Overload Avoidance Model using Optimal Placement of Virtual Machines in Cloud Data Cetres

Narander Kumar
Department of Computer Science, B. B. Ambedkar University,
Lucknow, U.P., India.

Shalini Agarwal
Department of Computer Science, B. B. Ambedkar University,
Lucknow, U.P., India.

Vipin Saxena
Department of Computer Science, B. B. Ambedkar University,
Lucknow, U.P., India.

ABSTRACT
Cloud data centres improve CPU utilization of their servers (physical machines or PMs) through Virtualization (virtual machines or VMs). Over virtualised and under virtualized PMs suffer performance degradation and power dissipation respectively. This work presents a stochastic modular scheme for allocating VM requests to a PM by avoiding overloading of PM and keeping the global load characteristics under specified QoS goal. The proposed approach categorizes PMs into three groups (Under Load, Normal Load, Over Load) in a way that minimizes number of PMs in Under Load and Over Load groups and maximizes number of PMs in Normal Load group. We compute VM request rejection probability, response time, service time and number of PMs that are overload or under loaded for evaluating the performance of our model. The results show that these parameters do not degrade with increasing arrival rate. Thus the proposed model is simple yet efficient approach for VM placement problem.

Keywords
Markov Chain, Virtual Machine, Physical Machine, Virtual Chunks, VM Consolidation.

1. INTRODUCTION
Cloud Computing is based on Autonomic Computing Model with access to a vast and shared pool of resources (e.g. servers, storage, applications, services) that are subject to rapid provisioning and deployment at the user's site with minimal effort on the part of resource service provider. It is becoming a pervasive model [1] in which achieving high reliability and scalability of applications and enabling the cloud resource provider to maximize CPU utilization subject to the constraints imposed by the need to optimize QoS are certain conflicting objectives [1][12][16]. Therefore, developing a model that captures resiliency and yet be tractable, is an area of active research [3].

The integral component of the next generation cloud data centers is the virtualization [9][10] of compute and storage resources which enables a number of virtual machines (VMs) to be deployed and scaled on a single physical machine (PM) or host, corresponding to the input workload. A VM refers to the software implementation of a computer that runs its own OS and applications as if it were a physical machine. Most cloud data centers implement automatic VM placement algorithms [2][4][5][8][9][10][12][15][19][20][22][23], in which the most suitable host is selected by categorizing VM resource requirements and its anticipated expansion while optimizing the placement goals. There are numerous placement goals which include maximizing the usage of available resources, power saving by switching idle hosts to sleep mode, meeting SLA and optimizing server consolidation [11][14][16][25]. In order to meet one or more of the placement goals and depending upon the dynamic workload submitted to the hosts, the placement controller component allows VMs to move from one physical machine to another, under state consistency preservation constraints. This form of movement is termed as live migration, in the literature[11][14][16][25](a sub process of VM consolidation), which is desirable, yet, being a resource intensive operation, consumes several CPU cycles and appreciable network bandwidth, affects the performance of applications as well as resource usage of the migrating and collocated VMs. Therefore, a model that avoids overloading and under loading of PMs directly influences VM migrations.

PMs become under loaded or overload by dynamic departure and deployment of VMs to PMs [13]. This work proposes a VM placement algorithm that places VMs to PMs in such a way so as to keep the number of overloaded and under loaded PMs to a minimum so that the interval between successive VM consolidations is maximized. The VM placement algorithm is described as a continuous time Markov Chain model, which has the knowledge of the current and future load characteristics of the PMs in the cloud data centre in terms of computed probabilities. The model is essentially a collection of interacting sub-models that exchange their outputs to minimize VM request rejection probability and the response time. The proposed model efficiently represents the dynamics of today's cloud centers as compared to the earlier monolithic models that were more restrictive in nature.

The rest of the paper is organized in the following manner. In section 2, we discuss the related work, followed by a CPU utilization model in section 3. We present System Model of VM placement technique in section 4 and the associated sub models with mathematical formulations in sections 5, 6, 7, 8 respectively Section 9 presents the VM placement Algorithm. Simulation and numerical result analysis is given in section 10. Finally we give conclusion and future directions in section 11.

2. RELATED WORK
Stochastic models for data centre performance have been proposed in literature [1][5][6]. In [1], the authors proposed a statistical model for PM overload detection for stationary workload that maximizes mean inter migration time. The authors give the categorization of dynamic VM consolidation as periodic, heuristic-based and thresh-hold based. A degree constraint is introduced in [2], and using this constraint a model of virtual machine allocation problem is developed. However, after theoritical modeling several heuristics has been proposed to solve the on-line version of the problem. An efficient and responsive economic resource allocation in high-

performance computing environments is proposed in [4] by avoiding repeated allocation and commitment of resources. But the associated penalty functions also play their role. An analytical performance model, similar to our work, is proposed in [5] that assigns 'supertasks' to hot, warm and cold PMs in the given order. This paper, along with [1] motivated us towards incorporating load aspects to managing load on cloud servers.

A heuristic methodology for resource mapping to federated infrastructures is formulated as mixed integer problem in [9]. Similar work is carried out in [8, 12] where several conflicting objectives are attempted to solve through a single aggregate function. The underlying concept is not comparable to ours because of the difference in the mode of solution. The work involve efficient control on VM migrations and are based on active strategies of balancing idle and overload hosts. We propose a pro-active solution to this problem by controlling workload right from the placement phase. The work in [10] supports green computing, optimizes the number of servers in use by introducing the concept of 'skewness' and overload avoidance on PMs. [11] proposes a LP formulation and heuristics to control VM migration by giving priority to those VMs that have steady state capacities. The approach is different from ours because the proposed non-eager migration based VM placement algorithm also controls overloaded servers. [14] gives an entirely different methodology that combines peak power demand energy consumption on cooling, designed for tightly coupled parallel applications. The load balancing is based on frequency aware dynamic voltage scaling approach.

The authors of [15][16][28], have taken energy constraints as their main consideration for management and consolidation of data centres. In [17][18] client server and multitier system performance is discussed by allowing applications to elastically scale.[21][25][26][27] investigate virtualization by bounding migrations to reduce overheads. The underlying idea in [20][22][23] is to built QoS constraints in server management models. [28][29] give the foundation for mathematical set-up of VM placement. In [30] a policy optimization for managing power is given.

3. CPU UTILIZATION MODEL

We assume that the total CPU capacity of each host is divided into small units of allocation and we call this division a Virtual Chunk (VC). Each Virtual Machine (VM) consumes certain number of Virtual Chunks on a given host, given that number of VCs required by VM is known apriori. Thus CPU utilization of PM is expressed in terms of the number of VC consumed on it. The division of C into VC also helps to measure the load characteristics of the PM. In order to give a mathematical formulation of VM placement algorithm, we assume that CPU utilization on a PM progresses by consuming VCs in three types of region on PMs which we name as, Under Load (UL), Normal Load (NL) and Over Load (OL). Let c1, c2, c3 be the number of chunks in UL, NL and OL regions respectively.

4. SYSTEM MODEL

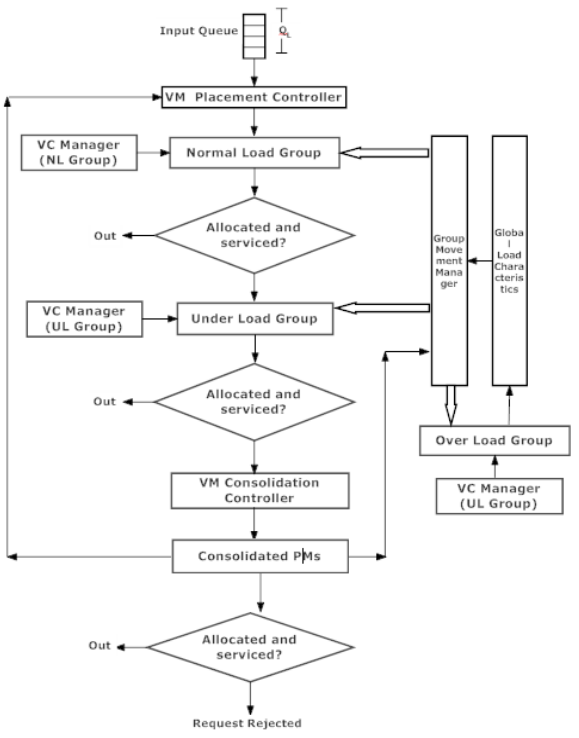

Figure1. System Model

Under this model, we categorize PMs into three groups, GUL (operating in Under Load region), GNL (operating in Normal Load region), GOL (operating in Over Load region) respectively. All VMs and PMs are homogeneous and uniform instantiation, provisioning and deployment delays are suffered by the VMs.

As the number of users in a cloud environment is quite high, the VM request arrival can be modeled as a Poisson Process with arrival rate λvm. A user submitted VM request joins the finite input queue (IQ) of length QL. A VM request is rejected if the number of VM requests in IQ equals QL. If the VM request is admitted to IQ, it waits for processing by the VM Placement Controller (VMPC). The VMPC tries to map the VM request at the head of IQ by looking up (with Look-up Delay βn) the required number of chunks in a PM belonging to GNL. If a PM cannot be found there, VMPC moves to GUL. If sufficient chunks cannot be found in GUL (with Look-up Delay βu) also, then VMPC calls for VM Consolidation Controller (VMCC) for further action. The VMCC can respond to VMPC (Response Delay (RD)) in two ways. First, it redistributes the workloads among the PMs in the group GUL, finds a PM with required VCs and reports it to the VMPC. In doing so, the VMCC may or may not initiate movement of PMs from GUL to GNL or GOL. Second, it tells the VMPC that required VCs cannot be made available. A VM request is eventually rejected in the second case. When a VM request is mapped to PM by VMPC, the VM must wait in the input queue for that PM until PM's Virtual Machine Chunk Management (VMM) module can actually deploy the required VCs. Thus the total response time (RT) for a VM request is the sum of all the delays in passing through groups as well as the Service Time ST, i.e.

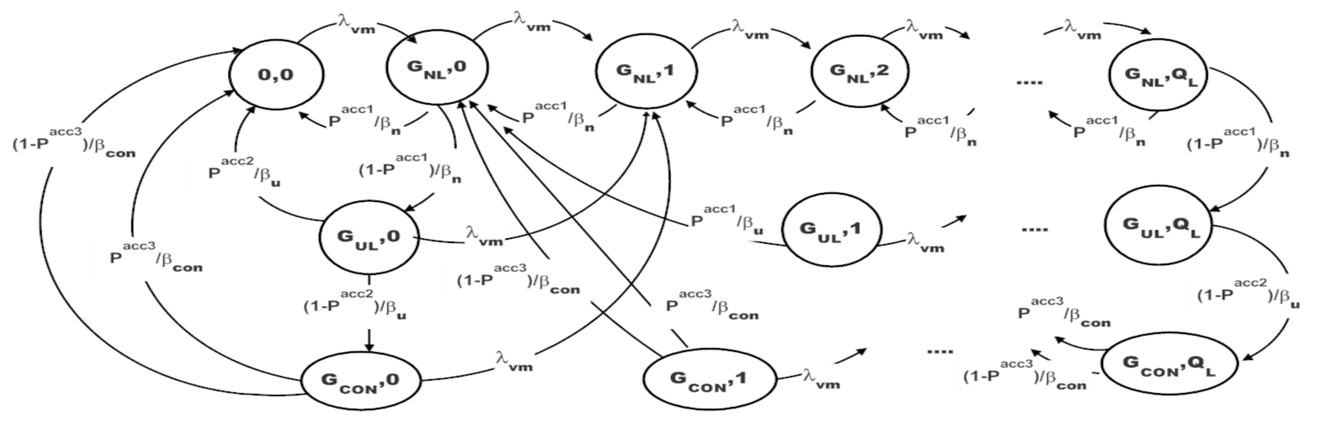

Figure 2. VM Placement Controller Sub Model

Response Time (RT) = βn + βu + RD + ST……………… (1)
When a VM finishes its assigned request, it releases the VCs held by it and the PM's operating region (OL, NL, UL) changes. Accordingly it may move to another group.
The movement of PMs in and out of groups is handled by Group Movement Controller (GMC). The GMC's prime objective is to minimize number of PMs in the groups GUL and GOL. This is desirable because overloaded PMs suffer performance degradation and under loaded PMs dissipate power. Hence GMC must ensure that the number of PMs in the group GOL must be maintained at a level 'α', where α is a design parameter which depends on traffic workload and level of performance required.
To give the mathematical formulation of the proposed system model, we give individual sub models for VMPC, VMCC, VCM and GMC. The sub models interact with each other by sharing their outputs to provide minimum task rejection and response time under overload avoidance of the PMs. We integrate the sub models to provide the overall optimum solution of VM placement under given constraints. Our integrated model is scalable to cover the dynamic nature of the cloud. It is also better than the earlier monolithic models as they are difficult to analyze and extend.

5. VM PLACEMENT CONTROLLER (VMPC) SUB MODEL

The placement of a VM request to a suitable PM is handled by VMPC as shown in the figure 2. As we mentioned above, the VM request arrival pattern is according to Poisson Process with arrival rate λ_{vm}. Therefore we model the VMPC as a continuous state Markov chain whose state space consists of 2D tuples (x, y). The attribute x shows the current group (G_{UL}, G_{NL}, G_{OL}) in which VCs are available and the attribute y shows the number of VM requests in IQ. At the initial stage the system is empty which means that neither the VM requests are being mapped to the groups nor they are waiting in IQ. It must be noted here that, initially all PMs belong to the G_{UL} group. Let us denote the first PM to be chosen for mapping VM requests by PM1. When the c1 chunks of PM1 are consumed, then PM1 joins the G_{NL} group. Thereafter, if further VM request for VCs can be accommodated on PM1 itself, then PM1 can either become overloaded after consuming c2 chunks or new PMs from G_{UL} group can be initiated to satisfy the requests. For the purpose of our experiment, we assume that G_{NL} group has at least one PM and no VM request is pending in IQ.

Under this assumption, when the '*first'* VM request arrives, the system moves to state (G_{NL}, 0) indicating that it is mapped to a PM in normal load group G_{NL}. Thereafter, depending upon the successive VM requests, the possible transitions are explained as follows.

1. A new VM request arrives at the rate of λ_{vm} and the state changes to (G_{NL}, 1).

2. The newly arrived VM request can be allocated required VCs in a PM belonging to G_{NL}, producing a transition back to the state (G_{NL}, 0).

3. There is not sufficient VCs on any of the PMs in G_{NL} group, so that VMPC checks the G_{UL} group for required VCs, producing the state change to (G_{NL}, 0).Let P^{acc1} be the probability of success of finding a PM with required number of VCs in the group G_{NL} and β_n be the average lookup delay for finding a PM in G_{NL}, then the transition to state (G_{UL}, 0) occurs at the rate of $(1- P^{acc1})/\beta_n$.

On the transition (G_{UL}, 0), the VMPC will attempt to map the request to a PM in G_{UL} group and if found(with the probability P^{acc2}), the system moves back to the state (0, 0) at the rate $(1-P^{acc2})/\beta_u$, where β_u is the average look-up delay for finding a PM in the group G_{UL}. Otherwise the VMPC invokes VMCC for a response. VMCC attempts to redistribute the workload among the PMs in G_{UL}. Redistribution of workload may (or may not) force transition of under loaded PMs to normal load region or overload region. Let G_{CON} be the group of PMs affected by VMCC. G_{CON} can be further grouped as G_{CON}(OL), G_{CON}(NL), G_{CON}(UL) respectively. The VMCC selects a suitable PM from G_{CON} (NL) or G_{CON}(UL) if possible, otherwise request is rejected due to insufficient resources.
Let P^{acc3} be the probability of success of finding a PM with required number of VCs in the group G_{CON} after consolidation and β_{con} be the average lookup delay for finding a PM in G_{CON}. If none of the PMs can accommodate VM request even after consolidation then a state transition occurs from (G_{CON}, 0) to (0, 0) at the rate of $(1-P^{acc3})/\beta_{con}$ which means that the VM request is eventually rejected.
While an accepted VM request is being mapped to a suitable PM, the VMPC will forward the waiting VM request at the head of the IQ to be considered for mapping in G_{NL} group in the same way as described above.
Above discussion brings out that the VM request can be rejected in two cases-

1) The input queue is full (number of requests waiting is Q_L). Request Rejection due to IQ being full (RR1) occurs with the probability –

Request Rejection 1=$\Pi_{(GNL, QL)} + \Pi_{(GUL, QL)} + \Pi_{(GUL, QL)}$…(2)

2)There are insufficient VCs in the PMs.

6. VIRTUAL CHUNK MANAGEMENT (VCM) SUB MODEL

The VCM sub model is responsible for allocating, servicing and releasing VCs corresponding to VM requests. The VCM sub model is described as a 2D CTMC. Each state in the state space of VCM is represented as a 2D tuple (w, z) in which w denotes the number of VM requests in the PM's queue, z denotes the number of VCs already consumed on the PM. If μ_{vm} is the service rate of each VM then the total service rate (TSR) of each PM is given as-

TSR = μ_{vm} * Number of VMs running on PM……….... (3)

or

TSR = μ_{vm} * Number of VCs consumed on the PM……. (4)

Since all PMs and VMs are homogeneous the working of VCM sub model for PMs in each of the groups is same. We give VCM sub model for the normal load group in the next section.

6(A). VCM SUB MODEL FOR NORMAL LOAD GROUP

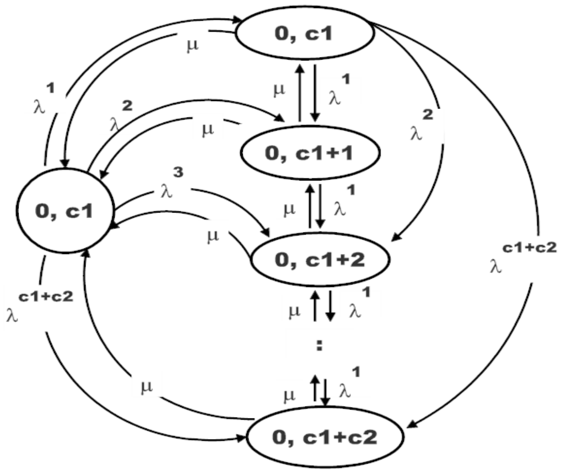

Figure 3. VCM Sub Model for Normal Load Group

The state transition diagram for VCM sub model for PMs in GNL group is shown in the figure 3. We set the queue size to c2. This avoids allocation of a new VM request beyond the normal load region. The initial state is (0, c1) which means that the PM queue is empty, c1 VCs are already consumed on the PM, and c2 VCs of the normal load region are free. The arrival rate of VM requests to PMs in GNL is λ_{GNL}. We calculate λ GNL as –

λ_{GNL}= λ_{vm}(1-RR1)/ NNL…………….(5)

Since a VM request can ask for more than one VC, we derive λ_t, the arrival rate of requests for t virtual chunks (t =1, 2,.., c2) for the group GNL as –

$\lambda_t = \lambda_{GNL} * p^t$ ………………………….(6)

Where, p^t is the probability of having a VM request for t chunks.

With these notations the state transitions in the figure are explained as follows. From the initial state (0, c1) the system can move to any of the state (0, c1+1), (0, c1+2), (0, c1+3)… (0, c1+c2) depending on the arriving VM request for 1, 2, 3, …, c1+c2 VCs respectively. State (0, c1+1) goes back to (0, c1) when VM finishes the service with service rate μ_{vm} and the corresponding VC is released. Hence with the admitted VM request the chunk utilization grows and with the completion of the service the chunk utilization the number of free chunks each state (w, z) is given as – shrinks. The steady state probabilities are given by Π (GNL, w, z) and

fc(w, z) = c2 – (w + z)………………………...(7)

The VM request will be rejected when number of VCs requested (t) are less than the number of free chunks (fc). Let U be the set of all those states (w, z) where t >fc. Then, the probability that G_{NL} will reject a VM request due to insufficient VCs in normal load group (P^{rej1}) is given by-

$P^{rej1} = \sum \Pi(_{GNL, U}) * p^t$ ……………………..…(8)

Therefore, probability that G_{NL} will accept a VM request (P^{acc1}) is given as-

P^{acc1}= 1- $(P^{rej1})^{NNL}$ …………………………. (9)

6(B). VCM SUB MODEL FOR UNDER LOAD GROUP

When a VM request is rejected from the Normal Load group, it is forwarded to Under Load group for allocation. As mentioned above, the state transitions in 2D CTMC VCM sub model for Under Load group works similar to Normal Load group. However, the starting state is (0, 0) which means that no VM requests are in the PM's queue and all VCs are free. The queue size is set to c1+c2, the service rate remains μ_{vm}.VM requests arrive at the rate of λ_{UNL} given by-

$\lambda_{UNL} = \lambda_{vm}(1\text{-RR1})(1\text{-} P^{acc1})/N_{UL}$……..(10)

Where, N_{UL} is the number of PMs in the group G_{UL}. From the steady state probabilities, $\Pi(_{UNL,\ w,\ z})$, the number VCs requested (t) and the number of VCs available (fc), in each state of VCM sub model for G_{UL} group, we compute the probability that G_{UL} will accept a VM request (P^{acc2}) as-

P^{acc2}= 1- $(P^{rej2})^{NUL}$………………………... (11)

7. VM CONSOLIDATION CONTROLLER SUB MODEL FOR CONSOLIDATED GROUP

When a VM request has been rejected by G_{UL}, the placement controller calls the VM consolidation controller to redistribute the workload among the PMs in G_{UL}. Redistribution of workload may (or may not) force transition of under loaded PMs to normal load region or overload region affecting the Global Load Characteristics of the system. The GLC are measured in terms of number of PMs in under load and overload regions respectively. If GLC are within the specified limits, the VMCC initiates VM consolidation and finds an appropriate PM for request allocation.

Let G_{CON} be the group of PMs affected by VMCC. G_{CON} can be further grouped as G_{CON}(OL), G_{CON}(NL), G_{CON}(UL) respectively. The VMCC selects a suitable PM from G_{CON}(NL) or G_{CON}(UL) if possible, otherwise request is rejected due to insufficient resources. Let the arrival rate to G_{CON} be denoted by λ_{CON} which is calculated as-

$\lambda_{CON} = \lambda_{vm}(1\text{-RR1})(1\text{-} P^{acc1})(1\text{-}P^{acc2})/N_{CON}$...........(12)

where, N_{CON} is the number of PMs affected by consolidation. The probability that G_{CON} will accept a VM request (P^{acc3}) is given as-

$$P^{acc3} = 1 - (P^{rej3})^{NCON} \qquad (13)$$

8. GROUP MOVEMENT CONTROLLER SUB MODEL

The GMC sub model is described as 2D CTMC as shown in figure 4. Each state (p, q) represents number of PMs in G_{NL} and G_{UL} respectively. If the new VM request finds allocation in the group G_{NL} then the state (p, q) does not change otherwise an under loaded PM may enter the normal load region (at rate r1) and the state changes to (p+1, q). When VCs are released from PMs in G_{NL}, p decrements and q increments (some PMs get under loaded at the rate r2).When VCs are allocated on PMs in G_{UL}, p increments and q decrements. The number of PMs in G_{OL} is A- (p + q), where A, is the total number of active PMs.

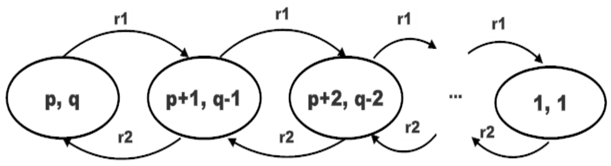

Figure 4. GMC Sub Model

The value of A serves as an important QoS metric. VMCC will not allow movement of PMs to G_{OL} if $A > \alpha$. In this way-

i. We control number of overloaded PMs in the group G_{OL}.
ii. We restrict CPU capacity utilization of overloaded PMs.

Therefore our objective of avoiding overload on PMs in cloud data centers is achieved. We summarize the discussion in the above sections and present our VM placement Algorithm, in the next section.

9. VM PLACEMENT ALGORITHM

```
Input:

    1. A set of PMs with at least one machine operating in the normal load region.
    2. A set of VM requests in terms of Virtual Chunks requirements.
    3. Initialize Size of input queue (Q_L), Maximum number of PMs allowed in Over Load group (α).
    4. For each VM request in the input queue-
    a. If (PM in Normal Load group is available with required number of Virtual Chunks)
{Allocate VCs to VM request}
Else
    b. If (PM in Under Load group is available with required number of Virtual Chunks)
{Allocate VCs to VM request}
Else
    c. If PMs in Over Load group < α, consolidate PMs and repeat the steps a, b.
Else
    d. Reject VM request.
Output:
        1. Probability of Task Rejection (P^rej2)
        2. Mean Service Time
        3. Number of PMs in each group (Normal Load, Under Load and Over Load )
```

10. SIMULATION RESULTS AND DISCUSSION

The proposed model as well as sub models is simulated using NS2 simulator. The performance of the model is studied by varying input parameters settings and collecting their influences on request rejection, service time, global load characteristics and probabilities of success of finding PMs in each of the three groups.

As can be seen from figure 5, as the arrivals of VM request increases, the service time does not degrade. Also, the number of PMs in the overload group does not exceed a given threshold value. Thus figure 5 justifies our assumptions that avoiding overload on PMs, keeps the performance of physical machines consistent.

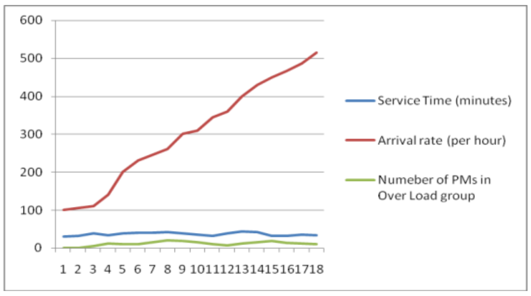

Figure 5. Effect of request arrivals on service time and number of PMs in overload group.

The request arrival does not built long queues in each of the groups and does not degrades the service time(figure 5), therefore request rejection is sufficiently low. This is depicted in figure 6. which shows that the request rejection with the same arrival rate as in the above case. The QoS parameter 'α' does not get violated.

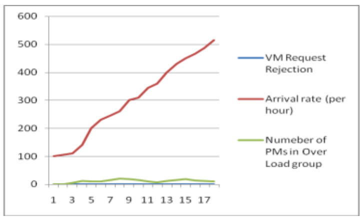

Figure 6. Effect of request arrivals on request rejection and number of PMs in overload group.

Figure 7, shows the Global Load Status of all the machines in each of the groups, under the different arrival rate as in figure 5. The number of machines in each of the groups is taken on y-axis while the x- axis represents time in hours. As can be seen from the graph, the number of PMs in normal load group is maximum at all points of time while the number of PMs in overload group is significantly small. The PMs in under load group lie intermediate between the normal load and overload groups. Thus the objective VM request placement while avoiding PMs in underload and overload groups is achieved.

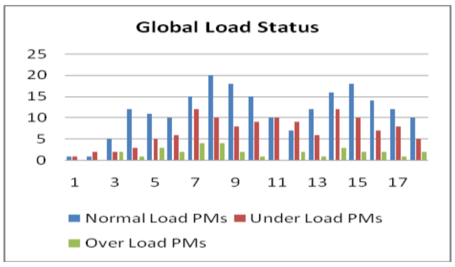

Figure 7. Number of PMs in each group(NL, UL, OL)

Figure 8, 9, 10 show the probabilities of success of finding a PM with required number of virtual chunks in the groups - GNL(normal load), GUL (under load) and GCON (consolidated PMs), respectively. Figure 8 depicts that there is a high probability of finding a PM in the normal load group while figure 10 shows that the probability of finding PM in overload group, Pacc3, is low. The probability Pacc2 (figure 9) lies intermediate to Pacc1 and Pacc3. The results show that the proposed model accurately describes global load characteristics and is successful in avoiding overload and under load conditions.

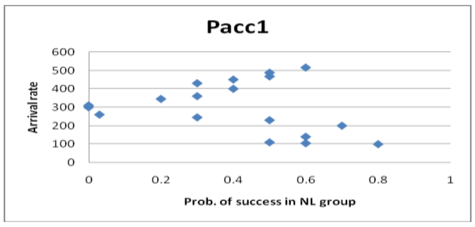

Figure 8. Effect of arrival rate on success prob. of finding VM in NL group

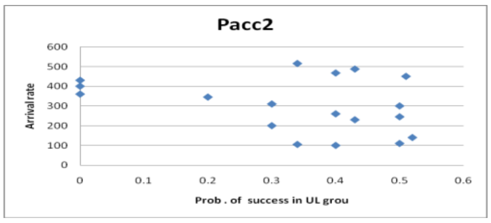

Figure 9. Effect of arrival rate on success prob. of finding - in UL group

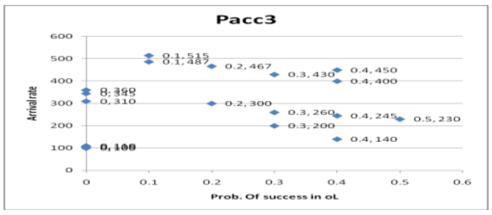

Figure 10. Effect of arrival rate on success prob. of finding VM in OL group

11. CONCLUSION AND FUTURE DIRECTIONS

In this paper we have used the Markov Chain Model to solve the problem of placing VM requests to PMs in such a way so as to minimize the number of over loaded and under loaded PMs in a cloud data centre. Our model helps to quantify CPU utilization and is therefore scalable and tractable. A work load based grouping scheme proposed by us limits the number of PMs in over load group to the value ‘α’. The parameter ‘α’ is used as QoS metric by us. This is because, as the workload on cloud data centres and hence overloaded servers increase in cloud data centres, the performance decreases due to insufficiency of resources. Under the proposed model, we calculated VM request rejection probability, response time as well service time of the incoming requests. The results of simulation confirm that as the arrival of requests grows, the service time does not degrade and request rejection is sufficiently low. Also, the number of PMs in overload group is minimum, number of PMs in normal group is maximum while number of PMs in under load group is intermediate between the two.

As a part of limitations, the model based on Markov Chains requires certain basic assumptions. Sometimes, the input workload may not satisfy Markovian property in which the future state is predicted only on the basis of the current state configurations. However, among the approaches for stationary workloads, the proposed model gives appreciable performance results and is scalable and tractable. The model is also extendible to cases where there are multiple CPU cores in cloud servers.

12. REFERENCES

[1] Anton Beloglazov, Rajkumar Buyya “Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers Under Quality of Service Constraints”, , IEEE Transactions on Parallel and Distributed Systems, vol. 24 no. 7, pp. 1366-1379, 2013.

[2] Olivier Beaumont, Lionel Eyraud-Dubois, Christopher Thraves Caro, and Hejer Rejeb, “Heterogeneous Resource Allocation under Degree Constraints” , IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 5, 2013 doi :10.1109/tpds.2012.175.

[3] Yongmin Tan, Venkatesh, V., Xiaohui Gu, "Resilient Self-Compressive Monitoring for Large-Scale Hosting Infrastructures," IEEE Transactions on Parallel and Distributed Systems, vol.24, no.3, pp.576, 586, 2013. doi: 10.1109/TPDS.2012.167

[4] Kyle Chard, Kris Bubendorfer, " High Performance Resource Allocation Strategies for Computational Economies”, IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 1, 2013, doi: 10.1109/TPDS.2012.102.

[5] Hamzeh Khazaei, Jelena M, Vojislav B. M., Saeed Rashwand ,"Analysis of a Pool Management Scheme for Cloud Computing Centers” IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 5, 2013, doi 10.1109/TPDS.2012.182.

[6] Zohar, E., Cidon, I., Mokryn, O., "PACK: Prediction-Based Cloud Bandwidth and Cost Reduction System," IEEE/ACM Transactions Networking, vol. PP, no.99, 2013, doi: 10.1109/TNET.2013.2240010.

[7] Bruneo, D., "A Stochastic Model to Investigate Data Center Performance and QoS in IaaS Cloud Computing Systems," IEEE Transactions on Parallel and Distributed Systems, vol. PP, no.99, 2013, doi: 10.1109/TPDS.2013.67.

[8] Sheng Di, Cho-Li Wang, "Dynamic Optimization of Multiattribute Resource Allocation in Self-Organizing Clouds," Parallel and Distributed Systems, IEEE Transactions on, vol.24, no.3, pp. 464,478, 2013. doi: 10.1109/TPDS.2012.144

[9] Papagianni, C., Leivadeas, A., Papavassiliou, S., Maglaris, V., Pastor, C., Monje, A., "On the Optimal Allocation of Virtual Resources in Cloud Computing Networks," IEEE Transactions on Computers, vol. PP, no.99, 2013, doi: 10.1109/TC.2013.31.

[10] Zhen Xiao, Weijia Song, and Qi Chen" Dynamic Resource Allocation using Virtual Machines for Cloud Computing Environment", IEEE Transactions on Parallel and Distributed Systems, vol. 24 no. 6, pp. 1107-1117, 2013, doi: 10.1109/TPDS.2012.283.

[11] Tiago C. Ferreto1, Marco A. S. Netto, Rodrigo N. Calheiros, and C´esar A. F. De Rose ,"Server Consolidation with Migration Control for Virtualized Data Centers" Future Generation Computer Systems, vol.28, i.8, pp.1350-1362, 2012, doi : 10.1016/j.future.2011.05.008.

[12] Fei MA, Feng LIU, Zhen LIU, "Multi-objective Optimization for Initial Virtual Machine Placement in Cloud Data Center", Journal of Information & Computational Science 9(16), 5029–5038, 2012.

[13] H. Khazaei, Jelena M, Vojislav B. M., "Performance Analysis of Cloud Computing Centers Using M/G/m/m + r Queueing Systems," IEEE Trans. Parallel and Distributed Systems, vol. 23,no. 5, pp. 936 943, 2012.

[14] Osman Sarood, Phil Miller, Ehsan Totoni, Laxmikant V. Kale "'Cool' Load Balancing for High Performance Computing Data Centers", IEEE Transactions on Computers, vol. 61, no. 12, December 2012, doi: 10.1109/TC.2012.143.

[15] Anton Beloglazov, Jemal Abawajy, Rajkumar Buyya, "Energy-aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing", Journal of Future Generation Computer Systems, vol. 28, issue **5**, pp-755–768, 2012.

[16] Anton Beloglazov, Rajkumar Buyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers", Concurrency And Computation: Practice And Experience, vol. 24 pp. 1397–1420, 2012, doi: 10.1002/cpe.1867.

[17] Bruneo, D., Distefano, S., Longo, F., Puliafito, A., Scarpa, M., "Workload-Based Software Rejuvenation in Cloud Systems," IEEE Transactions on Computers, vol. PP, issue 99, 2012. doi: 10.1109/TC.2013.30.

[18] Al-Azzoni, Douglas G. Down, "C-MART: Benchmarking the Cloud", IEEE Transactions on Parallel and Distributed Systems, vol. 12 pp. 045-9219, 2012, doi: 10.1109/TPDS.2012.335.

[19] Johan Tordsson, Rubén S. Montero, Rafael Moreno-Vozmediano, Ignacio M. Llorente, "Cloud Brokering Mechanisms For Optimized Placement Of Virtual Machines Across Multiple Providers", Future Generation Computer Systems vol. 28, pp-358–367, 2012.

[20] Sheng Di, Cho-Li Wang,"Error-tolerant Resource Allocation and Payment Minimization for Cloud System", IEEE Transactions on Parallel and Distributed Systems, vol. 28, pp-358–367, 2012, doi: 10.1109/TPDS.2012.309

[21] Hamzeh Khazaei, Jelena M.,Vojislav B.M, "Performance of Cloud Centers with High Degree of Virtualization under Batch Task Arrivals", IEEE Trans. on Parallel and Distributed systems, doi: 10.1109/TPDS.2012.318.

[22] Qian Zhu,, Gagan Agrawal, "Resource Provisioning with Budget Constraints for Adaptive Applications in Cloud Environments", IEEE Transactions on Services Computing, vol. 5, no. 4, 2012, doi: 10.1109/TSC.2011.61.

[23] Buyya, R., S.K. Garg and R.N. Calheiros, "SLA Oriented Resource Provisioning for Cloud Computing: Challenges, Architecture and Solutions", Proceedings of the International Conference on Cloud and Service Computing, IEEE, Australia, pp: 1-10, 2011.

[24] R. Ghosh, K.S. Trivedi, V.K. Naik, and D.S. Kim, "End-to-End Performability Analysis for Infrastructure-as-a-Service Cloud: An Interacting Stochastic Models Approach," Proc. IEEE 16th Pacific Rim Int'l Symp. Dependable Computing.

[25] Peter Sanders, Naveen Sivadasan, and Martin Skutella, "Online Scheduling with Bounded Migration", Journal of Mathematics of Operation Research, vol. 34 no. 2 481-498, 2009, doi: 10.1287/moor.1090.0381.

[26] S. Kumar, V. Talwar, V. Kumar, P. Ranganathan, and K. Schwan, "v Manage: Loosely coupled platform and virtualization management in data centers," in Proc. of the 6th Intl. Conf. on Autonomic Computing (ICAC), pp. 127–136, 2009.

[27] A. Verma, P. Ahuja, and A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems," in Proc. of the 9th ACM/IFIP/USENIX Intl. Conf. on Middleware, pp. 243–264, 2008.

[28] R. Nathuji, K. Schwan, "Virtual Power: Coordinated Power Management in Virtualized Enterprise Systems", ACM SIGOPS Operating Systems Review, vol. 41, no. 6, pp. 265–278, 2007.

[29] K.S. Trivedi, Probability and Statistics with Reliability, Queuing and Computer Science Applications, second ed. Wiley, 2001.

[30] L. Benini, A. Bogliolo, G. A. Paleologo, and G. D. Micheli, "Policy optimization for dynamic power management," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 18,no. 6, pp. 813–833, 1999.

AUTHORS' PROFILE

Narander Kumar received his Post Graduate Degree and Ph. D. in CS & IT, from the Department of Computer Science and Information Technology, Faculty of Engineering and Technology, M. J. P. Rohilkhand University, Bareilly, Uttar Pradesh, INDIA in 2002 and 2009, respectively. His current research interest includes Quality of Service (QoS), Computer Networks, Resource Management Mechanism, in the networks for Multimedia Applications, Performance Evaluation, Cloud Computing, Software Engineering,. Presently he is working as Assistant Professor, in the Department of Computer Science, Babasaheb Bhimrao Ambedkar University (A Central University), Lucknow, UP, INDIA.

Shalini Agarwal received her Master of Technology (M. Tech) in Computer Science and Engineering from Amity University, Uttar Pradesh, India in 2010. She also did her M.Sc (Computer Science) at J. K. Institute of Applied Physics and Technology, University of Allahabad, Allahabad, UP, India. She was a faculty member in the Department of Computer Science and Engineering, Sri Ram Swaroop Group of Professional Colleges, Lucknow, UP, India. Currently, she is a research scholar in the department of Computer Science at Babasaheb Bhimrao Ambedkar University (A Central University), Lucknow, UP. India. Her research area is computer networking and resource management for grid and cloud computing infrastructures.

Vipin Saxena is a Professor and Head, Department of Computer Science, Babasaheb Bhimrao Ambedkar University, Lucknow, India. He got his M.Phil. Degree in Computer Application in 1992 & Ph.D. Degree work on Scientific Computing from University of Roorkee (renamed as Indian Institute of Technology, Roorkee, India) in 1997. He has more than 17 years of teaching experience and 20 years of research experience in the field of Scientific Computing & Software Engineering. He has published more than hundred International and National research papers and authored four books in the Computer Science field. Dr. Saxena is a life time member of Indian Science Congress