Performance Improvement of Deep Learning Training on Large-scale Manycore Cluster

Toshihiro Hanawa
Kohei Tamura*
hanawa@cc.u-tokyo.ac.jp
The University of Tokyo
Tokyo, Japan

ABSTRACT

To shorten a large-scale training for deep learning, the distributed deep learning are widely applied to the massive clusters using accelerators such as GPUs. In contrast, manycore processor such as Intel Xeon Phi is also suitable for computing deep learning operation and it is easy to expand to large-scale cluster. In this study, to utilize deep learning training on large-scale many core cluster, we conduct performance evaluation of large-scale deep learning framework ChainerMN on Oakforest-PACS system operated by JCAHPC, and optimize the Allreduce communication latency. As a result, the improved communication of ChainerMN is 2.1x faster than the original one on Oakforest-PACS system.

KEYWORDS

Deep Learning, Intel Xeon Phi, Knights Landing, MPI, OmniPath, Oakforest-PACS

ACM Reference Format:
Toshihiro Hanawa and Kohei Tamura. 2019. Performance Improvement of Deep Learning Training on Large-scale Manycore Cluster. In *ICPP '19: International Conference on Parallel Processing, August 05–08, 2019, Kyoto, Japan.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Today, Deep Learning (hereafter, DL) has been utilized in many fields. DL requires more complex and deeper network model and a large amount of dataset for supervised training to achieve high accuracy. GPU is widely used for such trainings, however the memory capacity is limited up to 16 GB or 32 GB at most, and the communication among GPUs has larger latency due to the bottleneck by PCIe I/F.

Manycore CPU can address these problems thanks to the self bootable feature and larger memory capacity. We are developing large-scale deep learning environment using Oakforest-PACS supercomputer system (OFP) with 8,208 nodes of Intel Xeon Phi processors (KNL)[4]. The performance of KNL processer is similar to NVIDIA P100 in terms of the peak FLOPS and memory bandwidth of MCDRAM, in addition, KNL has DDR4 memory as well [5]. OFP also employs high performance interconnect and storage system, and as the result, these features of OFP are helpful for large-scale DL training. In this study, we target ChainerMN, which is the multi-node version of Chainer [1, 2].

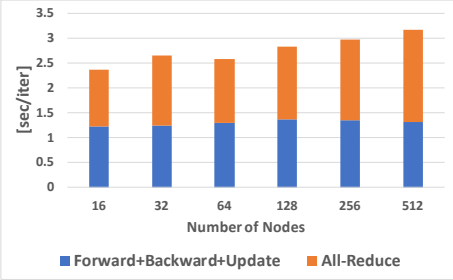

Figure 1: Breakdown of Execution Time per Iteration

2 PERFORMANCE ANALYSIS

First, we investigate the performance on OFP using ImageNet. We used ResNet-50 as the neural network model, and iDeep developed by Intel is adopted as the backend of the chainer for KNL's AVX-512 instructions. The parallelization can be specified as the combination between multithreading by OpenMP and multiprocessing by MPI.

Fig. 1 shows the breakdown of the execution time per iteration with various number of nodes. The learning process is performed using the fixed number of mini-batch size, therefore, it is weak-scaling problem. As the result, the elapsed time of Allreduce procedure exceeds half of the entire execution time, and Allreduce becomes bottleneck as the number of nodes increases.

3 OPTIMIZATION

To reduce the communication time of Allreduce part, we propose the new communication algorithm instead of Allreduce. This method is based on the 2-D Torus allreduce proposed by Mikami for the GPU cluster[6]. As shown in Fig. 2, the number of processes is decomposed by $M \times N$. In this algorithm, first, Reduce-scatter is performed in column-direction, then, Allreduce is done in row-direction, and finally, Allgather collects from and shares with M members in column-direction.

*Currently he is in NTT DATA Corporation.

ICPP '19, August 05–08, 2019, Kyoto, Japan

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Table 1: Specification of Oakforest-PACS

	Oakforest-PACS (OFP)
CPU	Xeon Phi 7250
Freq., # of cores, Peak Perf.	1.4 GHz, 68c, 3.04 TF
Memory	16 GB (MCDRAM)+ 96 GB (DDR4)
Mem. BW	490.0 GB/s (effective), 115.2 GB/s
Interconnect	OmniPath 100 Gbps
Parallel File System	Lustre File System
Capacity	26 PB
Fast File Cache	DDN Infinite Memory Engine (IME)
Capacity	940 TB
Data Transfer Rate	1.56 TB/s

Table 2: Software Environment

Software	version
Python	3.6.3
Intel MPI	2018.1.163
MPI4py	3.0.0
Chainer	5.0.0
iDeep4py	2.0.0

In this approach, the advantage is the average calculation of each element can be performed in parallel manner with M after the Allreduce on the second step.

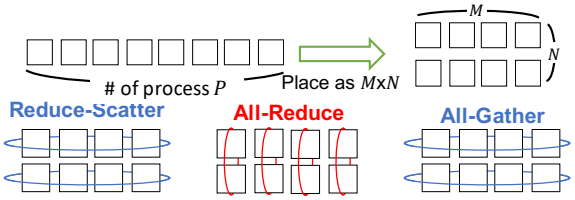

Figure 2: Communication Algorithm of 2-D Torus Allreduce

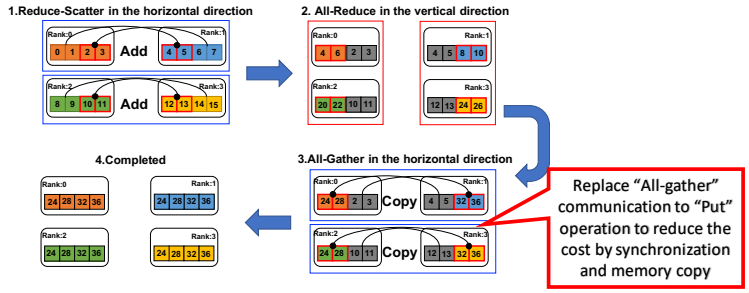

Figure 3: Proposed Communication Algorithm Based on 2-D Torus Allreduce

Moreover, we propose the new method using one-sided communication mechanism instead of collective communications. Fig. 3 indicates the example of the communication using Put as one-sided communication instead of Allgather. In this case, Put is naturally suitable for the data placement rather than Allgather operation.

Fig. 4 shows the elapsed time of each Allreduce process. As the result, in the case of 256 processes, we achieved over 2.1 times speedup by the communication algorithm using Reduce-scatter, Allreduce, and Put.

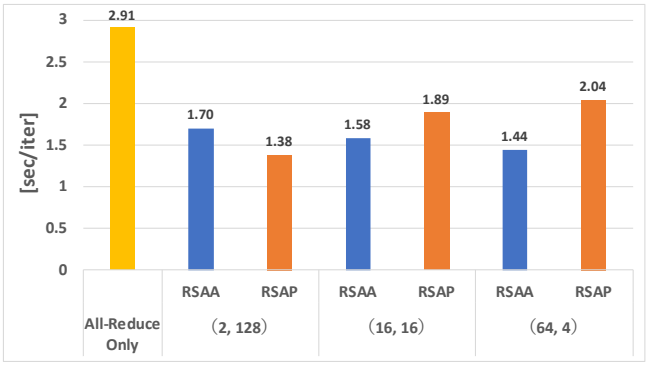

Figure 4: Comparison of Elapsed Time of Allreduce Process with 256 processes (Allreduce only: original MPI Allreduce, RSAA: Reduce-scatter, Allreduce, Allgather, RSAP: Reduce-scatter, Allreduce, Put)

REFERENCES

[1] S. Tokui, K. Oono, S. Hido, and J. Clayton: "Chainer: a next-generation open source framework for deep learning", NIPS, 2015
[2] Akiba, T., Fukuda, K. and Suzuki, S., "ChainerMN: Scalable Distributed Deep Learning Framework," Proc. of Workshop on ML Systems in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS), 2017
[3] T. Kurth, J. Zhang, N. Satish, et al.: "Supervised and Semi-Supervised Classification for Scientific Data", Supercomputing, 2017.
[4] Joint Center for Advanced HPC: "Oakforest-PACS," http://jcahpc.jp/eng/ofp_intro.html .
[5] A. Sodani: "Knights Landing (KNL): 2nd Generation Intel® Xeon Phi™ Processor," IEEE Hot Chips 27 Symposium, 2015.
[6] H. Mikami, et al., "ImageNet/ResNet-50 Training in 224 Seconds," arXiv preprint arXiv:1811.05233, 2018.