



Performance Evaluation of Server Consolidation Algorithms in Virtualized Cloud Environment with Constant Load

Susheel Thakur*, Arvind Kalia, Jawahar Thakur
Dept. of CS, HP University, Shimla
India

Abstract— Server Consolidation is an approach towards the efficient usage of computer server resources in order to reduce the total number of servers or server locations that an organization requires. To prevent the problem of server sprawl, server consolidation aims at reducing the number of server machines used in the data centers by consolidating load and enhancing resource utilization of physical systems. Server consolidation through live migration is an efficient way towards energy conservation in cloud data centers. Virtualization provides the ability to migrate virtual machines (VM) between the physical systems using the technique of live migration mainly for improving the efficiency. Live machine migration transfers “state” of a virtual machine from one physical machine to another, and can mitigate overload conditions. Although a lot of literature exists on server consolidation, a range of cases involved have mostly been presented in isolation of each other. The aim of this research paper is to present the details of the server consolidation algorithms over a common software framework and their usage toward dynamic resource management in the virtualized cloud environment. Here, we present a performance analysis of these heuristics and fundamental insights obtained when constant load is generated over the servers, aimed at reducing server sprawl, reducing power consumption and load balancing across physical servers in cloud data centers.

Keywords— Cloud computing, live migration, Load balancing, Server Sprawl, Virtual Machine Monitor (VMM).

I. INTRODUCTION

With the rapid development in the processing and storage technologies and the success of the internet, computing resources have become reasonable, powerful and globally available than ever before. Personnel in businesses are trying to find out methods to cut costs while maintaining the same performance standards. Their aspirations to grow have led them to try new ideas and methods even under the peer pressure of limiting computing resources. This realization has enabled the actualization of a new model for computing called cloud computing, in which the resources (e.g. cpu, n/w, etc.) are provided through the internet to user as general utilities in a pay-as-you-go and on-demand basis [18].

For simplicity, a cloud is a pool of physical computing resources i.e. a set of hardware, processors, memory, storage, networks, etc which can be provisioned on demand into services that can grow or shrink in real-time scenario[21].

Cloud Computing is defined by NIST[15] as a model for enabling convenient, on demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. Virtualization plays a vital role for managing and coordinating the access from the resource pool. A virtualized environment that enables the configuration of systems (i.e. compute power, bandwidth and storage) as well as the creation of individual virtual machines is the key features of the cloud computing. Virtualization is ideal for delivering cloud services. Virtualization Technology enables the decoupling of the application payload from the underlying physical hardware and provides virtualized resources for higher-level applications. Virtualization can provide remarkable benefits in cloud computing by enabling VM migration to balance load across the data centers [11].

In the surge of rapid usage of virtualization, migration procedure has been enhanced due to the advantages of live migration say server consolidation and resource isolation. Live migration of virtual machines [5, 13] is a technique in which the virtual machine seems to be active and give responses to end users all time during migration process. Live migration facilitates energy efficiency, online maintenance and load balancing [12]. Live migration helps to optimize the efficient utilization of available cpu resources.

Server consolidation is an approach to the efficient usage of computer server resources in order to reduce the total number of servers or server locations that an organization requires. This approach was developed in response to the problem of “server sprawl”. Server sprawl is a situation in which multiple underutilized servers accommodate more space and consume more resources that can be justified by their workload. Many organizations are turning to server consolidation to reduce infrastructure complexity, improve system availability and save money. With increasingly powerful computing hardware, including multi-core servers; organizations can run large workloads and more applications on few servers. Reducing the numbers of servers has tangible benefits for the data center as well.

Server Consolidation can be categorized into:

1. Location Consolidation-The number of physical locations containing hardware is reduced.
2. Physical Consolidation-The number of physical hardware is reduced.

3. Rationalized Consolidation—implementing multiple applications on fewer, more powerful problems, usually through workload management and partitioning.

Workload management describes a set of techniques which enable different applications to run together on a single instance of an OS. The aim of this technique was to balance the resource demands that each of the applications places on the system so that all of them co-exist. Partitioning involves the division of a server, which might ordinarily run on a single instance of an OS, into smaller systems each of which can run its own copy of an OS. In short, workload management allows several applications to run on one OS, partitioning allows several OS to run on one machine.

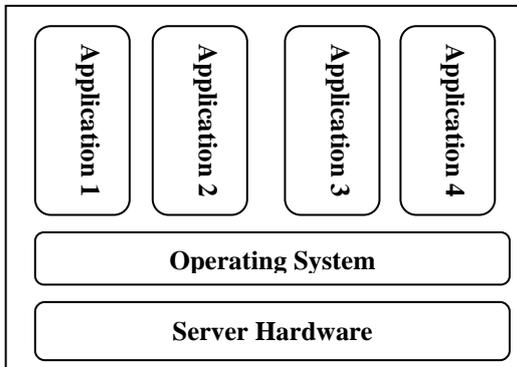


Fig.1.1 Workload Management

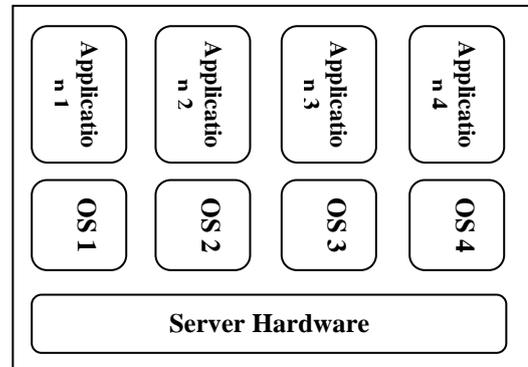


Fig. 1.2 Partitioning

Consolidation will result in reduced power consumption and thus reducing overall operational costs for data center administrators. Live migrations achieve this. Based on the load conditions, under-utilized machines having resource usage above a certain threshold are identified and migrations are triggered to tightly pack VMs to increase overall resource usage on all PMs and free up resources/PMs if possible[4], [18].

The rest of this paper is organized as follows: Section II describes the survey of existing literature of various server consolidation algorithms for cloud computing environment. Section III provides the overview of the chosen server consolidation algorithms for performance analysis in virtualized cloud environment. Section IV gives details about the experimental test bed used in performance analysis when a constant load is generated. Section V discusses the experimental evaluation and results. Section VI provides the conclusion and future work.

II. LITERATURE SURVEY

Server Consolidation through Live migration is a vital technology in modern cloud data centers to achieve load balancing, online maintenance and reducing energy consumption. Many efforts have been made towards server consolidation in academia and industry. Sandpiper [19] had their own centralized way of monitoring, profiling and detecting hotspots on a prototype data center consisting of twenty 2.4 GHz Pentium-4 servers connected over a gigabit Ethernet with control plane and nucleus, Khanna et al.[10] had their test-bed of IBM blade center environment with VMWare ESX server as the hypervisor deployed on three HS-20 Blades, Aameek et al.[2] used HARMONY set-up with ESX server and SAN (Storage Area Network) controller integrating both server and storage virtualization technologies to design an agile data center. Memory buddies [20] collocated similar VMs exploiting page sharing to mitigate hotspots. They too had a control plane with nucleus on the PMs like Sandpiper [19] performing initial VM placement, server consolidation and offline capacity planning using VMware ESX hypervisor. Their testbed is a cluster of P4 2.4 GHz servers connected over gigabit Ethernet. Emmanuel et al. [6] proposed a dynamic resource allocation framework based on their load balancing VM migration algorithm on a test-bed of three ESX servers, a SAN, VMware VC (Virtual Center) and VIBM Migration Handler. Autocontrol [14] proposed an automatic resource control system using two test-beds for evaluation, first consisting of HP C-class blades each equipped with two dual-core 2.2 GHz 64 bit processors and second Emulab-consisting of pc3000 nodes with single Xeon 3 GHz 64 bit processor as servers and pc850 nodes as clients. Similarly, Verma et al. [1] presented the pMapper architecture and a set of server consolidation algorithms for heterogeneous virtualized resources. The algorithms take into account power and migration costs and the performance benefit when consolidating applications into physical servers. Jung et al. [8], [9] have investigated the problem of dynamic consolidation of VMs running a multi-tier web-application using live migration, while meeting SLA requirements. The SLA requirements are modelled as the response time precomputed for each type of transactions specific to the web-application. A new VM placement is produced using bin packing and gradient search techniques. The migration controller decides whether there is a reconfiguration that is effective according to the utility function that accounts for the SLA fulfilment. However, this approach can be applied only to a single web-application setup and, therefore, cannot be utilized for a multitenant IaaS environment. Zhu et al [22] have studied a similar problem of automated resource allocation and capacity planning. They have proposed three individual controllers each operating at a different time scale: longest time scale (hours to days); shorter time scale (minutes); and shortest time scale (seconds). These three controllers place compatible workloads onto groups of servers, react to changing conditions by reallocating VMs, and allocate resources to VMs within the servers to satisfy the SLAs. Such different kinds of algorithms considering different parameters and set-up performing different jobs obviates the need to perform a structured analysis to find out which algorithm should be used when, which performs better and under what cases etc.

III. SERVER CONSOLIDATION ALGORITHMS

Server consolidation is an approach to the efficient usage of computer server resources in order to reduce the total number of servers or server locations that an organization requires. This approach was developed in response to the problem of “server sprawl”. In order to reduce server sprawl in the data centers, server consolidation algorithms are implemented. These algorithms are VM packing heuristics which try to pack as many VMs as possible on the physical machine (PM) so that resource usage is improved and under-utilized machines can be turned off.

A. Sandpiper

Sandpiper is a system that automates the task of monitoring and detecting hotspots, determining a new mapping of physical resources to virtual resources, by resizing or migrating VM’s to eliminate the hotspots. Sandpiper makes use of automated black-box and gray box strategies for virtual machine provisioning in cloud data centers. Specifically the black-box strategy can make decisions by simply observing each virtual machine from the outside and without any knowledge of the application resident within each VM. The authors present a gray-box approach that assumes access to OS-level statistics in addition to external observations to better inform the provisioning algorithm. Sandpiper implements a hotspot detection algorithm that determines when to resize or migrate virtual machines, and a hotspot migration algorithm that determines what and where to migrate and how many resources to allocate. The hotspot detection component employs a monitoring and profiling engine that gathers usage statistics on various virtual and physical servers and constructs profiles of resource usage. These profiles are used in conjunction with prediction techniques to detect hotspots in the system. Upon detection, Sandpiper grants additional resources to overloaded servers if available. If necessary, Sandpiper’s migration is invoked for further hotspot mitigation. The migration manager employs provisioning techniques to determine the resource needs of overloaded VMs to underloaded servers.

Sandpiper supports both black-box and gray-box monitoring techniques that are combined with profile generation tools to detect hotspots and predict VM Resource requirements. Hotspots are detected when CPU usage values are violated with respect to the CPU thresholds set. Physical machines (PMs) are classified as underloaded or overloaded. The PMs are sorted based on the descending order of their volume metric, and VMs are sorted based on the descending order of their vsr metric, where volume and vsr are computed as:

$$vol = \left(\frac{1}{1 - cpu}\right) * \left(\frac{1}{1 - mem}\right) * \left(\frac{1}{1 - net}\right)$$

$$vsr = \frac{vol}{size}$$

Where cpu, memory and n/w refers to cpu, memory and n/w usages of the PMs and VMs respectively and size refers to the memory footprint of the VM.

To mitigate hotspot on an overloaded PM, the highest vsr VM is migrated to a least loaded PM amongst the underloaded ones. If the least loaded PM can’t house the VM, next PM in the sorted order is checked. Similarly, if the VM cannot be housed in any of the underloaded PMs, next VM in the sorted order is checked. This way sandpiper tries to eliminate hotspots by remapping VMs on PMs through migration. The experimental results showed that migration overhead is less than that of swapping overhead; however, swapping increases the chances of mitigating hotspots in cluster with high average utilization [18], [19].

B. Khanna’s Algorithm

Khanna et al., in [10], [18] proposed Dynamic Management Algorithm (DMA) that is based on Polynomial-Time Approximation Scheme (PTAS) heuristic algorithm. The algorithm operates by maintaining two types of ordering lists, which are migration cost list and residual capacity list. The PMs are sorted according to the increasing order of their residual capacities across any resource dimension like CPU. The VMs on each PM are sorted according to the increasing order of their resource utilization like CPU usage. Migration costs of the VMs are determined based on their resource usage i.e. high usage implies high costly migration. Whenever a hotspot is detected on a PM due to violation of upper threshold, VM with least resource usage is chosen for migration to target host which has the least residual capacity to house it. If a PM cannot accommodate the VM, next PM in the sorted order is checked. Similarly, if the VM cannot be accommodated by any of the candidate target PMs, next least usage VM from the sorted order is checked.

Whenever coldspots are detected, the least usage VMs across all the underloaded PMs is chosen and migrated to a targeted PM, only if addition of the new VM increases the variance of residual capacities across all the PMs, else we choose the next VM in order. If there is no residual space left for the chosen VM, then the heuristic for coldspot mitigation stops. Variance is defined as follows:

$$variance, R(t) = \frac{(mean - rescpu)^2 + (mean - resmem)^2 + (mean - resnet)^2 \dots}{(m - 1)}$$

$$mean = \frac{rescpu + resmem + resnet + \dots}{m}$$

$$r_n = \sqrt{var_{p1}^2 + var_{p2}^2 \dots + var_{pn}^2}$$

In above equation, mean is defined as the average of normalized residual capacities across ‘m’ different resources like cpu, memory, networks, etc. res_{cpu}, res_{mem}, res_{net} ... stands for residual capacities across different resource dimensions. r_n is the magnitude of the vector which comprises of the individual variances across ‘n’ physical machines.

TABLE 2.1: Chosen Migration Heuristics

Algorithms	Goal	Metrics Used	Virtualization	Resource Considered	Platform
Sandpiper[14]	Hotspot Mitigation	Volume, Volume/size	Yes	cpu, memory & network	Xen 4.1
Khanna's Algorithm[6]	Server Consolidation	Residual Capacity, Variance	Yes	cpu, memory	Xen 4.1
Entropy[5]	Server Consolidation	No. of Migrations	Yes	cpu, memory	Xen 4.1

Khanna's Algorithm packs the VMs as tightly as possible trying to minimize the number of PMs by maximizing the variance across all the PMs. Thus, Khanna's algorithm minimizes power consumption by detecting underutilization in the managed using Max-Min thresholds selection model. When the resource usage of a running PM violates a minimum predefined threshold value, the algorithm tries to pack the running VMs as close as possible thus trying to minimize the number of running physical machines.

C. Entropy

Entropy proposes a consolidation algorithm based on constraint problem solving. The main idea of the constraint programming based resource manager is to formulate the VM resource allocation problem as constraint satisfaction problem, and then applies a constraint solver to solve the optimization problem. The ability of this solver to find the global optimum solution is the main motivation to take this approach. Entropy resource manager utilizes Choco constraint solver to achieve the objectives of minimizing the number of the running nodes and minimizing the migration cost. Entropy iteratively checks optimality constraint i.e. the current placement uses minimum number of the running nodes. If Entropy is successful in constructing a new optimal placement (uses fewer nodes) at VM packing problem (VMPP) phase, it will activate the re-allocation. Entropy employs a migration cost model that relates memory and CPU usage with migration context. High parallelism migration steps increases the cost. Using constraint programming techniques facilitates the task of capturing such context in two phases. In the first phase, Entropy computes a tentative placement (mapping of VMs to PMs) based on the current topology and resource usage of PMs and VMs and reconfiguration plan needed to achieve the placement using minimum number of PMs required. In the second phase, it tries to improve the reconfiguration plan by reducing the number of migrations required. Since obtaining the placement and reconfiguration may take a considerable amount of time, the time given to the CSP solver is defined by the users, exceeding which whatever immediate value the solver has computed is considered for dynamic placement of VMs. VMs are classified as active or inactive based on their usage of CPU with respect to thresholds set. The author define a viable configuration as one in which every active VM present in the cluster has access to sufficient cpu and memory resources on any PM. There can be any number of inactive VM on the PM satisfying the constraint. The CSP solver takes this viable condition into account in addition to the resource constraints, while procuring the final placement plan. However, considering only viable processing nodes and CPU-Memory Resource model is the limitation of the Entropy model [7], [18].

TABLE 2.2: VM Migration Heuristics [18]

Algorithms	Goal	When to migrate?	Which to migrate?	Where to Migrate?
Sandpiper[14]	Hotspot Mitigation	Resource usage exceed thresholds set	Most loaded VM	Least loaded PM
Khanna's Algorithm[6]	Server Consolidation	Resource usage violate the thresholds set	VM with lowest resource usage	Best first PM by residual capacity
Entropy[5]	Server Consolidation	No. of Migrations	Whichever minimizes reconfiguration cost.	Whichever minimizes reconfiguration cost.

IV. EXPERIMENTAL TEST BED

The experiment was performed on Six Xen hypervisor based machines. Four PMs with Xen 4.1 hypervisor installed were used to serve as the physical hosts and another machine was used as NFS Server to house the VMs images. Physical machines which worked as clients were used simultaneously to generate load on the virtual machines hosted on the PMs. We use Ubuntu 11.10 in Domain-0 and Seven VMs were created with Ubuntu 10.04, lucid host operating system. The virtual machines were configured with 2 VCPU and 256 MB memory size. They all have Apache, PHP and MySQL configured on them to act as web and Database servers. A separate machine has been configured which acts as the Management node, which runs the controller and the Decision Engine. The VIRT-M cluster management tool was implemented on this management node. Python programming was used to prototype these heuristics. Apart from these, RRD tool was installed on the Management node running the Decision Engine, for storage of resource data. Httperf was used to generate constant rate of load. Our Experimentation takes these heuristics into consideration and implements these on idle VMs to investigate their impacts on performance of live migration in both source and target machine.

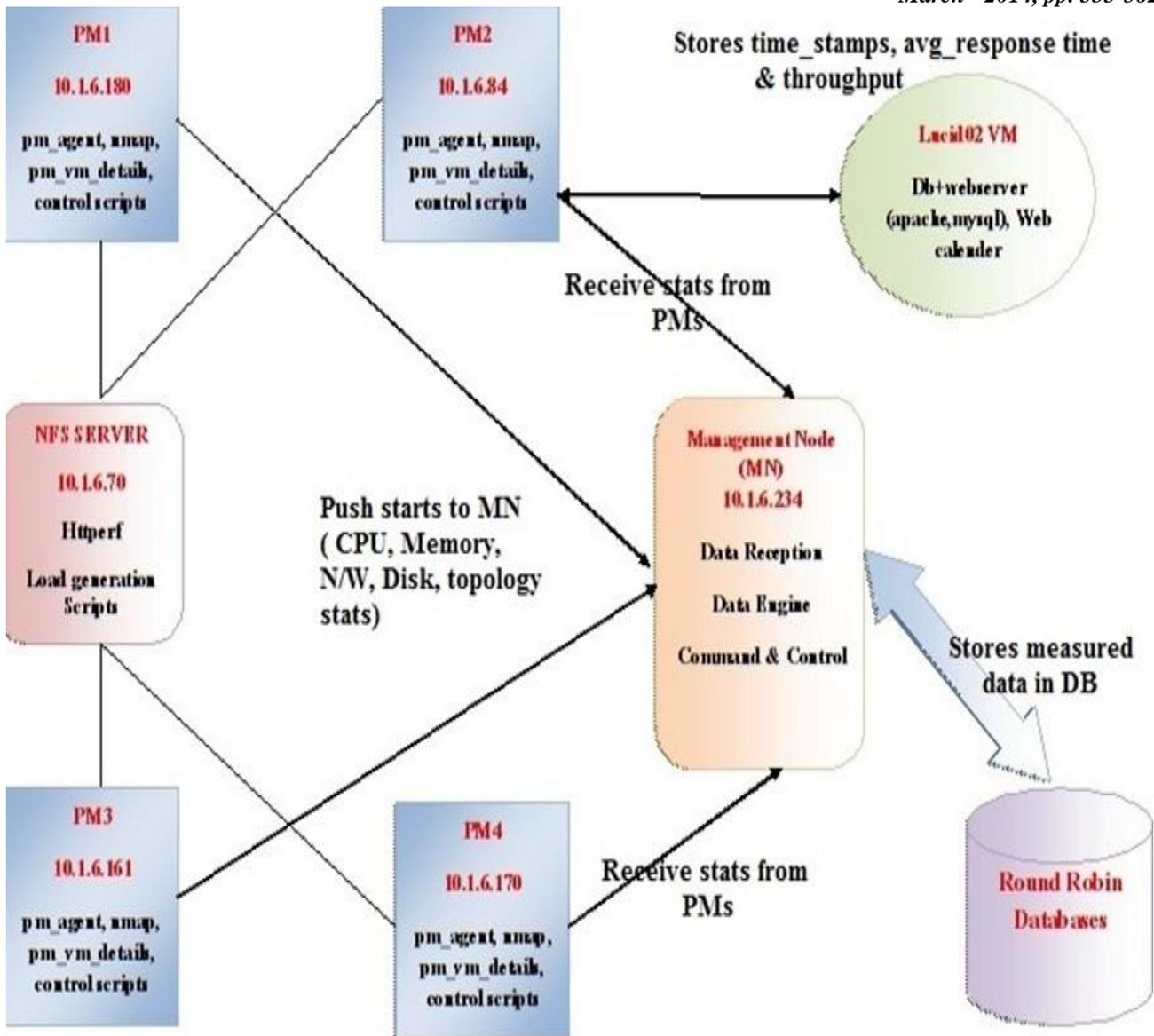


Fig. 4.1 Experimental Setup

V. EVALUATION AND RESULTS

An experiment was performed with four VMs, two of which were idle and two of moderate workload. “Httpperf” was used to inject workload using Web Calendar PHP scripts. The workload rate was set to 20 requests per second for lucid12 and lucid10. The following scenario was created:

- PM84 - empty
- PM161 - lucid12 and lucid13 - load with rate of 20 requests/sec on lucid12
- PM170 - lucid14 - no load
- PM180 - lucid10 - load with rate 20 requests/sec

The experimentation was performed for 15 minutes duration for all the algorithms. The upper cpu threshold was set to 60%. Under moderate levels of load what happens is of interest, because it is not obvious that if in the naive case Khanna’s Algorithm takes 2 minutes 45 seconds more time than entropy to consolidate, the same will happen in the case low levels of workload as well. We feel that this experiment is interesting because we want to answer questions like - Which VM does they choose to pack the VMs in low workload case? Is it same as the base case? What happens to the time taken to achieve consolidation? By how much does it increase/decrease?

From figures 5.1, 5.2 and 5.3, it was observed that, since upper cpu threshold was not violated which was a design goal, there were no events triggered by sandpiper. Figures 5.1, 5.2 and 5.3 depicts that Khanna’s Algorithm migrates lucid13 from PM161 to PM84, after detecting a coldspot on PM170, and sorting all four VMs based on their utilizations, clearly lucid13 and lucid14 has lower utilization than lucid12 and lucid10. Next lucid14 is migrated from PM170 to PM84, lucid10 from PM180 to PM170 and finally lucid12 from PM161 to PM170. This series of migrations were triggered based on the heuristic as a part of coldspot mitigation. After such migrations, the cpu usage levels were within the thresholds and hence no more coldspots were detected. Entropy again packs all the VMs in 1 PM by triggering just 2 migrations. The new topology generated was:

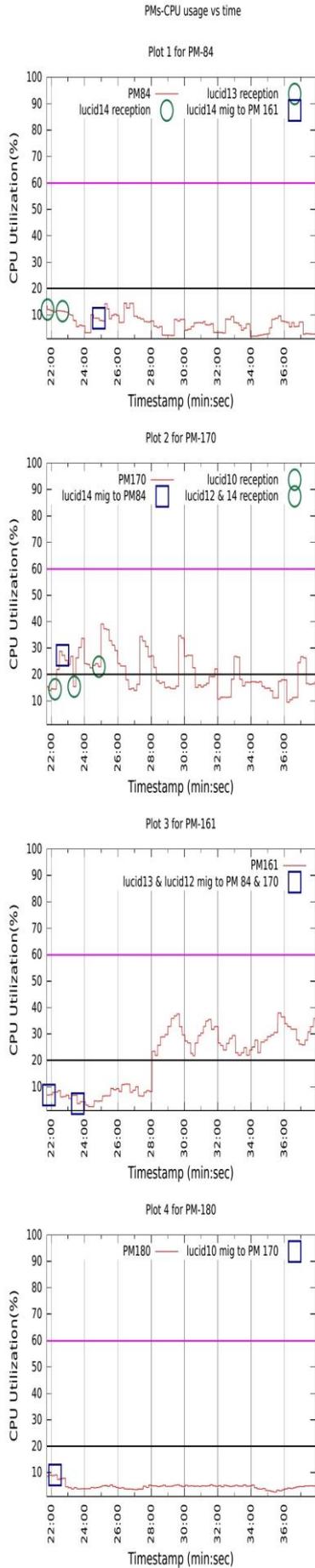


Fig. 5.1 Khanna's Algorithm

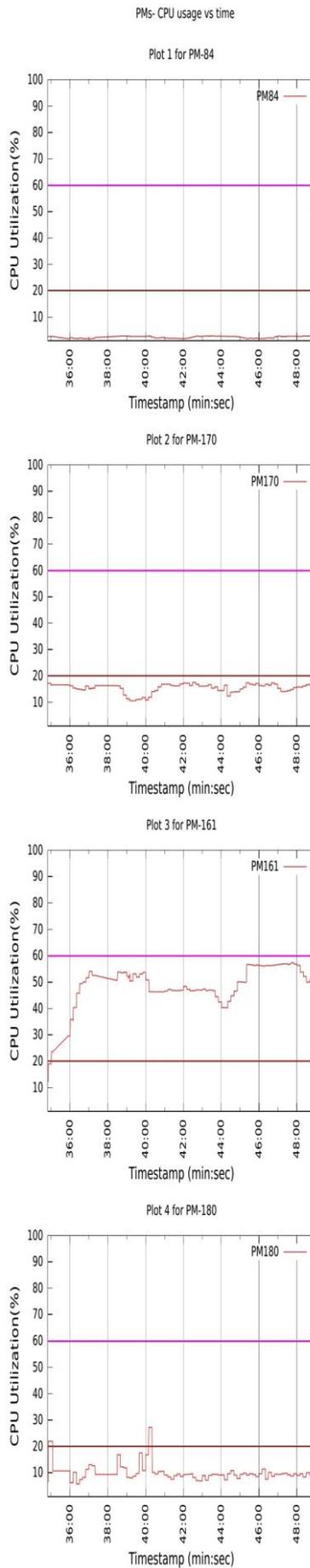


Fig. 5.2 Sandpiper

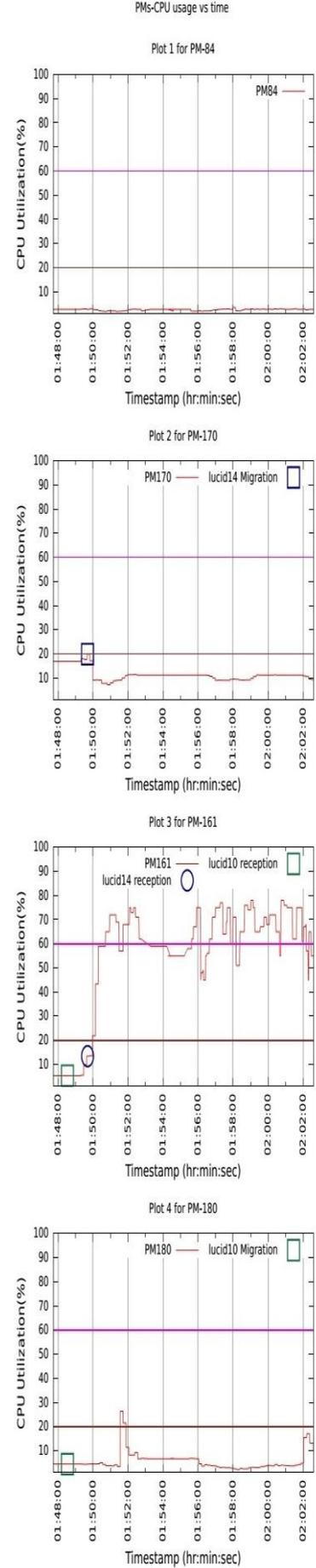


Fig. 5.3 Entropy Algorithm

- Sandpiper - No change
- Khanna’s Algorithm - PM170 hosts lucid10, lucid12 & lucid14 and PM84 host’s lucid13
- Entropy - all the VMs are moved to PM161.

From figures 5.1, 5.2 and 5.3, it was observed that whenever there has been a VM reception at any of the PMs, the cpu usage levels have increased. Also, since PM170 houses 3 VMs, its usage increases to as high as 37%. But even though PM161 migrated its VMs, a rise in its cpu usage was observed in the figures. It can be said that sometimes the PM cpu usages are reported higher than it was expected to be because of variation in migration time of VM from/to PMs, in presence of several scripts running on the PMs. The remote logging in PMs where the cpu usage is fluctuating might also have some minor effects on the host from where remote logging is done. Entropy again finds that the optimal number needed to house all the VMs is 1, and puts all the VMs on PM161, as cpu usage is not considered as migration cost metric here. So, even though cpu usage of lucid10 is expected to be higher than lucid13, it migrates lucid10 on PM161. It is easy to observe that since PM161 has already 2 VMs running on it, moving lucid14 from PM170 and lucid10 from PM180 will incur just 2 migrations to consolidate all the VMs on 1 PM. This increases the cpu usage of PM161 to as high as 70% exceeding the threshold of 60% when all the other PMs have low utilization levels.

Table 5.1 enlists the parameters of comparison, Khanna’s Algo reduces the number of PMs by 1, and PM84 was already devoid of VMs when we started. Entropy reduces the number of PMs used by 2, triggering just 2 migrations in 2.39 minutes. Khanna’s Algo triggers 5 migrations in total and uses 2 PMs by 3.33 minutes from the start of the algorithm.

Table 5.1 Experiment with low workload - Measured Evaluation Metrics

Algorithm	No. of PMs	No. of Migrations	Time Taken(mins)
Sandpiper	3	0	N/A
Khanna’s Algo	2	5	3.33
Entropy	1	2	2.39

VI. CONCLUSION AND FUTURE WORK

Live virtual machine migration will be great beneficial tool for dynamic resource management in the modern day data centers. To prevent server sprawl and minimizing power consumption, server consolidation aims at reducing the number of server machines by consolidating load, enhancing resource utilization of physical systems along with provision of isolation & security of the application hosted. In this paper, we presented a performance analysis of the server consolidation heuristics over a common software framework in a virtualized cloud environment.

Sandpiper, and Khanna’s Algorithm make use of threshold based techniques for triggering VM migrations. Entropy relies on CSP solver 4.4.1 to perform consolidation by providing a set of constraints, optimizing the number of PMs needed to house the VMs and the migration cost to determine the selection of configuration plan. In sandpiper, the migration cost is in terms of vsr metric whereas Khanna’s algorithm considers the resource utilization as the migration cost metric. Both of them intended to reduce migration cost in terms of the memory allocated to the VMs. Unlike other algorithms, Entropy attempts to obtain a globally optimal solution, which distinguishes itself in its consolidation approach. Unlike other algorithms does, Entropy takes into consideration all the hosts in the topology and based on their current resource usages, finds out an optimal solution which tries to decrease the migration overhead in terms of memory. The other algorithms try to achieve consolidation on a per host basis, making sure that resource violations are prevented every time each host is scanned, and then the VMs are packed as closely as possible.

It was observed from the graph that the variance of utilization levels across Khanna’s Algorithm is less than Entropy. Although Khanna’s Algorithm tries to maximize variance, it ensures that the cpu usage levels are within the defined bounds, hence the requirement of defined bounds, affects the migration decision making process keeping the overall utilization levels across the PMs higher than they were observed in the case of Entropy. In entropy, at the cost of 1 PM having very high usage, there were other PMs which can be freed up because of underutilization. The risk in entropy is that, in face of sudden bursty data after such rigorous consolidation, it might trigger many migrations to other destination PMs or it might need some new PMs to be started to house the VMs, since very less residual capacity will be left in the PM where all the VMs are packed. In case of Khanna’s Algorithm, such a crisis may not arise as there are residual capacities left in the PMs which may accommodate VMs in future.

If the nature of workload in data center can be known beforehand, it can be said that, Entropy may not be a good choice for too bursty workloads, as every time the algorithm tightly packs the VMs in a PM, after some discrete interval of time, if there is a sudden surge of high workload, there may arise a situation where new PMs need to be added to the system, if they had been shut down earlier. However, Khanna’s Algorithm takes care of the future possibility of VM accommodation in a PM, for which it may be able to accommodate VMs in face of bursty data.

These algorithms try to efficiently prevent server sprawl and ensure non-disruptive load balancing in data centers. Efficiency of the algorithm depends on the resource parameters and metrics considered. Hence, a comparative performance analysis was carried out to analyse their applicability, goodness and incurred overhead. In near future, Evaluation of these algorithms with variable and mixed load can facilitate in figuring out the distinct cases where an algorithm will behave well and hence can be used in those cases only to leverage the maximum benefits.

Moreover, more algorithms which does similar jobs like consolidation can be chosen in near future and their relative behaviour can be analysed with the already chosen algorithms.

REFERENCES

- [1] A. Verma, P. Ahuja, A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems", in *Proceedings of the ACM/IFIP/USENIX 9th International Middleware Conference*, 2008.
- [2] Aameek Singh, Madhukar Korupolu, Dushmanta Mohapatra. "Server-Storage Virtualization: Integration and Load Balancing in Data Centers", in *Proceedings of the SC 2008 ACM/IEEE conference on Supercomputing*, 2008.
- [3] A. Souvik Pal and B. Prasant Kumar Pattnaik, "Classification of Virtualization Environment for Cloud Computing", *International Journal of Science and Technology*, vol 6, Jan. 2013.
- [4] Anju Mohan and Shine S, "Survey on Live VM Migration Techniques", *International Journal of Advanced Research in Computer Engineering & Technology*, vol 2, Jan. 2013.
- [5] C. Clark, K. Fraser, S. Hand, J. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live Migration of Virtual Machines", in *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation*, vol. 2, pp. 286, 2005.
- [6] Emmanuel Arzuaga and David R. Kaeli, "Quantifying load imbalance on virtualized enterprise servers", in *Proceedings of the first joint WOSP/SIPEW International Conference on Performance Engineering IEEE/ACM Trans. Netw.* Pages 235-242, 2010.
- [7] Fabien Hermenier, Xavier Lorca, Jean-Marc Menuad, Gilles Muller and Julia Lawall, "Entropy: A Consolidation Machine Manager for Clusters", In *Proceedings of the 2009 ACM SIGPLAN/SIGOPS Internal Conference on Virtual Execution Networks, VEE, 2008*, pp.41-50, 2009.
- [8] G Jung, KR Joshi, MA Hiltunen, RD Schlichting RD, C Pu, "Generating adaptation policies for multi-tier applications in consolidated server environments", in *Proceedings of the 5th IEEE International Conference on Autonomic Computing (ICAC 2008)*, Chicago, IL, USA, 2008; 23–32.
- [9] G Jung, KR Joshi, MA Hiltunen, RD Schlichting RD, C Pu, "A cost-sensitive adaptation engine for server consolidation of multitier applications", in *Proceedings of the 10th ACM/IFIP/USENIX International Conference on Middleware (Middleware 2009)*, Urbana Champaign, IL, USA, 2009; 1–20.
- [10] Gunjan Khanna, Kirk Beaty, Gautam Kar and Andrzej Kochut, "Application Performance Management in Virtualized Server Environments", *10th IEEE/IFIP Conference on Network Operations and Management in Virtualized Server Environments, NOMS*, 2006.
- [11] Jyothi Sekhar, Getzi Jeba and S. Durga, "A survey on Energy Efficient Server Consolidation through VM Live Migration", *International Journal of Advances in Engineering and Technology*, vol 5, pp.515-525, Nov. 2012.
- [12] Kejiang Ye, Xiaohong Jiang, Dawei Huang, Jianhai Chen, and Bei Wang, "Live migration of Multiple Virtual Machines with Resource Reservation in Cloud Computing Environments" , in *Proceedings of 2011 IEEE 4th International Conference On Cloud Computing*, pp. 267-274, 2011.
- [13] M. Nelson, B. Lim, and G. Hutchins, "Fast transparent migration for virtual machines", in *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, pp. 25, 2005.
- [14] M Uysal, Zhikui Wang, Pradeep Padala, X. Zhu' "Automated Control of Multiple Virtualized resources", in *Proceedings of the 4th ACM European Conference on Computer Systems, Eurosys*, 2009.
- [15] NIST Definition of Cloud Computing v15, www.nist.gov/itl/cloud/upload/cloud-def-v15.pdf
- [16] Sarnpreet Kaur and Ravi Bhushan, "Resource Optimization of Servers using Virtualization", *International Journal of Advance Research in Computer Science and Software Engineering*, vol 3, pp.323-326, Jun.2013.
- [17] Sarnpreet Kaur and Ravi Bhushan, "Review Paper on Resource Optimization of Servers using Virtualization", *International Journal of Advance Research in Computer Science and Software Engineering*, vol 3, pp.327-332, Jun.2013.
- [18] Susheel Thakur, Arvind Kalia and Jawahar Thakur, "Server Consolidation Algorithms for Cloud Computing Environment: A Review", *International Journal of Advanced Research in Computer Science and Software Engineering*, vol 3(9), September 2013.
- [19] Timothy Wood, Prashant Shenoy, Arun Venkataramani and Mazin Yousif, "Sandpiper: Black-box and Gray-box Resource Management for Virtual Machines", *Journal of Computer Networks*, vol.53, pp.2923-2938, Dec.2009.
- [20] T.Wood, G. Tarasuk-Levin, Prashant Shenoy, Peter desnoyers, Emmanuel Cecchet and M.D.Corner "Memory Buddies:Exploiting Page sharing for Smart colocation in Virtualized Data Centers" in *Proceedings of the ACM SIGPLAN/SIGOPS International conference on Virtual execution environments,VEE*,pages 31–40, New York, NY, USA, 2009.
- [21] V. Sarathy, P. Narayan, and Rao Mikkilineni, "Next Generation Cloud Computing Architecture- Enabling Real-time Dynamism for Shared Distributed Physical Infrastructure", *19th IEEE International Workshops on enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE*, pp.48- 53, June 2010.
- [22] X Zhu, D Young, BJ Watson, Z Wang, J Rolia, S Singhal, B McKee, C Hyser, D Gmach, R Gardner, et al., "1000 islands: Integrated capacity and workload management for the next generation data center" in *Proceedings of the 5th International Conference on Autonomic Computing (ICAC 2008)*, Chicago, IL, USA, 2008; 172–181.