Performance Evaluation of Hypervisors and the Effect of Virtual CPU on Performance

Hafiz ur Rahman[†], Guojun Wang[*†], Jianer Chen[†], and Hai Jiang[‡]
[†]School of Computer Science and Technology, Guangzhou University, Guangzhou, China, 510006
[‡]Department of Computer Science, Arkansas State University, United States of America
hafiz_rahman@e.gzhu.edu.cn, csgjwang@gmail.com, jianer@gzhu.edu.cn, hjiang@astate.edu

[*]Correspondence to: csgjwang@gmail.com

***Abstract*—Organizations are adopting virtualization technology to reduce the cost while maximizing the productivity, flexibility, responsiveness, and efficiency. There are a variety of vendors for the virtualization environments, all of them claim that their virtualization hypervisor is the best in terms of performance. Furthermore, when a system administrator or a researcher want to deploy a virtual machine in a cloud environment, which vCPU-VM configuration is the best for better performance? In this paper, prior to evaluating the latest version of hypervisors (commercial and open source), the best virtual CPU to virtual machine (vCPU-VM) configuration as well as the effect of virtual CPUs on performance is analyzed for each hypervisor. We used Phoronix Test Suite (PTS) benchmarking tool as a traffic generator and analyzer. The results have shown that commercial and open source hypervisors have similar performance. As per our observation, the performance of a system would degrade by improper allocation of vCPUs to VMs, or when there is a massive over-allocation of vCPUs.**

***Index Terms*—cloud computing, hypervisor, virtual CPU mapping, CPU utilization**

I. Introduction

Traditional data centers suffer from server proliferation, low resource utilization, increased physical infrastructure costs, decreased scalability and agility, diminished disaster recovery, and migration challenges [1], [2]. Virtualization is used to ease computing resource management, resource utilization and running multiple heterogeneous or homogeneous operating systems on a single physical machine. In the last decade, virtualization has attracted many different research groups working on server consolidation, security, and computing [3], [4], [5]. For example, distributed data centers are now being utilized by using virtualization technology which was not possible in the past. Thus, virtualization plays a vital role in mitigating such challenges [6], [7]. Virtualization makes use of server resources in a well-organized manner by setting up different servers within different cloud types [2]. As a result, organizations can access and manage their data more efficiently. Therefore, many organizations are adopting virtualization technology to reduce the cost while maximizing the productivity, flexibility, responsiveness, and efficiency.

Virtualization can be done to various resources such as CPU, memory, or I/O devices. Virtualization vendors use different technologies to provide virtualization environments. Hypervisors are used in virtualized environments as agents facilitating virtual machines and hardware [8], [9], [10], [11]. In a regular system, the hardware resources are used by single operating system (OS). While in virtualization environments, hypervisor is responsible to manage hardware resources and virtual machines. Moreover, each guest OS is in charge of virtual resources and concurrently share and access the hardware resources [4]. Therefore, virtualization systems face challenges such as hypervisor selection, virtual machines (VMs) allocation, virtual CPU to Virtual Machine (vCPU-VM) configuration, and virtual CPU to physical CPU (vCPU-pCPU) mapping. Such challenges may lead to system performance degradation [12], [13], [14].

We were motivated by the fact that virtualization suffers from drawbacks. In addition, researchers evaluate and analyze hypervisors without investigating the best vCPU-VM configuration through which better CPU utilization and performance for each hypervisor can be expected. They assigned vCPU to VM based on non-suitable vCPU-VM configuration [9], [15], [16], [17]. Furthermore, there are a variety of vendors for the virtualization environments and all of them claim that their virtualization hypervisor is the best for virtualized environments , however they depend on the used application [18], [19], [20].

The main contributions of of the paper are as follows. Building a private cloud using the latest version of commercial and open source hypervisors (Citrix xenServer version 7.4 and KVM version 4.4.0). In this research work, we focus on the effect vCPU on performance prior to evaluating commercial and open source hypervisors. As another contribution, we provide recommendations for vCPU-VM configuration through which better CPU utilization on each hypervisor can be achieved. As a result, cloud service providers will get the benefits, when they deploy VMs in a cloud environment or evaluate open source and commercial hypervisors (KVM, Citrix xenServer, VMware, and Hyper-V). Our finding will be a road map to assist cloud service providers to choose the best hypervisor and vCPU-VM configuration for their specific needs.

The rest of the paper is organized as follows. Section II, provides an extensive literature review. Section III, describes our research methodology and experimental design. We analyze the evaluation results in Section IV and draw conclusions in Section V.

II. Related Work

In this sections, we discuss recently published papers in order to evaluate different hypervisors and investigate the impact of vCPU on performance on CPU utilization. Other factors such as the nature of virtualization type (i.e., para virtualization, full virtualization, or hardware assistance virtualization) are also investigated and summarized.

Charles David [15] analyzed two types of virtualization namely paravirtualization (i.e., Xen 3.1.2) and hardware assisted virtualization (i.e., KVM) using open sources virtualization platforms Xen 3.1.2 and KVM (RHEL 5.3 64bit) hypervisors on Chip Multiprocessor (CMP) Architecture. The author measured the throughput and overall performance of the hypervisors using PTS benchmarking tool under various levels of workload and compared different system attributes including CPU usage, memory access rate, and I/O operations. Unfortunately, the author randomly assigned vCPUs to VMs which caused performance degradation. The author did not analyze the root cause of the performance degradation.

Babu et al. [16] evaluated the system performance of three hypervisors. The authors had opted Xen-PV, OpenVZ, and XenServer for para virtualization, container virtualization, and full virtualization respectively. They compared the performance of these techniques based on Unixbench benchmarking tool. They observed that the hypervisor which supports full virtualization has a comparatively higher system performance in terms of file copy, pipe based context switching, process creation, shell scripts, and floating point operation than the other two virtualization types. However, the authors did not investigated the effects of vCPU-VM and vCPU-pCPU. Moreover, the authors only used one virtual machine for their evaluation.

C. Mancas [17], used Passmask benchmarking tool and evaluated VMware and KVM hypervisors for CPU, memory, and I/O performance. They observed that overall VMware behaves better than KVM. However, there are cases, such as memory and HDD in which KVM overtakes VMware. Like [16], the author used a simple test case in which he used XP as a guest OS.

S. Varette et al. [9] evaluated energy-efficiency of VMware ESXi 5, KVM 0.12 and Xen 4.0, using Non-uniform Memory Access (NUMA) architecture through HPC implementation. The authors used HPL benchmarking tool and the Grid 5000 platform to investigate the performance of different hypervisors in a well-regulated and similar to HPC environment. The authors concluded that there is a sustainable performance impact introduced by the virtualization layer across all types of hypervisors.

Hwang et al. [8] investigated open source and commercial hypervisors (Hyper-V 2008R2, vSphere 5.0, KVM 2.6.32-279, and Xen 4.1.2). The authors stated that there is no impact by increasing the number of virtual CPUs on performance from one vCPU up to four vCPUs on all hypervisors. In our work, we will show that there is a high impact of vCPU on performance.

TABLE I
Specification of the Servers

Specifications	Server 1 Dell Power Edge R620	Server 2 Dell Power Edge R620
Hardware Model	Intel Xeon	Intel Xeon
Processor Speed	2 GHz	2 GHz
CPU Processor	12 Cores	12 Cores
Logical Processors	24 cores	24 cores
Main Memory	64 GB	64 GB
Storage Capacity	1024 GB	1024 GB

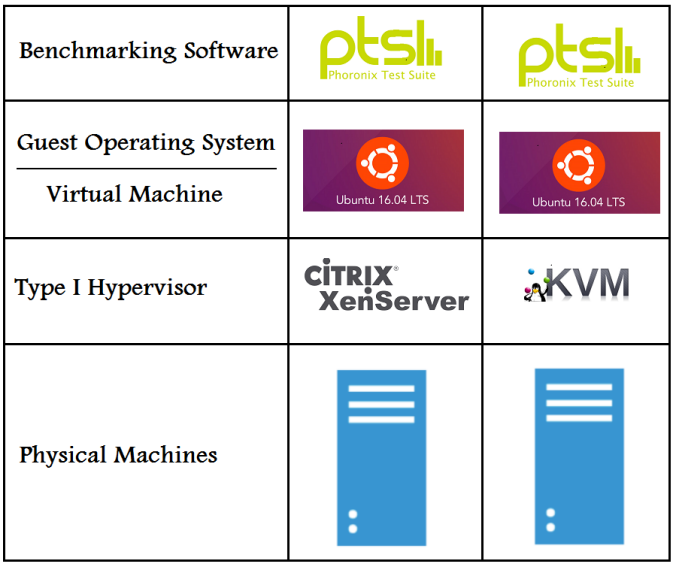

Fig. 1. Experimental Platform.

As a summary, most of the authors analyzed and compared various hypervisors without investigated the effect of vCPU-VM configuration. However, some of them analyzed either vCPU-VM configuration or vCPU-pCPU mapping by using a certain hypervisor. In this paper, before evaluating the latest version of hypervisors (Citrix XenServer version 7.4 and KVM version 4.4.0) NUMA architecture; firstly, we analyzed the effects of vCPU on performance along with vCPU-VM configuration for each hypervisor using NUMA architecture. Secondly, we investigated the effects of hypervisor and vCPU-VM configuration on performance.

III. Research Methodology and Experimental Design

In this section, we describe our experimental methodology and classification of experiments.

A. Experimental Platform

Our experimental platform, as shown in Fig. 1, starts with two identical physical servers. The two physical machine have similar architecture and specifications in order to achieve a fair assessment. Each physical node (Dell Power Edge R620) is equipped with two Intel Xeon hexacore CPU and 64 GB of DDR2 DRAM. The specification of these two servers are

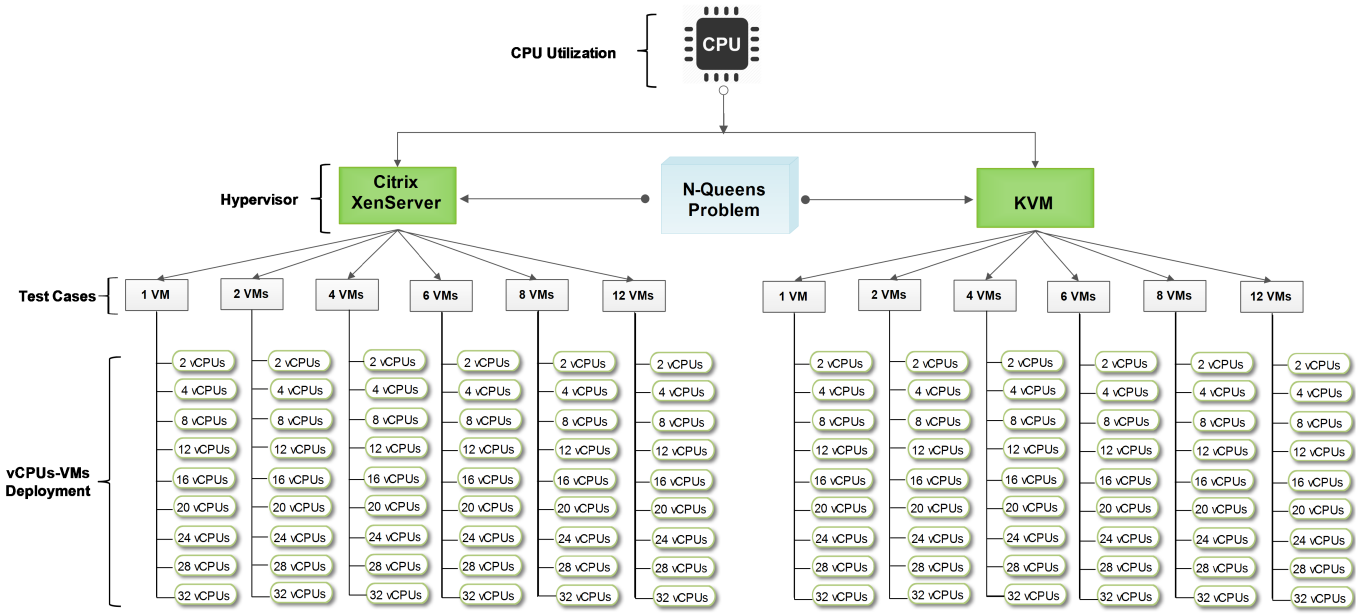

Fig. 2. Classification of experiments (experimental design).

given in Table I. We installed the latest versions of hypervisors (Citrix XenServer version 7.4 and KVM version 4.4.0) on the physical servers. Next, we created virtual machines running Ubuntu 16.04 as a guest operating system. We built virtual machines on each hypervisor in order to provide the test environment. Finally, the PTS benchmarking tool is installed on each virtual machine as a traffic generator and analyzer.

B. Classification of experiments

There are many factors in the platform to affect the CPU performance, such as the degree of overcommitment (that is, the ratio of vCPU-to-pCPU), the number of virtual machines concurrently running at the top of hypervisor, mapping strategy (static or dynamic allocation of vCPU-to-pCPU, and the workload or application running inside of VM. This evaluation is composed of two main experiments; Citrix XenServer-based setup and KVM-based setup as shown in Fig. 2. We have four main factors in our experimental design namely: type of hypervisor, VMs, vCPUs, and workload. The objectives of these test configurations are to investigate the effects of hyperviosr, VM, and vCPU on performance prior to evaluating different hyperviosrs.

In order to evaluate CPU utilization in a Cloud environment clearly, various experiments were conducted. We first investigate the effect of virtualization technology layer. Secondly, to systematically investigate the effect of VMs on performance, we performed three main experiments: under allocation (i.e., the number of vCPUs less than available logical CPUs), balance allocation (i.e., equally divided available logical CPUs among VMs), and over allocation (the number of vCPUs more than available logical CPUs) of computing resources. To investigate and choose the best vCPUs-VMs configuration and better CPU utilization, we also focused our test measurement with some restrictions, e.g., we scale the number of vCPUs from 2 to 32 (2, 4, 8, 12, 16, 20, 24, 28, and 32), and concurrently boot 1, 2, 4, 6, 8, and 12 VM.

For every experiment setup, six test cases and nine vCPU-VM configurations are presented. Fig. 2, illustrates the experimental design and the details of each test configuration. In Fig. 2, there are two different hypervisors, each hypervisor has six different test cases. Each test case has nine different deployments i.e., allocation of vCPUs to VMs. In every deployment, we run N-Queens benchmark for sixteen different workloads. In order to evaluate the effect of virtualization technology and vCPU-VM configuration, total 1728 (2 x 16 x 6 x 9) observations were obtained where (2) is the number of hypervisors used in our experiments, (16) represents different workloads, (6) shows different test cases of VMs running on top of each hypervisor, and (9) represents different vCPU-VM configurations.

Each experiment is conducted on an identical separate server. Therefore, all the hardware resources of the server are fully dedicated for each hypervisor and the results obtained are fairly and reliably analyzed.

C. Benchmarking Tools

The benchmarks used in our experiments are PTS [21] and Linux Top Command. PTS is used to generate the workloads and analyze the results for CPU utilization. PTS contains a variety of test profiles. For CPU bound operations, we chose two important test profiles called N-Queens and John-the-Ripper benchmarks. Based on elapsed time in seconds, different workloads (low, medium, and high) were generated using N-Queens benchmark [21], which report the elapsed time in seconds. In addition, we measured the CPU utilization at hypervisor level for both hypervisors using Linux Top

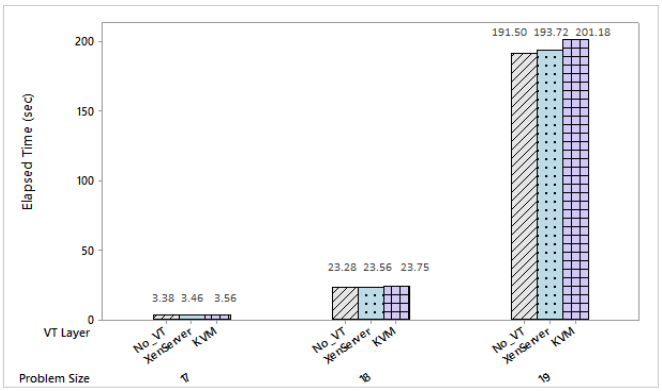

Fig. 3. The effect of Virtualization Layer using N-Queens benchmark.

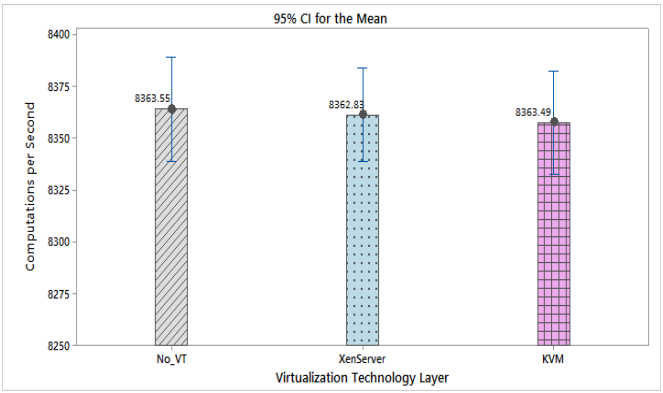

Fig. 4. The effect of Virtualization Layer using John-the-Ripper benchmark.

command. Furthermore, for each experiment, the average CPU utilization in percentage at hypervisor level is measured using Linux Top Command. But, only CPU utilization in percentage is insufficient to investigate the effects of VMs allocation, vCPU-VM configuration, and vCPU-pCPU mapping on performance, especially when the CPU utilization level is 100%. Then, we can not judge the effect of VMs on performance. Therefore, the elapsed time to solve the N-Queens problem was calculated to trace how much actual work is performed by CPU.

N-Queens is an open-source OpenMP benchmarking tool [21] that solves the N-Queens problem. N-Queens problem is a classical combinatorial problem, widely used as a benchmarking tool by researchers for CPU-intensive calculation that have different workloads and simple structure [22], [23]. The problem involves placing N queens on an N x N chessboard such that no queen can attack any other. Thus, a solution requires that no two queens share the same row, column, or diagonal. As the problem size increases (number of queens), the corresponding possible solutions and the elapsed time to solve the problem also increasing. In this paper, we tested each hypervisor for different queens size ranges from 4 to 19. Based on the possible solutions to solve N-Queens problem and corresponding elapsed time, we chose problem size 17, 18, and 19 as low, medium, and heavy workload respectively.

IV. Results and Discussion

In this section, the results have been discussed that were obtained using the PTS benchmarking tool. Each experiment was repeated five time and the results are averaged. The objectives of these experiments are to investigate the effects of hyperviosr and vCPU on performance prior to evaluate different hypervisors. The experimental results are shown in Figures 3 - 13.

A. The Effect of Virtualization Technology

This test is designed to investigate the effect of virtualization technology layer (hypervisor) on performance in terms of CPU bound operations. For this test, only one VM allocated 24 vCPUs (e.g., all the hardware resources of the server are allocated to one VM), having Ubuntu 16.04 as a guest OS, running at the top of both hypervisors, as well as a host OS on a bare-metal machine (i.e., non virtualized machine (No_VT)). This test is performed using two powerful servers, server specifications are given in Table I.

In this experiment, the N-Queens benchmark is used as a stress test to judge the virtualization overhead for CPU bound operations running one VM. The performance (elapsed time) of non virtualized machine against commercial and open source hypervisor are given in Fig. 3 and 4. Fig. 3, illustrates the effect of virtualization on performance using three different workloads (17, 18, and 19; which is low, medium, and high workload) of N-Queens benchmark. The results illustrate that for low and medium workload there is no significant performance overheard but for heavy workload a low performance overhead is observed i.e., performance is decrease by 0.01% and 0.04% using Citrix XenServer and KVM respectively. One of the reason of performance reduction for heavy workload is that when we used heavy workload there are more context switching due to high elapsed time (i.e., CPU cycle are wasted instead of being utilized by vCPUs) and NUMA processor affinity between vCPUs as compared to low and medium workload.

To ensure that the CPU utilization (i.e., elapsed time) seen with N-Queens benchmark was not an anomaly, John-the-Ripper benchmark (for CPU bound operation) is used as a benchmark with the same settings. Fig. 4, shows the result of Jon-the-ripper benchmark. Both benchmarks were run six times, and the results were averaged (the results were exactly the same for Fig. 3 therefore there is no error bar for each scenario). Both results illustrated that for CPU bound operations using only one VM and consuming all CPUs cores, the virtualization overhead is almost minimal and the performance of commercial and open source hypervisor are almost similar.

B. The Effect of Virtual CPUs on Performance

The cloud service providers are interested to know how much resources (vCPUs-VMs) should be allocated for maximum performance. Since large number of VMs are running in cloud environment and sharing physical computer resources,

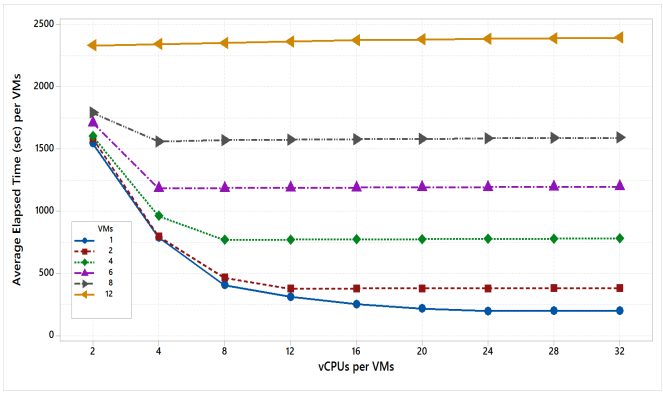

Fig. 5. The effect of Virtual CPUs on Performance - Citrix xenServer.

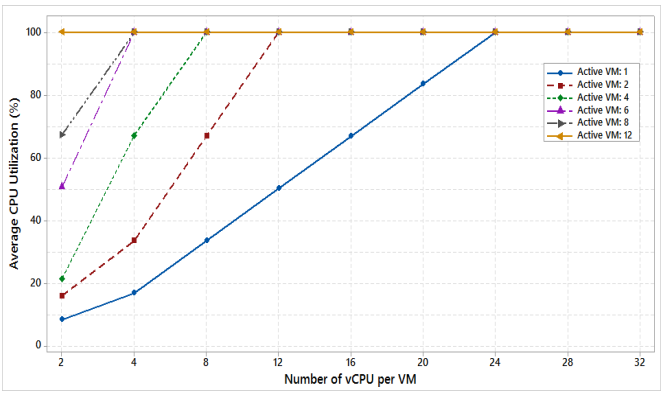

Fig. 7. Total CPU Utilization at Hypervisor Level - Citrix xenServer.

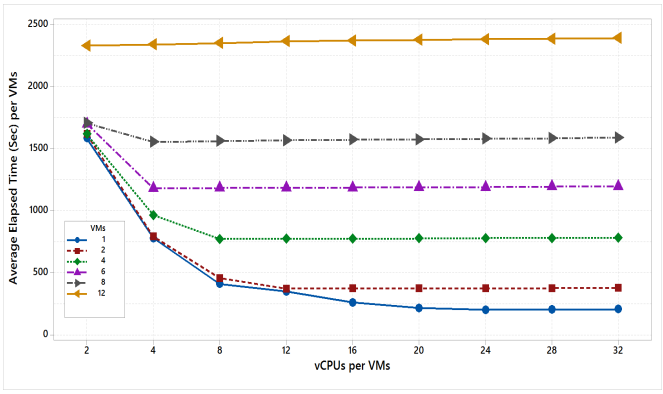

Fig. 6. The effect of Virtual CPUs on Performance - KVM.

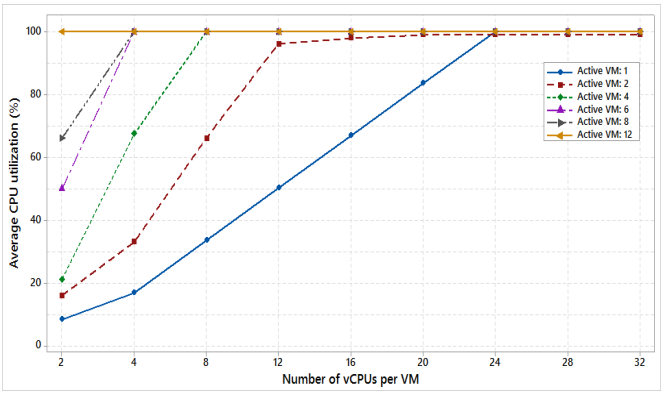

Fig. 8. Total CPU Utilization at Hypervisor Level - KVM.

a risk of poor performance arises due to over allocation of physical CPU (pCPU) resources. These performance bottlenecks should be investigated, quantified, and avoided.

Here, we are testing the impact of vCPU assigned to VM. Previous studies [14], [24] showed that the system performance could be affected by using different ways of the pCPUs. Each virtual machine is configured with a number of vCPUs. The performance of a VMs having eight vCPUs will be doubled as compared to four vCPU-VM configuration (e.g., balance and under allocation). One can decide to use available pCPUs in two opposite ways such as by using few VMs having large number of vCPUs, or large number of VMs having small number of vCPU.

Figures 5 and 6, show the elapsed time in second for different vCPUs configurations (i.e, CPU-VM configuration are 2, 4, 8, 12, 16, 20, 24, 28, and 32 vCPUs) using heavy workload (problem size 19). In both figures, the number of active VMs increases from 1 to 12 with nine different vCPUs-VM configuration. From both figures, it is clearly seen that, the performance is improved by allocating more vCPUs to VMs. However, there is a performance threshold for vCPUs i.e, 24 out of 24 logical CPUs are consume. After the threshold, no significance improvement was observed. However, the performance was decreased after crossing the threshold.

In one VM test case, the effect of over allocation of vCPUs to VMs was low, but it was significantly high for other test cases. The time sharing of CPU resources by VMs, in case of over allocation, could be the possible reason. That why, there was no or very small time sharing in balance allocation. Moreover, time sharing increases the number of context switches among VMs. The overhead due to excessive context switching between VMs and NUMA processors affinity will result performance reduction and CPU cycle will be wasted instead of being utilized by the VMs. After further analysis, higher performance implication of over allocation was observed in 6, 8, and 12 VMs tests cases as compare to the other test case (1,2, and 4). In addition to this, the CPU utilization for Citrix xenServer and KVM are given in Figures 7 and 8. Both figures show CPU utilization in percentage at hypervisor level i.e., how much the VMs are using the physical CPU resources.

C. The Significance of Over Allocation

The over allocations of vCPUs to VMs is also important in a cloud environment. If a cloud service provider did not use over allocation of vCPUs, they may not be able to use all the physical cores after live migration or idle VMs. For instance, there is one host with 24 CPU cores, and there are two VMs running on the host and each VM has 12 vCPUs. If

one VM is migrated to another host, crashed, or become idle then twelve physical CPU cores will not be used, although one VM is overloaded. However, if each VM configure more than 24 vCPUs then there would be enough vCPUs to utilize by overloaded VM after live migration or idle VM.

To highlight the significance of over allocation of vCPUs on performance, we performed two more experiments namely: uniform vCPUs and non-uniform vCPU-VMs configuration.

1) Uniform vCPUs-VMs Allocation: In uniform vCPU configuration, each active VM has the same number of vCPUs but no over allocation. In this experiment, we vary the pinning strategies and fixed the number of VMs (eight VMs), vCPUs (i.e., each VM allocated three vCPU, total vCPUs = 8 x 3 = 24), and also fixed the number of workloads (low, medium, and high). Two out of eight VMs will run a low workload, two of them with medium workload while the remaining four VMs will run the heavy workload.

Figure 9, illustrates the uniform vCPU-VM configuration for two vCPU pinning strategies (i.e., pinning and no pining strategy). In no pinning strategy, the hypervisor is free to schedule domain's vCPUs on any pCPUs. While pinning strategy the hypervisor is free to schedule the Dom0 (hypervisor) vCPUs on any pCPUs and other active VM's vCPUs are statically pinned to user defined logical CPUs.

In order to investigate the effect of vCPUs on uniform vCPUs, we run two different experiments each with eight VMs. The purpose of assigning different workloads to VMs, while keeping the same vCPUs configuration, is to investigate the effect of over allocation and pinning strategies. Figure 9, shows that after 26 and 28 seconds (depending on pinning strategies), the VMs having low workload become idle, due to low workload they finished their task early as compare to medium and high workload VMs. The other six VMs are still busy. However, the CPU utilization level dropped from 100% to 77% as shown in Figure 10. After 180 and 182 seconds (depending on pinning strategies), the VMs having medium workload become idle. Now four VMs out of eight VMs are idle while other four are still busy due to heavy workload. Thus, the average CPU utilization level drops to 52%. So, there was no significant difference among pinning strategies (i.e., pinning strategies have no effect on under allocation and balance allocation of vCPUs-VMs). For better CPU utilization, the optimum vCPU-VMs configuration is needed. In the next subsection, we will discussed the non-uniform vCPUs test configuration by which CPU utilization and performance (in terms of elapsed time) can be improved.

2) Non-uniform vCPUs-VMs Allocation: In non-uniform vCPU configuration, each active VM has the same vCPUs like uniform configuration, but we used over allocation. In this experiment, we vary the pinning strategies and fixed the number of vCPUs (6 vCPUs per VM), the number of VMs (8 active VMs), and the number of workloads (low, medium, and high). Two out of eight VMs will run a low workload, two of them with medium workload while the remaining four VMs will run the heavy workload. In this experiment, the aim of over allocation of vCPUs, is to utilize all the physical cores after idleness of VMs. As shown in Uniform vCPUs-

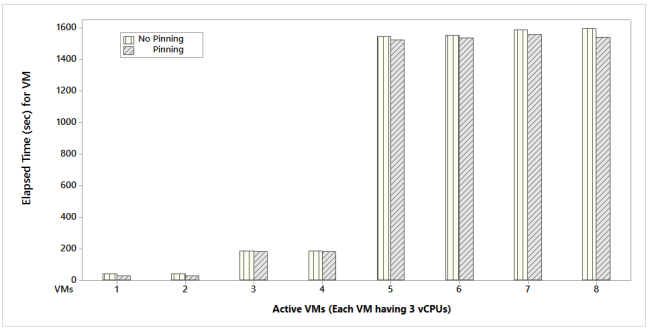

Fig. 9. The effect of Virtual CPUs on Performance - Uniform vCPUs per VMs

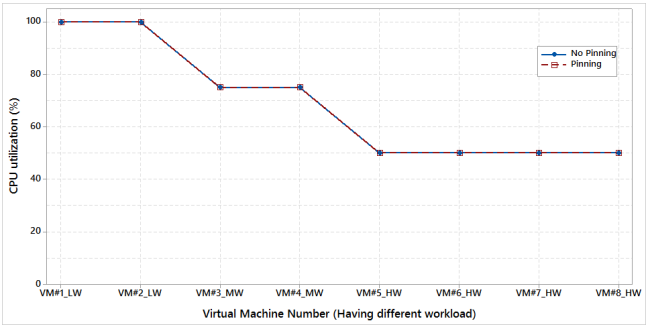

Fig. 10. CPU Utilization for Uniform vCPUs

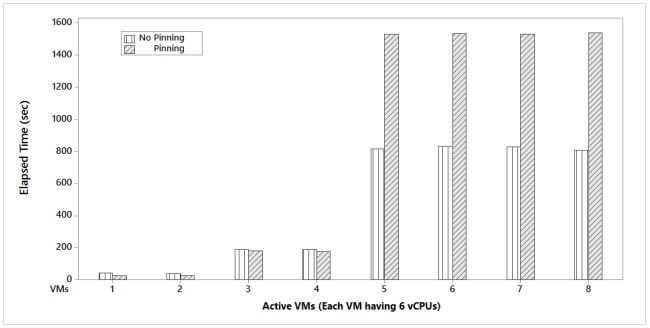

Fig. 11. The effect of Virtual CPUs on Performance - Non-uniform vCPUs per VMs.

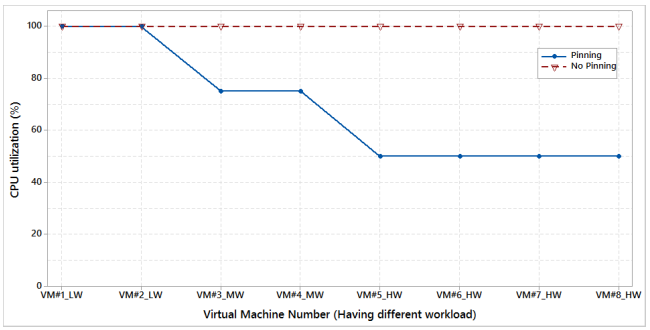

Fig. 12. CPU Utilization for Non-uniform vCPUs.

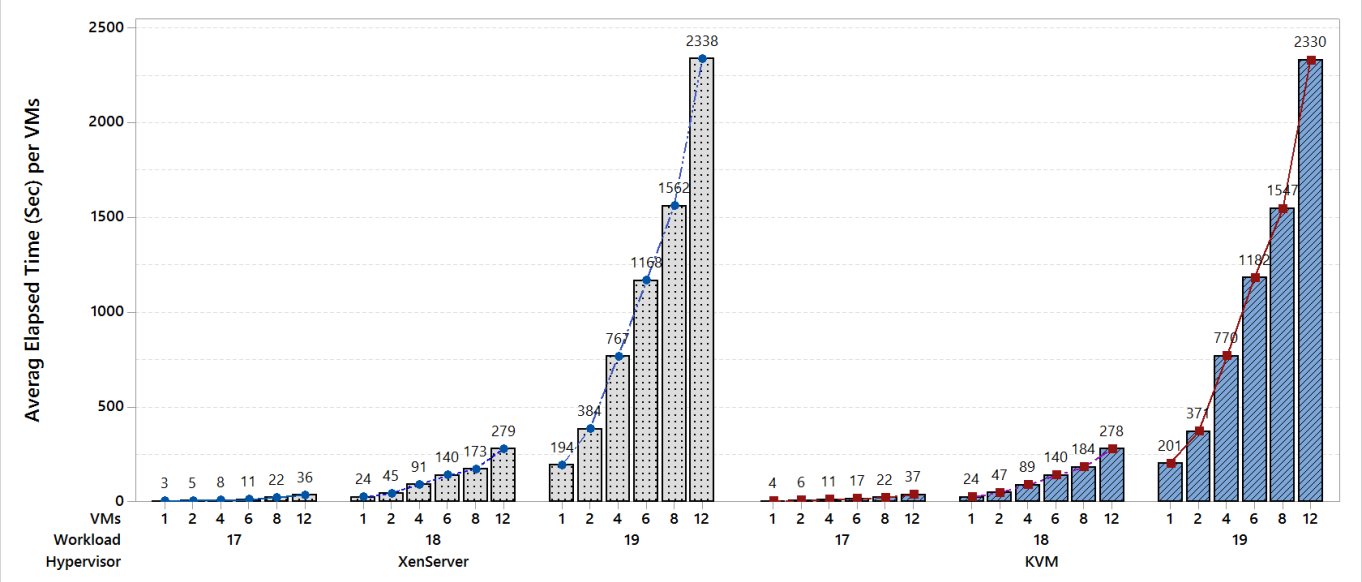

Fig. 13. The effect of Virtual Machines on Performance and the Comperision between Commercial and Open source Hypervisors

VMs Allocationin, half VMs were busy due to heavy workload while other four VMs which have low and medium workload became idle due to low and medium workload.

Figure 11, shows the significance of over allocation of vCPUs and the result of two pinning stratagies. It is shown, that after 28 and 26 seconds (depending on pinning strategies), the VMs having low workload became idle. The other six VMs having medium and high workload were still busy. But, this time the CPU utilization level for no pinning strategy did not drop from 100% as shown in Figure 12. Based on pinning strategies, the hypervisor may or may not assign the idle vCPUs of two idle VMs (having low workload) to VMs having the medium and high workload while no pinning strategy assign the idle vCPUs to medium and heavy workload VMs. After 180 and 178 seconds (depending on pinning strategies), the VMs having medium workload also became idle. In the case of pinning strategy, four out of eight VMs are idle while due to the heavy workload four VMs are still busy. In this experiment, no pinning strategy takes the advantage of over allocation of vCPUs, by using the free vCPUs.

In addition, the CPU utilization level for no pinning strategy did not drop from 100% as shown in Figure 12 as compare with Uniform vCPUs-VMs Allocation shown in Figure 10. As a result, the performance (elapsed time) is improved and the elapsed time to solved the N-Queens problem for the heavy workload is minimized (i.e., almost half as compare with other two pinning strategies). Moreover, using pinning strategy we can not utilized the free vCPUs, because pinning strategy restricted VMs to run on user defined vCPUs. In conclusion, both pinning strategies have only effect on over allocation of computing resources and it will give better performance (i.e., elapsed time). In addition, the higher CPU utilization can be achieved using over allocation of vCPUs-VMs with no pinning strategy. As a future direction, we need to investigate the effect of pinning strategies on performance as well.

D. Hypervisors Comparison

After investigating the effect of hypervisor and vCPU on performance. In this experiment, we compared commercial and open source hyperviosrs. In this experiment, we vary the number of VMs running on the top Citrix xenServer and KVM hypervisors. We also vary the number of vCPUs allocation to VMs as well as the workloads. As we already discussed, based on the possible solutions to solve N-Queens problem and the corresponding elapsed time, we chose problem size 17, 18, and 19 as low, medium, and heavy workload.

The maximum performance (i.e., low elapsed time) can be achieved if 100% CPU is utilized, which is one of the main objectives of cloud computing i.e., 24 out of 24 logical CPUs are utilized regarding our machines specification. However, we can not run many VMs per our need, because the performance of the system decreases with increases in number of VMs. In order to achieved better performance (i.e., low elapsed time to solved N-Queens problem) and maximum CPU utilization, we need to consume all CPU cores in such a way that the performance of the system will not be affected. Therefor, we carried out experiments, where all the CPU cores were allocated to active VMs.

As shown in Figures 7 and 8, physical CPUs are utilized 100%. In addition, Figure 13, depicts the effect of VMs on performance as well as the comparison between Citrix xenServer and KVM hyperviosr for three different workload. It can be observed that no significant difference was identified for each workload in either of the hypervisors. In this experiments, the number of vCPUs were kept fixed, while the number of workload and active VMs were varied. Furthermore, 24 out of 24 logical CPUs were equally divided among VMs, such

as: 24 CPUs cores were assigned to one VM; 12 vCPUs, 6 vCPUs, 4 vCPUs, 3 vCPUs and 2 vCPUs were allocated to other five test cases.

If we compare the average elapsed time of each test case (i.e., 1 VM with 2 VMs; 2 VMs with 4 VM; 4 VMs with 8 VMs) using any workload, the average elapsed time is almost double, although the CPU utilization level is 100%. In one VMs test case, total available physical CPU resources (24 out of 24 logical CPUs) are allocated to one VMs. Therefore the total elapsed time is minimum. For two VMs test case, the VMs time share the CPU resources such as 50% CPU is be used by VM1 and 50% is used by VM2, therefore the elapsed time is almost double (98% increase) by comparing with one VM test case and so on.

V. Conclusions and Future Work

In this paper, we analyzed the latest version of commercial and open source hypervisors. We also analyzed the implication of virtualization technology layer and vCPU on performance in terms of CPU utilization. In addition, we proposed a suitable vCPUs configuration for VMs in cloud environments. Cloud service providers and researchers will get the benefits when they deploy VMs in a cloud environment or evaluate open source and commercial hypervisors.

The results obtained from this evaluation showed that commercial and open source (KVM) hypervisors have similar performance in terms of elapsed time and CPU utilization. As per our observation, the performance of a system would degrade by running many VMs, improper allocation of vCPUs to VMs, or using unsuitable vCPUs-pCPUs pinning strategies. Moreover, we have found that elapsed time increases when there is a massive over allocation of vCPUs.

We suggested that the cloud service providers and researchers should consider the effects of massive over allocation of vCPUs and VMs when they choose deployment strategies for better performance and best CPU resources allocation. The obtained results from our evaluation experiments can be validated using other commercial hypervisors (VMware, Hyper-V, and RHV). This will aid the Cloud service providers in choosing which hypervisor to use for their specific needs.

ACKNOWLEDGEMENTS

This work is supported in part by the National Natural Science Foundation of China under Grants 61632009 & 61472451, in part by the Guangdong Provincial Natural Science Foundation under Grant 2017A030308006, and High-Level Talents Program of Higher Education in Guangdong Province under Grant 2016ZJ01.

References

[1] D. Minarolli, E. K. Meçe, E. Sheme, and I. Tafa, "Simplecloud: A simple simulator for modeling resource allocations in cloud computing," in *International Conference on Emerging Internetworking, Data & Web Technologies*. Springer, 2018, pp. 572–583.

[2] u. R. Hafiz, Farag, A. Shawahna, F. Sajjad, and A. S. Abdulrahman, "Performance evaluation of vdi environment," in *Innovative Computing Technology (INTECH), 2016 Sixth International Conference on*. IEEE, 2016, pp. 104–109.

[3] P.-J. Chuang and C.-Y. Chou, "Srvc: An efficient scheduler for concurrent virtual machines over the xen hypervisor," *Journal of Applied Science and Engineering*, vol. 20, no. 3, pp. 355–365, 2017.

[4] J. Sahoo, S. Mohapatra, and R. Lath, "Virtualization: A survey on concepts, taxonomy and associated security issues," pp. 222–226, 2010.

[5] T. Wang, Y. Li, G. Wang, J. Cao, M. Z. A. Bhuiyan, and W. Jia, "Sustainable and efficient data collection from wsns to cloud," *IEEE Transactions on Sustainable Computing*, 2017.

[6] P. Aghaalitari, "Development of a virtualization systems architecture course for the information sciences and technologies department at the rochester institute of technology (rit)," *Master Thesis, Rochester Institute of Technology*, 2014.

[7] Q. Liu, G. Wang, X. Liu, T. Peng, and J. Wu, "Achieving reliable and secure services in cloud computing environments," *Computers & Electrical Engineering*, vol. 59, pp. 153–164, 2017.

[8] J. Hwang, S. Zeng, F. y Wu, and T. Wood, "A component-based performance comparison of four hypervisors," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. IEEE, 2013, pp. 269–276.

[9] S. Varrette, M. Guzek, V. Plugaru, X. Besseron, and P. Bouvry, "Hpc performance and energy-efficiency of xen, kvm and vmware hypervisors," in *2013 25th International Symposium on Computer Architecture and High Performance Computing*. IEEE, 2013, pp. 89–96.

[10] A. Elsayed and N. Abdelbaki, "Performance evaluation and comparison of the top market virtualization hypervisors," in *Computer Engineering & Systems (ICCES), 2013 8th International Conference on*. IEEE, 2013, pp. 45–50.

[11] Z. Zhou, M. Dong, K. Ota, G. Wang, and L. T. Yang, "Energy-efficient resource allocation for d2d communications underlaying cloud-ran-based lte-a networks," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 428–438, 2016.

[12] A. Zhong, H. Jin, S. Wu, X. Shi, and W. Gao, "Performance implications of non-uniform vcpu-pcpu mapping in virtualization environment," *Cluster Computing*, vol. 16, no. 3, pp. 347–358, 2013.

[13] K. Kourai and R. Nakata, "Analysis of the impact of cpu virtualization on parallel applications in xen," in *Trustcom/BigDataSE/ISPA, 2015 IEEE*, vol. 3. IEEE, 2015, pp. 132–139.

[14] S. Shirinbab and L. Lundberg, "Performance implications of over-allocation of virtual cpus," in *Networks, Computers and Communications (ISNCC), 2015 International Symposium on*. IEEE, 2015, pp. 1–6.

[15] C. D. Graziano, "A performance analysis of xen and kvm hypervisors for hosting the xen worlds project," 2011.

[16] A. Babu, M. Hareesh, J. P. Martin, S. Cherian, and Y. Sastri, "System performance evaluation of para virtualization, container virtualization, and full virtualization using xen, openvz, and xenserver," in *Advances in Computing and Communications (ICACC), 2014 Fourth International Conference on*. IEEE, 2014, pp. 247–250.

[17] C. Mancaş, "Performance improvement through virtualization," in *2015 14th RoEduNet International Conference-Networking in Education and Research (RoEduNet NER)*. IEEE, 2015, pp. 253–256.

[18] W. Graniszewski and A. Arciszewski, "Performance analysis of selected hypervisors (virtual machine monitors-vmms)," *International Journal of Electronics and Telecommunications*, vol. 62, no. 3, pp. 231–236, 2016.

[19] M. ARIF, G. WANG, and V. E. BALAS, "Secure vanets: Trusted communication scheme between vehicles and infrastructure based on fog computing," *Studies in Informatics and Control*, vol. 27, no. 2, pp. 235–246, 2018.

[20] Q. Liu, Y. Guo, J. Wu, and G. Wang, "Effective query grouping strategy in clouds," *Journal of Computer Science and Technology*, vol. 32, no. 6, pp. 1231–1249, 2017.

[21] PTS, "Phoronix test suite," *[Online]. Available: http://www.phoronix-test-suite.com , [Accessed 21 April 2018]*, 2018.

[22] K.-L. Du and M. Swamy, "Tabu search and scatter search," in *Search and Optimization by Metaheuristics*. Springer, 2016, pp. 327–336.

[23] A. Fonseca, N. Lourenço, and B. Cabral, "Evolving cut-off mechanisms and other work-stealing parameters for parallel programs," in *European Conference on the Applications of Evolutionary Computation*. Springer, 2017, pp. 757–772.

[24] S. Shirinbab, L. Lundberg, and D. Ilie, "Performance comparison of kvm, vmware and xenserver using a large telecommunication application," in *Cloud Computing*. IARIA XPS Press, 2014.