Optimum Resource Allocation using Specification Matching and Priority Based Method in Cloud

Akhil Chaurasia
Department of Computer Science and Engineering
Sant Longowal Institute of Engineering and Technology
Deemed University, Longowal, Sangrur, India
e-mail:akhilchaurasia47@gmail.com

Jaspal Singh
Department of Computer Science and Engineering
Sant Longowal Institute of Engineering and Technology
Deemed University, Longowal, Sangrur, India
e-mail:safrisoft@yahoo.com

Abstract— Cloud computing is summed up as a different model for allowing favorable, network as per demand to use shared devices of computational resources which are collected and then released with marginal management effort or interaction with any client or any service provider. Cloud computing is a well-known technology in the pasture of information technology that provides computing as a service. In cloud computing environment the resources are provisioned on the basis of demand, as and when required. A large number of cloud users can request a number of cloud services at the same time. Due to increase in the usage of cloud computing there is a need for a efficient and effective resource allocation algorithm which can be used for proper usage of the resources and also check that the resource is not wastage. In this we propose a priority based resource allocation algorithm which can be used for proper allocation of resources and also the resources are allocated efficiently and effectively. In this paper, two strategies are proposed for the purpose of optimum resource allocation in which the first approach uses the concept of specification matching and second uses the concept of priority based approach. In the first approach, different types of resources (virtual machine) are allocated by taking three parameters into consideration: processing element, main memory, and network bandwidth. In the second approach, one parameter is considered namely: Priority. In both strategies, users are allowed to submit the parameters during cloudlet submission. The user inserted parameters will then be considered while allocating resources to them. The objectives of this research are to improve utilization of resources and reduce the request loss.

Keywords- *cloud computing, specification matching , priority*

*****

I. INTRODUCTION

Cloud computing "refers to both the applications delivered as services over the Internet, and the hardware and system software in the data centres that provide those services", according to Armbrust et al.[1], and "is a utility oriented distributed computing system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers" according to Buyya et al. [2]. Cloud computing can be considered as an extension of grid computing. One of the main characteristics of cloud computing is on-demand self-service. That means Cloud computing characteristically has provision for on-demand IT resource allocation and instantaneous scalability. Unlike Grid computing that typically provides persistent and permanent use of all available IT resources, the cloud computing is very specific on the consumer's demand, based on his current computing requirements and therefore eliminates over-provisioning of available IT resources.

The organizations can save the huge amount of expense by avoiding build and manage large data centers for in-house applications or data storage. The consumers of the cloud computing do not have to own the IT infrastructure and therefore need not care about the maintenance of servers and networks in the cloud. They just pay for services on demand that is based on the running of application instances normally varying depending upon the use of Internet bandwidth, a number of instances of action and amount of data transferred at a specific time.

There are a variety of background activities in cloud computing such as allocation of virtual machines (VMs), load sharing, load balancing, process migration, and shared memory access etc. which are completely abstracted from the users view [3]. If the workload is shared by all the available resources, it is known as load sharing. An even distribution of workload on available resources is called load balancing [4].

One of the elementary aspects of virtualization technologies engaged in cloud environments is resource consolidation and managing. Using hypervisors inside a bunch environment allows for a number of standalone physical machines to be consolidated to a virtualized situation, thereby need less Physical Machine (PM) physical resources than ever before. The perception of Cloud computing has not only adjusted the field of distributed systems but also fundamentally changed how businesses consume computing today. While Cloud computing offers many advanced features, it still has some shortcomings such as the moderately high working cost for both public and private Clouds. Better resource allocation is depends on maneuver over utilization of physical machines and virtual machines [5].

In this paper, we present an efficient resource allocation technique that will help cloud owner to reduce wastage of

resources and to achieve maximum profit. Efficient resource allocation in the cloud is a very challenging task as it needs to satisfy both the user's requirements and server's performance equally. Resource allocation in cloud computing environment is defined as the assignment of available resources such as CPU, memory, storage, network bandwidth etc in an economic way. It is the main part of resource management. Yet, an important problem that must be addressed effectively in the cloud is how to manage Quality of Services (QoS) and maintain Service Level Agreement (SLA) for cloud users that share cloud resources.

The proposed policy based resource allocation strategy provides maximized resource utilization and reduced completion time of user requests. Four parameters are used by the proposed approach for resource allocation to user requests namely: Processing Element (PE), Main Memory (RAM), Network Bandwidth (BW) and Priority (P). CloudSim-3.0.3 simulator is used for the purpose of simulation of results and to analyze the behavior of proposed work.

The rest of this paper is organized as: Section II describes different related works regarding allocation policies in a cloud environment and about the parameters which have been used by the researchers. Section III proposes a policy based resource allocation approach (PRAA), by taking four parameters into consideration for allocation of the resources. Section IV describes the simulation evaluations on CloudSim, which confirms the effectiveness of the proposed approach. Section V presents the conclusions and future scope.

II. RELATED WORK

Garima et al .[6] proposed a priority based earliest deadline first algorithm with task migration. The algorithm that discussed in this paper used two job scheduling algorithm one is priority based and second is the earliest deadline first scheduling algorithm. The tasks were scheduled on the basis of priority and high priority task gets scheduled first. The earliest deadline first scheduling algorithm had scheduled the tasks according to their deadline. The task having earliest deadline get scheduled first and then the other tasks having the earliest deadline will be scheduled next. The tasks were scheduled pre-emptively and preemptable tasks would be migrated to the other virtual machine.

Pawar and Wagh [7] had been present dynamic resource allocation for preemptable task execution in the cloud. The proposed priority based algorithm which considered multiple SLA parameters such as memory, network bandwidth, and required CPU time. In order to achieve the agreed SLA objective, the proposed algorithm dynamically provisioned the resources by preempting the low priority task with high priority task.

Natasha and Gill [8] proposed a priority based resource allocation method for handling a situation where two or more requests at a particular instance of time had the same priority. The work flow of this model was discussed in three stages. In stage I, initial parametric values were generated such as attaching priority to the request of Cloud user and the user requests were sorted in descending order based the priority. After priority assigned, requests with same priority were grouped into a group known as an open group. A ready queue was generated for all the requests which were in open group and had not been executed yet in stage II. In stage III, grouped requests were executed. A threshold value of available resources was set and when the load needed by one or more requests in ready queue exceed the threshold limit; request should wait in the waiting queue.

Santhosh and Ravichandran [9] proposed a scheduling algorithm with pre-emptive execution to overcome the non-pre-emptive scheduling limitations. In non-pre-emptive scheduling, if any high priority task arrived and wait because of unavailability of the virtual machine then system performance degrades. In this work, When a high priority task arrived in between execution of other task and the deadline of the task which was about to miss then the task would be migrated to another virtual machine. This work was compared with traditional EDF and other non-pre-emptive scheduling algorithms. The results show that response time was reduced and overall system performance was improved.

K C Gouda et al. [10] proposed a priority based resource allocation approach which allocates the resource with minimum wastage and provides maximum profit in a dynamic cloud environment. This algorithm used different parameters like cost, time, no.of processors request etc. This priority algorithm decides the allocation sequence for different task requested by the different user after considering the priority based on some optimum threshold decided by the cloud service provider.

Walsh et al. [11], proposed a general two-layer architecture that uses utility functions, adopted in the context of dynamic and autonomous resource allocation, which consists of local agents and global arbiter. The responsibility of local agents is to calculate utilities for given current or forecasted workloads and range of resources, for each AE and results, are transfer to the global arbiter. Where global arbiter computer near-optimal configuration of resources based on the results provided by the local agents. In global arbiter, the new configurations applied by assigning new resources to the AEs and the new configuration computed either at the end of fixed control intervals or in an event triggered manner or anticipated SLA violation.

Wazir Y. O. et al. [12] proposed a new approach for dynamic autonomous resource management in the computing cloud. In this paper, the author's contribution is two-fold. First, distributed architecture is adopted where resource management is decomposed into independent tasks, each of which is performed by Autonomous Node Agents that are tightly coupled with the physical machines in a data center. Second, the Autonomous Node Agents carry out configurations in parallel through Multiple Criteria Decision Analysis using the PROMETHEE method. This approach is potentially more feasible in large data centers than centralized approaches.

Savani and Amar [13] proposed a priority based resource allocation algorithm which can be used for proper allocation of resources effectively. In this algorithm, the resources in the cloud are allocated according to the priority which is assigned to each user request.

Zhen Xiao [14] presents design and implementation of an automated resource management system that can avoid overload in the system while minimizing numbers of servers being used, it introduces the concept of skewness measure the convent utilization of the server and improves utilization of servers of multidimensional resource constraints.

Gaganjot and Sugandhi [15] proposed a preemptive priority based job scheduling algorithm in green cloud computing (PPJSGC). In this paper a green energy efficient scheduling algorithm which makes the use of preemptive priority job scheduling algorithm in cloud computing. This algorithm focuses on reducing the power cost. The computing server is selected on the basis of which satisfies the minimum resource requirement of a job as per the best fit. Resources are allocated based on the best allocation scheme. This method creates a balanced between energy consumption and a load of the server. This paper focuses on to design such an algorithm that minimizes the carbon footprints and maximizes the resources according to the suitability of the servers.

Dorian Minarolli and Bernd Freisleben [16] represents a VM resource allocation in cloud computing via multi agent fuzzy control, it focused on line grained dynamic resource allocation of VM locally on each physical machine of a cloud provider and consider memory and CPU as a resource that can be managed. Fuzzy control is used to minimize a global utility function as n a hill climbing heuristic implemented over fuzzy rules. The problem considers is how to resource of a cloud provider should be reallocated to VM dynamically where workload changes to keep the performance according to SLA's.

Kazuki and Shin-ichi [17] proposed an optimal resource allocation algorithm with limited energy power consumption. Three parameters were taken into consideration to allocate resources: processing element, network bandwidth, and electric power consumption. The maximum numbers of the request were processed by this approach. Total consumption of electric power was reduced by aggregating requests being processed in multiple areas.

Swachil and Upendra [18] proposed an improvement Priority based Job Scheduling Algorithm in Cloud Computing using Iterative Method. Improved priority based job scheduling algorithm uses an iterative method to find priority of jobs and resources and also finds priority of jobs to achieve better performance. The proposed scheduling algorithm consists of three levels of priorities: scheduling level (goal), resources level (attributes) and jobs level (alternatives). Scheduling level is the goal to be achieved by the scheduler, resources level are the attributes that are available to achieve the desired goal and the last level is the job level which are the available alternatives from which the best job should be scheduled first. This algorithm has better makespan and consistency than the other algorithm like priority based job scheduling algorithm and prioritized round robin algorithm.

Satveer and Birmohan [19] proposed an Optimum Resource Allocation Approach (ORAA) in cloud computing. In this paper, different types of resources (virtual machine) are allocated by taking three parameters into consideration: processing element, main memory, and network bandwidth. Users are allowed to submit the parameters during job submission. The user inserted parameters will then be considered while allocating resources to them. The objective of this paper is to make optimum resource allocation and achieve efficient utilization of resources over public cloud.

Jiayin Li [20] presents a resource optimization mechanism in heterogeneous IaaS federate multi cloud systems, that enable preemptable task scheduling with resource allotment model, cloud system model, local mapping, and energy consumption, and application model. It is suitable for autonomic future in cloud and VMs. The proposed online dynamic algorithms for resource allocation and task scheduling. In proposed cloud resource phenomenal every data center has a manager server, the communication and resource allotment scheme works between various servers of each data center for share workloads among multiple data servers. The workload sharing makes a large resource pool of flexible and cheaper resources to resource allocation.

It has been observed from the literature that the strategies used by the researchers provide a scope for improvement in resource allocation. To eradicate the limitations of work done by the researchers, a new priority based policy for resource allocation has been proposed in this paper.

III. PROPOSED WORK

This paper proposes a method to handle the cloudlet requests by managing resources using priority based approach for resource allocation. The proposed approach is used to attain better utilization of resources and reduced completion time of user requests by means of optimized allocation of resources at the virtual machine level. The Cloud parameters basically represent the different types of resource: four parameters have been chosen because of its dynamic nature. In the proposed work, users give the parameter values during request submission and these parameters are then be considered while allocating VMs. The demand for any of these resources may differ from user to user and hence are allocated dynamically. The information about VMs with different parameters is maintained by the data center broker in the form of resource matrices and the VMs are allocated to the user at run time with the help of these resource matrices. These matrices contain merely two values either ‘1’ or ‘0’. Here '1' indicates that the virtual machine of specific configuration is available and '0' indicates its unavailability.

At the starting of the approach, to acquire some service, user sends the request for resource allocation; the broker checks for the available VM to process that request. When the request with required parameter values arrives, it starts searching a VM that fulfils its requirement with the help of the matrices maintained by the broker. If a virtual instance of required type exists, the arrived user request will be allocated to that VM. After allocating all matching VMs to the user requests, the broker will check unallocated user requests and available VMs in the datacenter. Here the priority parameter submit by user is used. The values of priority parameter submitted by user are assumed. If user gives priority value '1' which represents the priority of PE type of resource required most by the user. In this condition, the broker search for the available VM which has PE value equal or more than required by the user and allocate to the user request. Accordingly, priority value '2' represents the RAM type of resource and '3' represents the BW type of resource. If the user gives priority value '0', then broker will allocate any available VM to the user request without checking any specification of the VM. So according to this allocation of resources, all of the user requests will get computing resource without waiting in waiting queue if VM with any configuration is available. To check the availability of matching VMs in the data center the essential condition is as follows:

$$Z1[k1_p][VM_i]=1 \qquad (1)$$

p ϵ U and i ϵ J

Where U is a set of PEs having "a" types and J is set of VMs. The availability of first parameter that is PE is measured with the help of a matrix of dimension (a × j). If the value of corresponding equation 1 is '1', that means requisite resource can be assigned to the request and if that value is '0', which means no resource can fulfil the requirement of the user request.

$$Z2[k2_q][VM_i]=1 \quad (2)$$

q ϵ V and i ϵ J

Where V is a set of RAMs having "b" types and J is set of VMs. Similarly, in order to check the availability of the second parameter that is different types of RAM, matrix of dimension (b × j) is used and

$$Z3[k3_r][VM_i]=1 \quad (3)$$

r ϵ W and i ϵ J

Where W is a set of BWs having "c" types and J is set of VMs. To check the availability of the third parameter that is the BW, a third matrix of dimension (c × j) is maintained, where PE, RAM, and BW are three parameters of a request that is send by the user to achieve a required type of service. In the above matrices Z1, Z2 and Z3, the type of PE is represented by k1, and it may vary from 1 to a, k2 represents the types of RAM and it may vary from 1 to b, and k3 represents the different kind of BW and it may vary from 1 to c.

When a user demand for a service with the requisite parameters, the availability of service is determined with the help of three resource matrices and each matrix will return a set of VM Ids namely T1, T2 and T3. The common VM Ids between T1, T2 and T3 are taken into a set T and these VM Ids can fulfil the requirement of request of a user. One of the VM Ids from set T is assigned to the request.

In the proposed approach, if all of three equations must be satisfied at the same time, then a VM Id returned. If the service is unattainable, that means VM with the required configuration does not exist. Once a VM is allocated to a request, the value of that VM in all three matrices is reset. For the resource allocation based on the priority, at a time only one condition must be satisfied like when priority is for PE then, Equation 1 only checked. Accordingly, Equation 2 checked for priority of RAM type of resource and Equation 3 will be checked for the priority of BW type of resource.

In the proposed approach, the space shared policy of CloudSim-3.0.3 simulator has been chosen over time shared policy to achieve the concept of load balancing of individual VM in the data center. The workload on a single virtual machine is balanced efficiently with the help of fair allocation of resources and space-shared policy of VM. The proposed approach which inculcates the above benefits is described in following algorithm.

Algorithm1: Optimum Resource Allocation (VMs, CRs)

Begin

1. Initialize CloudletRequest-Completed
2. Formulate matrices Z1, Z2, Z3 for parameters PE, RAM, BW.
3. Update VM-List (based on VM parameters)
4. Update CR-List (based on CR parameters)
5. Repeat for i = 1 to length (CR-List)
    a. Call SMA (VM-List, CR-List)
    b. Call Update (VM-List, Request-Completed, CR-List)
6. If (Request-Completed != CR-List && VM-List != Null)
   Repeat for i = 1 to length (CR-List != Request-Completed)
    a. Call PBRAA (VM-List, CR-List)
    b. Call Update (VM-List, Request-Completed, CR-List)

End

Algorithm2: SMA (VM-List, CR-List)

Begin

1. For Id = 1 to length (VM-List)
   if (Z1 [Cloudlet-List.PE] [Id] and
   Z2 [Cloudlet-List.RAM] [Id] and
   Z3 [Cloudlet-List.BW] [Id] = available)
2. Allocate VM to CR (from VM-List)
3. Call Update (VM-List, CloudletRequest-Completed, CR-List)

End

Algorithm3: PBRAA (VM-List, CR-List)

Begin

1. For Id = 1 to length (VM-List)
   If (CR.P = 1 && Z1 [CR-List.PE] [Id] = Available)
   Allocate VM to CR (from VM-List)
   Else if (CR.P = 2 && Z2 [CR-List.RAM] [Id] = Available)
   Allocate VM to CR (from VM-List)
   Else if (CR.P = 3 && Z3 [CR-List.BW] [Id] = Available)
   Allocate VM to CR (from VM-List)
   Else if (CR.P = 0 && Z1 [CR-List.PE] [Id] = Available || Z2 [CR-
   List.RAM] [Id] = Available || Z3 [CR-List.BW] [Id] = Available)
   Allocate VM to CR (from VM-List)
2. Call Update (VM-List, CloudletRequest-Completed, CR-List)

End

Algorithm4: Update (VM-List, CloudletRequest-Completed, CR-List)

Begin

1. If (CloudletRequest-Completed = True)
   Set (Z1 [CR.PE] [Id] and Z2
   [CR-List.RAM] [Id] and Z3
   [CR-List.BW] [Id] = 1)
   Else
   (Z1 [CR-List.PE] [Id] and

Z2 [CR-List.RAM] [Id] and
Z3 [CR-List.BW] [Id] = 0)

2. Update VM-List
3. Update CR-List

End

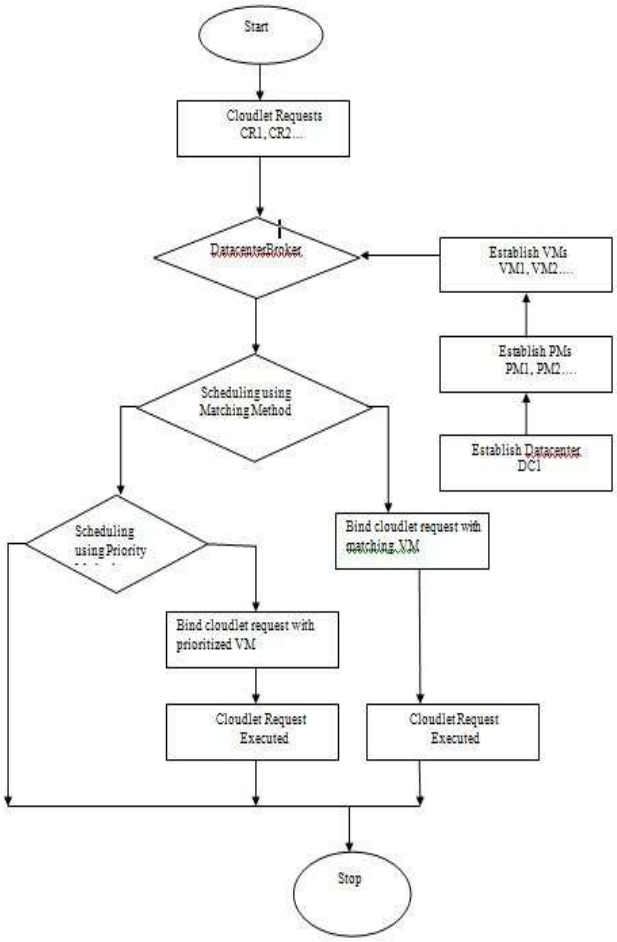

Figure 1. Dataflow Diagram of Proposed Method

The proposed algorithm in this chapter takes O (m^2* n^2) time and O (n^2) space to process all the Cloudlets in the virtual Cloud in worst case where n is the number of virtual machines and m is the number of Cloudlets.

IV. RESULT AND DISCUSSION

For the purpose of simulation, the CloudSim-3.0.3 simulation toolkit has been used in the proposed work as the primary objective of this toolkit is to provide a generalized and extensible simulation framework that enables seamless modeling, simulation, and experimentation of emerging Cloud computing infrastructures and application [26].

The Cloud service system with proposed policy is shown in figure 4.1. Only one data center is considered in the public Cloud. There are different types of user requests and VMs in this CloudSim-3.0.3 environment. A new broker policy is implemented for resource allocation in which space-shared policy is used by Cloudlets on VMs. The value of parameters PE, RAM, and BW that a user can submit in its request are listed in table 4.1 and the values of Priority parameter is listed in table 4.2

TABLE 4.1 The values of Three Parameters

S.N.	Type	PE (No. of Cores)	RAM (in MB)	BW (in Mbps)
1	1	1 Core	1024	1000
2	2	2 Core	2048	2000
3	3	4 Core	4096	4000
4	4	8 Core	8192	8000

TABLE 4.2 The Values of Priority Parameter

S. No.	Value of Priority (P) Parameter	Demand for (Resource Type)
1	0	Any available VM
2	1	PE type of resource
3	2	RAM type of resource
4	3	BW type of resource

In this simulation environment, three different VM-sets in which VM-set 1 consist of ten VMs and VM-set 2 consist of eleven VMs and VM-set 3 consist of twelve VMs. Three different data-sets in which each data-set consist of ten or twelve user requests are created. All three data-sets are executed on VM-sets separately which confirms the simulation results. The specifications of the VMs in VM-sets and specifications of cloudlet requests in data-sets are listed in table 4.3 and table 4.4 respectively

The proposed approach for allocation of resources has been verified with the help of three datasets. The Cloudlets requested by the users of dissimilar types due to variation in input parameters and these are allocated to the three VM-sets. Table 4.5, Table 4.6 and table 4.7 show the number of requests that has been accepted using matching method and using priority separately, to provide the services, and this is because a machine with required configuration has been found.

As a result of this exact matching and by using priority given by user, a mapping has been performed between the requests and the virtual machines. In other words, user requests have

been assigned to the virtual machine so as to get its required service. Sometimes, if there is no available virtual machine and in case a virtual machine is available, which does not fulfils the need of user request, then only a request will be unallocated. Using this information about user requests, the average utilization of virtual machines has been calculated.

TABLE 4.3 The Specifications of Virtual Machines

S. N.	VMs	VM Set 1			VM Set 2			VM Set 3		
		PE Type	RAM Type	BW Type	PE Type	RAM Type	BW Type	PE Type	RAM Type	BW Type
1	VM 0	Type 1	Type 1	Type 1	Type 1	Type 2	Type 1	Type 1	Type 2	Type 1
2	VM 1	Type 2	Type 2	Type 2	Type 1	Type 2	Type 2	Type 1	Type 2	Type 2
3	VM 2	Type 2	Type 1	Type 2	Type 2	Type 1	Type 2	Type 2	Type 1	Type 2
4	VM 3	Type 4	Type 4	Type 8	Type 4	Type 2	Type 2	Type 2	Type 2	Type 2
5	VM 4	Type 2	Type 4	Type 4	Type 2	Type 4	Type 2	Type 2	Type 4	Type 2
6	VM 5	Type 4	Type 8	Type 2	Type 2	Type 8	Type 4	Type 2	Type 4	Type 4
7	VM 6	Type 4	Type 4	Type 4	Type 8	Type 4	Type 4	Type 4	Type 4	Type 4
8	VM 7	Type 8	Type 4	Type 8	Type 4	Type 8	Type 2	Type 4	Type 8	Type 4
9	VM 8	Type 8	Type 8	Type 8	Type 8	Type 8	Type 8	Type 8	Type 8	Type 8
10	VM 9	Type 8	Type 4	Type 8	Type 4	Type 8	Type 2	Type 8	Type 8	Type 2
11	VM10	-	-	-	Type 4	Type 1	Type 8	Type 4	Type 1	Type 8
12	VM 11	-	-	-	-	-	-	Type 8	Type 4	Type 8

TABLE 4.4 The Specification of Cloudlet Request

CR Id	Data Set 1				Data Set 2				Data Set 3			
	PE Type	RAM Type	BW Type	P	PE Type	RAM Type	BW Type	P	PE Type	RAM Type	BW Type	P
0	1	2	1	0	1	2	1	0	1	2	1	1
1	1	2	2	0	1	2	3	2	1	2	1	0
2	2	1	2	0	2	2	2	1	1	2	2	3
3	2	3	2	0	2	2	3	3	2	1	2	0
4	2	3	3	0	2	3	3	2	2	3	2	2
5	3	4	2	0	3	4	2	1	2	3	3	0
6	3	3	3	0	3	3	3	0	2	3	3	1
7	3	2	3	0	3	2	4	1	3	3	3	0
8	4	3	4	0	4	3	4	0	3	4	2	3
9	4	4	3	0	4	3	3	2	4	4	4	0
10	4	4	4	0	4	4	1	3	4	4	2	2
11	4	4	2	0	4	4	3	1	4	1	2	2

TABLE 4.5 The Result on VM Set 1

Data Set	No. of CR	No. of VMs	Allocation of User Requests			Average Utilization (P_u) (In %)
			Using Matching Method	Using Priority	Unallocated Requests	
01	12	10	10	00	02	100.00
02	12	10	05	05	02	90.00
03	12	10	08	02	02	95.00

TABLE 4.6 The Results on VM Set 2

Data Set	No. of CR	No. of VMs	Allocation of User Requests			Average Utilization (P_u) (In %)
			Using Matching Method	Using Priority	Unallocated Requests	
01	12	11	09	01	01	90.00
02	12	11	05	06	00	100.00
03	12	11	08	03	00	95.00

TABLE 4.7 The Results on VM Set 3

Data Set	No. of CR	No. of VMs	Allocation of User Requests			Average Utilization (P_u) (In %)
			Using Matching Method	Using Priority	Unallocated Requests	
01	12	12	09	03	00	100.00
02	12	12	06	05	01	85.34
03	12	12	07	05	00	90.00

It has been observed, the proposed approach has achieved an average utilization of 100% for dataset 1 on VM Set 1, VM Set 2 and on VM Set 3. Dataset 2 and Dataset 3 also achieved nearly 90% to 95 % average utilizations on VM sets. In the table 4.6 and 4.7, there is case in which one cloudlet request is unallocated when a VM is available in VM set. It is because; the resource priority given by user can't fulfill using available VM.

V. CONCLUSION

The optimum resource allocation using configuration matching and priority approach is used for the purpose to improve the resource utilization and to reduce the completion time of user requests. Basically, this approach is proposed to eradicate the limitations of scheduling of resources of optimum allocation. At the time when priority given to the resources, there has a situation where there are available computing resources with miss-matched configuration to serve the user request, but the resource is not allocated to the request. In this situation, the request loss is more and utilization of resources is somehow not maximized as well. So, to eliminate this limitation of that situation this method of optimum allocation of resource by using SMA and priority is proposed.

VI. REFERENCES

[1] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities", 10th IEEE International Conference on High Performance Computing and Communications (HPCC 2008), Sept. 25-27, 2008.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of Cloud Computing", Communications of the ACM, Apr. 2010.

[3] Gouda, K. C., T. V. Radhika, and M. Akshatha, "Priority based resource allocation model for cloud computing", International Journal of Science, Engineering and Technology Research , pp-215, 2013.

[4] Ren, Haozheng, Yihua Lan, and Chao Yin, "The load balancing algorithm in cloud computing environment", Computer Science and Network Technology (ICCSNT), 2012.

[5] Dixit, Rituraj, Prashant Buyan, and Vijay Kumar. "Priority Based Dynamic Resource Allocation for Green Cloud Environment: A Survey" International Journal of Engineering and Management Research, 2014.

[6] Gupta, Garima, Vimal Kr Kumawat, P. R. Laxmi, Dharmendra Singh, Vinesh Jain, and Ravinder Singh, "A simulation of priority based earliest deadline first scheduling for cloud computing system" In Networks & Soft Computing (ICNSC) International Conference on, pp. 35-39, 2014.

[7] Pawar, Chandrashekhar S., and Rajnikant B. Wagh, "Priority based dynamic resource allocation in cloud computing" In Cloud and Services Computing (ISCOS), International Symposium, pp. 1-6, 2012.

[8] N. Gill, "Enhanced priority based resource allocation in Cloud computing," In The Next Generation Information Technology Summit (4th International Conference) , pp. 121-126, September 2013.

[9] Santhosh, R., and T. Ravichandran, "Pre-emptive scheduling of on-line real time services with task migration for cloud computing" In Pattern Recognition, Informatics and Mobile Engineering (PRIME), pp. 271-276, 2013.

[10] Gouda, K. C., T. V. Radhika, and M. Akshatha, "Priority based resource allocation model for cloud computing" International Journal of Science, Engineering and Technology Research, pp-215, 2013.

[11] Walsh, William E., Gerald Tesauro, Jeffrey O. Kephart, and Rajarshi Das, "Utility functions in autonomic systems" In Autonomic Computing International Conference, pp. 70-77, 2004.

[12] Yazir, Yagiz Onat, Chris Matthews, Roozbeh Farahbod, Stephen Neville, Adel Guitouni, Sudhakar Ganti, and Yvonne Coady, "Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis" 3rd International Conference IEEE, pp. 91-98, 2010.

[13] Savani Nirav M, Amar Buchade, "Priority Based Resource Allocation in Cloud Computing" International Journal of Engineering Research & Technology (IJERT), 2014.

[14] Xiao, Zhen, Weijia Song, and Qi Chen, "Dynamic resource allocation using virtual machines for cloud computing environment" IEEE transactions on parallel and distributed systems, 2013.

[15] Kaur, Gaganjot, and Sugandhi Midha, "A Preemptive Priority Based Job Scheduling Algorithm in Green Cloud Computing" 6th International Conference IEEE, 2016, pp. 152-156.

[16] Minarolli, Dorian, and Bernd Freisleben, "Utility-based resource allocation for virtual machines in cloud computing" In Computers and Communications (ISCC), IEEE Symposium, pp. 410-417, 2011.

[17] K. Mochizuki and S. I. Kuribayashi, "Evaluation of optimal resource allocation method for Cloud computing environments with limited electric power capacity," In Network-Based Information Systems (NBiS), 14th International Conference on IEEE, pp. 1-5, September 2011.

[18] Patel, Swachil, and Upendra Bhoi, "Priority based job scheduling techniques in cloud computing: a systematic review" International journal of scientific & technology research, 2013.

[19] S Satveer, and Birmohan Singh. "Optimum resource allocation approach in cloud." In Advanced Communication Control and Computing Technologies (ICACCCT), International Conference on IEEE, pp. 600-605, 2016.

[20] Li, Jiayin, Meikang Qiu, Zhong Ming, Gang Quan, Xiao Qin, and Zonghua Gu, "Online optimization for scheduling preemptable tasks on IaaS cloud systems" Journal of Parallel and Distributed Computing, 2012.

.