Resource Allocation Model for Efficient Management in Cloud Computing

Radhika T V[1] & K C Gouda[2]

[1]Dayananda Sagar College of Engineering
[2]Scientist, NAL-CMMACS Bangalore
E-mail : [1] radhikacord@gmail.com, [2] kcgouda@gmail.com ,

Abstract – **Cloud computing is a computing technology where a large pool of computers are connected in private or public network which provides dynamically scalable infrastructure for application hosting, storage and delivery and it must be supported with minimal management effort or service provider interaction. A cloud environment consists of multiple customers requesting for resources in a dynamic environment with possible constraints. In the existing economy based models of cloud computing, allocating the resource efficiently is a challenging job. In this paper we propose a new approach that allocates resource with minimum wastage and provides maximum profit. In order to minimize the cost of resources, it is also important to satisfy a minimum service level to customers. Therefore, this paper proposes profit model algorithms for SaaS providers who want to minimize infrastructure cost and SLA violations. The developed resource allocation algorithm is based on different parameters like time, cost, No of processor request etc. The developed priority algorithm is used for a better resource allocation of jobs in the cloud environment and for the simulation of different models or jobs in an efficient way. After the efficient resource allocation of various jobs, a profit model has been developed which calculates maximum profit gained by cloud administrator by serving to each user request. A performance study of all the algorithms in various systems and case studies are also presented.**

Index Terms-cloud computing, Resource allocation, profit model, priority algorithm, SaaS provider, SLA violation.

I. INTRODUCTION

Cloud computing is a new computing paradigm in which an IT user does not have to physically own any computing Infrastructure other than, perhaps, workstations. Contrastingly, the user "rents or leases" computational resources (time, bandwidth, storage, etc.) from some external entity, e.g., "pay-as-you-go" rather than. Originally, cloud computing was proposed as a solution to deliver large-scale computing resources to the scientific community for individual users who could not afford to make the huge investments in permanent infrastructure or specialized tools, or could not lease needed infrastructure and computing services. It evolved, rapidly, into a medium of storage and computation for Internet users that offers economies of scale in several areas. Rather than providing the user with a permanent server to connect to when application execution is required, cloud computing provides "virtualized servers" chosen from a pool of servers at one of the available data centers. A user's request for execution of a web application is directed to one of the available servers that has that application locally installed. Within a data center, almost any application can be run on any server. The user neither knows the physical server nor, in many cases, where it is located, i.e., it is locationally irrelevant. Cloud computing is a viable business computational model for small to medium businesses who cannot afford large investments in permanent infrastructure. The scalability, flexibility, pay-as-you-go and minimal upfront investment make Cloud Computing Environments an attractive option for computing services. But, for cloud computing to succeed, they must be able to run their own tailored or uniquely configured applications in the cloud, a capability available in the scientific community, but now becoming available in the personal and corporate usage community.

The cloud computing concept arises from the notion of "software as a service" (SaaS). A set of services are provided on a set of platforms at various locations. The user determines the service he requires and shops for the best value for that service based on specified defined criteria. The first model is Software as a Service(SaaS) in which the application runs entirely in the cloud, where “software that’s owned, delivered, and managed remotely by one or more providers”. The second model is attached services in which a local application acquires common services that allow it to interoperate with other applications or users at other sites. The third model treats cloud platforms as application execution engines, e.g., Hardware as a Service (HaaS), which communicate with applications.

An increasing number of organizations (e.g., research centers, enterprises etc.) benefit from Cloud computing to host their applications. In contrast to previous paradigms (Clusters and Grid computing), Cloud computing is not application-oriented but service-oriented it offers on-demand virtualized resources as measurable and billable utilities. 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 consumers demand, based on his current computing requirements and therefore eliminates over provisioning of available IT resources.

Primary advantage with the cloud computing is that the business enterprises can scale up to requisite capacities instantaneously without having to invest in new IT infrastructure that includes computers, network or database administrators and new licensed software.

The business enterprises can save 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 maintenance of servers and networks in the cloud. They just pay for services on demand that is based on running of application instances normally varying depending upon use of Internet bandwidth, number of instances in action and amount of data transferred at specific time.

In this paper, we present the methods for efficient resource allocation 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 assignment of available resources such as CPU, memory, storage, network bandwidth etc in an economic way. In this paper we have proposed priority algorithm that mainly decides priority among different user request based on many parameters like cost of resource, time needed to access, task type, number of processors needed to run the job or task etc.

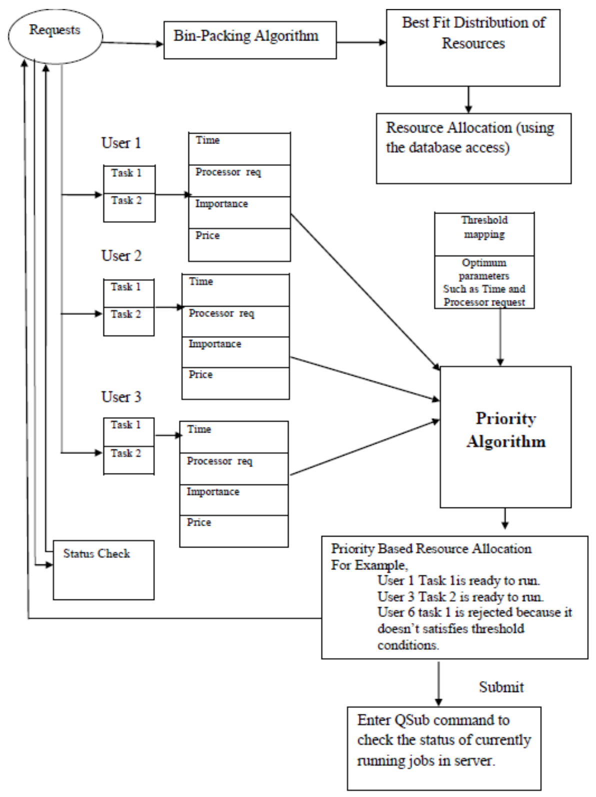
Before providing service to a customer, there should be a Service Level Agreement (SLA) between SaaS provider and customer. Suppose SaaS provider makes SLA violation or if he doesn’t provide service in time then he has to pay penality inturn. Finally Profit model algorithm is discussed that is basically used to calculate total profit gained by cloud owner by serving to all customer’s request. It considers some parameters such as contract Length (ConLen), virtual machine cost (VMcost), price of each service (PriServ), service intiation time(iniTimeSev), penalty cost.

II. RESOURCE ALLOCATION MODEL

The Resource Allocation Model presented in this paper is basically an algorithm for efficient resource allocation in a cloud computing Environment. The Proposed model has been developed by considering various parameters such as cost, profit, user, time, Number of processor request, resource assigned, resource availability, resource selection criteria etc. In Resource allocation Model, clients are customers or users of cloud which sends service request that is client sends job request that is to be executed or run in cloud server. Server in cloud computing environment is the cloud service provider which will run the task or job submitted by client. The cloud administrator plays key role in efficient resource allocation because he decides the priority among the different user request. This priority based resource allocation considers the parameters discussed above.

Virtualization is another important topic in cloud computing. It is a computing technology that enables a single user to access multiple physical devices. Another way to look at it is a single computer controlling multiple machines, or one operating system utilizing multiple computers to analyze a database. With cloud computing, the software programs that are used aren’t run from your personal computer, but rather are Stored on servers housed elsewhere and accessed via the Internet. The resource allocation model that decides priority among different user request is shown in figure.

In dynamic cloud environment different users are submitting their request. Each request consists of different task. For each task different parameters are considered such as time, Processor request, Importance and price. Time refers to computation time needed to complete the particular task, Processor request refers to number of processors needed to run the task. More the number of processor, faster will be the completion of task.


Requests
Bin-Packing Algorithm
Best Fit Distribution of Resources
Resource Allocation (using the database access)
User 1
Task 1
Task 2
Time
Processor req
Importance
Price
User 2
Task 1
Task 2
Time
Processor req
Importance
Price
User 3
Task 1
Task 2
Time
Processor req
Importance
Price
Threshold mapping
Optimum parameters Such as Time and Processor request
Priority Algorithm
Status Check
Priority Based Resource Allocation
For Example,
User 1 Task 1is ready to run.
User 3 Task 2 is ready to run.
User 6 task 1 is rejected because it doesn't satisfies threshold conditions.
Submit
Enter QSub command to check the status of currently running jobs in server.

Importance refers to how important the user to a cloud administrator(admin) that is whether the user is old customer to cloud or new customer. Finally price parameter refers to cost charged by cloud admin to cloud users.

Earlier Bin-Packing algorithm were used for best fit distribution of resources in cloud environment. Bin Packing is a mathematical way to deal with efficiently fitting resources into Bins. A formal definition of the Bin Packing (BP) problem can be defined as given a list of objects and their weights, and a collection of bins of fixed size, find the smallest number of bins so that all of the objects are assigned to a bin. Now, a Bin is something that can hold inside itself a certain amount (it's Bin Height). Every Resource is of a certain, nonzero, and positive value (Resource Height).

Based on all the parameters considered above and also based on some threshold parameters, priority algorithm decides priority among different task submitted by different users. The user's task with higher priority will be given first chance to run. The user's task with next higher priority will be given second chance and so on. The task which exceed threshold will be aborted. Cloud admin can also check the status in order to know which the running tasks are and which are in queue. In this way by using priority algorithm, cloud administrator can efficiently allocate the resources among the users with minimum wastage and provides maximum profit.

III. PRIORITY ALGORITHM

In a cloud computing environment, multiple customers are submitting job request with possible constraints that is multiple users are requesting same resource. For example in a high performance computational environment which mainly deal with scientific simulations such as weather prediction, rainfall simulation, monsoon prediction and cyclone simulation etc which requires huge amount of computing resources such as processors, servers, storage etc. Many users are requesting these computational resources to run their model which is used for scientific predictions. So at this situation it will be problem for cloud administrator to decide how to allocate the available resources among the requested users.

The proposed priority algorithm helps cloud admin to decide priority among the users and allocate resources efficiently according to priority. This resource allocation technique is more efficient than grid and utility computing because in those systems there is no priority among the user request and cloud administrator is randomly taking decision and he is giving priority to those user who have submitted their job first that is based on first come first serve method. But with the advent of cloud computing and by using this implemented priority algorithm, the cloud admin can easily take decision based on different parameters discussed earlier to decide priority among different user request so that admin can efficiently allocate the available resources and with cost-effectiveness as well as satisfaction from users. The table 1 shows the parameters considered for job/task submission cloud computing environment.

In order to run a particular model huge computational resources such as server, memory in terms of storage disk, processors, software etc are needed. Also some jobs are to be executed in parallel and some others in sequential manner. In that situation job type is also very important parameter. In a cloud environment type of user that is whether the user is internal to a cloud (in case of private cloud) or he is external to cloud(in case of public cloud) is also another important parameter to be considered during job submission. So the developed priority algorithm discusses in detail how efficiently it will help cloud admin to decide or calculate priority among the user requests.

No. of Users	eg: 10 users
Servers	eg: S1, S2,S3
Time to run	eg: 4 Hours
No Processors requested	eg: 8 Processors
Amount of memory requested	eg: 5 GB, 1 TB etc.
Time of request	eg:1:30 am
Software to be used	eg: Matlab, Grads,NetCDF
Job type	eg:Sequential or parallel.
User type	eg: Internal or External

Table 1: Parameters considered for job submission

In order to run a particular model huge computational resources such as server, memory in terms of storage disk, processors, software etc are needed. Also some jobs are to be executed in parallel and some others in sequential manner. In that situation job type is also very important parameter. In a cloud environment type of user that is whether the user is internal to a cloud (in case of private cloud) or he is external to cloud(in case of public cloud) is also another important parameter to be considered during job submission. So the developed priority algorithm discusses in detail how efficiently it will help cloud admin to decide or calculate priority among the user requests.

The main difficulties in the resource allocation in a cloud system are to take proper decision for job scheduling, execution of job, managing the status of job etc. Apart from traditional best fit and bin packing algorithm in this paper an algorithm is developed for the job allocation in the cloud environment to be decided by the cloud administrator. Several parameters listed in the table 1 are considered for the priority based on the client and server requirements and requests by the users.

In the present algorithm to decide the resource allocation in a better and impartial way, a technique based on threshold of all the parameters (both client and server side) is considered. For example the requested number of processors cannot be more than 20 etc. (server) and a job maximum run time will be 200 hrs (user).The step wise explanation of the above said algorithm is discussed below.

```
Algorithm: To compute and assign the priority for each request based on the threshold value and allocate the service to each request‘s.
Step 1: [Read the clients request data i.e, time, importance, price, node and requested server name]
        Insert  all values into the linked list
Step 2: [For each request and its tasks find the time priority value based on the predefined conditions]
        Assign priority value to each task for the client‘s request.
        t_p[i] = priority valu
Step 3: [For each request and its tasks find the node priority value based on the predefined conditions]
        Assign priority value to each task for the client‘s request.
        n_p[i] = priority value;
Step 4: [For each client‘s input data check whether it is within the threshold value or not]
        if ( input value is within the threshold limit and total node <= available node)
        [Add respective computed time and node priority value and other parameters like
        importance and price]
Sum[k] = t_p[i] + n_p[i] + importance + price
        Print —Ready to execute
available node = available node – total node
        else if (input value is within the threshold limit)
sum[k] = t_p[i] + n_p[i] + importance + price
        print —within the limit but it is in queue
        else
        print —Exeed the condition
Step 5: [Sort the sum[k] values]
Step 6: Client‘s request is ready to execute from least values of sum[k]
Stop
```

Figure 2: priority Algorithm

IV. SYSTEM MODEL FOR PROFIT ALGORITHM

Customer can send request for software service to SaaS provider after agreeying to predefined SLA clauses. Customers can dynamically change their requirements and usage of the hosted software services. The SaaS provider can use their own infrastructure or outsourced resources from public IaaS providers. In this section, we explain the detailed system model from both the customers' and the SaaS providers' perspective and also describe the progit model algorithm to maximize profit in cloud computing environment.

A. Actors

The actors involved in our system model are described

below along with their objectives, activities and constraints.

1) SaaS Providers

SaaS providers lease enterprise software as hosted services to customers. They are interested in maximizing profit and ensuring QoS for customers to enhance their reputation in the marketplace. In our context, an example of the business process between a SaaS provider and a customer is where a service provider (SaaS X) offers CRM or ERP software packages, which are offered as three types of products (for example, Standards, Professional and Enterprise) and accounts (for example, Group, Team and Department). When a customer (Company X) submits its first time rent request with product type (Standards), account type (Group), and the required number of accounts (m), the provider will allocate resources to serve this customer.

At anytime, Company X may require an upgrade in the service by adding more accounts or software editions. Customers can request an upgrade of services dynamically at any time in practice. Thus a SaaS provider has to handle these requests intelligently in line with the requirements as set out in the SLA. From a SaaS provider's point of view, there is a legal contract-SLA with any customer and if any party violates SLA terms, the defaulter has to pay for the penalty according to the clauses defined in the SLA. The SLA properties include SaaS provider pre-defined parameters and the customer specified QoS parameters.

The properties defined in the SLA are as follows:

- *Request Type (reqType):*

  It defines the customer's request type, which is 'fist time rent' or 'upgrade service'. 'First time rent' means the customer is the customer who is renting a new service from this SaaS provider. 'Upgrade service' includes two types of upgrade, which are 'add account' and 'upgrade product'.

- *Product Type (proType):*

  The software product offered to customers. For example, SaaS X offers Standard, Professional, and Enterprise product. The Standard product includes Order and Sales functions. The Professional contains all functions of Standard plus Accounting function. The Enterprise includes all features of Professional plus Report functions.

- **Account Type (accType)**:

  It constrains the maximum number of accounts a customer can create. For example, SaaS X offers three types of account: Group, Team, and Department, which allows each customer to create up to m, 2m and 5m number of accounts respectively.

- **Contract Length (conLen):**

  How long the software service is legally available for a customer to use(minimum is one month).

- **Number of Accounts (accNum):**

  The actual number of accounts that a customer wants to create. (Must be ≤ the maximum number of accounts for particular account type). For example, a customer Company X wants to rent Standard software and Group account type, then it can request [1, m] number of accounts.

- **Number of Records (recNum):**

  The maximum number of records a customer is able to create for each account during the transaction and this will impact the data transfer time during the service upgrade. (The value of this parameter is predefined in SLA).

- **Response Time (respTime):**

  It represents the elapsed time between the end of a demand on a software service and the beginning of a service. Violation occurs when actual elapsed time takes longer than pre-defined response time in SLA. We consider three types of response time: (i) response time for the first time renting of the service - respTime(ftr), (ii) response time for adding new.

2) Customers

When a customer agrees with pre-defined SLA properties (such as response time), a request for an enterprise application is sent to the SaaS provider's application layer with the customer's QoS requirements (including request type, product type, account type, contract length, and number of accounts).

Finally in cloud computing environment Profit calculation is important from service provider point of view. Profit Model Algorithm calculates profit gained by SaaS providers by serving to different customers request. Let *C* be the number of customer requests and *c* indicates customer request id. Let at a given time instance *t*, a customer *c* submits a service request to the SaaS Provider. The customer specifies a product type, account type, contract length (*conLen*), number of accounts after agreeing with the predefined SLA clauses (response time). After the agreement, based on the SLA, the SaaS provider will reserve the requested software services which are translated at the infrastructure level as VM capacity. Let *I* be the number of initiated VMs, and *i* indicates the VM id. Profit Model algorithm is explained below in step wise.

Step 1: Customer C with request id 'c' submits a job request to SaaS service provider. Customer specifies all the specifications or requirements including product type,account type, contract Length (ConLen) and the number of accounts.

Step 2: SaaS provider will serve all the requested services to all customers request and it will calculate Total profit gained by it using following equation.

$$\sum_{c=1}^{C} \mathrm{prof}_{il}^{c} = \sum_{c=1}^{C} \mathrm{priServ}^{c} * \sum_{c=1}^{C} \mathrm{ConLen} - \sum_{c=1}^{C} \mathrm{Cost}_{il}^{c}$$

Where, $\forall\, i \in I, l \in L, i \in I$

Where,

- $\sum_{c=1}^{C} \mathrm{prof}_{il}^{c}$ is total profit gained by SaaS provider by serving to C number of customers request. $\sum_{c=1}^{C} \mathrm{priServ}^{c}$it is the final price charged per month by SaaS provider to each customer.
- $\sum_{c=1}^{C} \mathrm{ConLen}$ is the amount of time SaaS provider is serving to each customer request C.
- $\sum_{c=1}^{C} \mathrm{cost}_{il}^{c}$ is the cost invested by SaaS provider for serving to C number of customers request.
- 'i' is virtual machine(VM) id and I is number of intiated VM's.
- 'L' is virtual machine(VM) type
- 'c' indicate customer request id.
- 'C' indicated number of customer request.

Step 3: Cost invested by SaaS provider depends on virtual Machine cost (VM cost) and also Penalty cost which is given by equation (2)

$\sum_{c=1}^{C} \mathrm{Cost}_{il}^{c} = \mathrm{VMcost}_{il}^{c} + \mathrm{penaltyCost}_{il}^{c}$-------------(2)

Where, $\forall\, i \in I, l \in L, i \in I$

Where,VMcost_{il}^{c} is the cost of each Virtual Machine(VM).

$\mathrm{Penaltycost}_{il}^{c}$ is the cost to be paid by SaaS provider for not serving customer request within requested amount of time.

Step 4: The VM cost depends on the VM type *l*, the price of VM *i* with type *l* (priVM), the service initiation Time ($\mathrm{iniTimeSev}_{il}^{c}$) and service length or contract length (ConLen^{c}) of customer request *c*. VM cost is given by equation(3)

$$\mathrm{VMcost}_{il}^{c} = \mathrm{priVM}_{il}^{c} * (\mathrm{iniTimeSev}_{il}^{c} * \mathrm{ConLen}^{c}) \quad \text{---- (3)}$$

Where, $\forall\, i \in I, l \in L, i \in I$

Where

- priVM_{il}^{c} is the price of each virtual machine with id i and type l
- $\mathrm{iniTimeSev}_{il}^{c}$ is the time needed to initiate service in each virtual machine(VM).
- ConLen^{c} is the contract length or service length of each customer.

Step 5: The penalty cost intern depends on the penalty delay time $\mathrm{DelayTime}_{il}^{c}$ penalty rate (*β*) and a constant number (*α*). The delay time and rate are subjected to the request type.

This is given by following equation.

$\mathrm{penaltyCost}_{il}^{c} =$
$\alpha + \beta(\mathrm{reqType}) * \mathrm{DelayTime}_{il}^{c}(\mathrm{reqType})$------------(4)

Where, α is constant number β is penalty rate

$\mathrm{DelayTime}_{il}^{c}(\mathrm{reqType})$ is the total time delayed by SaaS provider for serving to customers with particular request type.

This algorithm is tested by considering a the total cost and machines initiated as a function of the variation in request arrival rate and it is found that based on the different requests the profit varies and optimised profit can be derived by multiple usage by users for long term SLA. This entire algorithm is tested in high performance computing platform.

V. PERFORMANCE EVALUATION

This section describes the scenarios considered for performance evaluation, performance metrics and experimental results. All experiments are performed on the various simulation carried out at high performance computing platform by different users. Mainly four systems or models have used for the various applications like monsoon simulations, cyclone simulation, atmospheric cloud simulation etc. These models are submittend or run in high computational servers. To evaluate the computational time variability for an model (Meso scale model, MM5) using different virtual cluster instances, we show the time spent in computation for seven different runs measured using the integrated performance monitoring.

It can be seen that there is 30% variability seen in compute time which can be explained by the processor distribution required for each run as shown in Figure 3. In the first run the job is very slow because it has used only two processors on the other hand the seventh run spent least amount of time in the computation because there 18 processors were used. The most important point to be noted is that the communication pattern also performs based on the overall run time. The difference between maximum and minimum is 120 seconds.

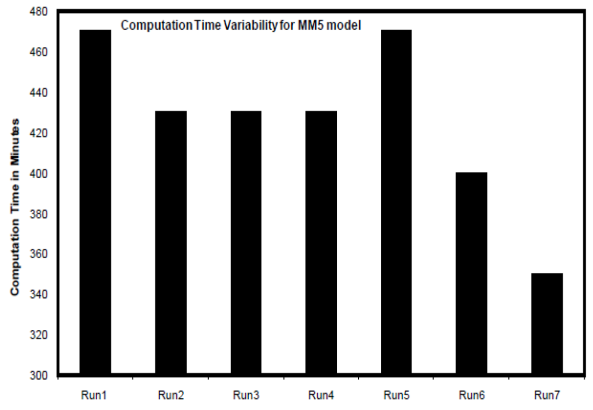


Figure 3 Computation Time Variability for MM5 model

The sustained performance per core for another application that is monsoon rainfall prediction using a General Circulation Model (GCM) is tested in different machines or servers (say A5, A4, A3, and A0).It is found that the performance is much better for the machine A5 in terms of sustained system performance as shown in Figure 4. So the main recommendation will be one should prefer to run in GCM in A5 rather than other machines.

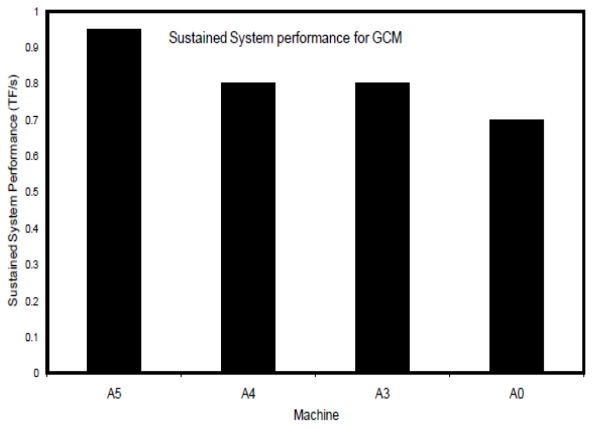


Figure 4. Sustained system performance for GCM

The same sustained performance is also carried out for different applications namely process, MM5, Weather Research Forecast (WRF) model, Non Hydrostatic Model (NHM) and GCM in four separate servers A5, A4, A3, A0. It is found that the performance is different for each application in different machines. The performance of process model, WRF and MM5 machine A0 is high where as for NHM , A5 is performing well and for GCM both A5 and A0 are comparable as shown in the Figure 5.

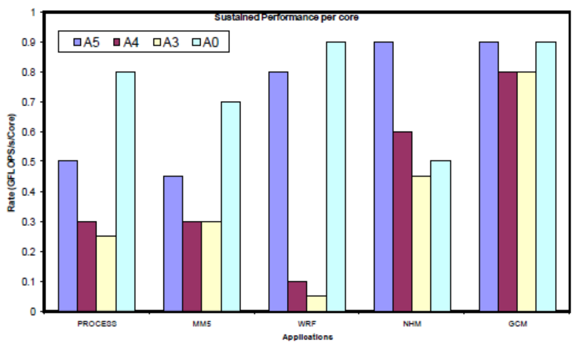


Figure 5. Sustained Performance per Core

VI. RELATED WORK

Cloud computing has been considered as a solution for solving enterprise application distribution and configuration challenges in the traditional software sales model. Linlin Wu, Saurabh Kumar Garg et al [11] has examined Migration from traditional software to Cloud which enables on-going revenue for software providers. However, in order to deliver hosted services to customers, SaaS companies have to either maintain their own hardware or rent it from infrastructure providers. This requirement means that SaaS providers will incur extra costs. In order to minimize the cost of resources, it is also important to satisfy a minimum service level to customers. Therefore, this paper proposes resource allocation algorithms for SaaS providers who want to minimize infrastructure cost and SLA violations.

Stephen Kaisler. Et al [2] examines service migration in a new computing paradigm, the cloud computing environment (CCE), by examining security and integration issues associated with service implementation. We postulate that a cloud architecture will evolve to be both more flexible and heterogeneous in resources because of the services complexity demanded byorganizations. This introduces additional, but tractable, complications when considering the service migration concept within three support areas; acquisition, implementation, and security that present significant challenges to service migration in the cloud.

A critical evaluation of current network resource allocation strategies and their possible applicability in Cloud Computing Environment which is expected to gain a prominent profile in the Future Internet are presented in work by M. Asad Arfeen [5]. Atsuo

Inomata et al. [6] has proposed a dynamic resource allocation method based on the load of VMs on IaaS, abbreviated as DAIaS. This method enables users to dynamically add and/or delete one or more instances on the basis of the load and the conditions specified by the user.

In a cloud computing environment, it is necessary to simultaneously allocate both processing ability and network bandwidth needed to access it. Tomita et al [8] proposed the congestion control method for a cloud computing environment which reduces the size of required resource for congested resource type, instead of restricting all service requests as in the existing networks.

VI. CONCLUSIONS

In this paper we have has described a work on the allocation of resources in a dynamic cloud environment by using priority algorithm which decides the allocation sequence for different jobs requested among the different user after considering the priority based on some optimum threshold decided by the cloud owner. This resource allocation technique is more efficient than grid and utility. With the advent of cloud computing and by using this implemented priority algorithm the cloud admin can easily take decision based on the different parameters discussed earlier and can efficiently allocate the available resources and with cost effectiveness as well as satisfaction from users. Finally cloud admin will decide how much profit he can be gained by allocating the available resources and prioritizing among the different user request. Various case studies are presented in order to evaluate the performance of the algorithm in terms of sustained time for many applications in many servers of different configuration. Simulation results also shows that on average, the profit algorithm optimized cost saving better when compared to other algorithms.

VIII. REFERENCES

[1] K C Gouda, Radhika T V, Akshatha M, "Priority based resource allocation model for cloud computing", Volume 2, Issue 1, January 2013,International Journal of Science, Engineering and Technology Research (IJSETR)

[2] Stephen Kaisler, SHK & Associates William H.Money, George Washington University. Service Migration in Cloud Architecture, Proceeding of the 44th Hawaii International Conference on System Sciences - 2011.

[3] Patricia Takako Endo, Andre Vitor de Almeida Palhares, Nadilma Nunes Pereira, 2011. Resource Allocation for Distributed Cloud: Concepts and Research Challenges, IEEE, july 2011.

[4] Hadi Goudarzi and Massoud Pedram University of Southern California, MaximizingProfit in Cloud Computing System via Resource Allocation.

[5] M.Asad Arfeen, Krzysztof Pawlikowski, Andreas Willig .2011, A Framework for Resource Allocation Strategies in Cloud Computing Environment, 2011 35th IEEE Annual Computer Software and Applications Conference Workshops.

[6] Atsuo Inomata, Taiki Morikawa, Minoru Ikebe. 2011, Proposal and Evaluation of a Dynamic Resource Allocation Method based on the Load of VMs on IaaS, IEEE 2011.

[7] Ikki Fujiwara, Isao ono, Kento Aida, Applying Double-sided Combinational Auctionsto Resource Allocation in Cloud Computing, 2010 10th Annual International Symposium on Applications and the Internet.

[8] Takuro TOMITA and Shin-ichi KURIBAYASHI, Congestion control method with fair resource allocation for cloud computing Environment . IEEE 2011.

[9] Kazuki MOCHIZUKI and Shin-ichi KURIBAYASHI, Evaluation of optimal resource allocation method for cloud computing environments with limited electric power capacity, 2011 International Conference on Network-Based Information Systems.

[10] Marco A. S. Netto and Rajkumar Buyya, Rescheduling Co-Allocation Requests based on Flexible Advance Reservations and Processor Remapping, IEEE 2008 9th Grid Computing Conference.

[11] Linlin Wu, Saurabh Kumar Garg and Rajkumar Buyya, SLA-based Resource Allocation for Software as a Service Provider (SaaS) in Cloud computing Environment, 2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing.

[12] Hadi Goudarzi and Massoud Pedram University of Southern California, Los Angeles, Multi-dimensional SLA-based Resource Allocation for Multi-tier Cloud computing Environment, 2011 IEEE 4th International Conference on Cloud Computing.

[13] T.R. Gopalakrishnan Nair, Vaidehi M, Efficient Resource Arbitration and and Allocation Stratagies in Cloud Computing through Virtualization, Proceedings of IEEE CCIS2011.

[14] Kejiang Ye, Xiaohong Jiang, Dawei Huang, Jianhai Chen, Bei Wang, Live Migration of Multiple Virtual Machines with Resource Reservation in Cloud Computing Environments, 2011 IEEE 4th International Conference on Cloud Computing.

[15] Moreno Marzolla, Ozalp Babaoglu, Fabio Panzieri, Server Consolidation in Clouds through Gossiping,IEEE 2011.

◈◈◈