2010 12th IEEE International Conference on High Performance Computing and Communications

Analyzing & Modeling the Performance in Xen-based Virtual Cluster Environment

Kejiang Ye, Xiaohong Jiang, Siding Chen, Dawei Huang, Bei Wang

College of Computer Science, Zhejiang University
Hangzhou 310027, China
{yekejiang,jiangxh,chensd,davidhuang,wangbei}@zju.edu.cn

***Abstract*—Virtualization technology is currently widely used due to its benefits on high resource utilization, flexible manageability and powerful system security. However, its use for high performance computing (HPC) is still not popular due to the unclearness of the virtualization overheads. It's worthy to evaluate the virtualization cost and to find the performance bottleneck when running HPC applications in virtual cluster. We first evaluate the basic performance overheads due to virtualization. Then we create a 16-node virtual cluster and perform a performance evaluation for both para-virtualization and full virtualization. After that, we evaluate the MPI (Message Passing Interface) scalability due to its importance to HPC applications. In addition to the macro assessment, we use the Oprofile/Xenoprof to investigate the architecture characterizations like CPU cycle, L2 cache misses, DTLB misses and ITLB misses which are auxiliary explanation to the performance bottleneck. Experimental results indicate that performance overheads of virtualization are acceptable for HPC, para-virtualization is very suitable for HPC due to the high virtualization efficiency and efficient inter-domain communication. Finally, we use the non-linear regression modeling technology to present a performance model for network latency and bandwidth to predict the performance in virtual cluster environment.**

***Keywords*-Virtualization; Xen; Performance; HPC; Modeling**

I. Introduction

Virtualization technology becomes increasing popular in modern data center and cloud computing. Applying virtualization technology in HPC environments is a new and challenging domain [1–4]. It is obvious that virtualization brings many benefits to HPC environments such as flexible resource management, high reliability, performance isolation and OS customization, etc. However, the trend of HPC applications running in virtualized environment is not so obvious, because whether the virtualized environment can meet the high performance requirements of HPC applications is not clear yet due to the virtualization overheads. What's more, since the presence of a variety of virtualization methods (classical trap-and-emulate, binary translation and hardware virtualization), it is necessary to choose a best suitable virtualization method for HPC area. Further, MPI (Message Passing Interface) scalability is a key attribute in HPC environment and how the virtualization can affect the MPI scalability is an important research topic. Based on the above analysis, it is essential to analyze the performance overheads, compare the virtualization efficiency of different virtualization methods and investigate the MPI scalability.

The performance of applications running in a virtual machine is different from that in the native environment due to the existing of virtual machine monitor. A lot of work has been done in performance evaluation on single virtual machine (VM) on single physical machine focusing on CPU, memory, disk I/O and network [5–9], and the performance issues of server consolidation [10–12]. However, to our knowledge, few work has been done on the performance overhead of HPC applications running in virtualization environment. Some of the researchers have investigated into Xen's para-virtualization performance with HPC benchmarks and real world HPC applications [1, 13–15]. However they didn't perform a deep analysis about the performance overhead and network I/O processing mechanism. What's more, they didn't refer to Xen's full virtualization mechanism and have no hardware profiling analysis from the hardware architecture perspective.

In this paper, we firstly study the basic virtualization overhead by comparing the performance of virtual machines running both in para-virtualized and full virtualized modes with a physical machine. Then we create a 16-node virtual cluster, and do a comprehensive performance comparison of para-virtualization and full virtualization to investigate virtualization efficiency for HPC applications, including computing performance, memory performance, data transfer rate, network bandwidth and latency. After that, we evaluate the MPI scalability of HPC applications running in virtual cluster. Besides, we also investigate the profiling data from the hardware characterization when running HPC applications in virtualization environment with Oprofile/Xenoprof toolkit. We focus on five hardware events CPU_CLK_UNHALTED, INST_RETIRED, LLC_MISSES, DTLB_MISSES and ITLB in our Intel platform for gathering CPU cycles, instruction number, L2 cache misses, DTLB misses and ITLB misses respectively which are helpful for explaining the performance bottleneck.

Experimental results show that: 1) performance overheads of virtualization are acceptable for HPC, 2) para-virtualization is more suitable for HPC due to the high virtualization efficiency and optimized network I/O processing mechanism compared to full virtualization, 3) the Front-

978-0-7695-4214-0/10 $26.00 © 2010 IEEE
DOI 10.1109/HPCC.2010.79

End/Back End network I/O mechanism of para-virtualization can cause fewer traps than emulated I/O mechanism of full virtualization and performs better performance in inter-domain communication, 4) the MPI and network communication overheads in HPC applications are the main bottleneck for full virtualized cluster, which cause huge L2 cache miss rate.

To predict the performance in large-scale virtual cluster environment, we present a performance model for network latency and bandwidth based on the non-linear regression modeling technology and pursue an accuracy with more than 98%, which indicates the model can effectively predict the performance.

The rest of the paper is structured as follows. In Section II, we introduce the background of applying virtualization in HPC environment, network I/O virtualization in Xen, and a system profile tool Oprofile/Xenoprof. In Section III, we present our experimental methodology to study the performance overheads of virtualization in HPC environment. In Section IV, we perform a comprehensive evaluation and analysis on the virtualization overheads and efficiency and use the profiling data to analyze the performance bottleneck. In Section V, we present a performance model for the network based on the experiment discovery. Section VI presents the related work. Finally we give our conclusion and future work in Section VII.

II. Background & Motivation

A. Requirements of Virtualization for HPC

There are several requirements should be satisfied when virtualization is applied to HPC system. Firstly, the performance overheads of virtualization must not have significant impact on the system performance, especially on the performance-critical HPC applications. Secondly, the virtualization technology should improve the administration efficiency for HPC system since virtualization holds the advantages of fast creating and shutdown the VM, centralized management of VM images, and flexible resource distribution. Thirdly, virtualization should ensure the reliability and security of HPC system which can minimize the downtime from crashes, isolate applications and implement the migration of workloads or whole VM easily.

B. Xen and it's Network I/O Virtualization

Xen is a popular open-source x86 virtual machine monitor (VMM) that allows multiple instantiation operating systems (OSes) running concurrently on a single physical machine. Xen supports both full virtualization and para-virtualization scheme. The full virtualization is implemented based on the hardware assistant method (Intel VT [16] and AMD-V [17]). Full virtualization technology virtualizes the underlying hardware and provides a unified abstraction for above level softwares, both the guest OS and the applications which are not required to be modified and are not aware of the virtualized environment when running in the VM. This approach provides high security because of the isolation between the applications in different VMs. However, the performance overheads of virtualization can not be neglected due to the frequent traps into the hypervisor. While para-virtualization avoids this phenomenon by modifying the kernel of guest operating system and host operating system to ensure that guest operating system can be aware of the virtual environment and achieves better performance.

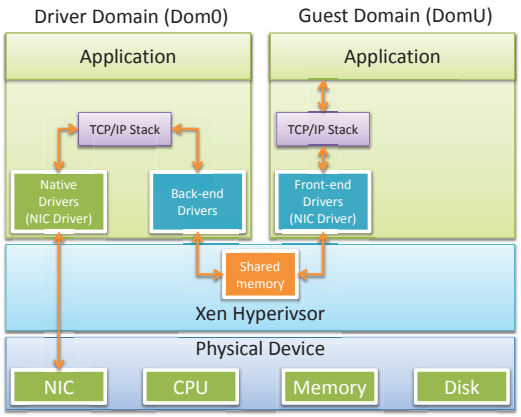

Figure 1. The Front-End/Back-End Virtualized Network I/O in Xen.

Xen uses two I/O rings for network virtualization that are used to transfer data between virtual machine and Xen, one for outgoing packets and one for incoming. When the NIC has a packet request, it first generates an interrupt. Then the VMM forwards the interrupt to the driver domain (Dom0), and the Dom0 DMAs the packet into the reception I/O ring. After the TCP/IP stack, Dom0 directly copy data from the back-end driver to the front-end driver in DomU. When the packet reaches the front-end driver, the back-end driver requests the VMM to send a virtual interrupt to notify the DomU of the new packet. Finally, the packet is processed in the DomU.

Fig. 1 shows the Xen's para-virtualized Front-End/Back-End virtualized network I/O mechanism. The privileged VM (Dom0) runs a modified version of OS that uses native driver to manage physical devices. Other VMs (DomU) communicate with Dom0 to transmit and receive packets through shared memory I/O channels.

C. Oprofile and Xenoprof

OProfile is a system profiling tool for Linux. It is capable of profiling all parts of a running system, from kernel to shared libraries and binaries. It is an ideal tool for profiling entire systems to determine bottlenecks in real-world systems.

Xenoprof [7] is an extended OProfile tool that is a system-wide statistical profiling toolkit implemented for Xen virtualization environments. It allows coordinated profiling of multiple VMs in a system to obtain the distribution of

hardware events such as clock cycles, instruction number, cache and TLB misses, etc. This is useful for finding the performance bottleneck and can help optimize the performance of Xen.

In this paper, we use the above tools to gather the events like CPU Cycles, L2 cache misses, DTLB misses and ITLB misses, etc. Based on the profiling data, we can find the bottlenecks while running HPC applications in Xen virtualization environment.

III. Experimental Methodology

A. Experimental Configuration

All experimental evaluations are performed on the Dell 2900 PowerEdge server, with 2 Quad-core 64-bit Xeon processors at 1.86 GHz. We use Ubuntu 8.10 with kernel version 2.6.27 in domain 0, and the version of Xen hypervisor is the 3.3.1, which has built-in support for Oprofile. Each virtual machine is installed with CentOS 5.2 as the guest OS with 4 VCPUs and 256MB memory size. We choose MPICH 2.1.0.8 as our MPI environment.

B. Profiling Tool

We use Oprofile 0.9.3 with Xen patch as our data gather tool. We set CPU counter frequency with 100000. This means that Oprofile will generate a sample for every 100000 occurrences of a specific event such as DTLB miss.

C. Benchmark

We use the HPC Challenge Benchmark suite (HPCC) [18] for our study that is commonly used for HPC measurements. The HPCC suite is a comprehensive set of synthetic benchmarks designed to profile the performance of several aspects of a cluster. The testing applications used in our study are listed in Table I:

Table I
Measurement Metrics of HPCC Benchmark Suite

Benchmarks	Measurement Metrics
HPL	floating point rate of execution for solving a linear system of equations
DGEMM	floating point rate of execution of double precision real matrix-matrix multiplication
FFT	floating point rate of execution of double precision complex one-dimensional DDF (Discrete Fourier Transform)
PTRANS	rate of transfer for large arrays of data from multiprocessor's memory
STREAM	memory bandwidth and the corresponding computation rate for simple vector kernel
RandomAccess	rate of integer random updates of memory
Latency	Communication latency
Bandwidth	Communication bandwidth

There are three running modes: **single** means that a single processor runs the benchmark, **star** means all the processors run separate independent copies of the benchmark with no communication, **mpi** means all processing elements run the benchmark in parallel using explicit data communications.

In our experiments, three problem sizes were evaluated, they are 1000MB, 2000MB, 3000MB. The block and grid sizes used are common-block : 80, 100, 120; grid: 2*2, 1*4, 4*1.

IV. Experimentation Results and Analysis

In this section, we describe the measurement results of HPCC Benchmark running both in para-virtualized environment and full virtualized environment.

In order to ensure the data precision, each of the showed experiment results was obtained via running benchmark five times on the same configuration, the highest and lowest values for each test were discarded, and the remaining three values were averaged.

A. Virtualization Overheads

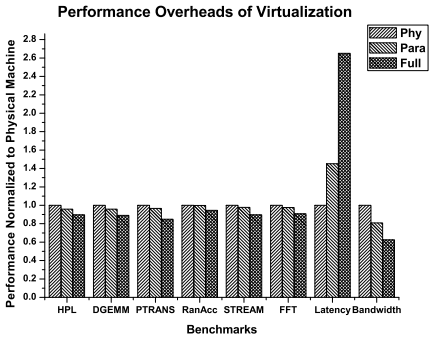

Figure 2. The Performance Comparison of Physical Machine, Para-virtualized VM and Full virtulized VM using HPCC Benchmark.

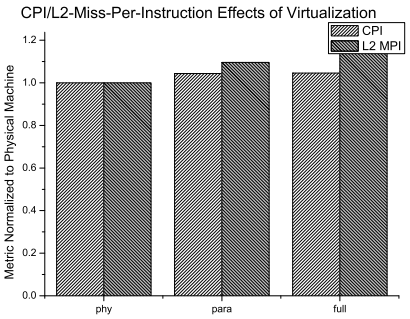

Figure 3. The CPI and L2-Miss-Per-Instruction Comparison of Physical Machine, Para-virtualized VM and Full virtualized VM.

Firstly, we investigate the basic virtualization overheads for high performance computing using HPCC benchmark. We compare the performance running in virtual machine with that in physical machine. We allocate 512MB virtual

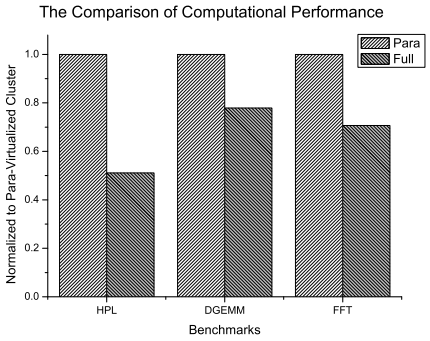

Figure 4. The Comparison of Computational Performance in a 16-node Para-virtualized and Full virtualized Cluster.

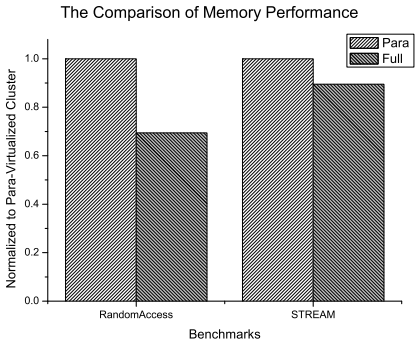

Figure 5. The Comparison of Memory Performance in a 16-node Para-virtualized and Full virtualized Cluster.

memory and 4 VCPU for the virtual machine when testing the virtual machine performance, and allocate 512MB physical memory and shutdown one physical CPU to maintain the similar environment with virtual machine when testing the physical machine performance.

Fig. 2 shows the performance of physical environment, para-virtualized environment and full virtualized environment that setting the running thread counter to 8 for both physical machine and virtual machine. We normalized the results of each test to a fraction of the physical performance. The HPL, DGEMM and FFT sub-benchmarks test the computation capability from different aspects. While RandomAccess and STREAM test the memory performance and PTRANS tests the data transfer rate. From the above 6 sub-benchmarks, we find that virtualizaiton causes a little performance loss so that para-virtualization perform better than full virtualization. However, from the network latency and bandwidth analysis, we find virtualization causes significant network overheads and full virtualization is particularly evident.

In order to find the performance overheads of virtualization from architecture perspective, we use the Oprofile/Xenoprof tool to gather the hardware events when running HPCC benchmark. Fig. 3 illustrates the CPI (Cycle per instruction), L2 cache misses per instruction, from that we find the L2 cache miss rate is one of the major reason degrading the performance.

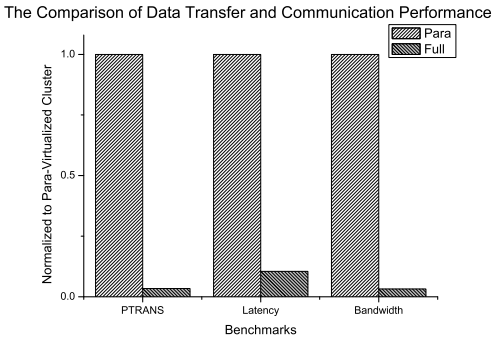

Figure 6. The Comparison of Data Transfer Rate and Communication Performance in a 16-node Para-virtualized and Full virtualized Cluster (the time property of latency benchmark is transformed into performance).

B. Virtualization Efficiency of Para-virtualization and Full Virtualization

In this section, we compare the virtualization efficiency between para-virtualization and full virtualization environments. To simulate the HPC environment, we create two 16-node virtual clusters, one is for para-virtualization, the other is for full virtualization and set the MPI running threads to 16. Due to the space limitation, we only present the results running in **single** mode that means only a single processor of each virtual machine runs the benchmark workloads. Fig. 4-6 show the detailed results from different aspects. We classify the results into computational performance, memory performance, data transfer rate and communication performance.

1) Computational Performance: Fig. 4 illustrates the computational performance with different benchmarks. From the figure, we found that the performance of full virtualized cluster is worse than para-virtualized cluster at a degradation of 48.86% in HPL testing, 22.06% in DGEMM testing and 29.35% in FFT testing. It indicates that the computational applications are sensitive to the virtualization in different degrees. The poor performance of full virtualization is due to the high MPI communication overheads when running parallel workloads in the virtual cluster which become the major factor affecting the full virtualization performance.

2) Memory Performance: Fig. 5 shows the memory performance of the virtual cluster. The STREAM performance of full virtualization is very close to the para-virtualization due to the high memory virtualization efficiency for both para-virtualization and full virtualization. While in the RandomAccess testing, full virtualization obtains very poor performance. It is because, RandomAccess incurs some

processor communication overheads which cause significant performance degradation in full virtualized cluster.

3) Data Transfer Rate and Communication Performance: From Fig. 6, we find the data transfer rate and communication performance of full virtualization is very poor which is not acceptable in the HPC environment. All the three benchmarks perform a large number of communication operations which cause high performance overheads. We note that the network latency and bandwidth performance is worse than that in Fig. 2. This is because there are additional communication overheads between virtual machines in the virtual cluster. It again indicates the communication overheads is a bottleneck in full virtualized cluster.

Profiling Data Analysis We collect the profiling data from several hardware events when running HPCC in the virtual cluster environment. Fig. 7 shows the CPI (Cycle per instruction) and L2 cache misses per instruction. It is obvious that the L2 cache misses is the main architecture reason affecting the virtualization efficiency.

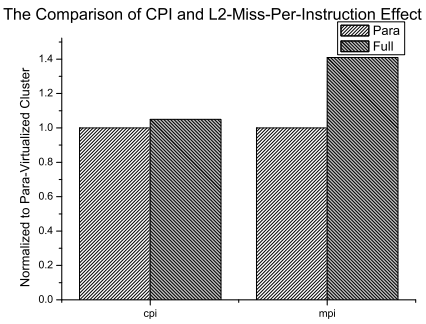

Figure 7. The Comparison of CPI and L2-Miss-Per-Instruction in a 16-node Para-virtualized and Full virtualized Cluster.

C. Measuring Scalability in Virtual Cluster

The MPI scalability is a crucial metric in high performance computing. In this section, we focus on the MPI scalability for our study since the HPC benchmarks commonly employ MPI for communication. We fix the MPI running threads to 8 when the virtual machine number scales from 1 to 16.

Fig. 8-15 illustrate the scalability of different benchmarks. We add the virtual machine into the virtual cluster one by one. From the figures, we found that the para-virtualizaiton keeps a stable state when the virtual machine number scales from 1 to 16 in all the testing cases. From Fig. 8, it is obvious that full virtualization shows a poor scalability when the virtual machine number scales. Although the performance of full virtualization is very close to the para-virtualization when in one virtual machine, there is a obvious sharp decline in the performance of full virtualization when the virtual machine number scales. The reason is that communication overheads between virtual machines become the bottleneck in virtual cluster environment when more virtual machines are added.

The scalability of DEGEMM in Fig. 9 and STREAM in Fig. 12 show a similar trend, but the para-virtualization is slightly better than the full virtualization. It is because they are CPU and memory intensive and hold the similar virtualization efficiency.

It is interesting to note the results of **mpi** mode in Fig. 11 and 13 and the communication performance in Fig. 10, 14 and 15 in which we find that para-virtualization has a slight increase in the performance when the virtual machine number scales from 1 to 16. It is because, in the para-virtulized Front-End/Back-End network I/O mechanism, the network interfaces can DMA the data into a buffer owned by the back end, and read the packet header and decide where to send the data, and it then has to either copy the data or remap the page, either of which is particularly cheap. It means the local VM to local VM traffic doesn't need to go via the network interface at all. This mechanism can achieve the best performance, because there is no Domain0 interaction required beyond the initial setup and pursues good MPI scalability. However, there are frequent traps into the VMM when the packet requests arrive and sacrifice a lot of performance in the emulated network I/O mechanism in full virtualization. We will further explain it by profiling analysis using Oprofile/Xenoprof.

Profiling Data Analysis We monitor the hardware events when running the HPCC benchmark. Fig. 16 and 17 show the CPI and L2 misses per instruction. Obviously, the CPI keeps unchanged while both full virtualization and para-virtualization have an increase in the L2 cache misses rate. As more virtual machines are involved in the calculation of applications, the full virtualization incurs a quick increase in L2 cache miss rate (Fig. 17). Considering to the macro analysis and micro analysis, we can conclude that communication of full virtualization can cause a poor scalability, and the increasing of L2 cache misses rate is the main architecture reason.

V. Characterization Analysis and Modeling

We present a more detailed analysis of the profiling data collected by Oprofile/Xenoprof tool to understand where the performance loss comes. Further, based on the fact that the communication is a key factor leading performance degradation in the virtual cluster, we construct a performance model for the network latency and bandwidth in Xen virtualization environment.

As described in Section III, samples gathered from Oprofile/Xenoprof are based on a frequency of 100,000 clock cycles. Oprofile/Xenoprof extracts CPU samples from CPU_CLK_UNHALTED event, DTLB samples from DTLB_MISSES event, L2 cache samples from LLC_MISSES event and ITLB samples from ITLB event.

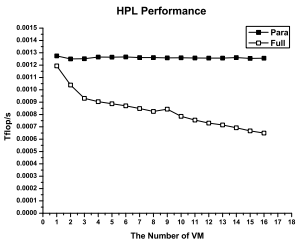

Figure 8. HPL Scalability

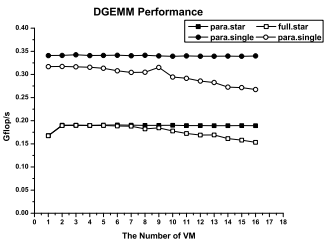

Figure 9. DGEMM Scalability

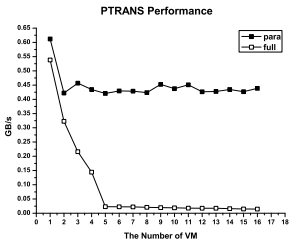

Figure 10. PTRANS Scalability

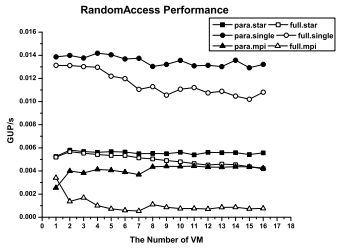

Figure 11. RandomAccess Scalability

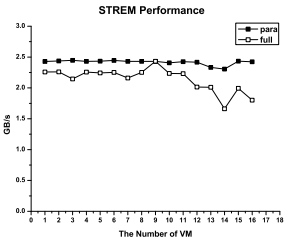

Figure 12. STREAM Scalability

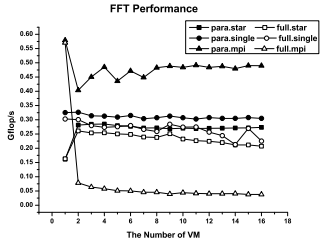

Figure 13. FFT Scalability

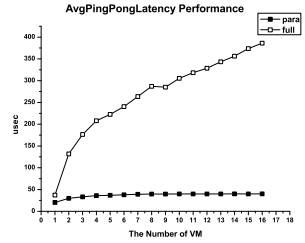

Figure 14. Scalability of Latency

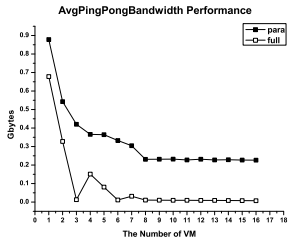

Figure 15. Scalability of Bandwidth

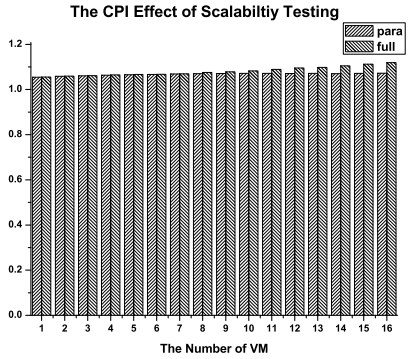

Figure 16. The CPI Effect of Scalability Testing in Para-virtualized and Full virtualized Environment

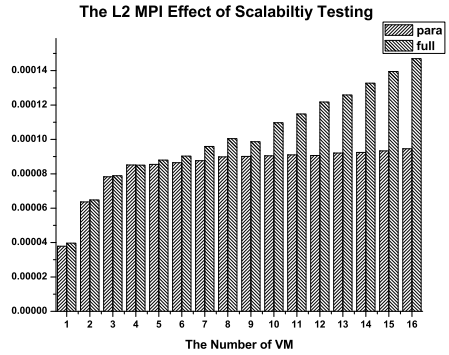

Figure 17. The L2-Miss-Per-Instruction Effect of Scalability Testing in Para-virtualized and Full virtualized Environment

A. Virtualization Performance Bottleneck Analysis

Table II
COMPARISON OF CPU CYCLES, L2 CACHE MISSES, DTLB_MISS AND ITLB SAMPLES ACROSS PLATFORMS

	CPU_CLK_ UNHALTED	**INST_** RETIRED	**LLC_** MISS	**DTLB_** MISSES	**ITLB**
Native	245647018	243455303	8421	145129	57
Para	266969704	253184882	9597	192273	159
Full	260904876	247263038	9802	255229	1932

In Section IV, we have already found that virtualization will lose a part of performance under the HPCC benchmark testing. We have shown that the CPI (cycles per instruction) and L2 cache misses rate increase after virtualization that help explain the performance penalty of virtualization. Table II shows the detailed virtualization performance penalty in native environment as well as on virtual machine in term of the number of samples of five events when running HPCC benchmark. From the Table II, the LLC_MISS samples of para-virtualization and full virtualization are 113.97% and 116.40% of native environment respectively, from which we can conclude that the performance incurs a certain degree of performance overhead. The DTLB_MISSES and ITLB samples can even account for it.

B. Modeling the Network Performance

Our primary motivation for creating a performance model for network performance is to predict the major performance bottleneck in virtual cluster. Considering the experimental results in the above section, we can conclude that the communication penalty is the bottleneck of virtual cluster. We use the regression modeling techniques that have been used to model application behavior in a non-virtualized en-

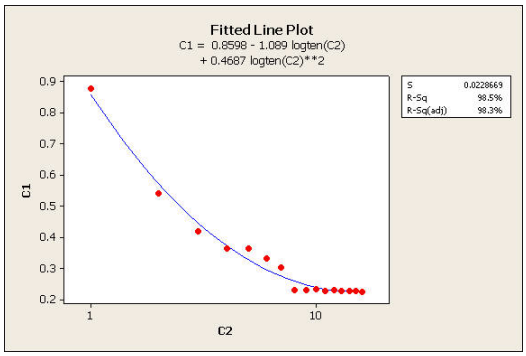

Figure 18. Model the Network Bandwidth Performance of Para-virtualized Cluster.

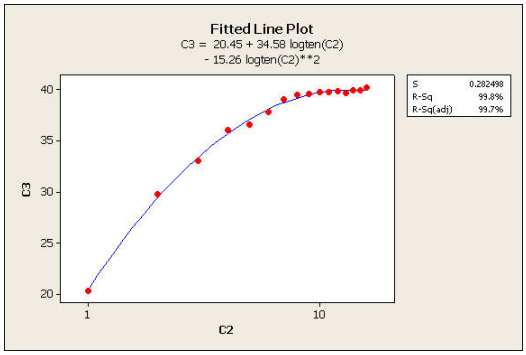

Figure 19. Model the Network Latency Performance of Para-virtualized Cluster.

vironments. We examine the suitability of several regression models including linear regression and non-linear regression. However, we find the accuracy of linear regression is not acceptable. So we try to use the non-linear regression model. We present the performance model of network bandwidth of para-virtualized cluster as below:

$$C1 = 0.8598 - 1.089lg(C2) + 0.4687lg(C2)^2 \quad (1)$$

and network latency as below:

$$C3 = 20.45 + 34.58lg(C2) - 15.26lg(C2)^2 \quad (2)$$

Here C1, C2 and C3 denote the network bandwidth, VM number and the network latency respectively. Fig. 18 and 19 illustrate the fitting process, the goodness of the fitting of network bandwidth and latency are 98.5% and 99.8% respectively. That is to say, our prediction of network performance can achieve very high accuracy. With these two models, we can easily predict the performance of virtual cluster with any number of virtual machines. We also apply this method to model the network performance in full-virtualized cluster and obtain a good result which means the modeling method can applied to other virtualization environment.

VI. Related Work

Many research has analyzed the performance overheads of virtualization in single virtual machine [5–7] and server consolidate [12] scenario using traditional benchmarks focusing on CPU, memory, I/O and network.

The virtualization used in HPC domain is a new and challenging topic that has been investigated in several recent research efforts. The work described in [2, 14] evaluate the performance impact of high performance codes running MPI in Xen para-virtualization environment. However, it leaves the full virtualization scheme untouched. Work in [1] gives the performance and management overheads of VM-based HPC framework based on VMM bypass I/O scheme and InfiniBand. Work in [19, 20] compared different virtualization technologies impact on HPC applications, including para-virtualization, full virtualization and OS level virtualization, however they only evaluate the macro performance using benchmarks and not refer to a deep analysis into architecture characterization.

Work reported in [7] is the first time to use Xenoprof to diagnose the performance overheads in Xen. The authors focus on the network applications running in the VM, and use the information extracted using Xenoprof to uncover bugs and optimize Xen. Work in [21] uses a real scientific application to evaluate the virtualization performance and also use the Oprofile tool to better understand the overheads of virtualization. Recently, Kundu et al. [22] modeled the application performance by using the CPU, memory and I/O parameters to train the artificial neural network model. However they didn't involve the HPC applications.

VII. Conclusion and Future Work

Our study was motivated by the interests in using virtualization technology in HPC environment. However, the trend of running HPC applications in virtualized environment is not yet lucid due to the virtualization overheads, which may affect the efficiency of performance-critical HPC applications. In this paper we have studied and analyzed the HPCC benchmark suite on native, para-virtualized and full virtualized environment, and did a comprehensive performance analysis of para-virtualization and full virtualization in a 16-node virtual cluster. We study virtualization efficiency from performance and scalability perspective. We not only concern with the final performance penalty produced by running the benchmark but also the detailed hardware event data that reflect the performance penalty using the Oprofile/Xenoprof tool.

Experimental results indicate that: 1) virtualization does bring performance overheads for HPC applications, but within the acceptable range. 2) Para-virtualization is very suitable for HPC due to the high virtualization efficiency which minimizes the traps into the VMM and an optimized Front-End/Back-End network I/O processing mechanism

which has an efficient inter domain communication mechanism without interacting with domain0. 3) The MPI and network communication overheads in HPC applications are the main bottleneck for full virtualization that cause huge L2 cache miss rate, but not for para-virtualized cluster. That is to say, para-virtualization technology is very suitable to be applied to high performance computing due to the other additional benefits brought by virtualization such as high management efficiency and excellent isolation and security.

In addition, we present a performance model for network communication overheads in virtual cluster that can be used to predict the network performance in virtual cluster with any number of nodes. The fitting data indicate the model has a high accuracy with 98.5% for bandwidth and 99.8% for latency.

Future work will include optimizing the virtualization performance for HPC, and creating a more comprehensive performance model to predict the performance in virtual cluster environment.

ACKNOWLEDGMENT

We are grateful to the anonymous reviewers for their comments and suggestions on the paper. This work is funded by the National 973 Basic Research Program of China under grant NO.2007CB310900 and National Natural Science Foundation of China under grant NO. 60970125.

REFERENCES

[1] W. Huang, J. Liu, B. Abali, and D. K. Panda, "A case for high performance computing with virtual machines," in *ICS '06: Proceedings of the 20th annual international conference on Supercomputing*, 2006, pp. 125–134.

[2] L. Youseff, R. Wolski, B. Gorda, and C. Krintz, "Paravirtualization for hpc systems," in *XHPC '06: ISPA Workshop on Xen in HPC Cluster and Grid Computing Environments*, 2006, pp. 474–486.

[3] M. F. Mergen, V. Uhlig, O. Krieger, and J. Xenidis, "Virtualization for high-performance computing," *SIGOPS Oper. Syst. Rev.*, vol. 40, no. 2, pp. 8–11, 2006.

[4] G. Vallée, T. Naughton, C. Engelmann, H. Ong, and S. L. Scott, "System-level virtualization for high performance computing," in *PDP '08: Proceedings of the 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP 2008)*, 2008, pp. 636–643.

[5] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 164–177, 2003.

[6] B. Clark, T. Deshane, E. Dow, S. Evanchik, M. Finlayson, J. Herne, and J. Matthews, "Xen and the art of repeated research," *USENIX annual Technical Conference*, pp. 135–144, 2004.

[7] A. Menon, J. Santos, Y. Turner, G. Janakiraman, and W. Zwaenepoel, "Diagnosing performance overheads in the xen virtual machine environment," in *VEE: Proceedings of the 1st ACM Conference on Virtual Execution Environments*, 2005, pp. 13–23.

[8] L. Cherkasova and R. Gardner, "Measuring cpu overhead for i/o processing in the xen virtual machine monitor," in *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, 2005, pp. 24–24.

[9] P. Apparao, S. Makineni, and D. Newell, "Characterization of network processing overheads in xen," in *VTDC '06: Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing*, 2006.

[10] M. R. Marty and M. D. Hill, "Virtual hierarchies to support server consolidation," in *ISCA '07: Proceedings of the 34th annual international symposium on Computer architecture*, 2007, pp. 46–56.

[11] P. Apparao, R. Iyer, and D. Newell, "Implications of cache asymmetry on server consolidation performance," in *Workload Characterization, 2008. IISWC 2008. IEEE International Symposium on*, Sept. 2008, pp. 24–32.

[12] P. Apparao, R. Iyer, X. Zhang, D. Newell, and T. Adelmeyer, "Characterization & analysis of a server consolidation benchmark," in *VEE '08: Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, 2008, pp. 21–30.

[13] A. Tikotekar, G. Vallée, T. Naughton, H. Ong, C. Engelmann, and S. L. Scott, "An analysis of hpc benchmarks in virtual machine environments," in *VHPC '08: Proceedings of 3rd Workshop on Virtualization in High-Performance Cluster and Grid Computing*, 2008, pp. 63–71.

[14] L. Youseff, R. Wolski, B. Gorda, and C. Krintz, "Evaluating the performance impact of xen on mpi and process execution for hpc systems," in *VTDC '06: Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing*, 2006.

[15] A. Ranadive, M. Kesavan, A. Gavrilovska, and K. Schwan, "Performance implications of virtualizing multicore cluster machines," in *HPCVirt '08: Proceedings of the 2nd workshop on System-level virtualization for high performance computing*, 2008, pp. 1–8.

[16] R. Uhlig, G. Neiger, D. Rodgers, A. Santoni, F. Martins, A. Anderson, S. Bennett, A. Kagi, F. Leung, and L. Smith, "Intel virtualization technology," *IEEE Computer*, vol. 38, no. 5, pp. 48–56, 2005.

[17] *Secure Virtual Machine Architecture Reference Manual*, AMD64 Architecture Codenamed Pacifica Technology, AMD, 2006.

[18] (2010) HPC Challenge Benchmark. [Online]. Available: http://icl.cs.utk.edu/hpcc

[19] W. Emeneker and D. Stanzione, "HPC Cluster Readiness of Xen and User Mode Linux," in *2006 IEEE International Conference on Cluster Computing*, 2006, pp. 1–8.

[20] J. P. Walters, V. Chaudhary, M. Cha, S. G. Jr., and S. Gallo, "A comparison of virtualization technologies for hpc," in *AINA '08: Proceedings of the 22nd International Conference on Advanced Information Networking and Applications*, 2008, pp. 861–868.

[21] A. Tikotekar, G. Vallée, T. Naughton, H. Ong, C. Engelmann, S. L. Scott, and A. M. Filippi, "Effects of virtualization on a scientific application running a hyperspectral radiative transfer code on virtual machines," in *HPCVirt '08: Proceedings of the 2nd workshop on System-level virtualization for high performance computing*, 2008, pp. 16–23.

[22] S. Kundu, R. Rangaswami, K. Dutta, and M. Zhao, "Application performance modeling in a virtualized environment," in *HPCA '10: Proceedings of 16th International Symposium on High-Performance Computer Architecture*, 2010.