DOI 10.12694/scpe.v18i2.1288
Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 18, Number 2, pp. 161–175. http://www.scpe.org © 2017 SCPE

IMPACT OF PROCESS ALLOCATION STRATEGIES IN HIGH PERFORMANCE CLOUD COMPUTING ON AZURE PLATFORM

HANAN A. HASSAN† AYA I. MAIYZA‡ AND WALAA M. SHETA§¶

Abstract. Nowadays, there is an increasing demand in the High-Performance Computing (HPC) community to make use of different public cloud service provider. The question of which cloud provider is superior for a certain application and usage configuration is very important for the successful deployment of HPC application on the cloud. In this paper, we evaluate the performance of HPC applications on Microsoft Azure cloud platform using the well-known NAS parallel benchmarks. These benchmarks are considered as examples of general scientific HPC applications to test the communication performance. Different process allocation strategies are performed in terms of MOPS and Speedup. Our results show that allocating one process per instance achieves higher scalability at the expense of the cost. The results compared with the same results with the same experiments in Amazon platform. We found that Azure platform has better shared-memory communication performance than Amazon platform. In contrast, Amazon is superior to Azure platform in terms of Ethernet bandwidth.

Key words: Cloud Computing, High Performance Computing, Message Passing Interface, Nas Parallel Benchmark

AMS subject classifications. 68M14, 68M20

1. Introduction.

Cloud computing is a revolutionary technology based on sharing resources to provide diverse types of e-services to end users. Cloud computing provisioning is basically based on virtualization techniques to obtain an abstract view of physical resources with the same interfaces. It offers several benefits such as the possibility of using same physical resources for different users and runtime environments simultaneously [1]. Additionally, dealing with Virtual Machine (VM) is much easier in management, maintenance setup, and administration. Moreover, its cost is optimized as payment is based on pay-as-you-go (PAYG) model. In PAYG model, the charge is only calculated for the actual usage of the physical resources.

The cloud service models are categorized as (i) Software as a Service (SaaS), which completely deals with applications such as iCloud. (ii) Platform as a Service (PaaS), which provides an environment for application development framework such as operating Systems. (iii) Infrastructure as a Service (IaaS), which provides the computing hardware virtual Infrastructure and virtual storage. Recently HPC as a Service (HPCaaS) is considered as one of the promising services on the cloud. The cloud computing offers advantages to HPC applications users including virtualization benefits, resources scalability, abstraction of cluster setup cost and time, and energy consumption [1, 2].

Nevertheless, there are challenges for running HPC application on cloud because of the performance diversity and poor network performance [2]. There are several IaaS public cloud Providers such as Amazon Elastic Cloud Compute (EC2), Microsoft Windows Azure, and Rackspace. These clouds are suitable for running HPC application on them [3]. Microsoft's Azure cloud provides an amiable development environment (.NET, SQL Server, and Visual Studio) with a wide group of capabilities for developers to construct robust applications over those [4, 5]. It provides powerful storage and resources through hardware level virtualization. In addition, it is possible to build virtual parallel clusters easily. This motivated us to study the performance of HPC on the Microsoft Azure Cloud.

In our previous work [6], the Ethernet and shared memory communication were measured in terms of bandwidth and latency on Azure platform. Furthermore, the scalability impact of running HPC applications on Azure platform was assessed using up to 128 cores [6]. As an extension, the impact of different process allocation strategies is studied in terms of performance and cost. The scalability is evaluated and assessed using up to 512 cores as well.

Furthermore, this paper is aimed to assess the performance of running HPC applications on different types of cloud platform, identify challenges that affect the performance, and presenting a set of possible solutions for

†Informatics Research Institute, The City of Scientific Research and Technological Applications (hali@srtacity.sci.eg).
‡Informatics Research Institute, The City of Scientific Research and Technological Applications (amaiyza@srtacity.sci.eg).
§Informatics Research Institute, The City of Scientific Research and Technological Applications (wsheta@srtacity.sci.eg).
¶CECS Department, University of Louisville, USA.

performance improvement. Scalability is evaluated by NAS Parallel Benchmarks (NPB) kernels [7] in terms of MOPS (millions operation per second) and Speedup, which is calculated by dividing serial over parallel execution time. The experiments were performed on one cluster of virtual machines on Microsoft's Azure cloud (A10 size). A10 size has specifications close to one of Amazon Elastic Compute Cloud (EC2) specifications. Then, the results are compared with the study performed on Amazon EC2 [8].

Intuitively, involving all the VM resources (virtual cores) to run a certain HPC process should result into the best cost/performance scenario (value verses money). This is called normal process allocation strategy (NPAS). However, this selection is not going to be constantly the best in terms of performance. So, this paper reveals that the strategy of allocating one process per instance (1ppi) is significantly much more expensive. However, it is able to achieve higher performance in several applications [8].

The NAS offers a set of HPC kernels, which varies in their types of being either communication or computation intensive. They also differ in several data types such as integer and floating point performance [3].

The rest of the paper is organized as follows: Section 2 shows the related work. Section 3 represents the evaluation methodology, which contains experiments configurations. Section 4 analyzes the experimental result, and compares between two process allocation strategies optimized for performance and cost. In addition, the section discusses the obtained result. Finally, section 5 recapitulates the conclusion and represents future work.

2. Related Work. This section is oriented toward two dominant issues: (1) Performance evaluation of running HPC on the public cloud providers using well-known benchmarks (2) The possibility of running HPC applications using a virtualized cluster on cloud platforms and evaluating its performance compared with traditional HPC platform.

Previous research of running HPC application on cloud platform focused on the performance of VM compared to physical HPC cluster. The result was not useful because of poor network performance, multi-tenancy, and resources disproportionate [2, 9, 10, 11]. Nevertheless, the performance improvement of running HPC applications on cloud is still a research target to get accepted performance compared to traditional HPC. In spite of its performance, it is suitable for some applications, which required scale up/down resources and can easily drop them out when the task is done at the end [12, 13, 14].

In general, several well-known benchmarks have been used to analyze the performance of parallel computing. But in particular, they use their own applications to compare the performance between different techniques or running in different infrastructures. NAS parallel benchmarks (NPB) was used in [2, 3, 6, 8, 15, 16] and High-Performance Linpack benchmark (HPL) was used in [5, 16, 17] to compare between the performance of using traditional HPC and of using HPC hosted on public or private cloud.

Akioka and Muraoka [15] used Amazon EC2 as an alternative to HPC environment using NPB benchmarks and HPL benchmark. The performance and cost efficiency were evaluated. Then, a performance comparison was performed between Amazon cloud instance and a physical machine with NPB benchmarks using the serial version (NPB-SER). They found that running HPC on cloud is not suitable for some kinds of applications, such as performance critical applications.

Gupta et al. [2] presented the performance analysis and the tradeoffs of cost for HPC applications with/without virtualization using NPB benchmarks. They found that running HPC on the cloud is considered more cost-effective for non-communication-intensive applications such as embarrassingly parallel and tree-structured computations over high processor count and for communication-intensive applications over low processor count and HPC-optimized clusters are outstanding for the rest.

Roloff et al. [3] provided an inclusive analysis of three important phases of HPC on the cloud: deployment, performance, and cost-efficiency. This study was performed on three public clouds, namely Amazon Elastic Compute Cloud, Rackspace and Microsoft Azure, as well as a traditional cluster using NPB benchmark. They showed that virtualized HPC had a better performance and cost efficiency than the cluster, up to 27% and 41%, for several benchmarks.

Strazdins et al. [16] presented performance results for two benchmarks and two large scientific applications running in private VM cluster, an Amazon HPC EC2 cluster and traditional HPC environment. They used the MPI micro-benchmark, the NAS Parallel macro-benchmarks, the UK Met Office's MetUM global climate model and the chaste multi-scale computational biology code. They succeeded to build application codes in a pure HPC environment and replicate these into VMs which ran on private VM cluster and on public HPC

cluster running on Amazon's EC2.

Expósito et al. [8] analyzed the major performance bottlenecks in HPC application running on Amazon EC2 Cluster Computing platform. They compared between two flavors of instances CC1 and CC2. First, they evaluated the communication performance on ten gigabits Ethernet network and shared memory using Intel MPI Benchmarks suite (IMB) on three HPC message-passing middleware, namely MPICH2, OpenMPI, and FastMPJ. They found that CC2 instances are somewhat better point-to-point communication performance and provide more computational power. Second, they appraised the scalability of representative message-passing codes using up to 512 cores using NAS Parallel Benchmarks (NPB) kernels. They found that CC2 instances have poorer scalability than CC1 instances for communication-intensive applications and CC1 instances are more effective than CC2 instances in terms of cost. Third, they achieved higher scalability using one process per instance only. Finally, several levels of parallelism have been used to achieve most scalable and cost-effective using hybrid technique between message-passing with multithreading.

Hassani et al. [12] implemented the MPI version for parallel Radix sort and evaluated its performance on Amazon cloud infrastructure. Then, they compared the result with traditional HPC platform. Parallel HPC applications use considerably Message Passing Interface (MPI) [18, 19, 20]. Previously, they implemented parallel Radix sort programming using MPI, Pthreads, and OpenMP and evaluated the performance using benchmarking test. Their experimental results proved that MPI is better than others at the time of parallel sorting, so they implemented the parallel Radix sort on Cloud. They spotted considerable improvement in speed up and scaled up for up to 8 virtual nodes. Cloud response rate was more 20 percent parallel efficiency than the traditional HPC cluster.

Zhang et al. [5] implemented a virtual HPC environment on Azure cloud for hydrological applications. They presented a case study on groundwater uncertainty analysis in Heihe River Basin. They proved that the Azure cloud can outperform traditional HPC infrastructure and can be useful for hydrological researchers to improve computing efficiency of the model.

Belgacem and Chopard [17] had successfully connected Amazon EC2 based cloud cluster situated in USA, to a private HPC cluster (Scylla) located at their university in Switzerland. They ran a large distributed multiscale application on this hybrid HPC infrastructure. They ran a distributed multiscale application using the software developed in their MAPPER project [21]. They found that the distributed computing performance is less than the monolithic one. Their analysis showed that this low performance because of the long-distance between the two continents thus resulting in very poor network performance.

Hassan et al. [6] evaluated the performance of the HPC applications on Azure cloud in terms of MOPS and Speedup. The performance was tested under several configurations of cluster sizes. In addition, the performance of point-to-point communication between processes was assessed in terms of bandwidth and latency. They found that the best performance was achieved using only a single VM (shared memory communication model), especially with IS and FT NPB-kernels.

Cala et al. [13] presented their experience in porting a genomics data processing pipeline (Next-Generation Sequencing Genetic test) from an existing scripted implementation deployed on a traditional HPC, to a workflow-based design deployed on the Microsoft Azure public cloud. Most of HPC systems used sophisticated MPI-based algorithms, but the current NGS tools do not require it. Rather an effective data splitting techniques are used to convert the Big Data to an embarrassingly parallel problem. They found that using public cloud provided several benefits such as speed, scalability, flexibility and cost-effectiveness for NGS Genetic test.

Mohrehkesh et al. [14] presented a feasibility study using cloud resources for Image Guided Neurosurgery (IGNS). They computed the deformable registration or non-rigid registration (NRR) of brain MR images using their local private cloud (Turing cluster) at Old Dominion University, as well as Microsoft Azure (Microsoft HPC pack) [22]. Using these clusters, they analyzed more than 6TB of images. They evaluated the accuracy of registration by speculative execution, the overhead time of running jobs on the Azure cloud and cost comparison of running jobs on a private versus public cloud. Their results indicated that the public cloud provides practical and cost-effective means for a hospital that supports IGNS solutions. Moreover, the accuracy of NRR could be improved up to 57% using cloud resources.

This diversity of research efforts to run HPC systems on cloud computing using various types of applications, benchmarks, and technologies. Table 2.1 summarizes the previous work of running HPC cluster on public cloud

TABLE 2.1
The relevant related work.

Authors Name	Benchmarks and HPC Applications	Libraries	Platform	Evaluated Metric	Recommendation for Running HPC on Cloud
S. Akioka, Y. Muraoka [15]	- NPB-MPI 3.3 (Class C) - HPL 2.0 Benchmark	- MPICH2 1.2	- Amazon EC2 - Physical machine (NPB-SER)	- Mops/second - Gflops - Cost	Not suitable for performance critical application
A. Gupta, M. Dejan [2]	- NPB-MPI (Class B) - NAMD - NQueens	- MPI	- Open Cirrus (Private cloud) - Eucalyptus (Private cloud) - Taub (Private HPC)	- Speedup - Cost	Cost-effective for non-communication-intensive application up to high processor count and for communication-intensive application up to low processor count
E. Roloff, M. Diener, A. Carissimi, P.O.A.Navaux[3]	- NPB-MPI 3.3.1 (Class B)	- MPI - OpenMPI	- Amazon EC2 - Rackspace - Microsoft Azure - Traditional HPC cluster	- Normalized average execution time - Cost efficiency	Higher performance and cost efficiency than the cluster up to 27% and 41% respectively for several benchmarks
P.E. Strazdins, J. Cai, M. Atif, J. Antony [16]	- MPI microbenchmark - NPB-MPI 3.3 (Class B) - memory intensive simulationapplications (Chaste 2.1 & UM 7.8)	- MPI	- Private VM cluster - Amazon EC2 - Traditional HPC cluster	- Bandwidth (MB/s) - Latency (s) - Speedup	Ability to successfully build large scale HPC applications on Cloud
R.R. Expósito, G.L. Taboada, S. Ramos, J. Tourio, R. Doallo [8]	- IMB - NPB-MPI 3.3 (Class C) - NPB-MZ 3.3.1 (Class C)	- MPICH2 1.4.1 - OpenMPI 1.4.4	- Amazon EC2	- Bandwidth (Gbps) - Latency (μs) - MOPS - Speedup - Cost efficiency (USD/GOPS)	Achieving higher scalability using 1ppi and using hybrid technique between message-passing with multithreading
R. Hassani, M. Aiatullah, P. Luksch [12]	- parallel Radix sort programming	- MPI	- Amazon EC2 - Dedicated HPC platform	- Execution times (sec)	MPI is better than others at the time of parallel sorting. Cloud response rate was more 20 percent parallel efficiency than the pure HPC cluster

Table 2.1 (continued)

Authors Name	Benchmarks and HPC Applications	Libraries	Platform	Evaluated Metric	Recommendation for Running HPC on Cloud
M.B. Belgacem, B. Chopard [17]	- large distributed multiscale application	- OpenMPI 1.4.5	- Hyprid Amazon EC2 cluster with private HPC cluster (Scylla)	- Execution times (sec)	The distributed computing performance is less than the monolithic one
H.A. Hassan, S.A. Mohamed, W.M. Sheta [6]	- IMB NPB-MPI 3.3 (Class C)	- MPICH2 1.4.1 - OpenMPI 1.4.4	- Microsoft Azure	- Bandwidth (Gbps) - Latency (s) - MOPS - Speedup - Cost efficiency (USD/GOPS)	Achieving higher performance when running entirely on a single VM (shared memory communication model), especially IS kernel and FT kernel.
J. Caa, E. Marei, Y. Xu, K. Takeda, P. Missier [13]	- Workflow-based application (Next-Generation Sequencing)		- local HPC cluster - Microsoft Azure	- Response time (Hours) - Throughput (samples/day) - Relative processing effectiveness (%) - Cost per sample	Public cloud could provide benefits such as speed, flexibility, scalability and cost-effectiveness for NGS.
S. Mohrehkesh, A. Fedorov, A. B. Vishwanatha, F. Drakopoulos, R. Kikinis, N. Chrisochoides [14]	- Image Guided Neurosurgery (IGNS)	- MPI - Microsoft MPI (built-in)	- private cloud (Turing cluster) - Microsoft Azure (Microsoft HPC pack)	- Registration accuracy (%) - Azure overhead time (s) - Monthly Cost (1000 US$)	Public cloud provides practical and cost-effective solution more than private cluster.

TABLE 3.1
Cloud instances Specifications [8, 23].

Cloud Platform	Instance Size	Cores	CPU Type	RAM	RAM Type
Azure	A10	8	Intel Xeon E5-2670 @ 2.6 GHz	56 GB	DDR3-1600 MHz
Amazon	cc1	8	Intel Xeon X5570 Nehalem @ 2.93 GHz	23 GB	DDR3

providers. These publications are classified according to the benchmarks types, evaluated metrics, used libraries, platform, and their main result. Concludes that the best cloud provider depends on the type and behavior of the application, in addition to the intended usage scenario. In our experiments, HPC system is assessed on Microsoft Azure Cloud using NAS Parallel Benchmarks (NPB) under different process allocation scenarios.

3. Evaluation Methodology. This section presents the used configurations and benchmarks for evaluating the performance of using HPC on Azure platform.

3.1. Experimental Configuration. Two experiments are conducted to evaluate the performance of a cluster of 64 VMs (512 cores) on Microsoft Windows Azure [23]. Table 3.1 shows the used Azure A10 instance specifications. MPICH and OpenMPI are used as a standard implementations of the Message Passing Interface (MPI) with GNU compiler. These two well-known implementations are used in our experiment with releases MPICH2 1.4.1 [24] and OpenMPI 1.4.4 [25].

The performance is evaluated using the most communication-intensive kernels of the NPB benchmarks on Azure platform (using 16 'A10' instances). The performance metrics are Million Operation Per Second (MOPS) and Speedups.

There are two process allocation strategies in our experiments. The first one is called normal process allocation strategy (NPAS). All cores in one instance must be used before extending the cluster with another instance. In this case, we used 8 process per instance (8ppi). The number of used A10 instances is the total number of cores divided by 8 cores per instance. For example, for 64 cores, eight instances were used.

On the other hand, the second process allocation is called expensive process allocation strategy (EPAS), which is used to improve the performance of the previous configuration. The performance is evaluated using the same kernels of NPB using only one process per instances (1ppi) until 64 cores (64 instances), then posteriorly 2, 4, and 8 processes per instance to reach 128, 256, and 512 cores (64 instances). For examples, 32 instances are used for 32 cores using 1ppi and 64 instances are used for 128 cores using 2ppi. This because, there is a limitation on the maximum used number of cores (512 cores).

For the two configurations, the cost is calculated to compare between them. The performance evaluated using the two implementations of message passing interface MPI and OpenMPI. It is evaluated using the result of NPB kernels as productivity in terms of USD per Giga Operations Per Second (USD/GOPS). In this paper, the results are the average of 6 measurements for NPB kernels on the explained experiment configuration. Finally, a comparison between our study and the previous study on Amazon platform [8] is performed taking in account a comparable instances specifications as shown in Table 3.1.

3.2. NAS Parallel Benchmarks (NPB). NPB benchmarks are a well-accepted and well-known HPC benchmark suites. In these experiments, the NPB-MPI version 3.3 [26] benchmarks are used to evaluate our two process allocation strategies on Azure platform. Most communication-intensive kernels have been chosen, namely CG, FT, IS and MG, with class C. Table 3.2 shows an overview of each used benchmark kernel [7, 26].

4. Experimental Results and Discussion. This section presents an analysis of the performance and scalability of HPC NPB kernels on Azure platform under two different process allocation strategies, using the selected benchmarks and their representative kernels, which were described before in sect. 3.2. In addition, there is a subsection for comparing the results with Amazon platform.

4.1. Normal process allocation strategy (NPAS). Fig. 4.1 (dotted line) shows the performance of CG, FT, IS and MG using up to 512 cores on Azure platform (hence, using 64 A10 Azure VMs). The used performance metrics are MOPS (left graphs), and Speedup (right graphs). The used number of VMs is the total number of cores divided by 8 cores (number of cores for A10 VM).

TABLE 3.2
NAS Parallel Benchmarks used Kernels [26].

NPB Kernel	Properties	Problem Size (Class C)	Message Size (Class C)
CG (conjugate gradient)	irregular memory access and communication	150000	146.5 Kbyte
FT (Fourier Transform)*	discrete 3D fast Fourier Transform, all-to-all communication	512^3	128 Mbyte
IS (Integer Sort)	random memory access	2^{27}	128 Mbyte
MG (Multi-Grid)**	Multi-Grid on a sequence of meshes, long- and short-distance communication, memory intensive	512^3	128 Mbyte

*denote the most communication-intensive

**denote the least communication-intensive

The execution of CG kernel on different cluster configurations show that the maximum performance is obtained by using four VMs (32 cores). The performance degrades significantly with the higher number of cores due to the communication overhead in the virtual network. The performance can be improved significantly by using a faster network such as InfiniBand and hardware based virtualization technology such as single-root I/O virtualization SR-IOV [27, 28, 29].

An optimum performance of IS kernels is achieved with 8 cores (1 VM). FT kernels show similar maximum performance with 8 cores (1 VM) and 128 cores (16 VMs). MG kernels have a maximum performance at a cluster of 256 cores (32 VM) using OpenMPI. NPB kernels performance shows that the evaluated applications obtain better results when running completely on only one VM (intra-communication) using up to eight cores, because of the higher scalability and performance of shared memory communications. However, when using greater than single VM, the kernels poorly perform, because of the network virtualization overhead. IS kernels obtain the poorest scalability.

CG kernel is characterized by multiple point-to-point data movements. The best speedup value is 7.3, which is achieved by using 4 VMs (32 cores). The performance is significantly dropped from that value on as it has to depend on ten gigabits Ethernet communications, due to the network virtualization bottleneck. FT kernel achieves a limited scalability on A10 VM. The best speedup value is approximately 5.6, which is achieved by using 8 and 128 cores. IS kernel is a communication-intensive code whose scalability is greatly affected by point-to-point communication start-up latency. Thus, this kernel obtains its highest results when using only one VM due to the high performance of shared memory transfers. It suffers a critical slowdown when using a cluster of VMs. Finally, MG kernel is a limited scalability. The speedup values are close to each other. The best speedup value is approximately 10, which is achieved by using 256 cores for MPICH2.

4.2. Impact of expensive process allocation strategy (EPAS). This set of experiments aims to analyze the impact of expensive process allocation strategy (1ppi) on HPC kernel performance. In the previous experiments, 8 processes per instance strategy are used to maximize the CPU utilization as Azure type A10 instance has 8 cores. This intuitive assumption is not necessarily valid in the case of virtualization, as there is a significant amount of CPU resources consumed to deal with the hypervisor and manage virtualized CPUs according to the type of the hypervisor used. Therefore, we investigate how the number of allocated processes will affect the overall performance in a virtualized HPC environment.

In this experiment, the performance of the NPB kernels using only one process per instance strategy (1ppi) is compared with the normal process allocation strategy (8ppi). This layout can be expensive as we use only one core per instance for each allocated process. For example, we use just 8 instances to allocate 8 processes on the cluster, with each process running on one instance (1ppi). Hence, 32 VMs are used to provide 32 cores with 32 processes on the cluster instead of 256 processes in the normal setting of experiments as the case of sect. 3. Two, four, and eight processes per instance are used to reach 128, 256, and 512 cores; respectively, because of limited resources (64 instances).

Fig. 4.1 (solid line) shows a significant improvement of both MOPS and speedup for all the four kernels. CG kernel's speedup has increased to 8 times its corresponding value of NPAS in MPICH2 environment. Moreover,

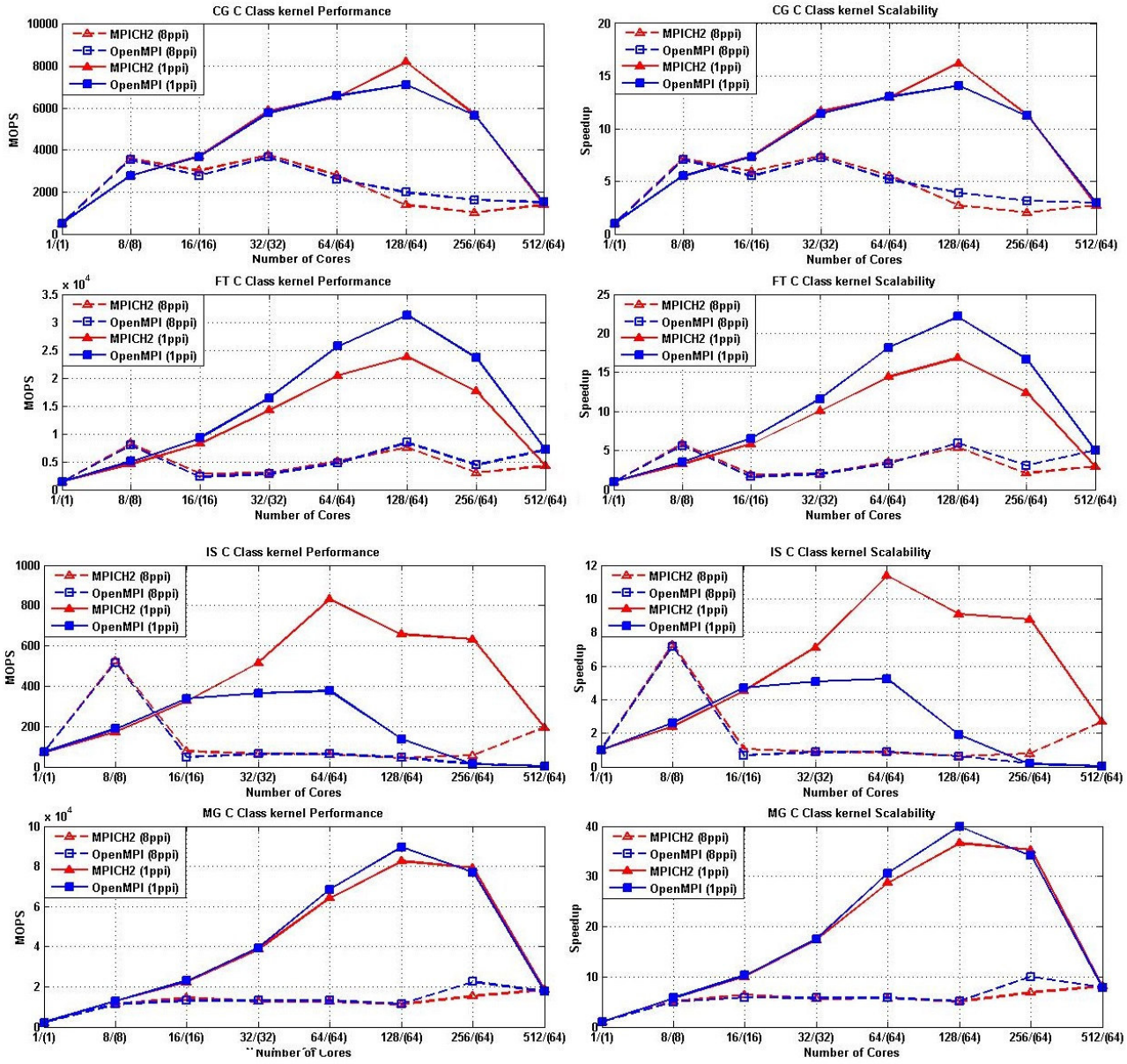

FIG. 4.1. *NPB kernels performance and scalability on A10 Azure instances. For 1ppi: horizontal axis presents x/(z), where x is number of cores and z is number of instances. For 8ppi: horizontal axis presents x, where x is number of cores (number of instances=(number of cores/8))*

the optimum configuration is achieved at 128 cores, i.e. a cluster of 64 instances running a total of 128 processes, instead of (32 cores) a cluster of 4 instances running 32 processes. FT kernel's speedup has improved to more than 4 times compared with NPAS in OpenMPI environment with 128 processes. IS kernel has progressed in MPICH2 more than OpenMPI environment. Its best speedup is achieved at 64 cores. MG kernel achieves a considerable betterment using EPAS. Its best speedup value obtained 40 by using 64 VMs (128 cores).

Fig. 4.2 compares the performance of NPB kernels using 8 cores on Azure platform using NPAS (hence, using one A10 VM) and EPAS (hence, using 8 VMs). It's clear to show that NPAS always gives better performance than EPAS at 8 cores for CG, FT and IS kernels. That is because intra-VM communication (shared memory)

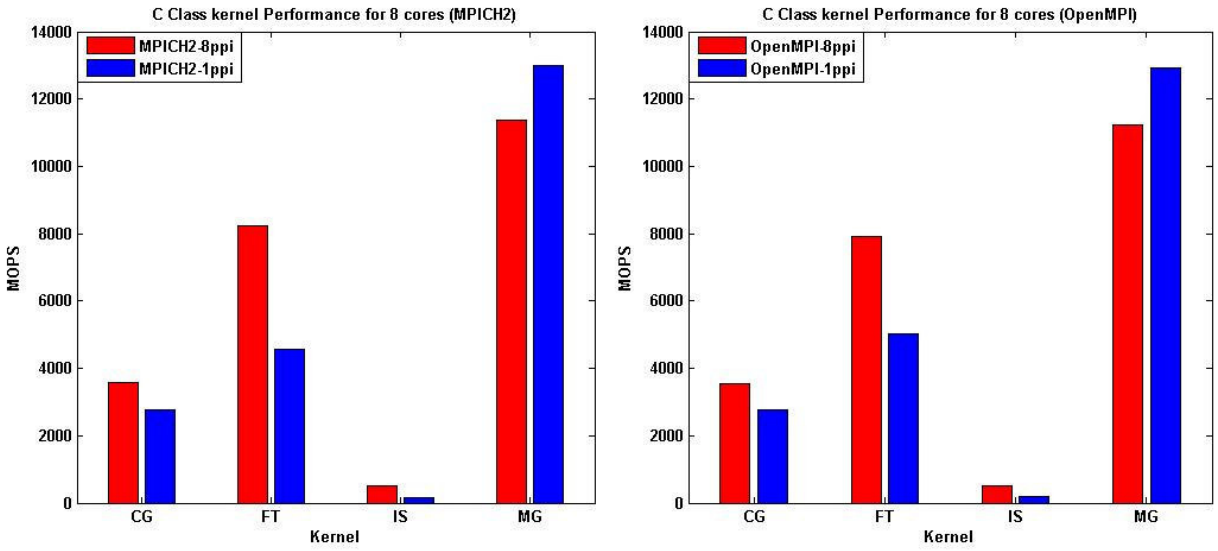

FIG. 4.2. *NPB kernels performance on A10 Azure instances for 8 Cores*

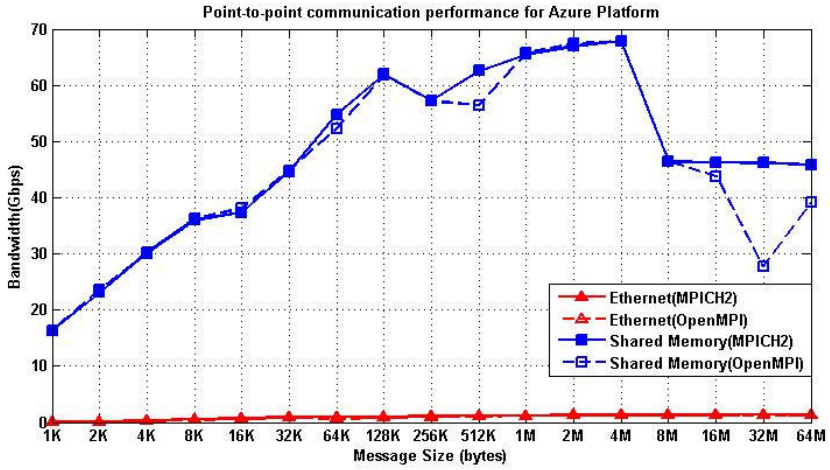

FIG. 4.3. *Point-to-point communication performance on Azure platform [6]*

is better than inter-VM communication (Ethernet) on that level of the cluster as shown in Fig. 4.3. But as the number of cores increases in the cluster using EPAS, its performance supersedes its corresponding NPAS. In contrast, MG kernel achieves higher performance using EPAS, because MG is least communication-intensive as mention in Table 3.2.

4.3. Cost Analysis. Using public cloud infrastructure like Azure, the cost has to be taken into account. Fig. 4.4 and Fig. 4.5 present the productivity in terms of USD per GOPS (Giga Operations per Second) of the already evaluated NPB kernels. This metric depends on the total number of used cores. The cost of each instance is 1.16 $/hour for each A10 instance [23]. When the cost behavior of kernels is inversely proportional to the cluster size, it means that it is better to use a large cluster rather than a small one from both perspectives of performance and cost. On the other hand, if the cost behavior is directly proportional to the cluster size, this indicates that the solution is getting expensive as the cluster is enlarged and therefore there is no need to use a large cluster. An efficient operation cost is obtained at a minimal value of the productivity curve which

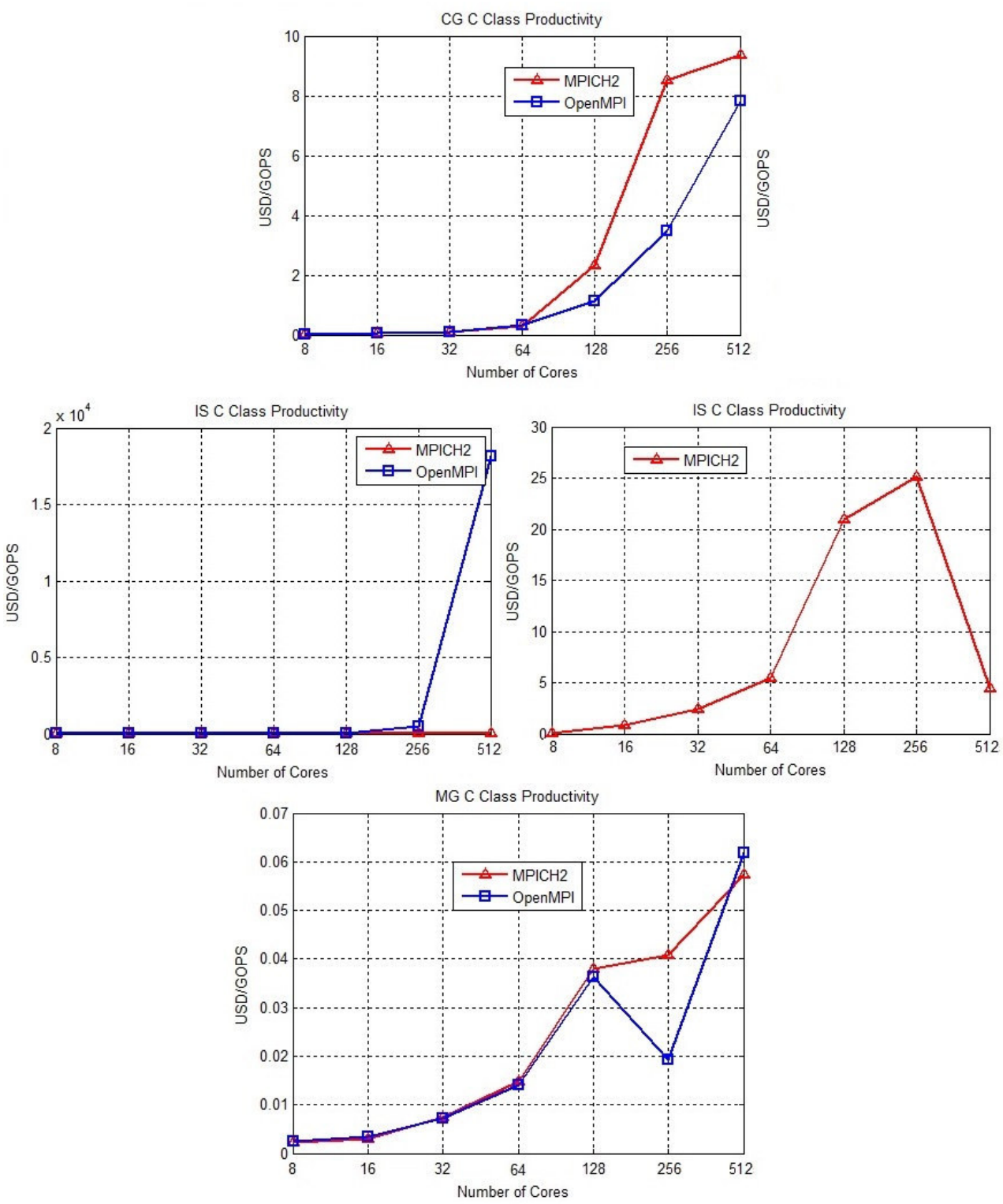

FIG. 4.4. *NPB kernels productivity on Azure instances (Normal process allocation strategy).*

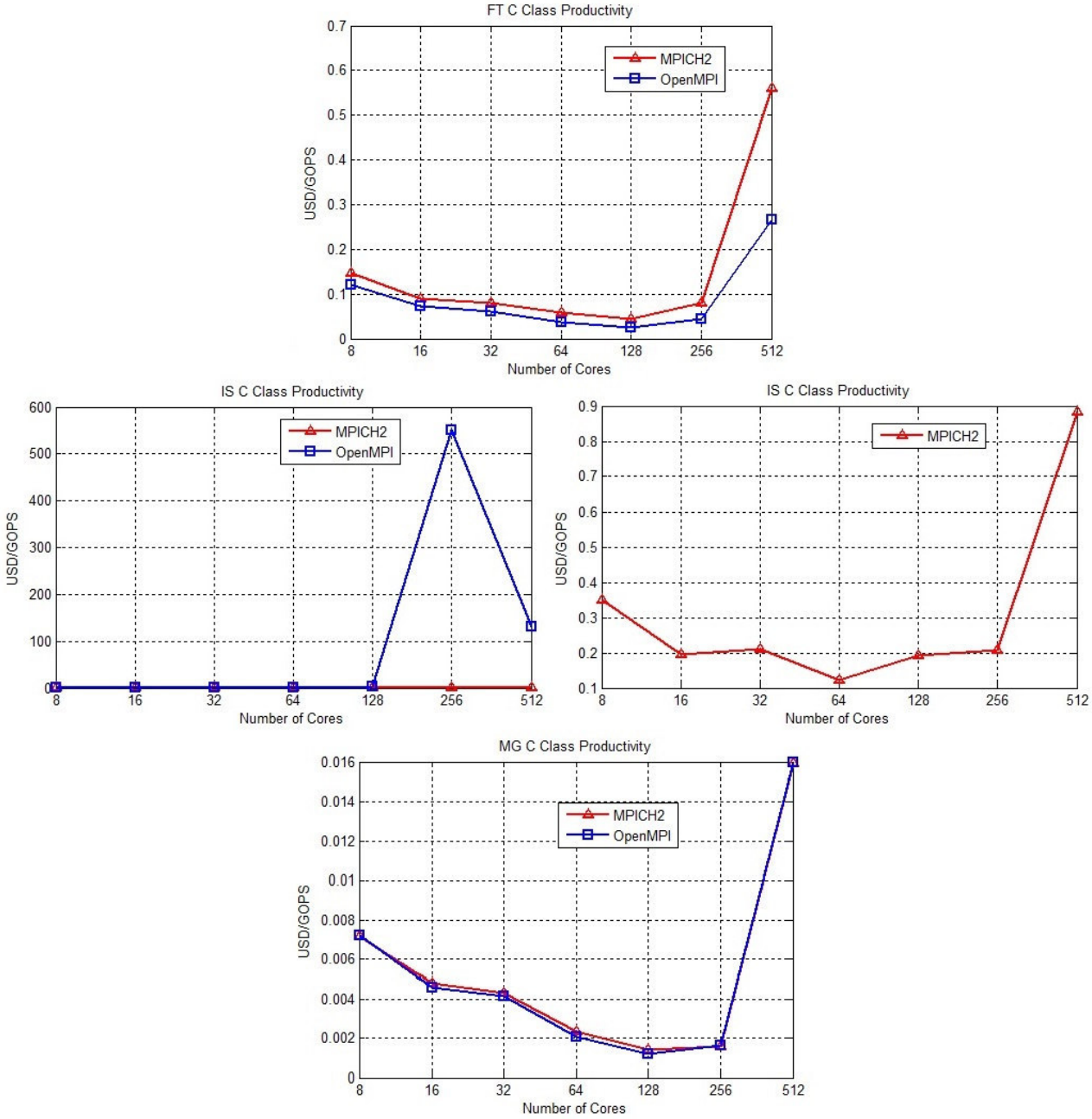

FIG. 4.5. *NPB kernels productivity on Azure instances (Expensive process allocation strategy).*

indicates a low cost per operation versus expensive operations at the maximum value of the curve.

Regarding the first strategy (NPAS), the cost behaviors of CG, IS, and MG kernels are directly proportional to the cluster size as shown in Fig. 4.4. The only exception is valid for IS Kernel using 512 cores (MPICH2 Environment) and MG Kernel using 256 cores (OpenMPI Environment). The reason for these exceptions is the performances of these two configurations, in terms of MOPS and Speedup, increase as shown in Fig. 4.1.

Fig. 4.5 shows the cost behavior in USD/GOPS for EPAS. An optimum value of the productivity curve of FT and MG kernels are achieved at 128 cores which are compatible with the results obtained from Fig. 4.1.

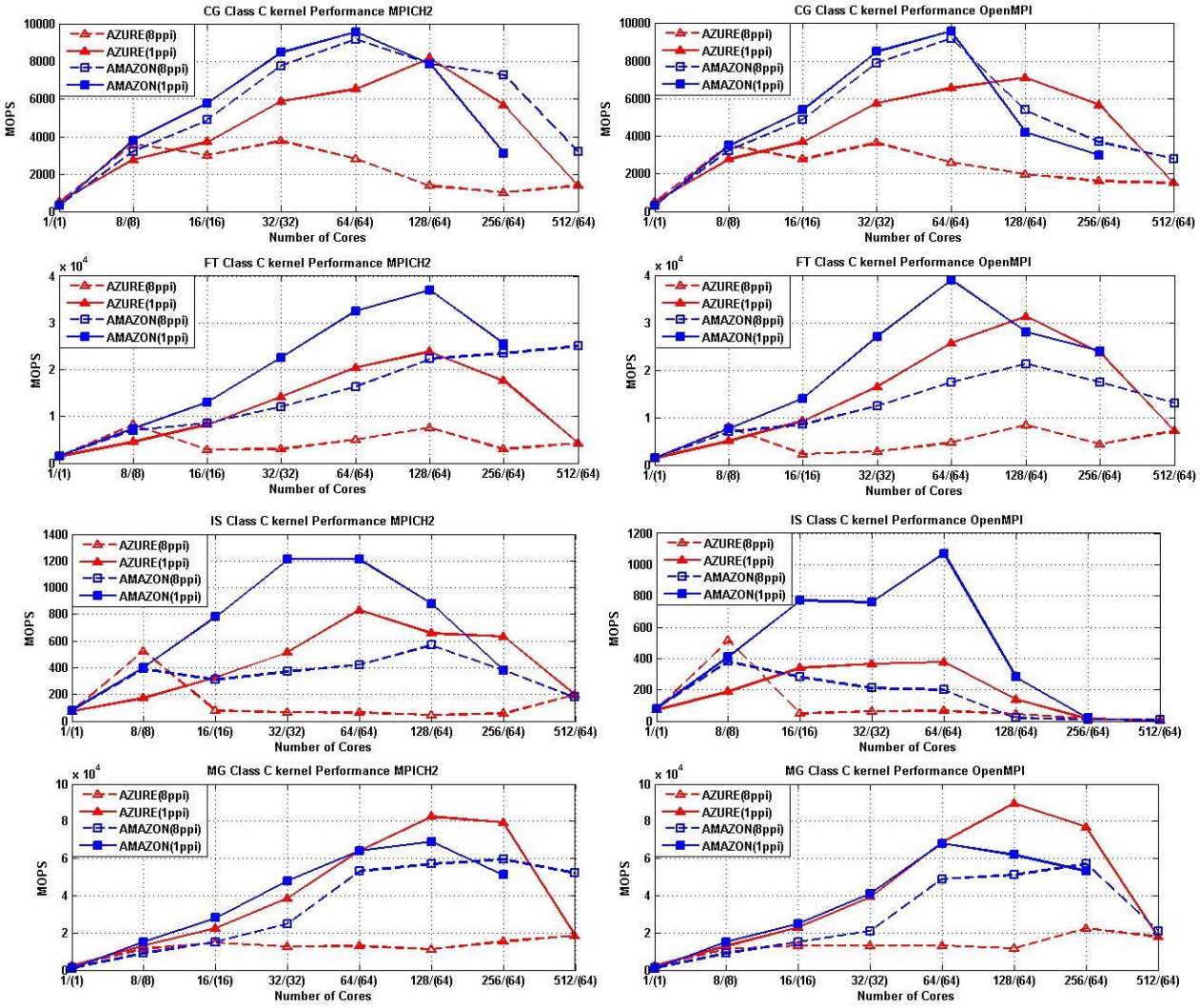

FIG. 4.6. *NPB kernels performance and scalability on A10 Azure instances and cc1 Amazon instances [8]. For 1ppi: horizontal axis presents x/(z), where x is number of cores and z is number of instances. For 8ppi: horizontal axis presents x, where x is number of cores (number of instances=(number of cores/8))*

Then, they dramatically increase using 512 cores, because MOPS and Speedup decrease with the same cluster size (64 instances). For IS kernel, the optimum solution can be achieved at 64 cores with MPICH2, because the cost increases and the performance decreases when the number of cores increases more than 64 cores as shown in Fig. 4.5.

4.4. Comparison between Azure and Amazon Performance. This section compares between the performance of our evaluated experiments on Azure platform and Amazon platform [8] using two different libraries, namely MPICH2 and OpenMPI.

Fig. 4.6 shows the performance of NPB kernels using up to 512 cores on Azure and Amazon platform (hence, using 64 of A10 instances in Azure and using 64 of cc1 instances on Amazon). The performance metric is MOPS only, which is available in the compared paper. The left and right graphs figure out the performance with MPICH2 and OpenMPI libraries, respectively. Dotted and solid lines show the improvements of process allocation strategy from NPAS (8ppi) to EPAS (1ppi), respectively.

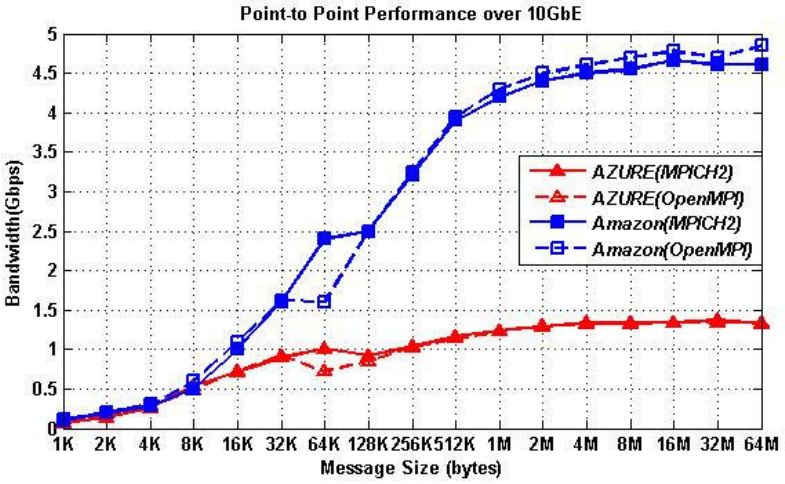

FIG. 4.7. *Point-to-point communication performance on Azure [6] and Amazon instances [8].*

4.4.1. Normal process allocation performance comparison (NPAS). For 8ppi, the overall performance is better in Amazon than Azure platform for all evaluated kernels except using 8 cores. This variance could be due to the different hypervisors which used in the two platforms. Amazon platform uses Xen paravirtualization guests for improving network and disk performance [8], whereas Azure platform uses Hyper-V virtual environments [30].

Fig. 4.7 compares between the bandwidth of Amazon and Azure platform. The bandwidth results of Azure [6] and Amazon [8] platform are performed with the same instance configurations using the same HPC messaging middleware such as MPICH2 1.4.1 [24] and OpenMPI 1.4.4 [25]. A communication is performed through an Ethernet network link using a ping-pong test. It is clear to see that the bandwidth of Amazon platform is better than Azure platform whether MPICH2 or OpenMPI. This observation considers another reason explaining why the overall performance of Amazon platform is better than Azure platform.

In contrast, the only configuration that gives better performance in Azure than Amazon platform is achieved using 8 cores allocated in one instance for all kernels.

Fig.4.8 compares the point-to-point performance of message-passing transfers in the intra-communication between Azure [6] and Amazon [8] platform, where data transfers are implemented on shared memory (hence, without accessing the network hardware). These results obtained with the Intel MPI Benchmarks suite (IMB). Table 3.2 shows the message size of the four evaluated kernels. For CG kernel, the shared memory message-passing communication performance is better using Azure platform over Amazon platform by 1 and 7 Gbps using MPICH2 and OpenMPI, respectively, where the message size for CG kernel is 146.5 Kbytes approximately for C class as mentioned in Table 3. For the three other kernels, the message size is 128 Mbytes. Unfortunately, the measured shared memory performance was performed in [8] and [6] up to 64 Mbytes message size. So, extrapolation is calculated to expect a value for intra-communication performance for this message size. The approximated shared memory communication performance is better using Azure platform over Amazon platform by 14 and 21 Gbps using MPICH2 and OpenMPI, respectively. These results explain why the performance is better using one Azure instance than Amazon instance for the four evaluated kernels with 8 core configuration.

4.4.2. Expensive process allocation performance comparison (EPAS). For EPAS, the most configurations using Amazon give better performance than Azure platform, nevertheless the overall improvement (from using NPAS to EPAS) is greater in Azure for CG and MG kernels than Amazon platform. That is because these two kernels are the least communication-intensive kernels. Amazon has the best performance, except for CG and MG kernels using 128 and 256 cores, FT kernels using 128 cores (OpenMPI), and IS kernels using 256 cores (MPICH2).

There is another important observation appears in Fig. 4.6, all the performance results drop as the number

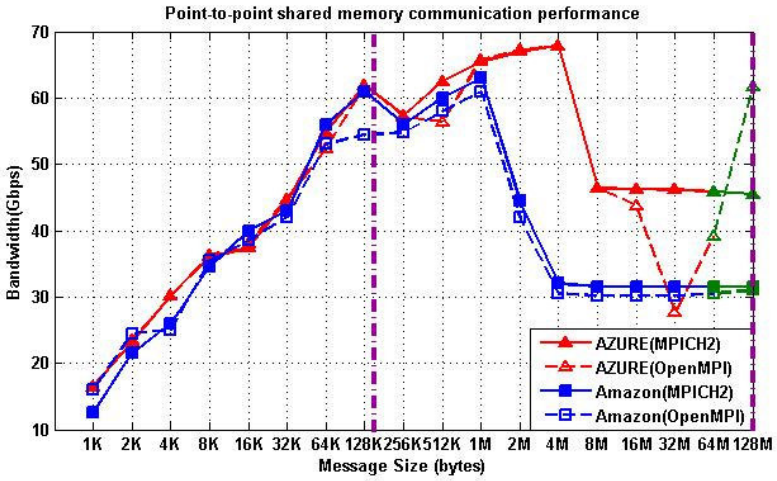

FIG. 4.8. *Point-to-point shared memory communication performance on Azure [6] and Amazon instances [8].*

of cores increase after using 64 or 128 cores. That is because 64 instances are used for 64, 128, 256 and 512 cores using 1, 2, 4, 8 processes per instances, respectively. These configurations are performed whether using Azure or Amazon platform. The performance degradation using Amazon platform is more than using Azure platform, because the shared memory bandwidth is better for Azure as previously explained in Fig. 4.6.

5. Conclusion and Future Work. The research and industry community would highly benefit from using HPC on the cloud. There is a high potential of running HPC on the cloud due to the capability of running large applications on powerful, scalable hardware without the need to physically have or maintain this hardware. Actually, there are several types of public cloud providers, allowing users to select the best provider for their needs. The best configuration for a certain application depends on its type and behavior, in addition to the usage scenario. In this paper, the performance of using HPC on Microsoft Azure cloud service is evaluated through NAS Parallel benchmarks. The scalability is assessed for representative message-passing kernels (NPB) using up to 512 cores for communication-intensive HPC application. Allocating only one process per instance helps HPC applications that are hosted on the cloud to get higher performance at the expense of cost.

This study helps HPC users to determine their own optimum configuration. HPC user could classify his application according to the presented NPB kernels, and then select the cluster size according to his budget and the nature of his application. In addition, it will be easy to select between Azure Cloud and Amazon EC2 platform depending on the deployed application.

For the future, it's intended to execute real HPC applications on Azure platform. The performance of another cloud platform could be compared with Azure and Amazon cloud too. Different process allocation strategies (2ppi and 4ppi) could be tested. Moreover, cost analysis could be performed using expensive process allocation strategy (EPAS) with Amazon platform. As a result, a comprehensive study could be presented in terms of performance and cost analysis between Azure and Amazon platform.

Acknowledgments. The authors are grateful for the partial support received by Microsoft to access Azure cloud computing facilities. We also would like to thank Mona Kashkoush and Shimaa Abdellatif Mohamed, from the Informatics Research Institute for their collaboration in conducting the experiments.

REFERENCES

[1] V. MAUCH, M. KUNZE AND M. HILLENBRAND, *High performance cloud computing*, Future Generation Computer Systems, 29, 2013, pp. 1408–1416.

[2] A. GUPTA AND D. MILOJICIC, *Evaluation of hpc applications on cloud*, in Open Cirrus Summit (OCS), 2011 Sixth, IEEE, 2011, pp. 22–26.

[3] E. Roloff, M. Diener, A. Carissimi, and P. O. A. Navaux, *High performance computing in the cloud: Deployment, performance and cost effciency*, in Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, IEEE, 2012, pp. 371–378.
[4] R. J. Dudley and N. A. Duchene, *Microsoft Azure: Enterprise Application Development*, Packt Publishing Ltd, 2010.
[5] G. Zhang, Y. Yao, and C. Zheng, *HPC Environment on Azure Cloud for Hydrological Parameter Estimation*, in Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on, IEEE, 2014, pp. 299–304.
[6] H. A. Hassan, S. A. Mohamed, and W. M. Sheta, *Scalability and communication performance of HPC on Azure Cloud*, Egyptian Informatics Journal, 2015.
[7] D. H. Bailey, E. Barszcz, J. T. Barton, D. S. Browning, R. L. Carter, L. Dagum, R. A. Fatoohi, P. O. Frederickson, T. A. Lasinski, R. S. Schreiber, et al., *The NAS parallel benchmarks*, International Journal of High Performance Computing Applications, 5, 1991, pp. 63–73.
[8] R. R. Expósito, G. L. Taboada, S. Ramos, J. Touriño and R. Doallo, *Performance analysis of HPC applications in the cloud*, Future Generation Computer Systems, 29, 2013, pp. 218–229.
[9] A. Iosup, S. Ostermann, M. N. Yigitbasi, R. Prodan, T. Fahringer and D. H. Epema, *Performance analysis of cloud computing services for many-tasks scientific computing*, Parallel and Distributed Systems, IEEE Transactions on, 22, 2011, pp. 931–945.
[10] R. Hassani, A. Fazely and P. Luksch, *Optimizing Bandwidth by Employing MPLS AToM with QoS Support*, in Networking, Architecture and Storage (NAS), 2012 IEEE 7th International Conference on, IEEE, 2012, pp. 104–108.
[11] R. Hassani and P. Luksch, *DMT: A new Approach of DiffServ QoS Methodology*, 2012.
[12] R. Hassani, M. Aiatullah, and P. Luksch, *Improving hpc application performance in public cloud*, in Networking, IERI Procedia, 10, 2014, pp. 169–176.
[13] J. Cała, E. Marei, Y. Xu, K. Takeda and P. Missier, *Scalable and efficient whole-exome data processing using workflows on the cloud*, Future Generation Computer Systems, 2016.
[14] S. Mohrehkesh, A. Fedorov, A. B. Vishwanatha, F. Drakopoulos, R. Kikinis and N. Chrisochoides, *Large Scale Cloud-Based Deformable Registration for Image Guided Therapy*, in Connected Health: Applications, Systems and Engineering Technologies (CHASE), 2016 IEEE First International Conference on, IEEE, 2016, pp. 67–72.
[15] S. Akioka and Y. Muraoka, *HPC benchmarks on Amazon EC2*, in Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on, IEEE, 2010, pp. 1029–1034.
[16] P. E. Strazdins, J. Cai, M. Atif, and J. Antony, *Scientific Application Performance on HPC, Private and Public Cloud Resources: A Case Study Using Climate, Cardiac Model Codes and the NPB Benchmark Suite*, in Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International, IEEE, 2012, pp. 1416–1424.
[17] M. B. Belgacem and B. Chopard, *A hybrid HPC/cloud distributed infrastructure: Coupling EC2 cloud resources with HPC clusters to run large tightly coupled multiscale applications*, Future Generation Computer Systems, 42, 2015, pp. 11–21.
[18] R. Hassani, A. Malekpour, A. Fazely, and P. Luksch, *High performance concurrent multi-path communication for MPI*, Springer, 2012.
[19] R. Hassani and P. Luksch, *Scalable high performance computing in wide area network*, in High Performance Computing and Simulation (HPCS), 2012 International Conference on, IEEE, 2012, pp. 684–686.
[20] R. Hassani, G. Chavan and P. Luksch, *Optimization of Communication in MPI-Based Clusters*, in Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2014 International Conference on, IEEE, 2014, pp. 143–149.
[21] "Mapper project,", `http://www.mapper-project.eu/`
[22] "Microsoft HPC pack (windows HPC server),", `https://technet.microsoft.com/en-us/enus/library/cc514029.aspx`
[23] `http://azure.microsoft.com/blog/2015/03/05/new-a10a11-azure-compute-sizes/`
[24] "MPICH2 Website:", `http://www.mcs.anl.gov/research/projects/mpich2.org/`
[25] "OpenMPI Website:", `http://www.open-mpi.org/`
[26] `https://www.nas.nasa.gov/publications/npb.html`
[27] T. Pais Pitta de Lacerda Ruivo, G. B. Altayo, G. Garzoglio, S. Timm, H.-W. Kim, S.-Y. Noh, and I. Raicu, *Exploring Infiniband Hardware Virtualization in OpenNebula towards Efficient High-Performance Computing*, tech. report, Fermi National Accelerator Laboratory (FNAL), Batavia, IL (United States), 2014.
[28] N. Regola and J.-C. Ducom, *Recommendations for virtualization technologies in high performance computing*, in Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, IEEE, 2010, pp. 409–416.
[29] Y. Dong, X. Yang, J. Li, G. Liao, K. Tian, and H. Guan, *High performance network virtualization with SR-IOV*, Journal of Parallel and Distributed Computing, 72, 2012, pp. 1471–1480.
[30] V. A. Gandhi and C. Kumbharana, *Comparative study of Amazon EC2 and Microsoft Azure cloud architecture*, International Journal of Advanced Networking & Applications, 2014.

Edited by: Dana Petcu
Received: Dec 26, 2016
Accepted: May 1, 2017