# Stochastic Simulator for Optimal Cloud Resource Allocation in a Heterogeneous Environment

P. K. Suri, Ph.D
Dean, Research and Development; Chairman
HCTM Technical Campus, Kaithal, Haryana, India

Himanshi Goyal
HCTM Technical Campus

## ABSTRACT
Cloud computing environment provides on-demand access to shared resources that can be managed with minimal interaction of cloud service provider. It is a heterogeneous environment where number of users request for shared resources with different possible conditions. Cloud computing provides reliable and validated services to the users on pay as-you-use basis. In a cloud computing environment, resources are allocated in terms of virtual machines and allocating the virtual machine to an appropriate user is very important so as to efficiently utilize scarce resources and to satisfy QoS requirements. In this paper, an attempt has been made to develop a stochastic simulator that allocates virtual machine to the user with efficient resource utilization and minimal investment. In present simulator, resource allocation strategy depending upon the time and cost has been proposed to allocate resources (virtual machines) in order to fulfil the requirements of both, cloud users and service providers. In additions, it has been assumed that each VM is capable of executing all requests and the execution times are generated as samples from a specific non-.uniform probability distribution i.e. by Exponential Distribution function. Simulation results demonstrate the better performance of clouds with minimum makespan of jobs on a given set of heterogeneous virtual machines (VMs).

## General Terms
Allocation Algorithm, Non-uniform distribution

## Keywords
Cloud Computing, Resource Allocation, Simulator, Execution time, Distribution, Makespan.

## 1. INTRODUCTION
Like a buzzword, cloud computing means different to different people. Cloud Computing offers variety of services such as on-demand service, virtual servers, tested environment, disk storage etc., due to which cloud environment becomes quite popular among a sodality of cloud users. It is an emerging technology that is reinforcing itself in the development and employment of large number of applications.

Unlike Cluster and Grid computing, Cloud computing is service-oriented that delivers computing resources as services on- demand and are billed on subscription basis i.e. on the basis of pay-as-per-usage. It can be considered as an enhancement of grid computing. Cloud computing provides all types of IT facilities as a service to the cloud users. Cloud environment is initially offered by private enterprises such as Google, Amazon etc.

Resource allocation is the process of allocating available resources to the user's requests. Different kind of users exists in a cloud computing system. Therefore, resource allocation must be done in a manner that fulfils the requirement of all its users and service providers. The objective of cloud user's is to complete the job as fast as possible i.e. minimum response time; and of service providers is to effectively utilize the scarce resources. An optimized allocation strategy must be able to meet SLA requirements, based on various factors such as availability, response time, throughput, cost of resources etc.

This study presents a new approach for resource allocation. In the present work, a simulator is proposed and implemented that allocates the resources (Virtual Machines) to cloud users with minimal investment and provides effective resource utilization.

The paper is organized as follows: Section I (i.e. present one) provides a concise introduction of topic of our interest. In section II, an overview of related literature is presented. Section III provides the description of the symbols used in the proposed algorithm. Section IV describes the proposed system. Section V presents the proposed resource allocation algorithm. In section VI, simulation results are presented and examined. This study is concluded in section VII.

## 2. LITERATURE
Resource allocation is a critical issue in establishing a cloud environment; where multiple cloud user's requests for resources simultaneously with different possible constraints and a cloud provider has to serve them with better performance. To optimally utilize the scarce resources within the limit of clouds, an efficient allocation strategy is needed. Many allocation strategies are already present in the cloud environment, developed by researchers. An overview of these strategies is presented in this section.

In [1], topology aware allocation strategy is proposed. The proposed strategy works on the basis of what-if methodology, to help the cloud computing system in allocation decision. In [2], allocation is done on the basis of virtual machines; the proposed strategy works for non-cooperative cloud environment. In [3], linear scheduling strategy is used to allocate resources. In the proposed strategy a threshold value is set on the basis of which allocation is done. In [4], allocation is done on the basis of most-fit processor policy; cluster that produces a left over distribution is allocated to the job. In [5], a new framework called Nephele is proposed, that provides dynamic resource allocation. In [6], gossip-based protocol is used for resource allocation in large-scale clouds; that allocates resources to the applications having time-dependent memory demands. In [7], allocation is done on the basis of priority where various parameters such as cost, task type, time etc. are used to decide the priority among different user request. In [8], auction mechanism is used to allocate the resources, in which cloud providers collect the bid from user's and on that basis takes allocation decision. In [9], market based allocation strategy is used, that uses the concept of equilibrium theory i.e. to maintain balance between supply and demand in the system.

In [10], allocation is done on the basis of workflow representation of the application and four strategies are designed to allocate the resources. In [11], queueing model based strategy is proposed to allocate the resources, where arrival of jobs follows non-homogeneous Poisson process. In [12], a new approach named as "pre-copy" is proposed in which memory pages are repeatedly copied to the destination host. In [13], a negotiate approach is presented, where both cloud users and providers automatically negotiate resource leasing contracts. In [14], location-aware allocation is performed that depends upon utilization level of physical machines and location of user and data center. In [15], resources are changed within an already set time interval according to the load modification. In [16] and [17], utility function is used to allocate the resources. In former, response time is used as a measure of utility function and in latter current workload of a system is used to compute utility function. In [18], an adoptive approach is presented to allocate resources in runtime for service based systems such as Grid computing, Cloud computing, utility computing etc. In [19], first clusters are categorized on the basis of parameters and the generalized processor sharing is used to allocate the resources. In [20], an adaptive approach i.e. based on the CPU consumption amount is presented.

# 3. NOTATIONS

**Table 1: Symbols used in the proposed algorithm**

| Symbol Used | Description |
|---|---|
| n | Number of jobs |
| m | Number of virtual machines |
| $t_{ij}$ | Expected Completion (or execution) time of $i^{th}$ job on $m^{th}$ machine |
| lamda[i] | Rate at which $i^{th}$ machine execute the jobs. |
| lr | Lower Range |
| hr | Higher Range |
| s[i][j] | Random Samples |
| $c_i$ | Per Unit Cost for each machine |
| counter | Number of jobs already allocated |
| max | Variable that stores result of Max(N,M) function |
| minimum | Variable that stores result of Min function |
| allval | Variable that stores value of allocated entry |

# 4. PROPOSED WORK

In a cloud computing system, multiple users can make a request at a same time therefore; an efficient and optimized strategy is required so that a cloud provider can serve all the requests with improved or better performance. An allocation strategy is needed that will be able to achieve maximum throughput and improved performance for the cloud computing system. Different allocation techniques are already available for the same. The simulator that we have proposed in this paper allocates the job to the best machine (VM) with minimal investment and provides reasonable performance to both cloud users and providers. The proposed strategy aims to provide better Qos and to meet SLA requirements in terms of availability, cost, time, performance etc. Our main objective is to minimize the overall makespan of jobs on machines and to provide efficient resource utilization.

In the present work few assumptions are made. It is assumed that each VM is having sufficient resources to fulfil the need of all kind of users i.e. able to execute all requests. But VM's will execute the job with a varying service rate, used as a measure of processing capabilities. As all nodes (VMs) participating in a cloud having different processing power; the proposed strategy is used for resource allocation of jobs on virtual machines in an efficient way for a heterogeneous environment.

The proposed model is described as: Let n be the number of jobs to be allocated on m number of machines (VMs). Each machine executes the job with a service rate say s i.e. $\{s_1,s_2,\ldots..s_m\}$ be the service rate of machines $\{m_1, m_2,\ldots m_m\}$ respectively and at a time only one job can execute on a machine. The execution of jobs on machines follows a non-uniform probability distribution i.e. Exponential Distribution for the proposed strategy and before starting the allocation process, expected completion time of jobs on each VM is computed. Let $T_{ij}$ be the expected completion time of job $j_i$ on machine $m_j$. The completion time of n number of jobs on m number of machines, having service rate s is represented in table 2.

**Table 2: Expected Completion Time**

| VMs / Jobs | $M_1$ ($s_1$) | $M_2$ ($s_2$) | …….. | $M_m$ ($s_m$) |
|---|---|---|---|---|
| $J_1$ | $T_{11}$ | $T_{12}$ | …….. | $T_{1m}$ |
| $J_2$ | $T_{21}$ | $T_{22}$ | ……. | $T_{2m}$ |
| ⋮ | ⋮ | ⋮ | … | ⋮ |
| Jn | $T_{n1}$ | $T_{n2}$ | …… | $t_{nm}$ |

The problem can be mathematically formulated as: two sets are given, J and M with a weight function (i.e. time function) T.

J → Set of n number of jobs, and
M→Set of m number of machines.
T: J*M → R

Then the job is allocated to the machine such that Time function is minimized i.e.

$\Sigma j \in J \; T(j,f(j))$ is minimized.

## 5. SIM_OPCLOUD_ALLOC ALGORITHM

Step 1: Generate expected completion time of N number of jobs on each machine $M_i$ as random samples within a specific range (lr---hr) from an exponential distribution with the service rate (lamda[i]).

```
for (i=1; i<=n; i++)
for (j=1; j<=m; j++)
t[i][j] = lr + (hr-lr) * ((-1.0)/lamda[j]) * log(s[i][j]).
```

Step 2: For each VM, Input per unit cost and initialize the value of counter at zero.

Step 3: Create a matrix T with n number of rows and m number of columns, where $T_{ij}$ is the execution time of job i on machine j.

Step 4: Convert the matrix T into T'' with max number of rows and columns, by keeping the value of additional entries equals to zero.

```
if (n>m)
        for ( j= m+1, max)
                t[i][j] = 0             for (i=1, n)
else
        for ( j= n+1, max)
                t[i][j] = 0             for (j=1, m)
```

Step 5: While (counter < n)
Do steps 6 to 17.

Step 6: For matrix T'', subtract the minimum execution time of each job from all the times of respective job i.e.

```
for (i=1; i<=max; i++)
        minimum = Min( t_ii)            (j= 1, max)
        t_ij = t_ij – minimum           (j= 1, max)
```

Step 7: Do same as step 6, for virtual machines.

Step 8: Examine the jobs successively until one with single zero is found, mark this entry to make an allocation. Then mark all other entries with value zero in corresponding machine so that cannot be considered for further allocation.

Step 9: After examine all jobs, do same for virtual machines.

Step 10: Now check if exactly one marked entry is obtained in each job and VM then go to step 14 else step 11.

```
for (i=1; i<=max; i++)
        for (j=1; j<=m; j++)
        if (MP[i][j] == 1)  go to 14
```

Step 11: Covered all the entries with mark zero by minimum number of lines.

Step 12: Determine the smallest time among those that are not covered.

Step 13: Subtract this smallest value from all the uncovered entries and add at the intersection of lines and go to step 8.

Step 14: Then according to the marked entries, the jobs allocated to virtual machines (1 to M) makes an optimal assignment and Update the counter value by that number of jobs.

Step 15: Now update the times as:

```
for (j=1; j<=max; j++)
        allval = t_ji
        t_ji = t_ji + allval
```

Step 16: Delete the jobs that are allocated to machines (1 to m) form the job matrix, and input the updated no of jobs to n'.

Step 17: Now create a matrix T with n' number of rows and m number of columns. And convert it into T' same as step 4.

Step 18: Print the computed statistics.

Step 19: End

## 6. RESULTS

In this section, the performance of the proposed algorithm is evaluated in terms of makespan, used as a measure of system throughput. Makespan is defined as the maximum amount of time a job takes to execute on any machine (VM) among those on which jobs are allocated. The performance is evaluated in following scenarios:

Scenario I: In this case, overall makespan is computed for fixed number of jobs (say, N=8) and machines (say, M=4). The expected completion (or execution) time of 8 jobs on 4 machines are recorded in table 3.

**Table 3: Expected Completion time; generated as random samples from an Exponential Distribution function.**

| | M1 | M2 | M3 | M4 |
|---|---|---|---|---|
| J1 | 148 | 112 | 144 | 105 |
| J2 | 143 | 120 | 637 | 175 |
| J3 | 109 | 274 | 107 | 235 |
| J4 | 206 | 171 | 201 | 177 |
| J5 | 141 | 225 | 102 | 278 |
| J6 | 101 | 110 | 109 | 100 |
| J7 | 376 | 100 | 108 | 203 |
| J8 | 306 | 232 | 268 | 131 |

Graph representing the allocation of eight jobs on four machines is shown below:

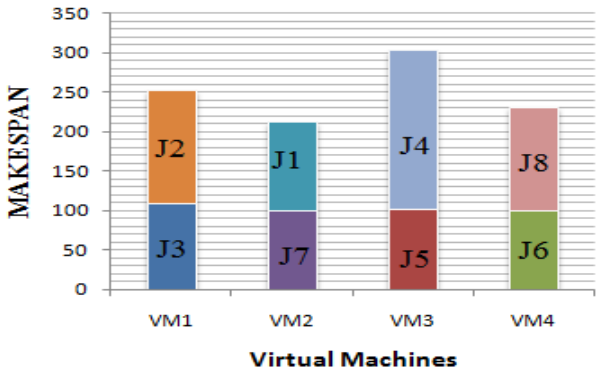


**Figure 1: Allocation of eight jobs on four virtual machines.**

It depicts that makespan of eight jobs on four machines is 303. As all the machines are equally utilized that means proposed strategy provides efficient resource utilization and load balancing.

Scenario II: In this scenario, makespan and the cost of corresponding allocation is computed for fixed number of virtual machines (say, M=4) at varying number of jobs (N). For the following analysis, we assume that cost of virtual machines {VM1, VM2, VM3, and VM4} is {600, 700, 900, 1000} respectively. The resultant values are recorded in table 4.

**Table 4: Performance parameter values for fixed number of VMs (say, M=4), at varying (N).**

| No. of Jobs (N) | 4 | 6 | 8 | 10 |
|---|---|---|---|---|
| Makespan | 161 | 274 | 280 | 397 |
| Execution Cost | 3200 | 5100 | 6400 | 7900 |

Graphs for the above table are represented in figure 2 and figure 3.

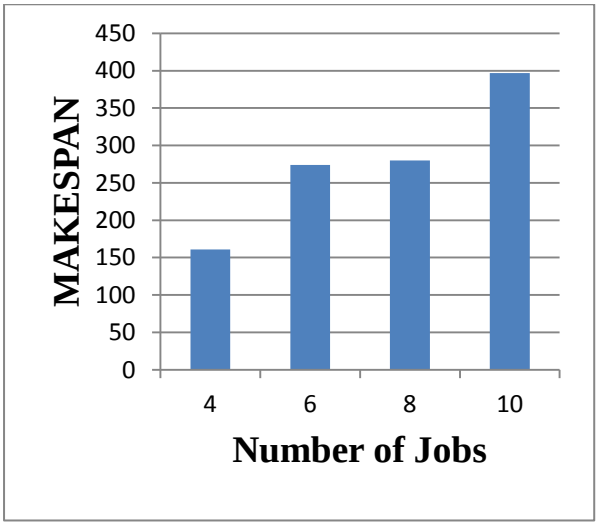


**Figure 2: Average Makespan analysis for fixed number of VMs (M=4), at varying (N).**

It depicts that increase in the number of VM's, increases the value of makespan (i.e. completion time of all jobs).

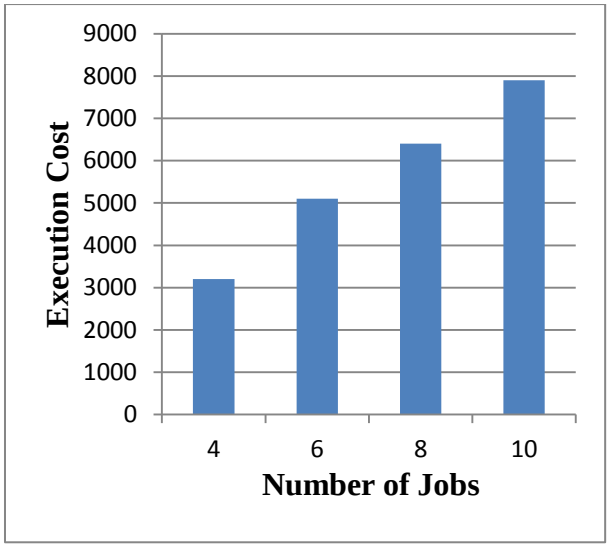


**Figure 3: Execution Cost analysis for fixed number of VMs (M=4), at varying (N).**

It depicts that increase in the number of VM's results into an increase in execution cost.

Scenario III: In this scenario, parameters makespan and the cost of corresponding allocation is computed for fixed number of jobs (say, N=10) at varying number of virtual machines (M). Cost of virtual machines (VMs) is already assumpted. The resultant values of both the parameters are recorded in table 5.

**Table 5: Performance parameter values for fixed number of Jobs (say, N=10), at varying (M).**

| No. of VMs (M) | 4 | 6 | 8 | 10 |
|---|---|---|---|---|
| Makespan | 445 | 294 | 252 | 122 |
| Execution Cost | 4300 | 5900 | 7100 | 8000 |

Graphs for the above table are represented in figure 4 and figure 5.

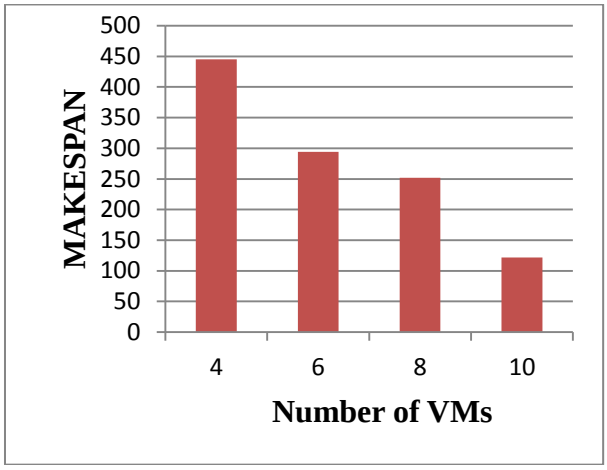


**Figure 4: Average Makespan analysis for fixed number of Jobs (say, N=10), at varying (M).**

It depicts that increase in the number of VM's, lowers the value of makespan (i.e. completion time of all jobs).

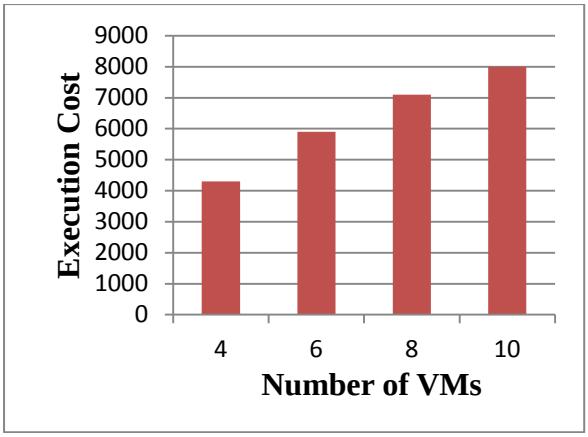


**Figure 5: Execution Cost analysis for fixed number of Jobs (say, N=10), at varying (M).**

It depicts that increase in the number of VM's also results into an increase in execution cost.

## 7. CONCLUSION

This study presents a successful implementation of the proposed simulator "SIM OPCLOUD ALLOC" that allocates the job to the best machine so that overall makespan of jobs on virtual machines minimizes. The proposed system provides efficient resource utilization and load balancing. The performance of the proposed strategy is evaluated in terms of maximum completion time i.e. makespan and execution cost. This simulator will be an asset for developing a dynamic stochastic simulator to uplift the cloud computing culture globally.

## 8. REFERENCES

[1] Gunho Lee, Niraj Tolia, Parthasarathy Ranganathan, and Randy H. Katz, "Topology-Aware Resource Allocation for Data-Intensive Workloads", ACM SIGCOMM Computer Communication Review, Vol. 41, No. 1, pp. 120-124, 2011.

[2] Zhen Kong et.al, "Mechanism Design for Stochastic Virtual Resource Allocation in Non-Cooperative Cloud Systems", IEEE 4th International Conference on Cloud Computing, pp. 614-621, 2011.

[3] Abirami S.P. and Shalini Ramanathan, "Linear Scheduling Strategy for Resource Allocation in Cloud Environment", International Journal on Cloud Computing: Services and Architecture, Vol. 2, No. 1, pp. 9-17, 2012.

[4] Kuo-Chan Huang and Kuan-Po Lai, "Processor Allocation Policies for Reducing Resource Fragmentation in Multi Cluster Grid and Cloud Environments", IEEE, pp. 971-976, 2010.

[5] Daniel Warneke and Odej Kao, "Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud", IEEE Transactions on Parallel and Distributed Systems, 2011.

[6] FetahiWuhib and Rolf Stadler, "Distributed Monitoring and Resource Management for Large Cloud Environments", IEEE, pp. 970-975, 2011.

[7] K C Gouda, Radhika T V, and Akshatha M, "Priority Based Resource Allocation Model for Cloud Computing", International Journal of Science, Engineering and Technology Research, Vol. 2, No. 1, 2013.

[8] Wei-Yu Lin et al, "Dynamic Auction Mechanism for Cloud Resource Allocation", IEEE/ACM 10th International Conference on Cluster, Cloud and Grid Computing, pp. 591-592, 2010.

[9] Xindong YOU, Xianghua XU, Jian Wan, and Dongjin YU, "RAS-M :Resource Allocation Strategy based on Market Mechanism in Cloud Computing", IEEE,pp. 256-263, 2009.

[10] Tram Truong Huu and John Montagnat, "Virtual Resource Allocations Distribution on a Cloud Infrastructure", IEEE, pp.612-617, 2010.

[11] Satyanarayana .A, Dr. P. Suresh Varma, Dr. M.V.Rama Sundari, and Dr. P Sarada Varma, "Performance Analysis of Cloud Computing under Non Homoeneous Conditions", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, No. 5, 2013.

[12] Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hanseny, Eric July, Christian Limpach, Ian Pratt, and Andrew Warfield, "Live Migration of Virtual Machines", 2nd Symposium on Networked Systems Design and Implementation, 2005.

[13] Bo An, Victor Lesser, David Irwin, and Michael Zink, "Automated Negotiation with Decommitment for Dynamic Resource Allocation in Cloud Computing", Proceedings of 9th International Conference on Autonomous Agents and Multi-agent Systems, Vol. 1, 2010.

[14] Gihun Jung and Kwang Mong Sim, "Location-Aware Dynamic Resource Allocation Model for Cloud Computing Environment", International Conference on Information and Computer Applications, Vol. 24, 2012.

[15] Nilabja Roy, Abhishek Dubey and Aniruddha Gokhale, "Efficient Autoscaling in the Cloud using Predictive Models for Workload Forecasting", Cloud Computing IEEE International Conference, pp. 500-507, 2011.

[16] HadiGoudaezi and MassoudPedram, "Multidimensional SLA-based Resource Allocation for Multi-tier Cloud Computing Systems", IEEE 4th International conference on Cloud Computing, pp. 324-331, 2011.

[17] Hien Nguyen et al, "SLA-Aware Virtual Resource Management for Cloud Infrastructures", IEEE 9th International Conference on Computer and Information Technology, pp. 357-362, 2009.

[18] Stephen S. Yau and Ho G., "An Adaptive Resource Allocation for Service-Based Systems", International Journal of Software and Informatics, Vol. 3, No. 4, pp. 483–499, 2009.

[19] HadiGoudarzi and MassoudPedram, "Maximizing Profit in Cloud Computing System Via Resource Allocation", IEEE 31st International Conference on Distributed Computing Systems Workshops, pp. 1-6, 2011.

[20] Patricia Takako Endo et al., "Resource Allocation for Distributed Cloud: Concept and Research Challenges", IEEE, pp. 42-46, 2011.