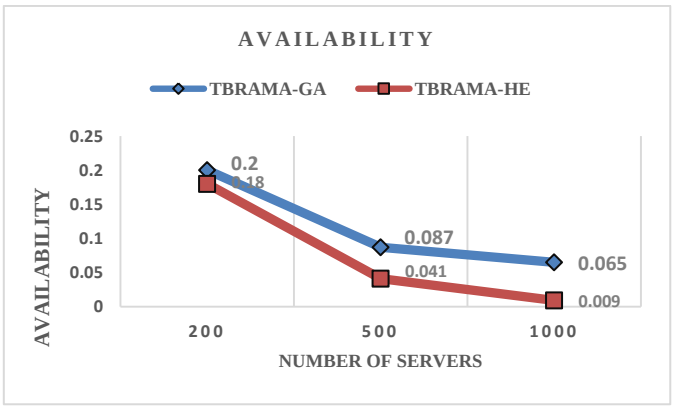

Figure 6. The average availability of resources in the three algorithms

Results of Figure 6 shows the better performance of TBRAMA-GA in terms of availability compared to TBRAMA-HE. According to the figure, whatever the number of requests increases availability of TBRAMA-GA algorithm increases compared to algorithm TBRAMA-HE.

4.3. Evaluation of the Second Senario

The number of tenants and weights in this simulation are equal to 5. The weights allocated to each tenant to use and find the number of VMs has been specified in Table 6.

Table 6. Weights allocated to each tenant

Tenant 1	Tenant 2	Tenant 3	Tenant 4	Tenant 5
370	195	80	493	210

In TBRAMA-GA algorithm, generation and initial population of 50, the combine operator value of 0.7 and mutation operator of 0.01 have been considered. Figure 7 shows the graph of mean values of fitness function during 50 generations. The number of VM is determined based on the weight allocated to each tenant. Figure 8 shows VM numbers per tenant in TBRAMA-GA and TBRAMA-HE algorithms.

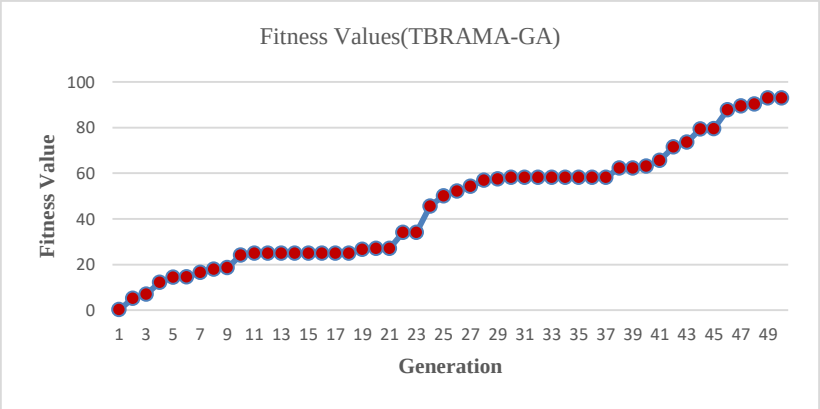

Figure 7. Graph of the fitness function during 50 generations

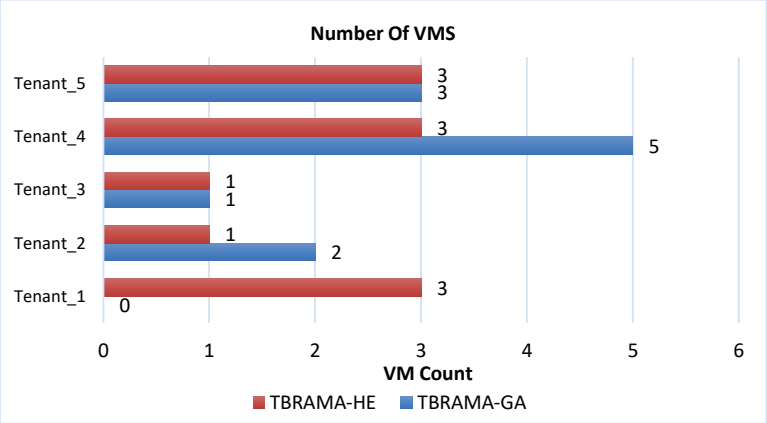

Figure 8. The number of VM per tenant

One of the most important indicators of the service quality is response duration. Figure 9 has investigated response time in TBRAMA-HE and TBRAMA-GA algorithms. Given that response speed indicator one of the main quality standards of resource allocation algorithm, by investigating Figure 8, it has determined that the proposed algorithm or TBRAMA-GA has the highest speed in responding to requests by its servers. Figure 10 shows the average waiting time of servers using two algorithms TBRAMA-GA and TBRAMA-HE. It is clear that the longer waiting duration to find the source will cause the longer response time.

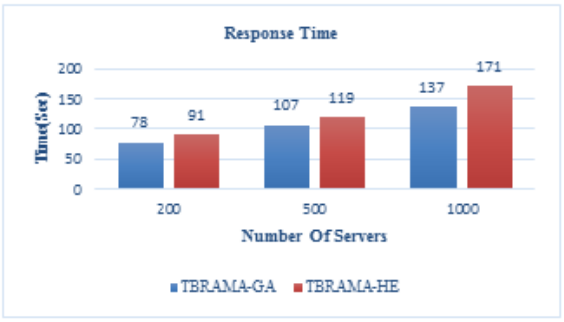

Figure 9. The response time in TBRAMA-GA and TBRAMA-HE algorithms

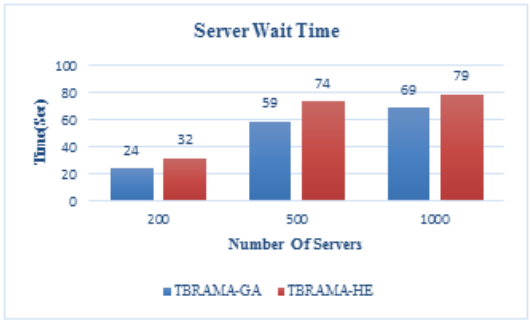

Figure 10. The server waiting time to find the source

The server waiting duration to receive confirmation from the Data Center to send the request is measured in this section. As Figure 10 shows, the server waiting duration in TBRAMA-GA method is less than another algorithm, which improves the efficiency of multi-tenant cloud computing system.

One of the most important factors in the selection of the optimal algorithm is the availability of resources. Figure 11 shows this factor in TBRAMA-GA and TBRAMA-HE algorithms.

Results of Figure 10 shows better performance of TBRAMA-GA in terms of availability than TBRAMA-HE algorithm.

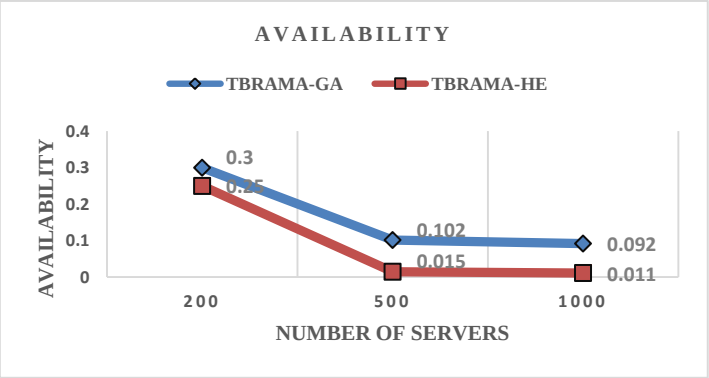

Figure 11. Average availability of resources in TBRAMA-GA and TBRAMA-HE algorithms

4.4. Evaluation of the Third Senario

The number of tenants in this simulation have been considered 10 tenants. Number of servers is variable and the number of 200, 500 and 1000 have been considered. Weight allocated to each tenant has been specified in Table 7.

Table 7. The weight allocated to each tenant

1	2	3	4	5	6	7	8	9	10
142	270	59	120	89	387	420	275	105	380

Using a genetic algorithm, the number of VMs is determined. And the number of generations of 50, the combine operator of 0.7 and mutation operator value of 0.01 have been considered. Figure 12 shows the average fitness function graph during 50 generations.

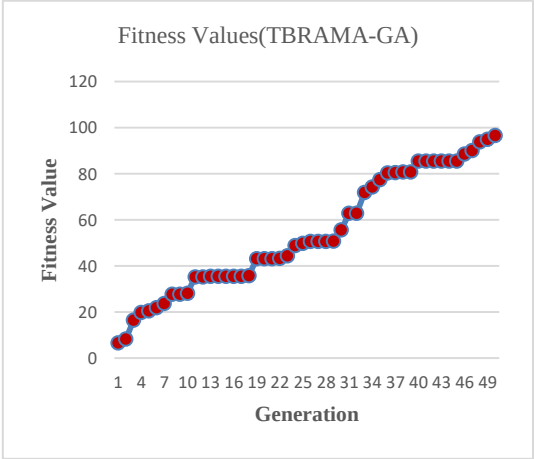

Figure 12. Graph of the fitness function during 50 generations

The number of VMs is determined based on the weight allocated to each tenant. Figure 13 indicates the number of VM per tenant in TBRAMA-GA and TBRAMA-HE algorithms.

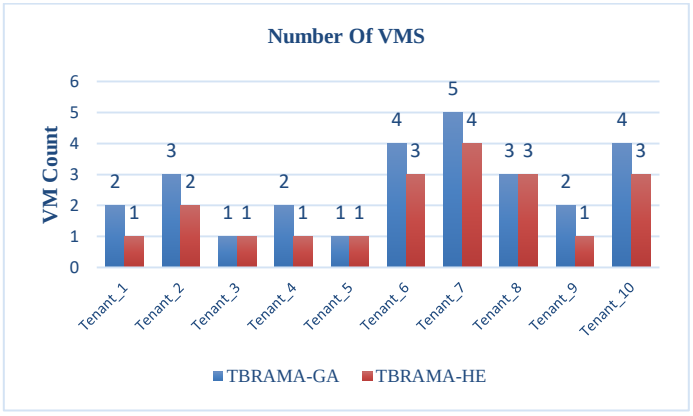

Figure 13. VM numbers per each tenant

The response time of server is, one of the most important indicators in this field. Figure 14 has investigated response time in TBRAMA-HE and TBRAMA-GA algorithms.

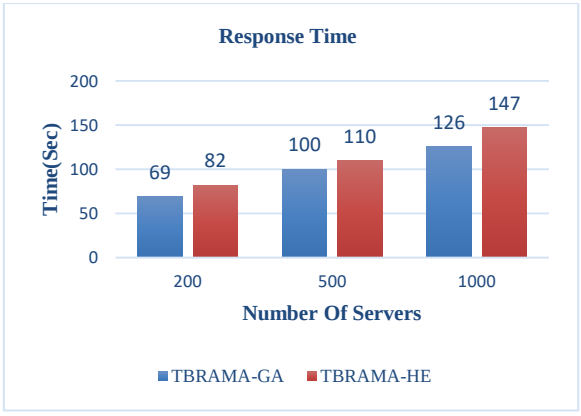

Figure 14. Response time in two algorithms

Figure 14 indicates that the proposed algorithm or TBRAMA-GA has the highest speed in responding to the requests sent by the server.

Figure 15 shows average waiting time of servers using two algorithms TBRAMA-GA and TBRAMA-HE. It is clear that how much the waiting duration is longer to find the source, the response time will be longer.

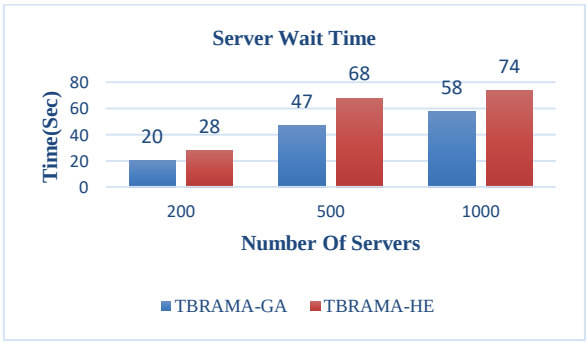

Figure 15. Servers waiting time for finding source in the two algorithms

Figure 15 shows, waiting time of servers in TBRAMA-GA method is less than another algorithm, which improves the efficiency of multi-tenant cloud computing system. One of the most important factors in the selection of optimal algorithm is availability of resources and Figure 16 shows this factor in two algorithms. The higher system availability is, the higher service quality we have. Given the number of requests and response rates per unit of time we can calculate availability of any source, and finally the availability of the entire system.

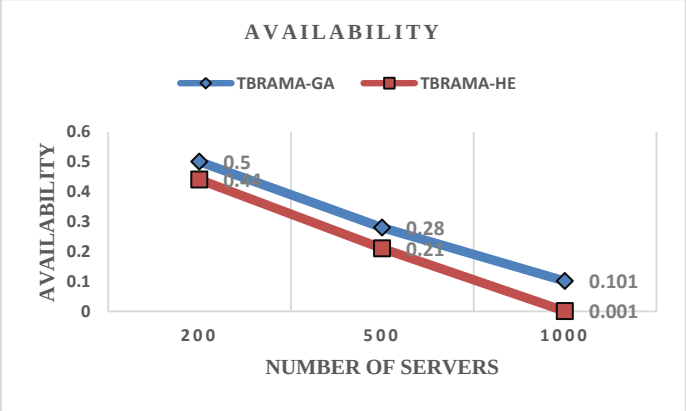

Figure 16. The average availability of resources in the three algorithms

Results of Figure 16 shows the higher performance of TBRAMA-GA in terms of availability compared to TBRAMA-HE. According to the figure, whatever the number of requests increases availability of TBRAMA-GA algorithm increases compared to algorithm TBRAMA-HE.

4.5. Total Evaluation

In multi-tenant structure resources management, response speed and availability are two important parameters in the quality of providing service. These two important criteria were assessed in previous evaluations and according to the results of assessments carried out in this section, the average of the two measures were calculated and compared. Figure 17 shows average response time in three modes with 3, 5 and 10 tenants. According to Figure 16, it is clear that the speed of response in TBRAMA-GA is higher than TBRAMA-HE.

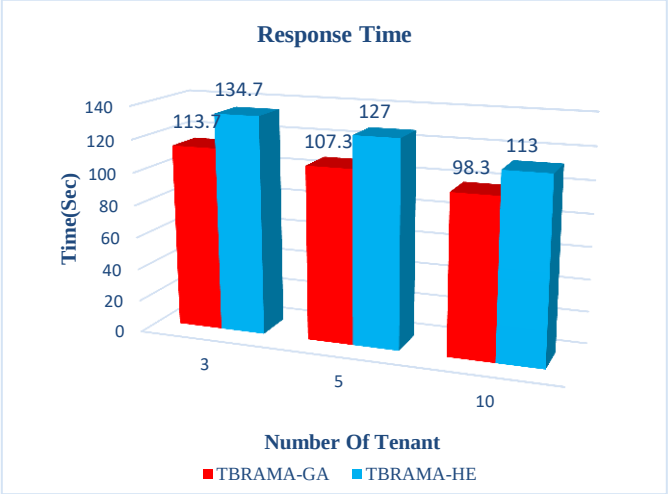

Figure 17. Average response time in two algorithms, TBRAMA-GA and TBRAMA-HE

Figure 18 show the average availability of resources in the modes with 3, 5 and 10 clients. Results of Figure 18 shows the higher performance of TBRAMA-GA than TBRAMA-HE in terms of availability.

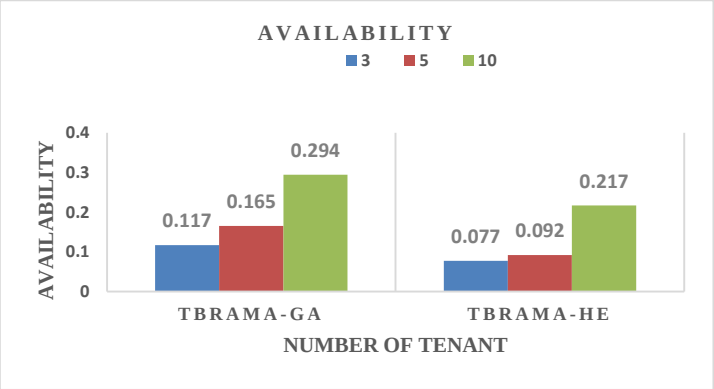

Figure 18. Average availability in two algorithms, TBRAMA-GA and TBRAMA-HE

5. CONCLUSION AND FUTURE WORK

Cloud computing is a new way of delivering services to users. It provides on-demand access to resource based on pay-per-use model. Having a cost-effective scalable resource allocation is a difficult issue. Resource allocation is directly related to the performance benefit of service providers and users' cost.

In this paper, the problem of resource allocation is discussed and an innovative approach for multi-tenant resource allocation is introduced. This approach guarantees service quality levels by provisioning the adequate resources. It also improves system performance and provides maximum efficiency in use of resources. In order to allocate resources for each tenant, the idea of a genetic algorithm so called TBRAMA-GA and a heuristic algorithm called TBRAMA-HE were employed.

During simulations, three different parameters including response time, servers waiting time and availability were evaluated in both TBRAMA-GA and TBRAMA-HE. The results showed that TBRAMA-GA performs better than TBRAMA-HE.

One of future research that will follow the current paper is to focus on the load balancer in the structure of TBRAM and evaluate the results of a SaaS system in terms of tenant-based resource test. Different approaches can be introduced to improve and validate tenant-based resource allocation model and various platforms can be used in the cloud infrastructure such as high-performance computing (HPC) or online trading programs. It is also possible to define other types of sources to be measured such as bandwidth, storage, data transfer or database connection. In this way, new models and mechanisms for measuring the status of virtual machines must be defined and implemented in order to collect results for this resources.

REFERENCES

[1] Tsai, W.T., Sun, X. and Balasooriya, J., 2010, April. Service-oriented cloud computing architecture. In *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on* (pp. 684-689). IEEE.

[2] Bana, A. and Hertzberg, D., 2015. Data Security and the Legal Profession: Risks, Unique Challenges and Practical Considerations. *Business Law International*, *16*(3).

[3] Tsai, W.T., Sun, X. and Balasooriya, J., 2010, April. Service-oriented cloud computing architecture. In *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on* (pp. 684-689). IEEE.

[4] Bezemer, C.P. and Zaidman, A., 2010, September. Multi-tenant SaaS applications: maintenance dream or nightmare? In *Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE)* (pp. 88-92). ACM.

[5] Krebs, R., Momm, C. and Kounev, S., 2012. Architectural Concerns in Multi-tenant SaaS Applications. *CLOSER*, *12*, pp.426-431.

[6] Arkaitz Ruiz-Alvarez, Marty Humphrey, "Review of Delay and Cost Efficient Methods in Cloud Computing", *in International Journal of Recent Technology and Engineering (IJRTE)*, ISSN: 2277-3878, Volume-2, Issue-5, November 2013.

[7] Espadas, J., Molina, A., Jiménez, G., Molina, M., Ramírez, R. and Concha, D., 2013. A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures. *Future Generation Computer Systems*, 29(1), pp.273-286.

[8] Cai, H., Zhang, K., Zhou, M., Gong, W., Cai, J. and Mao, X., 2009, September. An end-to-end methodology and toolkit for fine granularity SaaS-ization. In *Cloud Computing, 2009. CLOUD'09. IEEE International Conference on* (pp. 101-108). IEEE.

[9] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I. and Zaharia, M., 2009. BAbove the clouds: A Berkeley view of cloud computing, [Electr. *Eng. Comput. Sci. Dept., Univ. California, Berkeley, CA, Tech. Rep. UCB/EECS-2009-28*.

[10] Huai, J., Li, Q. and Hu, C., 2007, September. Civic: a hypervisor based virtual computing environment. In *Parallel Processing Workshops, 2007. ICPPW 2007. International Conference on* (pp. 51-51). IEEE.

[11] Yang, J., Qiu, J. and Li, Y., 2009, September. A profile-based approach to just-in-time scalability for cloud applications. In *Cloud Computing, 2009. CLOUD'09. IEEE International Conference on* (pp. 9-16). IEEE.

[12] Stolarz, W. and Woda, M., 2013. Proposal of cost-effective tenant-based resource allocation model for a SaaS system. In *New Results in Dependability and Computer Systems* (pp. 409-420). Springer International Publishing.

[13] Wang, W., Huang, X., Qin, X., Zhang, W., Wei, J. and Zhong, H., 2012, June. Application-level CPU consumption estimation: Towards performance isolation of multi-tenancy web applications. In *Cloud computing (cloud), 2012 IEEE 5th international conference on* (pp. 439-446). IEEE.

[14] Zhang, Q., Cherkasova, L. and Smirni, E., 2007, June. A regression-based analytic model for dynamic resource provisioning of multi-tier applications. In *Autonomic Computing, 2007. ICAC'07. Fourth International Conference on* (pp. 27-27). IEEE.

[15] Kraft, S., Pacheco-Sanchez, S., Casale, G. and Dawson, S., 2009, October. Estimating service resource consumption from response time measurements. In *Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools* (p. 48). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[16] Krebs, R., Spinner, S., Ahmed, N. and Kounev, S., 2014, May. Resource usage control in multi-tenant applications. In *Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on* (pp. 122-131). IEEE.

[17] Rajkumar, R., Lee, C., Lehoczky, J.P. and Siewiorek, D.P., 1998. A QoS-based Resource Allocation Model. *Proceedings of the IEEE Real-Time Systems Symposium.*

[18] LIU, A.F., CHEN, Z.G. and ZENG, Z.W., 2004. Heuristic Optimization Algorithm of Resource Load Balancing for Web Cluster [J]. *Mini-micro Systems*, *12*, p.005.

[19] Yu, H. and Wang, D., 2011, December. System resource allocation algorithm for multi-tenant SaaS application. In *Cloud and Service Computing (CSC), 2011 International Conference on* (pp. 207-211). IEEE.