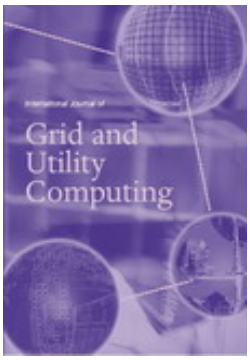


International Journal of Grid and Utility Computing

ISSN online: 1741-8488 - ISSN print: 1741-847X
https://www.inderscience.com/ijguc

An evaluation environment for high-performance computing combining supercomputing and cloud

Yusuke Gotoh, Toshihiro Kotani

DOI: 10.1504/IJGUC.2023.10054822

Article History:

Received:	17 October 2020
Last revised:	18 February 2021
Accepted:	22 February 2021
Published online:	21 March 2023

An evaluation environment for high-performance computing combining supercomputing and cloud

Yusuke Gotoh*

Faculty of Natural Science and Technology,
Okayama University,
Okayama 7008530, Japan
Email: y-gotoh@okayama-u.ac.jp
*Corresponding author

Toshihiro Kotani

Graduate School of Natural Science and Technology,
Okayama University,
Okayama 7008530, Japan
Email: kotani@s.okayama-u.ac.jp

Abstract: In recent computer environments that use a wide variety of data, users need a system that can efficiently process data according to the scale and type of data. There are two types of typical computing environments: supercomputers and cloud systems. Cloud services can provide computer resources based on the scale of the computer environment desired by users. On the other hand, in conventional large-scale computer environment that only consists of CPUs or GPUs, the processing time greatly increases according to the scale of the calculation processing. Based on the characteristics of the supercomputer and the cloud system installed in the Hokkaido University Information Initiative Center, Japan, we aim to construct a high-performance computing environment by linking the two types of systems. In this paper, we propose an evaluation environment for high-performance computing combining supercomputing and cloud systems. Our proposed system can construct a high-performance computing environment based on the scale of the computing process by the cooperation of the supercomputing and cloud systems with short physical distance and short network distance. In our evaluation of deep reinforcement learning using our proposed system, we confirmed that computer resources can be effectively used by allocating suitable processing for the supercomputer and the cloud according to the usage situations of the CPU, the GPU, and the memory.

Keywords: cloud service; high-performance computing; processing time; supercomputer.

Reference to this paper should be made as follows: Gotoh, Y. and Kotani, T. (2023) 'An evaluation environment for high-performance computing combining supercomputing and cloud', *Int. J. Grid and Utility Computing*, Vol. 14, No. 1, pp.29–36.

Biographical notes: Yusuke Gotoh received a Bachelor's degree from Okayama University, Okayama, Japan, in 2005 and Master's and Doctor's degrees from Kyoto University, Kyoto, Japan, in 2007 and 2009, respectively. In April 2009, he joined the Graduate School of Natural Science and Technology, Okayama University, Okayama, Japan as an Assistant Professor and in February 2014, he became an Associate Professor. From April 2011 to November 2012, he was a Visiting Researcher at La Trobe University, Melbourne, Australia, as a JSPS Postdoctoral Fellow for Research Abroad. His research interests include broadcast computing, spatial computing, high-performance computing, and scheduling.

Toshihiro Kotani received Bachelor's and Master's degrees from Okayama University, Okayama, Japan, in 2016 and 2018, respectively. His research interests include high-performance computing and scheduling.

This paper is a revised and expanded version of a paper entitled 'High-Performance Computing Environment with Cooperation Between Supercomputer and Cloud' presented at the '10th International Workshop on Streaming Media Delivery and Management Systems (SMDMS 2019)', Antwerp, Belgium, 8 November 2019.

1 Introduction

In recent computer environments that use a wide variety of data, users need a system that can efficiently process data according to the scale and type of data. In order to analyse such a wide variety of data, users need a large computing environment, such as a supercomputer or the cloud. For example, the RIKEN Center for Computational Science (R-CCS) (RIKEN, 2012) operates a supercomputer system called "K computer" using a CPU. Also, AlphaGO (Silver et

al., 2016) can analyse large-scale data. By using the supercomputer, users can predict the future of large-scale natural environments (MRI, 2003), such as tsunamis and climatic variations. On the other hand, users can evaluate the calculation processing with a cloud service, such as Amazon Web Services (AWS, 2004). In these simulation evaluations, cloud services use CPU-based virtual servers and many GPUs.

In the case of a conventional large-scale computing environment where only the CPUs and GPUs are used for computation, the computation time increases significantly depending on the scale of computation processing. In cloud services, users can use the resources according to the computing environment. For example, distributed deep reinforcement learning requires simulation evaluation using memory and many CPUs, and then training using GPUs based on the results of the simulation evaluation. Therefore, users need to use a computing environment that combines CPU and GPU to reduce processing time.

When users process large amounts of data in the cloud, the sheer volume of computation increases the computation time. Using a supercomputer to process such data may reduce the computation time. However, there are several problems in combining traditional supercomputers with cloud systems.

Based on the characteristics of the supercomputer and the cloud system installed in the Hokkaido University Information Initiative Center, we aim to construct a high-performance computing environment by linking the two types of systems. Specifically, we will build a mechanism for cooperation system in Hokkaido University, taking advantage of the geographical and network proximity of the supercomputer and the unified processor architecture used in the system.

In this paper, we design an evaluation environment for high-performance computing combining supercomputing and cloud systems. In the proposed environment, we make the mechanism that cooperates with the supercomputer and the cloud with short network and short physical distances. Our evaluations show that the performance of deep reinforcement learning according to the increase number of CPUs is improved. Our contribution is to construct an evaluation environment that realises load balancing according to the scale of computation by combining the supercomputer and the cloud.

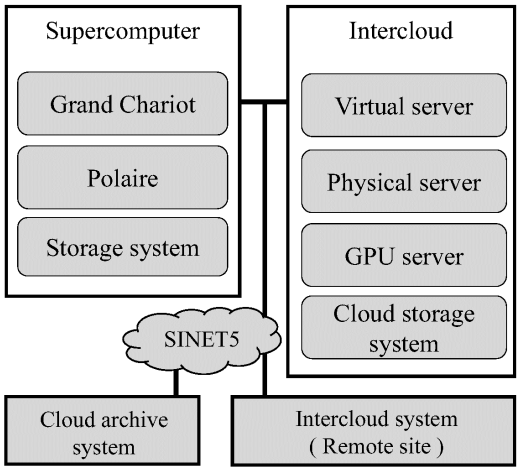
The rest of the paper consists of the following. We explain the configuration of the network system in Hokkaido University in Section 2. In Section 3, we design the functions with the supercomputer and the cloud. Related works are introduced in Section 4. In Section 5, we introduce the distributed deep reinforcement learning. We implement the proposed environment using deep reinforcement learning in Section 6. In Sections 7 and 8, we evaluate the performance of the proposed environment and discuss it. Finally, in Section 9, we conclude our paper.

2 Network system

2.1 Hokkaido University Information NEtwork System (HINES)

In Figure 1, we show the configuration of the Hokkaido University Information NEtwork System (HINES, 2004). HINES is the system in operation at Hokkaido University Information Initiative Center, which consists of a supercomputer system and an intercloud system. HINES is constructing a widely distributed cloud system using servers in various parts of Japan connected by a Scientific Information NETwork 5 (SINET5) (NII, 2016).

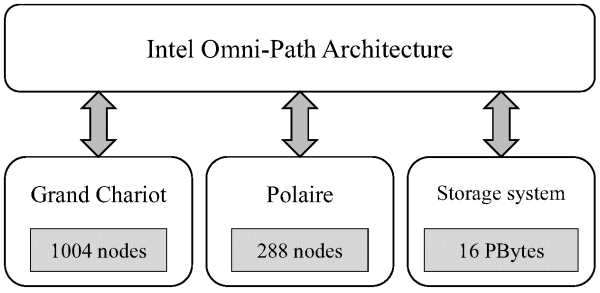
Figure 1 Hokkaido University Information NEtwork System (HINES)



2.2 Supercomputer

The supercomputer system managed by the Hokkaido University Information Initiative Center is shown in Figure 2. The system is connected by an Omni-Path network and is operated by a storage system and two computer systems called the Grand Chariot and Polaire.

Figure 2 Supercomputer system



The specifications of the supercomputers in the Hokkaido University Information Initiative Center are shown in Tables 1 and 2, respectively. The operation performance of this system is 3.96 PFLOPS, which is more than 20 times higher than that of the supercomputer previously used at Hokkaido University. This supercomputer can accelerate a variety of applications related to computational science. In addition, by installing an Intel processor and Linux OS, the Intel C/C++/Fortran compiler and various libraries can be used on the supercomputer.

2.3 Intercloud

The intercloud system managed by the Hokkaido University is shown in Figure 3. This intercloud system is operated by several universities in Japan as a globally distributed cloud

system with multiple servers. In Table 3, the specifications of cloud-launched storage and online storage are shown. Kitami Institute of Technology also provides a remote backup function with magnetic tape archiving equipment. We show the specifications of the magnetic tape archiving system at Kitami Institute of Technology in Table 4.

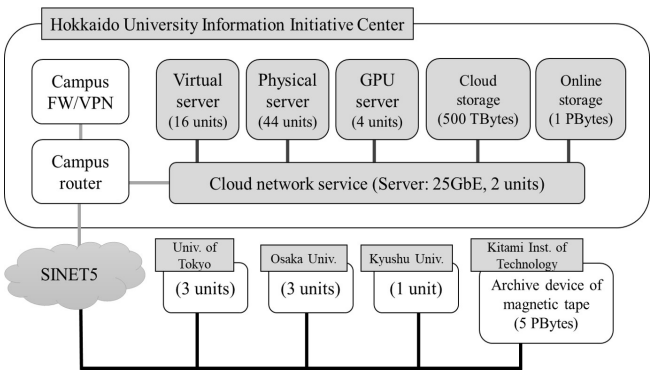
Table 1 Grand Chariot

Total	Nodes	1,004
	Spec	3.08 PFLOPS
	Memory	376.5 TBytes
	Network topology	Full-bisection Fat Tree
Node	Processors (Cores)	2 (2 × 20 = 40)
	Spec	3.072 TFLOPS
	Memory	384 GBytes
	Storage	240 GBytes SSD × 1
	Interconnect	Omni-Path × 2 ports
	OS	CentOS
Processor	Clocks	2.4 GHz
	Cores	20
	Spec	1.536 TFLOPS

Table 2 Polaire

Total	Nodes	288
	Spec	0.87 PFLOPS
	Memory	31.5 TBytes
Node	Processors (Cores)	1 (1×68 = 68)
	Spec	3.0464 TFLOPS
	Memory	96 GBytes
	Storage	64 GBytes SATA Flash × 1
	Interconnect	Omni-Path × 1 port
	OS	CentOS
Processor	Clocks	1.4 GHz
	Cores	68
	Spec	3.0464 TFLOPS

Figure 3 Intercloud system



The cloud servers, cloud-activated storage, and online storage in the Hokkaido University Information Initiative Center are connected to the cloud network system. The cloud server group consists of three types of servers: virtual servers, physical servers, and GPU servers, which are managed to start and stop using OSS Openstack.

Table 3 Cloud storage

Cloud storage A	Name	ETERNUS DX200 S4
	Capacity	SSD and HDD (320 TBytes)
Cloud storage B	Name	ETERNUS DX200 S4
	Capacity	SSD and HDD (220 TBytes)
Online storage	Name	DDN GS7K + SS8462
	Capacity	1.0 PBytes

Table 4 Cloud archive

Cloud Archive	Tape Library	Name	ETERNUS LT270 S2
		Archive capacity	5.0 PBytes
	Device	Cartridge	LTO Ultrium8
	Tape back Server	Name	PRIMERGY R2530 M4
		Nodes	4

3 Design

In the case of processing large scale data at high speed in a cloud system, using a supercomputer may reduce the computation time. In this research, we aim to make a high-performance computing environment that combines the advantages of both supercomputers and cloud systems.

As an example of the current supercomputer/cloud collaboration system, we consider the collaboration of the SR16000 (HITACHI, 2014) of the Hokkaido University Information Initiative Center and a cloud system by a private company. Also, we explain an overview of this collaborative system. The following three issues can be identified as problems with our collaborative system.

3.1 Location of system

In Japan, supercomputers are mainly used in universities and research institutes, and most of them are installed in universities. On the other hand, cloud systems have been installed not only in Japan but also all over the world, considering the environment in which the systems are operated. Users choose the most suitable environment for their use. Therefore, when a supercomputer and a cloud system work together, their locations are far from each other physically, which increases the processing time for data communication.

3.2 Security risk of data leakage

In many cases, the supercomputer and the cloud system belong to different network systems, so they need to go through an external network when communicating. Therefore, users need to process data beyond the firewalls installed by their respective networks, which poses a risk of data leakage.

3.3 Operating environment of program

If the processor architecture and OS of the supercomputer and the cloud system are different, a problem arises where the service programs running on the cloud system cannot be executed on the supercomputer. The supercomputer uses a processor with the POWER architecture, which is less consistent with many civilian cloud systems.

3.4 Requirements

The requirements for the environmental design of our research are as follows.

1 The supercomputer and the cloud can work together to perform deep learning.

2 The cloud can execute the program at any time.

3 The timing of the job execution in the supercomputer can be clarified.

3.5 Design policy

The design policy of task and resource allocation between supercomputer and cloud is as follows.

1 Increase the task/resource allocation of the cloud for the processing of applications that can be accelerated by using the GPU.

2 When the difference of processing performance between CPU-only and CPU + GPU is small, the task/resource allocation between the supercomputer and the cloud is determined by considering the status of other applications running in the environment.

3 When the computational resources are insufficient for processing by the supercomputer alone, the cloud is combined with the supercomputer to temporarily increase the resources for computation.

4 Related works

Nicolas et al. (2019) create a scalable cloud supercomputer software architecture for photo-realistic terapixel visualisation, and rigorously evaluate their application on the cloud supercomputing. They confirmed that the GPU compute resource available in the cloud provides access to globally competitive resources and the direct financial cost of access was low compared to procuring and running these systems.

To provide a set of baseline measurements showing the kind of performance actually obtainable from storage systems under real-world conditions, Michael et al. (2017) ran similar measurements against a modern high performance compute cluster's storage array in a distributed manner with multiple client nodes to measure obtainable performance in a massively parallel configuration. They provide a baseline assessment of the performance and capabilities that can be expected when choosing a storage solution.

Antonenko et al. (2019) present an environment for academic multidisciplinary research called MC2E that aggregates heterogeneous resources such as private/public clouds, HPC clusters and supercomputers under a unified easy-to-use interface. MC2E can schedule parallel applications between clouds and supercomputers based on their performance and resource usage.

Ritu et al. (2019) have extended the BOINC software infrastructure that is the most popular software framework for Volunteer Computing (VC) that uses donated computing cycles on the devices and have made it amenable for integration with the supercomputing and cloud computing environments. They discussed the need for harnessing the available computing power through the cloud computing systems so that if there is a need for special hardware that is not available on the other volunteered resources, they can still service the jobs submitted through BOINC with a reasonable guarantee for the quality of service.

Michael et al. (2016) compare with the early days of clustered computing systems and the current state of cloud computing based on their background in high-performance computing. They have demonstrated that cloud computing architectures are suitable for data-intensive scientific analysis.

In the task and workflow scheduling in inter-cloud environments, Mohammad et al. (2019) put forward a comprehensive analysis and survey of the scheduling frameworks provided for the inter-cloud systems. They provide general background knowledge about the task and workflow scheduling and classifies various methods and features that are considered in the scheduling process.

In order to develop a Supercomputing Cloud Platform (SCP) prototype system using Service-Oriented Architecture (SOA) and Petri nets, Ziyun et al. (2016) researched some technologies for Web service composition. They showed that the reliability of the Web service composition has a correlation with the number of Web services and the range of reliability transition values.

Andrew et al. (2017) investigate the use of containers that is OS-level virtualisation with advanced supercomputing and HPC system software. They evaluated the performance of their containerisation methods by comparing native HPC benchmarks to those running in a container.

In utilising containers for HPC applications and the current approaches used in many HPC container runtimes, Shane et al. (2019) have outlined the challenge and trade-offs in detail, with current example problems from real-world supercomputing deployments. They also explore several potential methods for solving or elevating container interoperability.

5 Methods for deep learning

5.1 Distributed deep reinforcement learning

In this section, we describe the distributed deep reinforcement learning used to evaluate the usefulness of the proposed evaluation environment.

In deep learning, which is used for image recognition and clustering, a neural network based on brain information is built on a computer. This neural network can discriminate against other inputs by learning the discrimination and predicted output for the input using the dataset.

In deep reinforcement learning based on deep learning, deep learning is performed on the reward-maximising process that occurs in reinforcement learning. Users learn by inputting the current state and then outputting the reward-maximising behaviour.

Distributed deep reinforcement learning is a distributed processing method in deep reinforcement learning. In distributed deep reinforcement learning, users get a large amount of training data by using an environment where training is performed in parallel.

Table 5 Evaluation environment

Supercomputer	Nodes	32
	Processors	2
	Theoretical performance	3.072 TFLOPS
	Intranodes Storage	240 GBytes SSD × 1
	OS	CentOS
Cloud	Name	PRIMERGY 2540 M4
	CPU	Intel Xeon Gold 6138 (20 C/2.0 GHz) × 2
	Memory	256 GBytes
	GPU	NVIDIA Tesla V100 (PCIe/16 GBytes) × 2
	Storage	SSD (3.84 TBytes) × 2
	OS	CentOS

5.2 Ape-X

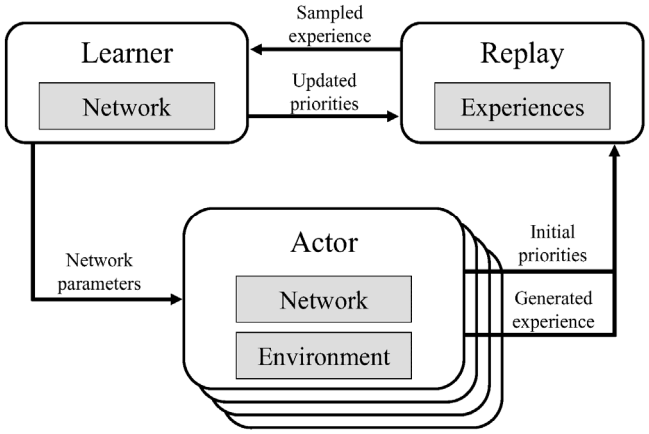
Ape-X (Horgan et al., 2018) is a learning method to improve the accuracy of distributed deep reinforcement learning by reducing the learning time.

In Figure 4, we show the configuration of Ape-X. In Ape-X, there are three types of learning models: Actor, Learner, and Replay Memory. The training data generated by the Actor is stored in Replay Memory. The Actor uses the network and the learning environment to determine their actions. The Actor periodically copies parameters from the Learner and updates the network. The Replay Memory transmits the data retrieved from the Actor to the Learner. The Learner uses the learning data received from the Replay Memory to perform deep learning. Finally, the Learner updates the parameters and provides a copy of the network to the Actor.

5.3 Deep Q-Network

A Deep Q-Network (DQN) (Bellemare et al., 2015) makes a neural network (NN) in which the input is set to the state and the output is set to the action. In addition, DQN learns the action value function Q by comparing the predicted result by NN with the actual execution result. However, DQN may overestimate the output because the NN for selecting the action and that for evaluating it are the same. Therefore, Double Deep Q-Network (DDQN) evaluates the action using the NN generated last time in order not to overestimate the output result.

Figure 4 Ape-X



6 Implementation

In this section, we implement a high-performance computing environment combining supercomputing and cloud systems.

6.1 Programs for evaluation

We use Distributed Tensorflow (Google, 2019) as a program to link deep learning between the supercomputer and the cloud. In Distributed Tensorflow, we create clusters for parallel distributed processing using multiple Tensorflow servers. By operating a parameter server and a worker server and assigning processes, the supercomputer and the cloud can be linked together.

horovod (Sergeev et al., 2018) is a mechanism for distributed processing in deep learning, which reduces costs in training and speeds up distributed processing. When Distributed Tensorflow implements an evaluation environment with distributed processing using parameter and worker servers, both the cost of adapting traditional code to distributed processing and the cost of training increase.

In this paper, we use deep reinforcement learning and the cloud through distributed supercomputer connections. The horovod automatically connects from the host server to other servers via SSH based on the configuration file. Therefore, horovod does not need to check the connection status. The Distributed Tensorflow requires a parameter server and a worker server in order for the supercomputer and the cloud to work together to run the program.

The supercomputer cannot set the opportunity to run a job because the supercomputer puts the job in a waiting state based on the occupancy status. In this paper, we collaborate with the supercomputer and the cloud by checking the execution status of programs in the supercomputer.

6.2 Network environment

We describe a network for connecting the supercomputer and the cloud. Distributed Tensorflow sets the server on which the cluster is built by port number and IP address. In this case, it is not possible to connect directly to the supercomputer from the cloud and vice versa. The cloud needs to be connected to a login node that sends jobs to the supercomputer. However, the supercomputer consists of multiple nodes and it is very difficult to connect to the cloud. On the other hand, the supercomputer cannot set the IP address of the cloud to connect to it.

In the proposed environment, the supercomputer is connected to the cloud at the login node. In addition, the supercomputer and the cloud can be connected to each other on the network through an SSH port.

7 Evaluation

The requirements for the evaluation environment in this paper are as follows:

1 In order to link the supercomputer and the cloud on the network, create an SSH port forwarding system that interconnects the supercomputer and the cloud by connecting the supercomputer to the cloud via a login node.

2 The inter-cloud system is built on a physical server that can occupy resources.

3 The cloud checks the execution status of the program on the supercomputer.

The specifications of the computers used for the evaluation are shown in Table 5. In performance evaluations, there are three types of programs: the Distributed Tensorflow, the horovod, and the Ape-X.

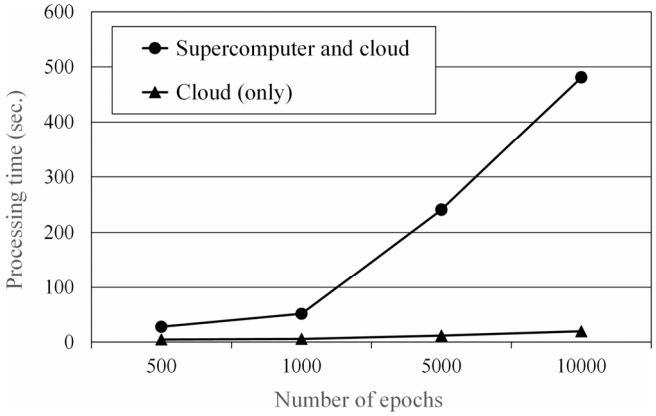
7.1 Computational processing using distributed tensorflow

We evaluate the performance of computational processing using a Distributed Tensorflow program to cooperate with the supercomputer and the cloud. Figure 5 shows the processing time with the Distributed Tensorflow. The horizontal axis is the number of epochs. The vertical axis is the processing time.

As for the comparison items, "Supercomputer and cloud" is the case in which Distributed Tensorflow is executed by integrating a cloud and a supercomputer. In the case where a supercomputer and a cloud are linked, one parameter server and one worker server are executed on the supercomputer and on the cloud, respectively. "Cloud-only" is the case in which Distributed Tensorflow is run only in the cloud. In this case, one parameter server and one worker server are executed in the cloud.

As shown in Figure 5, the processing time in the case of the cloud alone is shorter than in the case of using the supercomputer and the cloud. For example, when the number of epochs is 1,000, the processing time for the cloud alone is about 19.6 seconds, and about 481.2 seconds for the supercomputer and the cloud.

Figure 5 Computational processing using distributed tensorflow



In the proposed environment, two types of servers, a parameter server to update parameters and a worker server to train the same model using multiple mini-batches of finer-grained data, need to run on a supercomputer and a cloud, respectively. In this case, the processing time to synchronise the parameter server and the worker server between the supercomputer and the cloud becomes long, and the execution time becomes long.

For distributed deep reinforcement learning in a supercomputer and cloud-based environment, the Actor operates in parallel and asynchronously. Learner learns in the cloud, which is closer to the network distance. Therefore, the network delay in training is less than the result of the execution time shown in Figure 5.

7.2 Performance of distributed processes using horovod

The results of the cloud distribution benchmark with horovod are shown in Figure 6. The horizontal axis is the number of GPUs used in the cloud. In this evaluation, we set up two cases: one and two. The vertical axis is the number of images processed by the cloud per second.

Figure 6 Performance of distributed processes of cloud using horovod

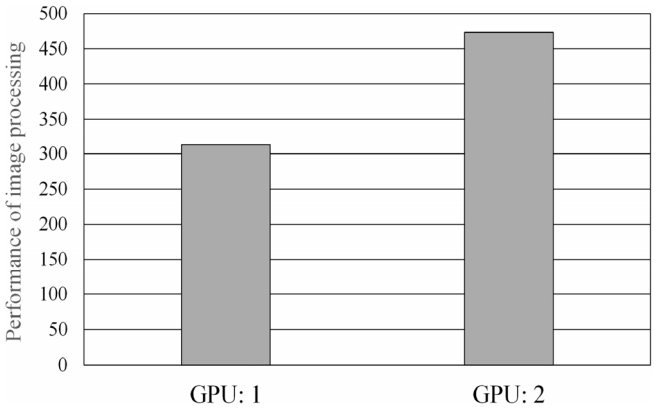
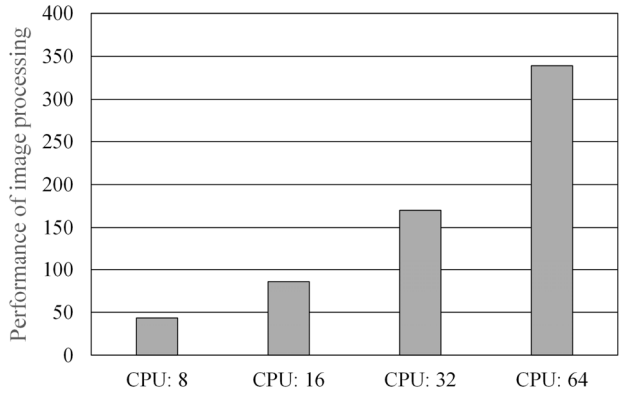


Figure 6 shows that the number of image processing is about 312.9 when there are one GPU and about 473.2 when there are two GPUs. In this case, increasing the number of GPUs improves the performance of the benchmark. When the number of GPUs is two, the number of image processing per GPU is about 236.6, which is about 24.4% lower in performance than the number of GPUs with one GPU. Thus, the performance per GPU decreases as the number of GPUs increases, compared to the case of a single GPU.

Next, the total benchmark values of the supercomputer in the distributed process using horovod are shown in Figure 7. The horizontal axis is the number of CPUs, and we have four types: 8, 16, 32, and 64. The vertical axis is the number of images processed per second. From Figure 7, the benchmark value increases proportionally with the increase in the number of CPUs. For example, the benchmark values for 8 CPUs are about 43.3, 86.0, 170.1, and 339.3 for 16, 32, and 64 CPUs, respectively.

Figure 7 Performance of distributed processes of supercomputer using horovod

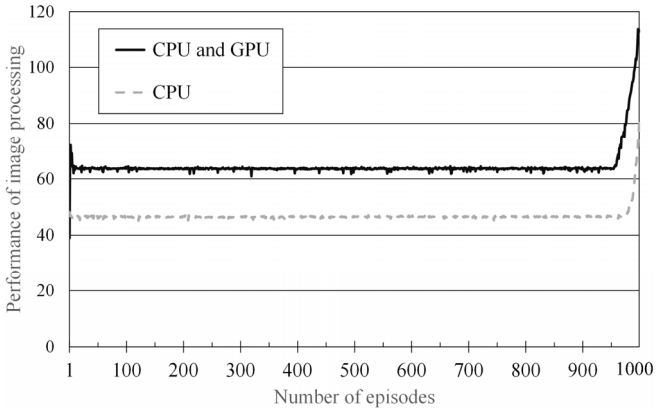


7.3 Performance of distributed processes using Ape-X

We evaluate the performance of computational processing using the Ape-X program. The results of the execution time when Ape-X is used are shown in Figure 8. The horizontal axis is the number of episodes and the vertical axis is the processing speed of Actor per second in Ape-X. The learning model of Ape-X is implemented by asynchronously running the parallel processing performed by Actor with Replay memory and the deep learning processing in Learner. There are two types of evaluation items. "CPU and GPU" is a case where the Actor uses both CPU and GPU. "CPU" is a case in which only the CPU is used without assigning a GPU. The number of Actors is 12. Therefore, in the case of "CPU + GPU", Learner uses 1 GPU server, Actor uses 1 GPU server and 11 CPU servers. In the "CPU" case, Learner uses 1 GPU server and Actor uses 12 CPU servers.

In Figure 8, the processing speed per second is about 63 steps/sec. between 1 and 960 episodes when both CPU and GPU are used. When using only the CPU, the processing speed per second is about 46 steps/sec. for the number of episodes between 1 and 970 or less. In this case, the processing speed of the CPU alone is about 26.9% lower than that of the case where both CPU and GPU are used. In both cases, the processing speed per second increases as the number of episodes approaches 1000, because the number of Actors decreases as the number of processes decreases.

Figure 8 Performance of distributed processes of cloud using Ape-X



Ape-X uses a network trained by Learner, so the performance is lower. On the other hand, the performance of the Actor is not much lower than the benchmark in horovod because of the high speed of the simulation using the reinforcement learning environment on the CPU. Therefore, even if the Actors do not use GPUs, the performance can be improved by using a larger number of CPUs in an environment where the supercomputer and the cloud are linked.

8 Discussion

8.1 Processing performance and execution time of distributed tensorflow in proposed environment

When distributed deep reinforcement learning is performed in the proposed environment where the supercomputer and the cloud are linked, the Actor running in parallel runs asynchronously. In addition, Learner runs on the cloud where the network distance is closer. Therefore, by using more parameter servers and worker servers for training, the processing performance of Distributed TensorFlow is improved and the execution time of Distributed TensorFlow is reduced.

8.2 Additional descriptions of costs

The authors show experiment results such as Figures 6, 7 and 8. However, they are natural results because more resources show higher performances. On the other hand, there are no mentions of those costs, e.g. expenses, setup time, and energy consumption. The reviewer requests the additional results or descriptions of the costs.

9 Conclusion

In this paper, we proposed a high-performance computing environment that combines the advantages of supercomputers and cloud systems. In the proposed system, we developed a high-performance computing environment for a variety of computational tasks by linking the supercomputer and the cloud at the Hokkaido University Information Initiative Center. In the evaluation, we performed distributed deep reinforcement learning using Distributed Tensorflow, horovod, and Ape-X, and confirmed that increasing the number of CPUs improved the performance.

In the future, we will design and implement the computing environment using horovod.

Acknowledgments

Authors would like to thank Prof. Takeshi Iwashita of Hokkaido University for helpful discussion. This work is supported by JSPS KAKENHI Grant Number 18K11265. In addition, this work is supported by "Joint Usage/ Research Center for Interdisciplinary Large-scale Information Infrastructures" in Japan (Project ID: EX17101).

References

Amazon (2004) *Amazon Web Services (AWS)*. Available online at: https://aws.amazon.com/jp/

Andrew, J.Y., Kevin, P., Ryan, E.G. and Ron, B. (2017) 'A tale of two systems: using containers to deploy HPC applications on supercomputers and clouds', *Proceedings of the 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, https://doi.org/10.1109/CloudCom.2017.40.

Antonenko, V., Petrov, I., Smeliansky, R. Huang, Z., Chen, M. Cao, D. and Chen, X. (2019) 'MC2E: meta-cloud computing environment for HPC', *Proceedings of the 27th International Symposium Nuclear Electronics and Computing (NEC'2019)*, pp.7–15.

Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. and Hassabis, D. (2015) 'Human-level control through deep reinforcement learning', *Nature*, Vol. 518, pp.529–533.

Google (2019) *Distributed TensorFlow*. Available online at: https://www.tensorflow.org/deploy/distributed

Hokkaido University Information Initiative Center (2004) *Hokkaido university Information NEtwork System (HINES)*. Available online at: https://www.hines.hokudai.ac.jp/

Hokkaido University Information Initiative Center (2014) *Supercomputer HITACHI SR16000/M1*. Available online at: https://www.hpci-office.jp/pages/e_h26_boshu_hpci_resource

Horgan, D., Quan, J., Budden, D., Barth-Maron, G., Hessel, M., Hasselt, H.V. and Silver, D. (2018) 'Distributed Prioritized Experience Replay', *Proceedings of the International Conference on Learning Representations (ICLR 2018)*. Available online at: https://arxiv.org/pdf/1803.00933.pdf

Meteorological Research Institute (MRI) (2003) *Supercomputer System*. Available online at: https://www.mri-jma.go.jp/Facility/supercomputer_en.html

Michael, J., Jeremy, K., William, A., David, B., Bill, B., Vijay, G., Michael, H., Matthew, H., Peter, M., Andrew, P., Albert, R., Siddharth, S. and Paul, M. (2017) 'Performance Measurements of Supercomputing and Cloud Storage Solutions', *Proceedings of the 2017 IEEE High Performance Extreme Computing Conference (HPEC)*, https://doi.org/10.1109/HPEC.2017.8091073.

Michael, S.W., Samuel, W.S. and Matthew, T. (2016) 'Data-intensive supercomputing in the cloud: global analytics for satellite imagery', *Proceedings of the 7th International Workshop on Data-Intensive Computing in the Cloud (DataCloud '16)*, pp.24–31.

Mohammad, M. and Mehran, Z. (2019) 'Efficient task and workflow scheduling in inter-cloud environments: challenges and opportunities', *The Journal of Supercomputing*, https://doi.org/10.1007/s11227-019-03038-7.

National Institute of Informatics (NII) (2016) *Science Information NETwork 5*. Available online at: https://www.sinet.ad.jp/en/top-en.

Nicolas, S.H., Manu, A., James, C., Stephen, D., Philip, J. and Mark, T. (2019) 'Petascale cloud supercomputing for terapixel visualization of a digital twin', *The Computing Research Repository (CoRR)*. Available online at: https://arxiv.org/pdf/1902.04820.

RIKEN Center for Computational Science (R-CCS) (2012) *K computer*. Available online at: https://www.r-ccs.riken.jp/en/k-computer/.

Ritu, A., Carlos R. and Gerald, J. (2019) 'Scalable software infrastructure for integrating supercomputing with volunteer computing and cloud computing', *Communications in Computer and Information Science*, Vol. 964, pp.105–119.

Sergeev, A. and Del Balso, M. (2018) 'Horovod: fast and easy distributed deep learning in TensorFlow', *arXiv*. Available online at: https://arxiv.org/abs/1802.05799.

Shane, R.C. and Andrew, J.Y. (2019) 'A case for portability and reproducibility of HPC containers', *Proceedings of the 2019 IEEE/ACM International Workshop on Containers and New Orchestration Paradigms for Isolated Environments in HPC (CANOPIE-HPC)*, https://doi.org/10.1109/CANOPIE-HPC49598.2019.00012.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Driessche, G. et al. (2016) 'Mastering the game of go with deep neural networks and tree search', *Nature*, Vol. 529, pp.484–489.

Ziyun, D., Lei, C., Tingqing, H. and Tao, M. (2016) 'A reliability calculation method for web service composition using fuzzy reasoning colored petri nets and its application on supercomputing cloud platform', *Future Internet*, Vol. 8, No. 47, https://doi.org/10.3390/fi8040047.