RESEARCH ARTICLE OPEN ACCESS

# Analysis of a Pool Management Scheme for Cloud Computing Centres by Using Partial Acceptance Policy

Mr. M. Karthick Selvam, Mr. S. Gopinathan
Asst Professor, Dept of CSE, Annamalaiar College of Engineering, Modaiyur
Asst Professor, Dept of CSE, Aksheyaa College of Engineering, Puludivakkam

**Abstract**
A monolithic model may suffer from and poor scalability due to large number of parameters. A cloud user may submit a super task at once. The user request is sent to the global queue and then to the Resource Assigning Module (RAM). A number of heterogeneous server pools placed in the RAM. First is Hot, in which the servers will be handling the jobs currently, second is Warm, in which the servers are kept in ideal state, then Finally Cold, in which the servers are Turned Off state. Initially the request is send to Hot, if those servers are busy the request is forwarded to warm, then finally if required to Cold if both the hot and warm server pools are busy. The user submitted supertask may split so that the individual task run on different physical machines, this is called as partial acceptance policy. So the supertask rejection ratio will be reduced.

**Keywords**—Partial Acceptance Policy, Total Acceptance Policy, RAM, FIFO, Mean Service Time, Rejection Ratio

## I. INTRODUCTION

The storing and accessing of applications often through a web browser rather than running installed software on your personal computer or office server. This is called as cloud computing. The cloud computing provides many different types of services. 1. Software as a Service (SAAS) – Consumers purchase the ability to access and use an application or service that is hosted in the cloud. 2. Platform as a Service (PAAS) – consumers purchase access to the platforms, enable them to deploy their own software and application in the cloud. 3. Infrastructure as a Service (IAAS) – consumers control and manage the systems in terms of the operating systems, applications, storage and network connectivity, but do not themselves control the cloud infrastructure.

A cloud user may submit a compound request which consists of two or more individual simple task at once, this is called as supertask. We assume that the cloud centre will consists of number of physical servers. Each physical server will consists of 'n' number of virtual machines. The user requested job will be allocated to these machines.

## II. RELATED WORK

The previous work does the analysis of pool management scheme by using the Total Acceptance Policy. The Total Acceptance Policy suggested that the size of the supertask and the number of virtual machines in the physical servers will be more or equal. Otherwise the supertask will be rejected. The supertask will be assigned to a single physical server, it cannot be split and run in to the different physical servers.

Quantifying resiliency of IAAS cloud measures the two key performances with respect to the job rejection rate and provisioning response delay. It measures the above performances by using stochastic reward nets an extension of generalized stochastic petri nets.

There are two optimization mechanism to improve the isolation property. They are performance isolation and fault isolation. Performance isolation is the one which indicates the effect of performance when consolidating several work loads into one physical servers. The fault is the another one which indicates the effect of performance when the misbehaviour work load which affects the other work loads.

The fine grained performance model that permits user to submit a supertask with a high degree of virtualization. Each pool has a fixed number of VM's A user submit a brust of task if there is enough room for the whole supertask then only it will be accepted, otherwise the supertask will be rejected

## III. CURRENT WORK

The current work will consists of five modules. These modules are explained as follows.

### 3.1 Client of the Network

In this module we are going to create an User application by which the User is allowed to access the data from the Server of the Cloud Service Provider. Here first the User want to create an account and then only they are allowed to access the Network. Once the User create an account, they are to login into their account and request the Job from

the Cloud Service Provider. Based on the User's request, the Cloud Service Provider will process the User requested Job and respond to them. All the User details will be stored in the Database of the Cloud Service Provider. In this Project, we will design the User Interface Frame to Communicate with the Cloud. By sending the request to Cloud Server Provider, the User can access the requested data if they authenticated by the Cloud Service Provider.

**3.2 Cloud Service Provider**

Cloud Service Provider will contain the large amount of data in their Data Storage. Also the Cloud Service provider will maintain the all the User information to authenticate the User when are login into their account. The User information will be stored in the Database of the Cloud Service Provider. Also the Cloud Server will redirect the User requested job to the Resource Assigning Module to process the User requested Job. The Request of all the Users will process by the Resource Assigning Module. To communicate with the Client and with the other modules of the Cloud Network, the Cloud Server will establish connection between them. For this Purpose we are going to create an User Interface Frame. Also the Cloud Service Provider will send the User Job request to the Resource Assign Module in Fist In First Out (FIFO) manner.

**3.3 Resource Assigning Module (RAM)**

In this Module, we will Process the User requested Job. The User requested Job will redirected to the RAM of the Cloud Server. The RAM will contain three Types of the Physical Servers. 1. HOT Server,2. WARM Server and 3.COLD Server. These Physical Servers will contain 'n' number of virtual Server to process the User requested Job. So that the Job can be efficiently processed. To communicate with the Physical Server and Virtual Server we will develop the network coding in the Java / .Net Platforms. We have to create a separate Interface Frame of each Physical Servers and Virtual Servers. For each Physical Servers and Virtual Servers will assign an IP address through Network Connection.

**3.4 Job Processing**

Once the RAM got the User requested Job from the Server of the Cloud Service Provider, it will first check the HOT Server, because the HOT server will handle the Current User requested Job. If the Virtual Machines of the HOT Server is busy then the Job will be transferred to WARM Server which will be idle state when they didn't have any Job to Process. So that the WARM Server will process the Job. But if the Virtual Server of the WARM Server is also busy, then the request will be passed to the COLD Server. By implementing this Job Processing Scheme, we can effectively process the User Requested Job and efficiently maintains the Resources of the Cloud Server. So that we can save the Energy of the Resources when they are not process the Job. The job processing will be carried out by using the Partial Acceptance Policy. The Successive Substitution Algorithm is used to process the user requested job.

**3.5 Cache Memory Management**

As a modification in this Project, we are creating a Cache Memory in the User requested job will be stored for the period time. If the another User requests the same Job to the Server of the Cloud Service Provider (CSP), the Server will check in the Cache Memory first. So that we can reduce the job processing time. If the request Data is presented, then the Server will provide the Data to the User immediately. If the request data is not in the Cache Memory, then the Server process the User requested job by transferring it to the RAM.

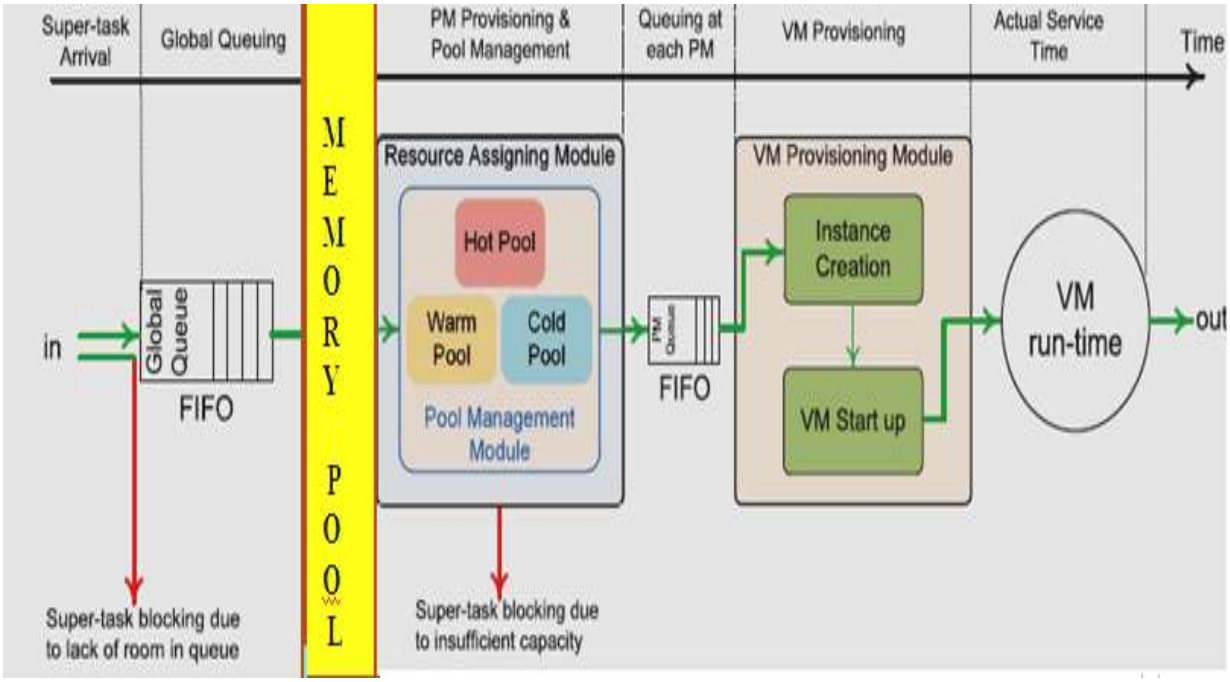


Fig 1.The Pool Management with Cache Memory

## IV. SYSTEM IMPLEMENTATION

In this project we are implementing an SSM algorithm by which we are able to allocate the resource more effectively. First this algorithm will the check the available resources on the Hot Pool, if there is no resources are available then transfer the request to the Warm pool and check the for the available resource and if there is no resources are available then check in the Cold Pool. If the resource are available then allocate the job to the virtual machine to process the User requested Job. So that the Jobs are processed in best manner.

There is an interdependency among submodels. This cyclic dependency is resolved via fixed-point iterative method using a modified version of successive substitution approach. For numerical experiments the successive substitution method (Algorithm 1) is continued until the difference between the previous and current value of blocking probability in the global queue

### 4.1. Successive Substitution Algorithm

**Algorithm 1:** Successive Substitution Method
**Input:** Initial success probabilities in pools: $P_{h0}$, $P_{w0,}$ $P_{c0}$
Initial idle probability of a hot PM: $P_{i0}$
**Output**: Blocking probability in Global Queue: $BP_q$
counter ← 0, max ← 10, diff ← 1
$BP_{q0}$ ← RASM ($P_{h0}$, $P_{w0}$, $P_{c0}$)
[$N_h$, $N_w$, $N_c$] ← PMM ($P_{h0}$, $P_{i0}$)
while diff ≥ $10^{-6}$ do
counter ← counter + 1
[ $P_h$, $P_i$ ] ← VMPSM_hot ($BP_{q0}$, $N_h$)
$P_w$ ← VMPSM_warm ($BP_{q0}$, $P_h$, $N_w$)
$P_c$ ← VMPSM_cold ($BP_{q0}$, $P_h$, $P_w$, $N_c$)
[ $N_h$, $N_w$, $N_c$] ← PMM ($P_h$, $P_i$)
$BP_{q1}$ ← RASM ($P_h$, $P_w$, $P_c$)
diff ← | ($BP_{q1}$ ─ $BP_{q0}$) |
$BP_{q0}$ ← $BP_{q1}$
if counter = max then
break
end if
end while
if counter = max then
return─ 1
else
return $BP_{q0}$
end if

## V. NUMERICAL VALIDATION

In this work we concentrated on various numerical validations such as the rejection ratio, load Vs throughput and mean service time.

The fig.2 shows that rejection ratio between the partial acceptance policy and the acceptance policy. Whenever the supertask size increases the rejection ratio in the partial acceptance policy will be very high when compared to total acceptance policy

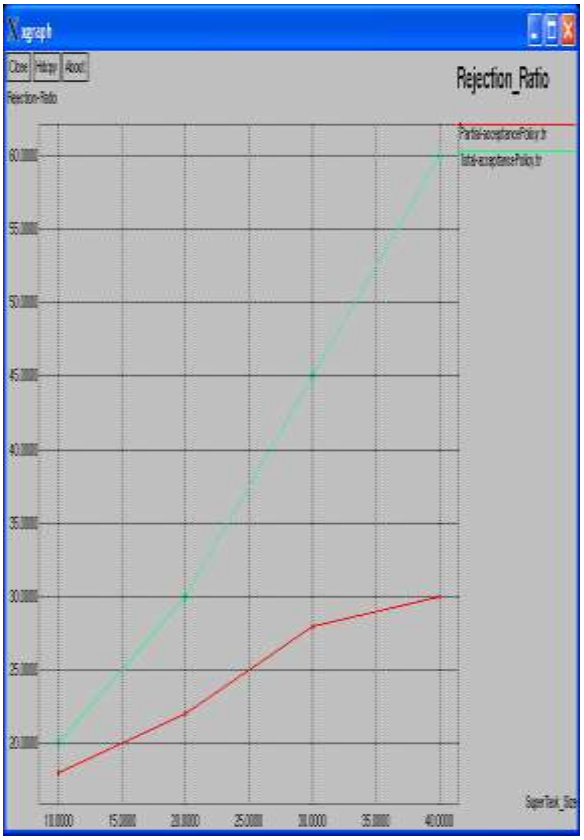


Fig2. Rejection Ratio: Partial Acceptance Policy Vs Total Acceptance Policy

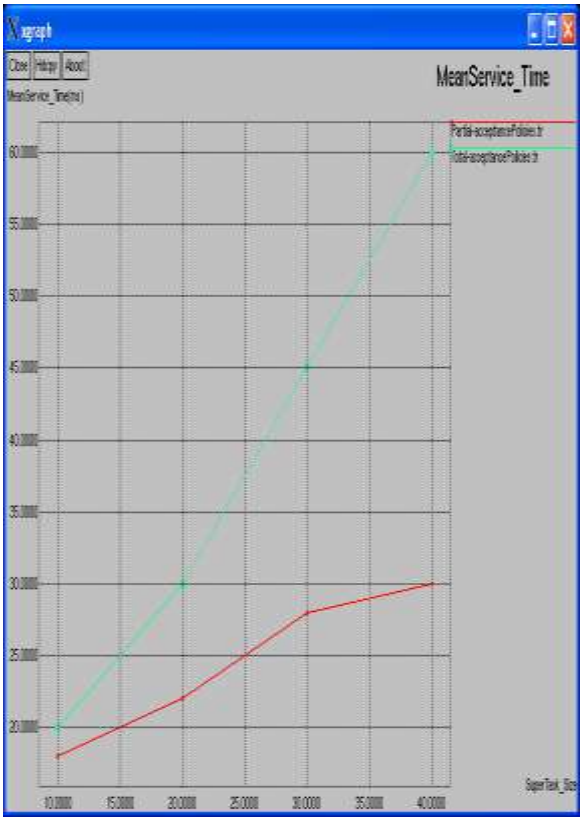


Fig3 Mean Service Time: Partial Acceptance Policy Vs Total Acceptance Policy

The fig 3 shows that the mean service time between the partial acceptance policy and the total acceptance policy.

Whenever the supertask size increases, the mean service time of the partial acceptance policy will be very low when compared to the total acceptance policy

## VI. CONCLUSION

In this paper, we have developed an interacting analytical model that captures important aspects including resource assigning process, virtual machine deployment, pool management, and power consumption of nowadays cloud centers. The performance model can assist cloud providers to predict the expected servicing delay, task rejection probability, steady-state arrangement of server pools, and power consumption. We carried out extensive numerical experiments to study the effects of various parameters such as arrival rate of supertasks, task service time, virtualization degree, supertask size, and pool check rate on the task rejection probability, response time, and normalized power consumption. The behavior of cloud center for given configurations has been characterized in order to facilitate the capacity planning, SLA analysis, cloud economic analysis, and tradeoffs by cloud service providers. Using the proposed pool management model, the most appropriate arrangement of server pools and the amount of required electricity power can be identified in advance for anticipated arrival process and super task characteristics.

## VII. FUTURE ENHANCEMENT

In the Project we are concentrated to process the User requested Job in partial acceptance manner. In the Partial Acceptance manner may split a super task so that individual tasks run on different PMs. While this policy may reduce the super task rejection probability, it may also increase intertask communication overhead and idle waiting, and, consequently, extend the overall service time. So in future, we can concentrated to reduce the intertask communication and idle waiting.

## REFERNCES

[1] HamzehKhazaei, Vojislav B. Misic, "Analysis of a Pool Management Scheme for Cloud Computing Centers," IEEE Transactions on Parallel and Distributed Systems, Vol. 24, No. 5, May 2013.

[2] An amazon.com Company, "Amazon Elastic Compute Cloud, Amazon EC2,"Website, http://aws.amazon.com/ec2. 2012.

[3] H. Khazaei, J. Misic, and V.B. Misic, "Performance Analysis of Cloud Computing Centers Using M=G=m=m þ r Queueing Systems," IEEE Trans. Parallel and

Distributed Systems, vol. 23, no. 5, pp. 936-943, May 2012.[4] IBM, "IBM Cloud Computing," Website, http://www.ibm.com/ ibm/cloud/, 2012.

[5] F. Longo, R. Ghosh, V.K. Naik, and K.S. Trivedi, "A Scalable Availability Model for Infrastructure-as-a-Service Cloud," Proc. Int'l Conf. Dependable Systems and Networks, pp. 335-346, 2011

[6] H. Khazaei, J. Misic, and V.B. Misic, "Performance Analysis of Cloud Centers under Burst Arrivals and Total Rejection Policy," Proc. IEEE GLOBECOM '11, pp. 1-6, Dec. 2011.

[7] A. Gandhi, V. Gupta, M. Harchol-Balter, and M.A. Kozuch, "Optimality Analysis of Energy-Performance Trade-Off for Server Farm Management," Performance Evaluation, vol. 67, no. 11, pp. 1155-1171, 2010.

[8] K. Ye, X. Jiang, D. Ye, and D. Huang, "Two Optimization Mechanisms to Improve the Isolation Property of Server Consolidation in Virtualized Multi-Core Server," Proc. IEEE 12th Int'l Conf. High Performance Computing and Comm. (HPCC), pp. 281-288, Sept. 2010.

[9] R. Ghosh, F. Longo, V.K. Naik, and K.S. Trivedi, "Quantifying Resiliency of IaaS Cloud," Proc. IEEE Symp.Reliable Distributed Systems, pp. 343-347, 2010. [10] S. Ferretti, V. Ghini, F. Panzieri, M. Pellegrini, and E. Turrini, "QoS-Aware Clouds," Proc. IEEE Third Int'l Conf. Cloud Computing (CLOUD), pp. 321-328, July 2010.

[11] K. Xiong and H. Perros, "Service Performance and Analysis in Cloud Computing," Proc. IEEE World Conf. Services, pp. 693-700, 2009.

[12] L.M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A Break in the Clouds: Towards a Cloud Definition," ACM SIGCOMM Computer Comm. Rev., vol. 39, no. 1, 2009.

[13] N. Sato and K.S. Trivedi, "Stochastic Modeling of Composite Web Services for Closed-Form Analysis of Their Performance and Reliability Bottlenecks," Proc. Int'l Conf. Service Oriented Computing (ICSOC '07), pp. 107-118, 2007.

[14] M. Martinello, M. Kaaniche, and K. Kanoun, "Web Service Availability-Impact of Error Recovery and Traffic Model," Reliability Eng. System Safety, vol. 89, no. 1, pp. 6-16, 2005.