IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-3, Issue-5 E-ISSN: 2347-2693

Hint Based Virtual Machine Placement in Cloud

Pankaj Kadam[1*] and Nilesh Alone[2]

[1*,2]*Department of Computer Engineering, R H Sapat College of Engineering Nashik Pune University, India*

www.ijcseonline.org

Received: Apr /25/2015 Revised: May/07/2015 Accepted: May/19/2015 Published: May/30/ 2015

***Abstract*—** the capacity of the cloud is allowing users to deploy complex infrastructure on the cloud due to huge pool of the resources offered by the cloud. In the infrastructure as service virtual machines are provided to the end users as the mean of the resource from the cloud. Complex application such as load balancing system or cluster system requires multiple virtual machines and their special connectivity for system realization. Current cloud providers do not allow users to deploy their complex workload. Users will need to deploy these virtual machines manually by selecting one at a time. In this paper we are proposing hint based virtual placement which accepts hints from the users and select the nodes from cloud. This system act as middleware between cloud provider and users and automate the complex workload deployment process. This system mutually understands the constraints placed by both users and cloud admins while deploying virtual machines. Such complex virtual machine deployment can be passed to the system using constraint like AssociateVM, DistantVM, BackupVM and PrivateNetVM. Cloud admins can also place constraints like out of service, backup node etc to influence the virtual machine placement. Our virtual machine scheduler considers these constraints from both the ends and selects the appropriate nodes on the basis of the constraints and scores assigned to the nodes. We have compared this strategy with available known cloud scheduler algorithms and found that available strategies do not allow any user passed hints while our algorithm allow users to pass such hints while deploying virtual machines.

***Keywords*—** Public Cloud; Virtual Machine Placement; Resource allocation; User Hints

I. INTRODUCTION

Cloud computing allows to access huge pool of resources to the end users. Most of the cloud providers are using the pay per use cloud model which ease of the burden of investing huge amount of money in building the infrastructure. This converts the capital cost to the operating cost which is way more less than the capital investment which is required otherwise. Many new startup companies are using public cloud as their launch pad due to cost model of the cloud. In infrastructure as service cloud providers provision the resources in terms of virtual machines which can be used by the users to run their custom application on those servers. Clouds can be categorized in public, private and hybrid type of the cloud.

Resource allocation is crucial part in the cloud and it is seen as mechanism which maps end users virtual machine requests to the public or private cloud service provider's resources. It also calculates the present state of the cloud resources and requested resources by the users. In Infrastructure as service cloud resources are provided in the form of virtual machines and resource allocation manages the placement of the virtual machines. Resource allocation can be divided into two stages, in first phase the user requests analysis is done and second phase analyses the current status of present resources in the cloud. This is handled using various strategies and algorithms and these algorithms place the virtual machines on the nodes.

II. BACKGROUND

A. *Resource Allocation*

In infrastructure as service various client requested virtual machines are created on the nodes. This process is also called as resource allocation or virtual machine placement algorithms in cloud. There are various types of such algorithms available and mainly vary due to their goals or constraints. These goals can be mainly categorized into power consumption based goals and quality of service goals. In power consumption based algorithm main goal is to reduce the power consumption by reducing number of nodes and overall hardware required to build the infrastructure. This also includes using the available infrastructure efficiently and optimize for the future needs so future resource requests can be fulfilled. Algorithms which are based on the quality of service approach focus mainly on the quality of resources delivered. Critical applications these days require quality resources to deliver the good performance and in such cases the power based approach can hamper performance of the applications by packing them in less number of nodes. QoS based approach deliver the user requested resources and considers their requirements too. This ensures that users are getting what they have opt for and meets the application requirements.

B. *Static vs Dynamic Resource Allocation*

VM placements can be further classified in static approach and dynamic approach [2]. In static approach VM placement

is assessed at the time of the user request and generally stays fixed afterwards while dynamic strategies can change virtual machine placements by triggering certain events. These events include host node spikes, any maintenance mode on the host and manual migration. The DRS system of VMware is best example of dynamic approach which assesses the load status of each node and then automatically migrate virtual machines from one node to another.

C. *Stochastic Integer Programming*

Stochastic integer programming can be used to implement virtual machine placement algorithms and strategies [3]. These algorithms breaks the resource allocation in the three phases namely reservation which reserve resources, utilization which try to utilized the reserved resources and on demand also known as spot instance. Reservation phase reserve the resource before the user virtual machine requests, utilization used the reserved resources and on demand can meet the extra resource requirements from the clients. On demand request can be in form of virtual machine resizing requests from the users.

D. *Bin Packing Approach*

Bin packing is oldest strategy which can also be used to realize the resource allocation problem [4]. In this approach available compute nodes are considered as bins of variable sizes and virtual machine requests are considered as objects. The goal of this algorithm is to simply pack those objects in the available bins. It simply packs requested and available virtual machines in the tightly fashion so that resource allocation problem can be satisfied with less number of nodes. To further improve resources utilization one can sort the virtual machine according to their sizes and then place them on the compute nodes for better utilization [6]. There are various variants of the bin packing algorithm which are known as first fit, next fit and best fit. Grouping Genetic algorithm can also be used to solve this resource allocation problem. Grouping Genetic algorithm finds the bin which can be utilized to its highest extend also can satisfy some additional constraints [5].

E. *Open Source Virtual Machine Placement Approach*

There are different open source cloud suites available in the market which also uses different types of resource allocation algorithms to assign the virtual machines on the appropriate hosting nodes. Open Nebula which is open source cloud suite use predefined ranks and score to priorities the nodes. Based on these priorities, nodes which have highest priorities and score are chosen and virtual machines are assigned on those nodes. User's virtual machine requirements and computed ranks are used as input for this algorithm and according to these ranks virtual machine placement is decided. Nimbus which is another open source cloud suite makes use of Static-greedy and round-robin resource selection algorithm. Round robin algorithm analyzes available resources as a connected link list and start from the last element where it had left in the previous allocation. This approach is random because it does not consider any power savings or user workflow requirements.

All above studied strategies only considers user resource requirements and place the virtual machines according to cloud administrator configuration or strategies. These algorithms consider virtual machines as object and do not care about the dependencies of the application which are running inside the virtual machines or association and distinctiveness of virtual machines with each other. Only users are aware about their applications running inside the virtual machines. Say if any user is using the 6 virtual machines for apache and mysql cluster then only users can understand the purpose of each VM but cloud provider will look them as objects and can move them or rearrange them according to their cloud scheduler algorithms. This can cause degradation of quality of service which is not expected by the end users of the cloud as it will affect the application performance. If user wants to deploy such complex associated virtual machines in current cloud then they may need to do this manually by requesting one virtual machine at a single time which can be pretty time consuming as you will need to check available specifications and features of each node and then match it with your requirements. Both the cloud provider and cloud consumer have their knowledge but as they are applying it separately this is not ending up in a fruitful way. Suppose any particular virtual machine is acting as a network bridge between the virtual machine deployed then that virtual machine should be placed on the node which has the backup facility enabled so that it can be restored using previous backup.

To tackle this problem we are proposing a system which can consider these signals from users while deploying the set of virtual machines in the clouds. These signals can also be called as user data or meta data associated with any virtual machine. Users can provide this data while requesting resources in the cloud. The proposed scheduler will consider these signals received from the users and the information available about the compute nodes. This will mutually understand the requirements of both the cloud provider and end users hence we have named it as mutual cloud scheduler. Considering the user requirements and compute node information mutual cloud scheduler will able to map the requested virtual machines to available physical nodes in effective fashion.

III. HINT BASED CLOUD SCHEDULING

Cloud users or clients can enter their workload information in the provided interface with help of any modern browser. Workload information generally includes the number of virtual servers required, their association with

each other, resources required for each virtual machine and the operating system or application which will be running on these virtual machines. This workload information is then forwarded to the Mutual cloud scheduler to create the possible mapping of these virtual machines to the compute nodes. Mutual cloud scheduler can work with both private cloud and public cloud. In the public cloud the information like node information and its features can be fetched using their API. The public cloud provider can return below information:

Node Information: The retrieved information from the API will includes node name, node ID, sizes of the VM it can hold and features which are available on that node.

Virtual Machine Information: API also provides the information retrieved about the virtual machines, this information includes hostname of the virtual machine, IP address of the virtual machine and size of the virtual machine.

Administrative constraints: Public cloud providers can also return the administrative constraints or features of the particular nodes which can help algorithm to select the proper nodes.

Let $N = \{N_1, N_2, N_3, \ldots N_n\}$

be the set of all the nodes available in our infrastructure or present with the cloud providers. Now suppose any client or end user requested some set of virtual machines

Let $V = \{V_1, V_2, V_3, \ldots V_n\}$

There can be various kind of constraints either placed by cloud provider or placed by user while requesting the virtual machine can be considered as set of constraints say

$C = \{C_1, C_2, C_3, \ldots C_n\}$

Here system will need to map the virtual machine to hosts as $V \rightarrow H$ such that all the constraints are satisfied as well as resource consumption is also optimal.
When there is any workload information entered by user then that information is processed by the mutual cloud scheduler. Mutual cloud scheduler checks the available nodes for each virtual machine which is requested by the client via the workload information. Then each virtual machine is mapped to the compute nodes which are called as deployment profiles. Deployment profile M is mapping between virtual machine to compute node.

Deployment Profile m is $V \rightarrow H$

Mutual cloud scheduler then find the satisfaction score of each compute nodes by considering the constraints entered by the users and constraints placed by the cloud admin. These constrains are stored in array say C which holds all the constraints. System calculates the score of each deployment profile and determines the profile with the highest score. Score is calculated on the basis of constraints placed by both users and cloud administrator. The score function can be realized by using:

Score (m) $= \sum w*c$

Where c denotes the set of constraints
And w denotes the predefined weight

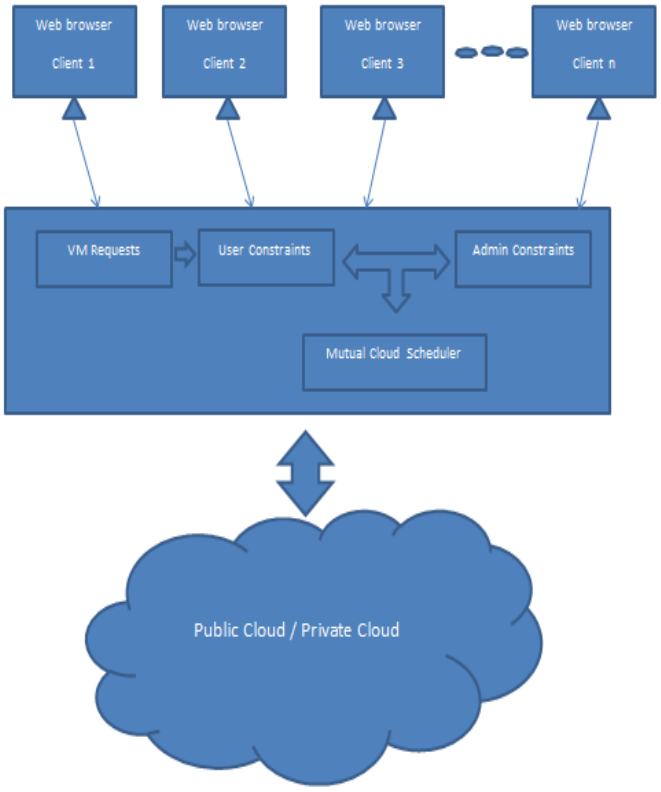

Figure 1 Proposed Framework

These scores are calculated for all the deployment profiles. The profile with the highest score is selected as the optimal profile for the VM deployment. Then this information is passed to the cloud provider for actual VM deployment using the cloud API. This process can be repeated for all the virtual machines requested by the user. While deploying the second virtual machine system will also consider the constraints placed for these two virtual machines.

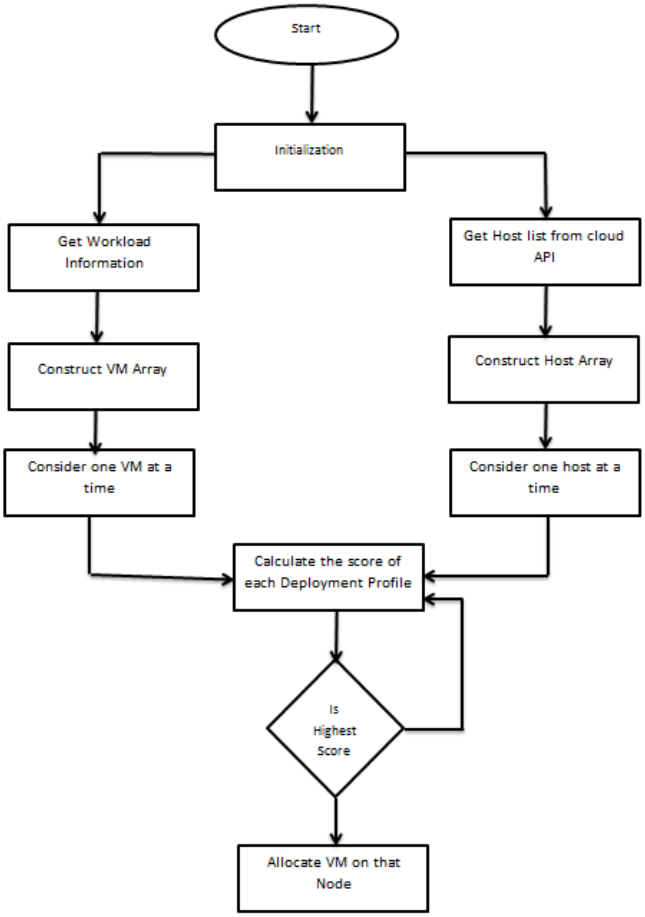

Figure 2 Flowchart of Proposed System

A. Psuedo Algorithm:

Algorithm 1: VM array and Host array Formation

Input: Workload information from client and node information from cloud API
Output: VM array with related constraints and Compute node array

1. N = number of virtual machines give in the workload
2. VM(i) represent the virtual machine where i ranges from 1 to N
3. Store the information entered by the client in form of the array VMarray[N]
4. Get the constraints entered by the user and save them in respective virtual machine entries in the VMarray[N]
5. Fetch the available nodes from the cloud provider and store the available numbers of nodes in Cn.
6. Fetch each compute nodes information and store them in Nodearray[Cn]
7. Return VMarray[] and Nodearray[]

Algorithm 2: Constraint satisfaction score:

Input: VMarray[] and Nodearray[]
Output: Constraints satisfaction Score for each VM to node mapping

1. Get VMid from VMarray[]
2. Get Nodeid from Nodearray[]
3. For each VMid to Nodeid mapping do;
4. Check constraints entered for VMid in the VMarray and match them with features of Nodeid
5. For each constrains from the from VMarray[] do
6. If VMid.constraint(i) == Nodeid.feature(i)
7. Then satisfaction + = 1
8. End if
9. VMid.constraint(i+1) == Nodeid.feature(i+1)
10. Then satisfaction +=1
11. End if
12. Return Satisfaction for each VMid to Nodeid mapping

Algorithm 3: Find optimal deployment profiles on the basis of satisfaction scores

Input: Satisfaction for each VMid to Nodeid mapping
Output: Vmid to Nodeid mapping which have highest score.

1. Compare the score of each Vmid to nodeid mapping
2. maximum = array[0]
3. for (c = 1; c < size; c++)
4. if (array[c] > maximum)
5. maximum = array[c];
6. Return the nodeid for particular VMid with max score.

B. Psuedo Algorithm:

Our proposed system will help users to deploy their complex workflows in the cloud in more efficient and more effective way also this will allow cloud providers to prevent over expose of the cloud internals. This simply place the requested virtual machines on the nodes according to the users requirements or constraints placed in the workload but also considers the specification of compute nodes so that resource consumption can be optimal. These task flows can be realized with the help of constraints or interrelation placed by the end user. Our system allows following constraints which can be entered by the users while entering the workload information:

AssociateVM: Allocate the same node for the VM's which have AssociateVM constraints placed by the users.

DistantVM: Allocate the different nodes for the VM's which have DistantVM constraints placed by the users.

PrivatenetVM: Enable the private networking in the VM's which have PrivatenetVM constraints placed by the users.

BackupVM: Allocate the virtual machine on the compute node which have backup enabled for the data redundancy.

These users placed constraints can help to realize any virtual machine work flow entered by cloud users. This will help users to communicate the infrastructure needs more effectively even in a public cloud provider. Suppose user want to create high availability apache cluster then user will need to make sure that these virtual machines are deployed on different nodes so that high availability can be achieved. In proposed system user can enter the virtual machines with the DistantVM constraints so that requested virtual machines can be deployed on the separate nodes.

Cloud administrators can control the cloud scheduler by placing some cloud administrative constraints which can priorities some nodes or can rule out some nodes from the process if there is any maintenance activity is going on. This gives cloud admin control over proposed mutual cloud scheduler and allows control in this automatic process. Cloud administrators can place below constraints on the nodes:

EvenNode: Distribute VMs on the compute nodes.

PowersaveNode: Cloud admin can place this constraint on compute node on which no new virtual machines will be setup.

MaintainanceNode: Cloud admin can place this constraint on the node if there is any maintenance going on the node so that VM will not be allocated on that node.

BackupNode: Cloud admin can mark the node which have backup facility available so while deploying virtual machine

IV. EVALUATION

We can implement this system with both the public and private cloud. Most of the public cloud providers have their own restful API system which allows integration of our code onto their system while in the private cloud we can calls direct hypervisor calls to allocate VM on a particular node.

We have implemented this system on a public cloud provider. We have used the API provided by them and integrated into our control panel. Our control panel allows users to enter the workload information in terms of the number of virtual machines and their inter relationships. This allows users to communicate their workload information more effectively even in a public cloud. The information which can be retrieved from public cloud API includes:

- Create a new virtual machine
- Fetch the available nodes Information
- Fetch the features of the nodes
- Fetch the constraints placed by public cloud provide on the compute nodes
- Destroy the virtual machine

Our proposed system will allow users to communicate their workload information with a simple control panel. Users can deploy set of virtual machines or workflow in the public cloud via single interface. Consider that user wants to build a load balancer + cache + replication infrastructure in the public cloud which involves set of virtual machines which are associated with each other.

This typical system will look like something as Figure 3

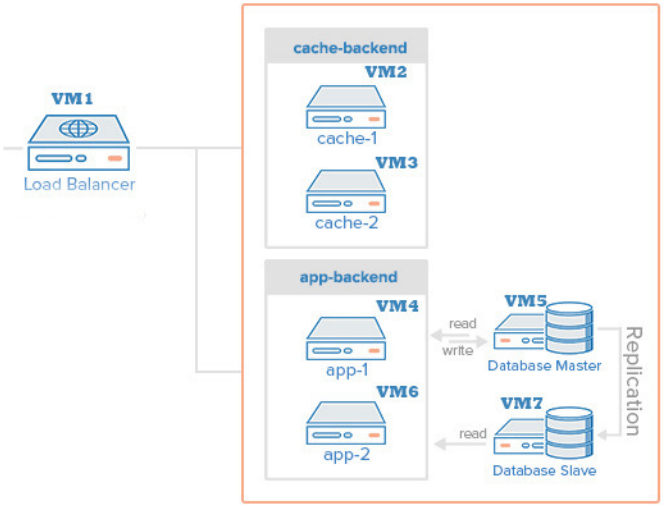

Figure 3 Sample User Requirements

To deploy these user requirements in our proposed system user can model this information in terms of number of virtual machines, and there interrelation. To realize this system user can request 7 virtual machines at a time with below constraints in-between:

VM2 and VM3 : DistantVM
VM4 and VM5 : PrivatenetVM
VM6 and VM7: PrivatenetVM

User can enter the above information from the control panel and proposed system will allocate these virtual machines on the compute nodes in such way that these constraints are satisfied as we all resource consumption on the nodes stays optimal too. They can also do this manually but that will involve a lot of work as users will need to check if these

features are available or not on a particular node then will manually select the node and will need to manually deploy the virtual machines. Our system saves users from all this hassle and quickly deployed such workloads in the cloud and also preserves the virtual machines interrelations

V. RESULT

To evaluate the proposed system we have compared the VM allocation patterns of our proposed system to some existing virtual machine placement algorithms. In the given sample user requirement user has requested below 7 virtual machines refer figure 3.

We have modeled various placement strategies using CloudSim simulator to test the performance of our proposed system. We have created hosts and virtual machine requests using the CloudSim API and calculated the results.

Compute Nodes Available:

Table 1: Size of compute nodes

Compute Node	Memory
C1	16 GB
C2	32 GB
C3	32 GB
C4	16 GB
C5	12 GB

Virtual Machines Requested:

Table 2: Size of virtual machines

Virtual Machine	Memory
VM1	8 GB
VM2	2 GB
VM3	2 GB
VM4	4 GB
VM5	4 GB
VM6	6 GB
VM7	6 GB

VM to host Mapping:

Table 3: VM to compute node mapping

Virtual Machines	First Fit algorithm	Round Robin	Ranking	Mutual Scheduler
VM1	C1	C1	C2	C3
VM2	C1	C2	C3	C2
VM3	C1	C3	C3	C3
VM4	C1	C4	C3	C4
VM5	C2	C5	C2	C4
VM6	C2	C1	C3	C2
VM7	C2	C2	C2	C2

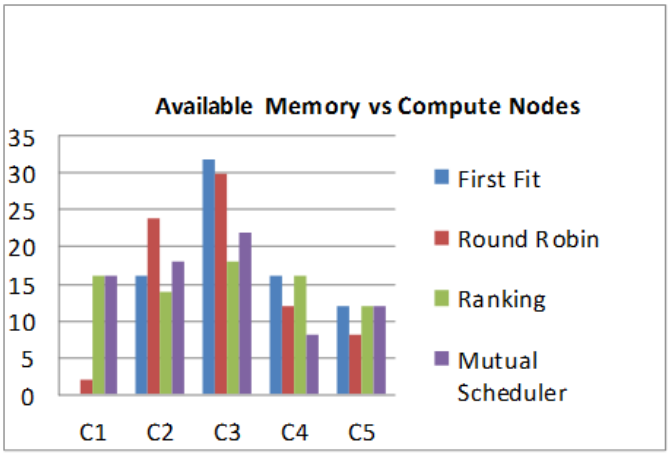

Figure 4 Experimental Result Free Memory vs Host

Figure 4 shows the free memory available on compute nodes after allocation of virtual machines using different allocation strategies. The experiment shows that our algorithm considers user entered constraints i.e. place virtual machine 2 and virtual machine 3 on distinct nodes, virtual machine 4 and virtual machine 5 on the same node for private networking and same for virtual machine 6 and virtual machine 7. Mutual schedulers also utilized all the nodes efficiently while first fit strategy and ranking algorithm utilized certain nodes to full extend while others are kept completely empty. Other algorithms also affect the application performance as they do not consider any virtual machine placement requirements passed by users.

VI. CONCLUSION

Public cloud is getting popular these days due to its costing models and scalability. Virtual machines are quickly deployed in the cloud as per user's requirements but sometimes users are not able to communicate their complex needs to public cloud providers due to many abstractions layers. In this paper we have proposed a system called mutual cloud scheduler which allows end users to pass various VM relations so that their requested virtual machines can be placed on the nodes according to their needs. Users can influence the placement strategy used by the cloud provider to deploy custom layout of the virtual machines. Mutual cloud scheduler also considers node side specifications while placing the virtual machine which results in efficient consumption of the nodes resources. We have compared the proposed strategy with existing strategies like first fit, round robin and ranking algorithm

and found that proposed scheme efficiently consumed resources and also satisfy users custom virtual machine layout requirements too. In future we can implement this system with multiple clouds API to realize the power of federated clouds.

ACKNOWLEDGEMENT

We sincerely thanks to all the peoples who have helped to prepare this research paper. We also thanks all the authors of the reference materials we have used and communities which maintains and keep running the open source software's which we have used in our work.

REFERENCES

[1] Konstantinos Tsakalozos, Mema Roussopoulos, and Alex Delis "Hint-Based Execution of Workloads in Clouds with Nefeli" IEEE transactions on parallel and ditributed systems, vol. 24, no. 7, July 2013

[2] Bobroff, N. ; T.J. Watson Res. Center, IBM, Hawthorne, NY ; Kochut, A. ; Beaty, K. "Dynamic Placement of Virtual Machines for Managing SLA Violations" Integrated Network Management, 2007. IM '07. 10th IFIP/IEEE

[3] Chaisiri, S. ; Sch. of Comput. Eng., Nanyang Technol. Univ., Singapore, Singapore ; Bu-Sung Lee ; Niyato, D. "Optimal virtual machine placement across multiple cloud providers" Services Computing Conference, 2009

[4] Remesh Babu, K.R. ; Dept. of Inf. Technol., Gov. Eng. Coll., Idukki, India ; Samuel, P. "Virtual Machine Placement for Improved Quality in IaaS Cloud" Advances in Computing and Communications (ICACC), 2014

[5] Jamali, S. ; Dept. of Electr. & Comput. Eng., Univ. of Mohaghegh Ardebili, Ardebil, Iran ; Malektaji, S. "Improving grouping genetic algorithm for virtual machine placement in cloud data centers" Computer and Knowledge Engineering (ICCKE), 2014

[6] Ricardo Stegh Camati, Alcides Calsavara, Luiz Lima Jr. "Solving the Virtual Machine Placement Problem as a Multiple Multidimensional Knapsack Problem" ICN 2014 : The Thirteenth International Conference on Networks

[7] Nskinc study on cloud computing, Nskinc white paper 2012

[8] Linlin Wu ; Dept. of Comput. & Inf. Syst., Univ. of Melbourne, Melbourne, VIC, Australia ; Garg, S.K. ; Versteeg, S. ; Buyya, R. "SLA-Based Resource Provisioning for Hosted Software-as-a-Service Applications in Cloud Computing Environments" Services Computing, IEEE Transactions on (Volume:7 , Issue: 3)

[9] Dong Huang ; Inst. for Infocomm Res., Agency for Sci. Technol. & Res, Singapore, Singapore ; Bingsheng He ; Chunyan Miao "A Survey of Resource Management in Multi-Tier Web Applications

[10] Pankaj R Kadam, Nilesh V Alone "Review on KVM Hypervisor" International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-3 Issue-4 September, 2014

[11] Mell, P. and Grance, T. 2011. "The NIST Definition of Cloud Computing" NIST Special Publication, 800-145

AUTHORS PROFILE

Pankaj R Kadam received his BE degree from Pune University. He is currently pursuing his master degree in computer science. He also has 3 Years of working experience in the cloud and worked on the various cloud technologies. He is currently working with a private firm and managing the cloud servers for the clients

Nilesh V Alone is currently working as Assistant Professor in Department of Computer Engineering, R. H. Sapat College of Engineering, Management Studies & Research, Nashik. He is having 13 years of experience in teaching. His area of interest is in Cloud Computing and Machine Learning.