First Experience and Practice of Cloud Bursting Extension to OCTOPUS

Susumu Date[1], Hiroaki Kataoka[2], Shuichi Gojuki[3], Yuki Katsuura[4],
Yuki Teramae[4] and Shinichiro Kigoshi[4]
[1]*Cybermedia Center, Osaka University, Japan*
[2]*1st Government and Public Solutions Division, NEC, Japan*
[3]*Customer Success Unit, Microsoft Japan Co., Ltd., Japan*
[4]*Department of Information and Communication Technology Service, Osaka University, Japan*

Keywords: Cloud Bursting, Supercomputer, High-performance Computing, Job Scheduler.

Abstract: In the Cybermedia Center at Osaka University, OCTOPUS (Osaka university Cybermedia cenTer Over-Petascale Universal Supercomputer) has provided for providing a high performance computing environment for scientists and researchers in Japan since April 2017. OCTOPUS enables a stable operation with only a few technical problems. However, due to higher and higher computing demands, the users' waiting time from job submission to job completion has been increasing. In this research, we explore the feasibility of solving this problem by offloading the computational workload to IaaS Cloud services, in the hope that we can alleviate the high utilization on OCTOPUS. Technically, we build a cloud bursting environment where OCTOPUS as an on-premise computing environment and Microsoft Azure as an IaaS Cloud service are integrated. Therefore, we investigate whether the cloud bursting environment is useful and practical for avoiding high utilization on OCTOPUS. In this paper, we summarize the first experience and practice of our cloud-bursting solution.

1 INTRODUCTION

In the Cybermedia Center at Osaka University (CMC), OCTOPUS (Osaka university Cybermedia cenTer Over-Petascale Universal Supercomputer) (CMC, Osaka University, 2019b), which is a hybrid cluster system of heterogeneous architectures, was installed in December 2017, in the hope that it could accommodate a variety of computing needs and demands on a scalar-type supercomputer and then provide scientists and researchers as users with a higher computing performance stably and constantly. As a result, OCTOPUS has operated with a fairly high utilization. However, this situation has given a rise to a new problem; namely, that users' waiting time from job submission to job completion has been long. Although the waiting time depends on the degree of parallelism users request for computation, the waiting time often reaches three days, and sometimes even a week.

Recently, however, the IaaS (Infrastructure as a Service) cloud service, which has allowed us to use a necessary amount of computing resources with our own preferable software stack based on our computing needs and requirements, has approached maturity. Amazon AWS (Mathew, 2019), Microsoft Azure (Microsoft, 2019b), and Oracle Cloud (Gong and Amin, 2018) are just some examples of IaaS cloud services delivered by public cloud vendors. Today, users can easily build a large-scale computing environment through the use of an intuitive graphical user interface on such IaaS cloud services, even if such users do not have any detailed knowledge on high performance computing, distributed computing, or networking, for example. Furthermore, users do not need be aware of the amount and utilization of computing resources on the cloud because of the enourmous amount of computing resources provided by such public cloud vendors. Moreover, users can purchase computing resources on the cloud anytime with ease through credit card transactions. On-demand features, simplicity and ease, brought by IaaS cloud services, have calmed scientists' fears about using IaaS cloud services instead of a part or all of computing resources in the supercomputing centers located in academic research institution and universities.

In this paper, we explore a cloud bursting environment where the computational workload on OCTOPUS as an on-premise environment is temporarily and urgently offloaded to the IaaS cloud when OCTOPUS

Date, S., Kataoka, H., Gojuki, S., Katsuura, Y., Teramae, Y. and Kigoshi, S.
First Experience and Practice of Cloud Bursting Extension to OCTOPUS.
DOI: 10.5220/0009573904480455
In *Proceedings of the 10th International Conference on Cloud Computing and Services Science (CLOSER 2020)*, pages 448-455
ISBN: 978-989-758-424-4

is overloaded. More technically and specifically, we aim to offload the computational workload on OCTOPUS to Microsoft Azure through the use of the cloud bursting technologies. Also, this research investigates the feasibility of the OCTOPUS-Azure cloud bursting environment as well as the technical issues that need to be tackled for the future use of IaaS Cloud for the future supercomputing environment at the CMC.

The structure of this paper is as follows. In Section 2, we briefly describe the structure of OCTOPUS and detail the current problems. Next, the technical requirements towards the envisioned cloud bursting functionality to be deployed on OCTOPUS are summarized. Section 3 presents the cloud bursting functionality which we have developed in this research. In Section 4, the experience and practice which we have obtained through this research are described as well as the evaluation result of the cloud bursting functionality. Section 5 describes related works and Section 6 concludes this paper.

2 TECHNICAL REQUIREMENTS

2.1 OCTOPUS Overview

Table 1: OCTOPUS System Configuration.

Compute Node	236 General purpose CPU nodes (471.24 TFlops)	CPU: 2 Intel Xeon Gold 6126 (SkyLake/2.6GHz/12C) Memory: 192 GB
	37 GPU nodes (858.28 TFlops)	CPU: 2 Intel Xeon Gold 6126 (SkyLake/2.6GHz/12C) GPU: 4 NVIDIA Tesla P100(SXM2) Memory: 192 GB
	44 Xeon Phi nodes (117.14 TFlops)	CPU: Intel Xeon Phi 7210 (Knights Landing/1.3GHz/64C) Memory: 192 GB
	2 Large-scale shared memory nodes (16.38 TFLOPS)	CPU: 8 Intel Xeon Platinum 8153 (Skylake/2.0 GHz/16C) Memory: 6TB
Storage	DDN EXAScaler (Lustre /3.1 PB)	
Interconnect	InfiniBand EDR (100Gbps)	

OCTOPUS is a 1.46 PFlops hybrid cluster system composed of general-purpose CPU (Intel Skylake) nodes, GPU (NVIDIA Tesla P100) nodes, many-core (Intel Knights Landing) nodes, large-scale shared-memory (6 TB memory) nodes, frontend servers and 3 PB Storage (Table 1). These nodes and storage are connected on an Infiniband EDR network and thus each node can take advantage of 100 Gbps full bi-directional bandwidth. Also, a Mellanox CS7500 switch, which is a 648-port EDR 100Gbps InfiniBand switch, houses all of the compute nodes and storage servers. This connection configuration allows each node to communicate with each other with 1-hop latency. All of compute nodes except the large-scale shared memory nodes have introduced DLC (direct liquid cooling) to cool their processers and accelerators to stably provide high performance.

For the operation of OCTOPUS, we charge users for the use of compute nodes on a per-node-hour basis. In other words, the CMC has introduced a service charge rule for OCTOPUS based on the cost for power consumed by using a compute node for an hour. For this reason, the CMC controls and operates OCTOPUS to avoid a situation where a job request cannot use a compute node simultaneously with other job requests from a perspective of fairness in service charge rules.

To allow users to efficiently and effectively use the computational resources of OCTOPUS as well as perform the above-mentioned operational manner, NEC NQSII and JobManipulator (NQSII/JM), which are the proprietary job management servers from NEC, has been adopted as the job management server in OCTOPUS. This NQSII/JM has the functionality of receiving job requests from users and then assigning them to an appropriate set of compute nodes. As described above, since OCTOPUS is composed of heterogenous architectures, NQSII/JM sends job requests to an appropriate queue for one of the job classes, each of which prescribes the degree of parallelism, set for each type of compute node. This mechanism is in charge of constantly maintaining higher job throughput as well as reducing user waiting time. For the actual daily operation of OCTOPUS, we dynamically change the configuration of the queue of the NQSII/JM to reduce user waiting time by monitoring the current status of job submission and user waiting time.

2.2 Problem in OCTOPUS

Despite the administrators' efforts to avoid the high utilization and a long user waiting time, general-purpose CPU nodes and GPU nodes, in particular, have been faced with extremely high utilization since their beginning of operation. In 2019, the second operational academic year, the general-purpose CPU and GPU nodes have been operated between 80 and 90 percent on average. Also, many-core nodes have been operated with high utilization due to the tendency of users to utilize many-core nodes as alternate compute nodes for the general-purpose CPU nodes for the purpose of workload offloading. In short, some of users prefer to get their own job completed quickly on many-core nodes without waiting long even if their jobs cannot obtain better performance than general-purpose CPU nodes. The administrators at the CMC welcomed this high utilization situation of OCTOPUS at first because we recognized that the procurement of OCTOPUS was successful. Currently, how-

ever, we see that this high utilization is a serious problem to be avoided because the CMC is expected to deliver a high performance computing environment in a national joint-use facility to scientists and researchers in Japan.

2.3 Technical Requirements to Cloud Bursting Functionality

In this research the following five technical requirements have been determined for the realization of cloud bursting functionality on OCTOPUS: On-demandness, Transparency, Selectivity, Equality, and High-throughput.

2.3.1 On-demandness

As described in section 2.2, long user waiting time as a result of high utilization on OCTOPUS has become a serious problem. For this reason, the administrators prefer to forward a part of the user submitted job requests to the cloud resources that the CMC prepares in advance. Also, when the administrators learn that high utilization and waiting time is alleviated, they prefer to stop the forwarding of job requests. Furthermore, they expect the job management server to automatically stops the forwarding of job requests when the cloud resources prepared in advance are almost consumed by the execution of job requests on the cloud. To support the above-mentioned administrators' scenario, the envisioned cloud bursting functionality must provide the cloud resources in an on-demand fashion only when the administrators want to make the cloud open to the users, by assuming that the administrators want the users to use OCTOPUS as the on-premise environment for monetary cost saving.

2.3.2 Transparency

The CMC wants to control the use of the cloud resources dynamically and forwards the submitted job requests waiting in a queue of the job management server to the cloud based on their judgement. In this way, the user's submission way of job requests should not differ between on-premise and cloud environments. For example, the difference of user IDs and passwords requires the additional development of a technical solution that maps user IDs and passwords in the on-premise environment to the environment in the cloud. Furthermore, since OCTOPUS is a service system that charges users as described in section 2.1, the peripheral systems essential for actual operation of OCTOPUS, such as user and statics management and portals for user support, also need to be modified and redeveloped if the differences in user ID and password occur. In addition, the differences of the job management server used in the cloud and in the on-premise environment force the users to be aware of the differences in job script description. Furthermore, the differences in system disk image views from users cannot be easily accepted because users are forced to be aware of the differences in data location. For these reasons, in our envisioned cloud bursting environment, it is ideal that the users do not have to be aware of any differences between the on-premise environment and the cloud environment.

2.3.3 Selectivity

Even in the case that the cloud resource is available when OCTOPUS as the on-premise environment is overloaded, some users might prefer the use of an on-premise environment rather than the cloud environment due to their concerns about exposing data to the cloud environment. Taking this situation into consideration, the cloud bursting functionality on OCTOPUS requires some means of checking the user's preferences and judgement regarding the use of the cloud environment.

2.3.4 Equality

The computational results should be identical or within a margin of error between on-premise and cloud environments, especially when the cloud bursting functionality allows users to submit jobs to the on-premise and cloud environments in a transparent manner. For example, many users check and verify the correctness and preciseness of their own computation on the on-premise OCTOPUS. Therefore, we cannot forward their job requests to the cloud without confirming that these criteria are satisfied.

2.3.5 High Throughput

The cloud bursting functionality has to actually reduce users waiting time even if it can alleviate an overloaded situation on the on-premise environment.

3 CLOUD BURSTING FUNCTIONALITY

3.1 Overview of OCTOPUS-Azure Cloud Bursting Environment

Microsft Azure is an IaaS cloud service that delivers a variety of virtual machines composed of Intel or AMD processors, NVIDIA GPU (Kepler, Maxwell,

Pascal and Volta), and InfiniBand in the world region. Today through the use of Azure, users can easily build their own supercomputers that satisfy their computing requirements and needs. Also, when any HPC-related technology such as brand-new processors, accelerators, and interconnects is released, Microsoft Azure plans to promptly deliver the corresponding virtual machine to users. In addition, Osaka University has tied an EES contract with Microsoft for improving the quality of the educational environment and therefore it is relatively easy to add Azure service to the contract. For this reason, we have adopted Microsoft Azure as an IaaS cloud service to realize the cloud bursting functionality on OCTOPUS.

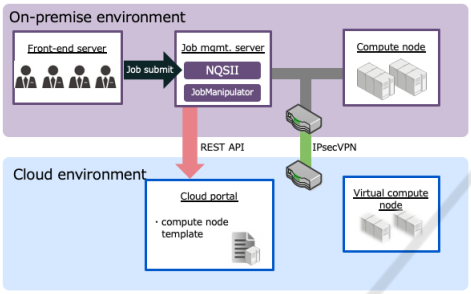

Figure 1: Overview of OCTOPUS-Azure Cloud Bursting Environment.

Figure 1 shows an overview of OCTOPUS-Azure cloud bursting environment which we have built in this research. To embody our envisioned cloud bursting functionality, we have developed the following three main extensions: (1) cloud bridge network, (2) job management server, and (3) deployment of compute node image on OCTOPUS to the cloud.

3.2 Cloud Bridge Network

As described in section 2.3, the envisioned OCTOPUS-Azure cloud bursting environment has to allow users to transparently use on-premise and cloud environments in the same way. To perform this, the virtual compute nodes deployed on Azure needs to be monitored, managed and controlled in the same way as the compute nodes on OCTOPUS. Also, the software stack needs to be identical on both the compute nodes on OCTOPUS and the virtual compute nodes on Azure. Furthermore, the large-scale storage on OCTOPUS needs to be used on both in an identical fashion. Taking this requirement into consideration, the virtual compute nodes to be deployed on Azure should be on the same logical network as the compute nodes on OCTOPUS.

Figure 2 shows an overview of the cloud bridge network between OCTOPUS and Azure. In this research, we have adopted IPSec VPN to connect the private logical network in OCTOPUS to the virtual network in Azure. For this purpose, NEC Univerge IX2106 has been newly deployed so that the traffic goes through the public Internet between the virtual network gateway on Azure and the IX2106 gateway on OCTOPUS. On the Azure side, the cloud management network used for the management of virtual compute nodes, and the cloud service network used for interprocess communication among virutal compute nodes and acccess traffic to storage on OCTOPUS have been built and configured.

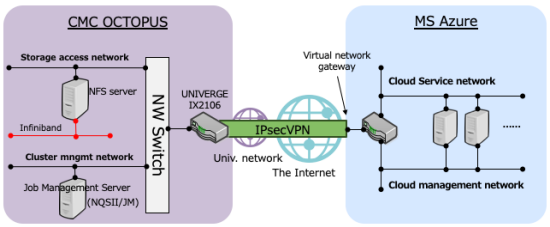

Figure 2: Structure of Cloud Bridge Network.

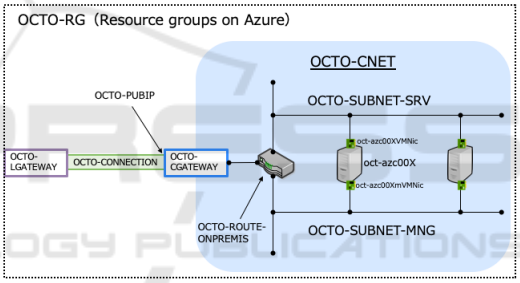

Figure 3: Structure of Network in Azure.

Figure 3 details the structure of the network built in Azure. *OCTO-LGATEWAY* and *OCTO-CGATEWAY* and *OCTO-CONNECTION* are Azure resources for the above-mentioned VPN connection between Azure and OCTOPUS. OCTO-CNET is a virtual network composed of two subnets: OCTO-SUBNET-SRV as the service network and OCTO-SUBNET-MNG as the cloud management network. Each of virtual compute nodes (oct-azx00X) on Azure is configured to have a virtual NIC for each subnet. OCTO-ROUTE-ONPREMIS is the routing table configured so that the traffic from the virtual NIC on virtual compute nodes is configured to flow to the network on OCTOPUS.

3.3 Job Management Server Extension

As described in section 2.3, it is ideal for users that there be no difference in terms of the way to submit jobs to the on-premise environment or the cloud environment. Also, it is desirable that the execution of

their jobs is completed quickly and promptly. It does not matter which the cloud or the on-premise environments on which the job is executed on. For this reason, the job management server needs to dynamically control job execution based on the utilization status of the on-premise and the cloud environment.

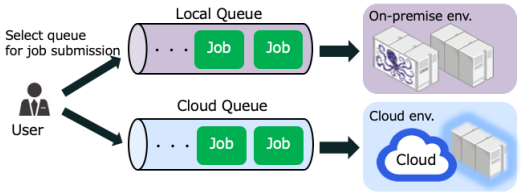

Figure 4: Queue Structure on Job Management Server.

In this stage of the research, however, we have not implemented any algorithm and operation policy for deciding which job requests in the queue should be forwarded to the cloud resources to improve the job throughput in the entire system. Therefore, in this research, we have set and configured a queue dedicated for job submission to the cloud in the job management server in addition to the queues for job submission to OCTOPUS as on-premise resources (Fig. 4). In the actual operation of OCTOPUS, we envision that the administrators enable and disable the cloud bursting functionality based on their consideration and judgement

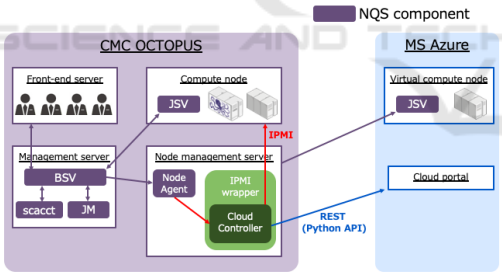

Figure 5: Structure of Job Management Server on the OCTOPUS-Azure environment.

Figure 5 shows the structure of the job management server on the OCTOPUS-Azure environment. NQSII/JM is composed of a batch server (BSV) that has the responsibility of receiving and executing jobs, a scheduler (JM) that is in charge of job scheduling, a job server (JSV) that monitors the compute nodes to be managed, an accounting (SCACCT) that manages the statics of compute node usage and a node management agent (NodeAgent) that is in charge of fault or failure detection and power management on compute nodes.

To satisfy the on-demandness requirement listed in section 2.3, we need to implement some function that invokes virtual compute nodes on Azure when any job request is submitted to the queue dedicated for job submission to the cloud and then stops the requests when virtual compute nodes are not used for a while. In this research, we have immplemented such a function on NQSII/JM because we can leverage the energy-saving function built into NQSII/JM that allows us to operate the compute node on the on-premise environment with less power consumption by turning on and off the compute nodes. Furthermore, we have developed an IPMI Wrapper and Cloud Controller triggered by the NodeAgent by extending this energy-saving function.

3.4 Deployment of the Compute Node

A general-purpose CPU node is composed of 2 Intel Xeon Gold 6126 (Skylake / 1.6GHz 12 cores) with 192 GB memory. On OCTOPUS, 236 general-purpose nodes are connected on an InfiniBand EDR network. In this research, we explore the offloading of the computational workload on these general-purpose nodes which have the highest utilization on OCTOPUS to the cloud. To allow all of the users' job requests submitted to OCTOPUS to be forwarded to the Azure cloud, the compute nodes to be deployed on Azure need to have higher performance in terms of memory size and the number of cores which users specify when requesting compute resources in their job script file in submitting their job requests. Also, the network between the on-premise and the cloud environments needs a higher bandwidth and low latency in the OCTOPUS-Azure environment for data communication between such environments. For this reason, we have adopted the Azure service provided in the western Japan region, since the western Japan region is the nearest region where the CMC is located and is expected to provide higher network performance. For the use of a virtual machine on Azure, we have selected F48s_v2 Linux virtual machine, which has 48 virtual processor cores with 96 GB memory on the top of Intel Xeon Platinum 8168 (Skylake / 2.7 GHz 24 cores) to make 32 virtual machines (oct-azc001 – oct-azc032) based on the OCTOPUS compute node image. Simultaneously, we have generalized the compute node image on Azure, so that users can perform their job executions with the same user ID and password regardless of the job executions on the cloud or the on-premise environment.

4 EVALUATION AND DISCUSSION

4.1 On-demandness

To offload the computational workload to the cloud environment, just enabling the queue for job submission to the cloud on the job management server is required. The configuration of enabling and disabling the queue can be easily conducted by the administrator through the use of *qmgr*, a command provided by NQSII/JM. Figure 6 shows an example of qmgr execution for enabling the queue for job submission to the cloud. This figure shows two queues of *INT-AZT* and *O-AZT* are configured targetting the virtual compute node on Azure. As this figure shows, the administrator can perform this operation quickly and promptly in response to the utilization or user waiting time status of OCTOPUS. For this reason, the administrator can reduce the cost for the cloud resources, by enabling the use of the cloud only when it is necessary.

```
octopus $  qmgr -Pm
 Mgr: enable interactive_queue=INT-AZT
 Mgr: start interactive_queue=INT-AZT
 Mgr: enable interactive_queue=O-AZT
 Mgr: start interactive_queue=O-AZT
 Mgr: exit
```

Figure 6: Control of Queue for Job submission to Cloud.

4.2 Transparency

As described in Section 2.1, the job submission to OCTOPUS is conducted through the use of the *qsub* NQSII/JM command with a job script. Figure 7 shows an example of the job script file. Also, in the current OCTOPUS-Azure environment where we have built in this research, the users can submit jobs to the cloud, just by specifying INT-AZT or OCT-AZT as the name of queue for job submission.

```
#!/bin/bash
#PBS -q (Queue Name)
#PBS -l elapstim_req=1:00:00
cd $PBS_O_WORKDIR
./a.out
```

Figure 7: Example of Job Script File.

In the current stage of this research, as described by the above example, the current OCTOUS-Azure environment does not have high transparency for users in terms of using the cloud-bursting environment because users cannot help being aware of the existence and difference of the cloud and on-premise environments. We recognize that this transparency issue needs to be reconsidered in addition to selectivity issue described next.

4.3 Selectivity

In the computing environment at supercomputing centers, users exist who cannot permit nor approve the use of the cloud for their own computation because, for example, they just do not want to use cloud or they are aware of data security. Under the OCTOPUS-Azure enviroment as of the writing this paper, users can specify the use of cloud or on-premise environments for job submission in the job script file and thus, users have selectivity in job submission to some degree. Since our research goal is to alleviate the high utilization of OCTOPUS, however, it is desirable that the job management server automatically choose and transfer user submitted job requests that users submit to OCTOPUS, to the cloud so that OCTOPUS utilization is lowered.

From this above viewpoint, we have to extend the user interaface to the job management server so that users can specify whether they want to use the cloud or not in submitting jobs. A possible solution might be to let users describe, for example, "*PBS -cloud yes*" in the job script file shown in Fig. 7. A tradeoff between transparency and selectivity issues in users' using the cloud is an important future issue in the cloud bursting functionality.

4.4 Equality

In this research, to make sure that there is no difference in the computation result between OCTOPUS and Azure, we have used *cpi.c*, which is a sample program shipped with MPICH 3.3.1, and a Fortran MPI code used at the CMC. cpi is coded to use an internode parallelism and our user's code only uses intranode parallelism.

These two programs were executed on both OCTOPUS and Azure. As a result, these two programs output the same result.

4.5 High Throughput

To evaluate whether the OCTOPUS-Azure environment can contribute to the alleviation of high utilization on OCTOPUS, we measured the performance of the OCTOPUS-Azure cloud bridge netowrk, in other words, communication performance between the virtual compute node on Azure and the computational

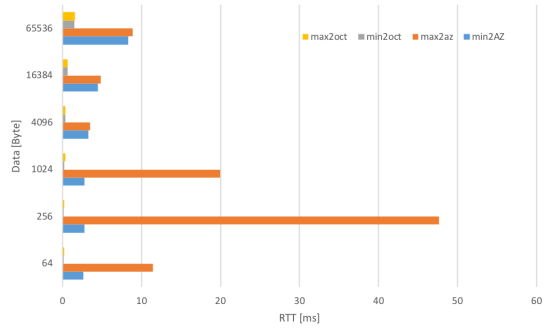

Figure 8: RTT from OCTOPUS Frontend server.

performance of the virtual compute node from an OCTOPUS Frontend server

4.5.1 Performance of Cloud Bridge Network

In this measurement test, we measured RTT (round-trip time) from a front-end server of OCTOPUS to the general-purpose compute node on OCTOPUS and the virtual compute node on OCTOPUS using the *ping* command. The measurement to each server was conducted 100 times on each packet size. Also, we repeated this measurement 5 times each in the morning, noon, and evening.

Figure 8 shows the measurement results. The graph in the figure plots the observed maximum and minimum RTT to OCTOPUS compute node and Azure virtual compute node. This result indicates that the RTT between the frontend server and the compute node on OCTOPUS was small and stable while the one between the frontend server and virtual compute node on Azure fluctuated greatly.

4.5.2 Communication Performance between Virtual Compute Node on Azure

To observe the communication performance among 32 virtual compute nodes on Azure deployed in this research, we measured the communication performance from oct-azc001 to each of the remaining virtual compute nodes with MPI PingPong. Figure 9 shows the measurement result when the size of data was 4 MB. This result indicates that there were variations in throughput performance. The maximum throughput performance observed was 510.35 MB/sec, and the minimum throughput performance was 188.6 MB/sec. The average performance was 432.687 MB/sec.

The observed variation in performance could impact on the total execution time of user jobs. Also, we observed that the performance also fluctuated depending on time factors. As the result of our investigation

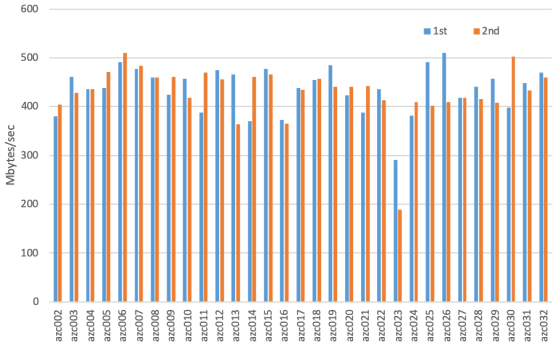

Figure 9: MPI PingPong Result (data size = 4MB).

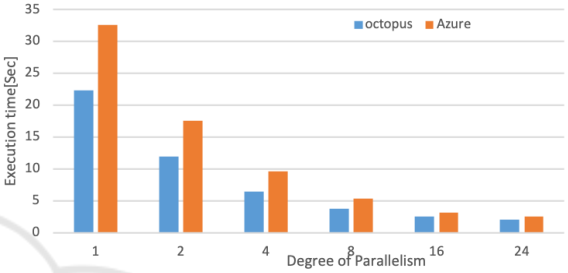

Figure 10: Performance Comparison with GROMACS.

after the measurement test, we believe that this performance characteristics was partly caused from the default virtual switch built into the virtual compute node on Azure. In the next step, we plan to perform the measurement test using an Accelerated Network (Microsoft, 2019a) configuration provided by Microsoft Azure.

4.5.3 Computational Performance of Virtual Compute Node on Azure

We measured the computational performance of GROMACS (GROMACS, 2019), used at the CMC, on an OCTOPUS general-purpose CPU node and a virtual compute node (F48s_v2) on Azure. The version of GROMACS used for this comparison is GROMACS 2016.1 installed onto OCTOPUS (CMC, Osaka University, 2019a).

Figure 10 shows the measurement result. The X axis shows the number of cores used for intranode parallelism, and the Y axis shows the execution time. Although virtual compute nodes built in this research have higher performance processors than the ones on OCTOPUS, the performance of GROMACS was not higher than that on OCTOPUS. As of the writing of this paper, we have not finished the investigation of performance bottleneck in GROMACS, but we plan to continue the investigation. Also, we still need to verify whether the OCTOPUS-Azure environment can alleviate the high utilization of OCTOPUS in op-

eration and then reduce user waiting times by offloading the computational workload to the cloud.

5 RELATED WORK

Many studies that aim to integrate computational resources from multiple administrative domains have been reported until today. Grid computing is an example of such studies. The Globus grid toolkit (Foster and Kesselman, 1997) was developed to integrate computational resources, each of which may be located on different administrative domains on the Internet to realize a world-wide supercomputing environment that solves unsolved scientific problems. For this purpose, the grid toolkit provides a suite of component services and APIs to allow the users and developers to use the underlying computational resources without being aware of the heterogeneity and complexity. GSI (Butler et al., 2000), GRAM (Stelling et al., 1998), and MDS (Fitzgerald et al., 1997) are just examples of such component services for metacomputing. The research summarized in this paper shares this vision, but the integration management of computational resources differs from the research in the Grid era. In this research we envision the emergent and on-demand use of computational resources in response to user demands, rather than the mutual sharing of computational resources.

6 CONCLUSION

In this paper we explored the feasibility of the cloud bursting environment composed of OCTOPUS and the IaaS cloud service. Specifically, we presented the OCTOPUS-Azure environment and show the evaluation result which we have obtained so far in terms of the following criteria: On-demandness, Transparency, Selectivity, Equality and High-throughput. Although confirming the feasibility of the cloud bursting functionality from OCTOPUS to the cloud, we have also shown that there are still several unsolved problems related to the unstable and fluctuating performance of virtual compute nodes on the cloud. In addition, there are selectivity and transparency issues of the cloud bursting functionality when actually deploying and operating the cloud bursting functionality as a service of the CMC. Furthermore, whether this functionality alleviates the workload on OCTOPUS in actual operation and then reduces user waiting time or not must still be evaluated.

ACKNOWLEDGEMENTS

This work was partly supported by JSPS KAKENHI Grant Number JP16H02802. Also, this work was achieved through the use of OCTOPUS at the Cybermedia Center, Osaka University.

REFERENCES

Butler, R., Welch, V., Engert, D., Foster, I., Tuecke, S., Volmar, J., and Kesselman, C. (2000). A national-scale authentication infrastructure. *Computer*, 33(12):60–66.

CMC, Osaka University (2019a). How to use GROMACS (OCTOPUS). http://www.gromacs.org/.

CMC, Osaka University (2019b). OCTOPUS. http://www.hpc.cmc.osaka-u.ac.jp/en/octopus.

Fitzgerald, S., Foster, I., Kesselman, C., von Laszewski, G., Smith, W., and Tueck, S. (1997). A directory service for configuring high-performance distributed computations. In *Proceedings of 6th IEEE Symposium on High-Performance Distributed Computing*, pages 365–375.

Foster, I. and Kesselman, C. (1997). Globus: A metacomputing infrastructure tookit. *High Performance Computing Applications*, 11(2):115–128.

Gong, C. and Amin, A. (2018). Best practices for deploying high availability architectures on Oracle Cloud Infrastructure. In *Oracle Technical White Paper*.

GROMACS (2019). GROMACS. http://www.gromacs.org/.

Mathew, S. (2019). Overview of amazon web services. In *AWS White paper*.

Microsoft (2019a). Create a Linux virtual machine with Accelerated Networking using Azure CLI. https://docs.microsoft.com/bs-latn-ba/azure/virtual-network/create-vm-accelerated-networking-cli.

Microsoft (2019b). Microsoft Azure. https://azure.microsoft.com.

Stelling, P., Foster, I., Kesselman, C., Lee, C., and von Laszewski, G. (1998). A resource management architecture for metacomputing systems. In *Proceeding of 7th IEEE Symposium on High Performance Distributed Computing*, pages 268–278.