Student Cluster Competition 2020, Team ShanghaiTech University: Reproducing performance of MemXCT

GeekPie_HPC 08[1]

No.1,Zhongke Road, Pudong New District, Shanghai, China

Abstract

For the Reproducibility Challenge of Student Cluster Competition 2020, the results presented in MemXCT: Memory-Centric X-ray CT Reconstruction with Massive Parallelization were similarly produced and verified. This paper describes the ShanghaiTech University team's methods for reproducing the optimized X-ray CT reconstruction on AMD architecture with the Intel Parallel Studio Cluster 2019.5 compiler and compares our results with those in the original paper. Motivated by the need to validate conclusions from previous work, we reproduce the numerical results, performance of the algorithm, and the scaling behavior of the algorithms as the number of MPI processes increases.

Keywords: Reproducible computation, Student Cluster Competition, Reproducibility, Scalability

1. Introduction

As part of the Student Cluster Competition, the Reproducibility Challenge focuses on replicating experimental results presented in publications. As computational models and software play increasingly important roles in scientific results, reproducibility is of more importance to validate and verify the results of articles. This year's Student Cluster Competition nominated *MemXCT: Memory-Centric X-ray CT Reconstruction with Massive Parallelization* to reproduce. This paper proposes a memory-centric algorithm for X-ray computed tomography, avoiding the redundant computations in the conventional method. The MemXCT system optimized data communication, partitioning, and accesses with an improved implementation of sparse matrix-vector multiplication, and demonstrated the changes in performance metrics with multi-stage buffering and two-level pseudo-Hilbert ordering.

The ShanghaiTech team reproduced the paper in 72 hours, using limited budget during the Student Cluster Competition at VSCC20. The experiment was aimed to validate the result of the paper and evaluate the reproducibility of the algorithm on a different cluster. The team succeeded in reproducing the performance gains caused by the optimizations outlined in the original paper. We assessed the performance of the MemXCT system by comparing performances of single CPU and GPU, then validated its scalability using multiple datasets.

The remaining of this paper is organized as follows: Section 2 describes our experimental methodology. Section 3 demonstrates the planned experimental run. Section 4 compares the single CPU and single GPU performance with given datasets. Section 5 discusses the results and scalability of MemXCT with strong scaling. Section 6 contains our conclusion.

2. Experimental Setup

2.1. Machine and Compiler

To validate the performance of this system, we utilized a cluster with the specification provided in Table.1. The machine used was comprised of eight nodes, each 60-core AMD EPYC 7551 processors. With these processors, we can use up to the 126 GB/s memory bandwidth of hexa-channel. Each node hosts an installation of CentOS 8.0. For node-to-node connectivity, we use Nvidia Infiniband HDR to ensure

[1]mengyx@shanghaitech.edu.cn
[2]liuyc1@shanghaitech.edu.cn

Table 1
Environment specification.

Nodes	eight CPU nodes two GPU nodes
CPU Model	AMD EPYC 7551
GPU per node	Tesla P100 × 4
Cores per CPU node	60
Memory per node	384GiB
Storage per node	1TB
Interconnect	Infiniband HDR
Operating System	CentOS 8.0
Kernel	Linux kernel 4.18.0

Table 2
Software specification.

Compilers	Intel parallel studio 2019.05 icc 19.0.5.281 gcc 8.3.1
Cuda and CuDNN	11.1
Fiji	2.1.0

efficient MPI communication between nodes. In comparison, the original paper utilized 4096 KNL nodes with 256K cores.

The shared storage subsystem of our cluster mounted over Lustre and NFS filesystem over raid on Premium SSDs and Ultra SSDs across the cluster.

2.2. Software

As listed in Table.2, the cluster installs a complete development environment, including version of our compiler and the visualization tool. We choose the intel-parallel-studio@cluster.2019.5 to compile the code. We edited the makefile to adapt to our environment, and wrote scripts to run and analyzing the scalability based on scaling results.

2.3. Datasets

The datasets used during the competition was shown in Table 1, the four datasets were given to us, with the input sinogram and the output tomogram. The size of four input images ranges from 360x256 to 2400x2048, while the image used in MemXCT paper has up to 24000x16384 file size.

Table 3
Data used to reproduce results

Dataset	Sinogram(M×N)	Regular Data
ADS1	360×256	0.21/0.21 GB
ADS2	750×512	1.75/1.75 GB
CDS1	750×512	1.75/1.75 GB
CDS2	375×1024	3.50/3.50 GB
CDS3	1501×2048	56.06/56.06 GB

3. Description of experimental run

We migrated source code of MemXCT for CPU and GPU to our cluster, and compiled the code with the Interl parallel studio compiler. Over 60 loops are vectorized with average scalar cost 6.59, vector cost 2.16 and estimated potential speedup 3.87. We organized experiments as follows. The MemXCT paper use 62.5 cores per node in average, and we have 60 cores for each node. The KNL processors have a bottleneck introdued by shared memory, therefore when the size of datasets exceeds the capacity of shared memory, there appears a dramatic drop of GFLOPS with large buffer size and block size. However, such bottleneck does not exist in our CPU architecture, since out CPUs do not have a shared memory and the provided datasets can fit in CPU memory. Such property of the architecture predicts a different scalability for strong scaling experiment.

Due to limited budget, we used Tesla P100 GPU, while the MemXCT paper used instead of V100. The single-precision performance is 9.3 TeraFLOPS for Tesla P100 , and 16.4 TeraFLOPS for V100. The GFLOPS and effective memory bandwidth of a single CPU and GPU are then tested and compared on different datasets. The performance on a single Tesla P100 is unsurprisingly much better than a the performance on single AMD EPYC 7551 processor.

Table 4
Data used to reproduce results

	CPU	GPU
GFLOPS on ADS1	36.6470	126.5434
Mem B/W on ADS1	7.3655	487.2815
GFLOPS on ADS2	33.6689	119.3763
Mem B/W on ADS2	6.8672	491.2889

4. Single CPU vs single GPU Performance Comparison

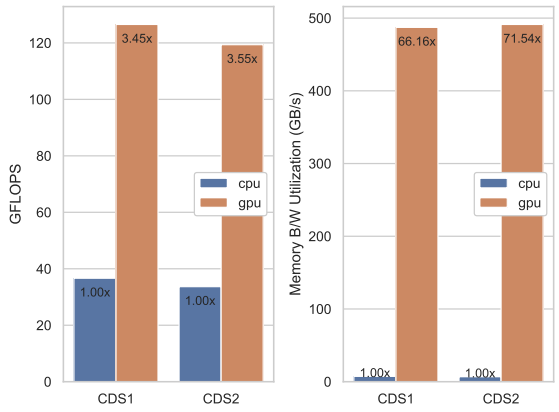

Figure 1 Test and compare GFLOPS and effective memory B/W on single CPU and GPU

To test the GFLOPS performance and effective memory bandwidth on single devices(CPU and GPU), we ran CDS1 and CDS2 on the total 60 cores of an AMD EPYC 7551 and a Tesla P100. We choose the block size and buffer size according to tuned parameters to obtain the best performance of them. The comparison between GFLOPS and memory bandwidth of a single CPU and GPU are shown in Figure.1, suggesting a great improvement on both metrics on GPU rather than CPU.

5. Strong Scaling on CPUs

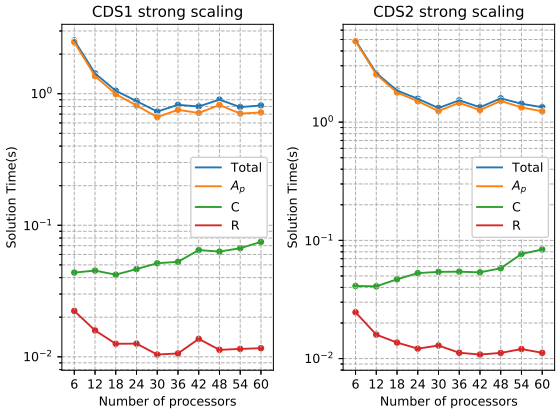

Figure 3 Strong scaling on CDS1 and CDS2 datasets,incresing number of processors

We reproduced the strong scaling experiment by reconstructing CDS1 and CDS2 samples. Fig.3 shows the strong scaling property of the application while running the CDS1 and CDS2 dataset. During the experiment, the sizes of the datasets remain the same as we increase the number of processors. The minimum number of processors we used to run the datasets is 60, to let the dataset fit within the system. Our cluster showed good scaling with up to 30 processors for CDS1 and CDS2, however, the performance of the MemXCT system drops with 48 CPU processors for both the CDS1 and the CDS2 datasets. Additionally, the MemXCT paper pointed out a super-linear speedup of A_p kernel, such a trend is not significant in our cluster after the number of processors increased to 30, probably due to the increasing communication overhead as the number of processors becomes larger.

We also performed strong scaling experiments with one, two, four and eight nodes, each node is equipped an AMD EPYC 7551 processor, while the MemXCT paper used up to 4096 nodes. The result for all three datasets are shown in Fig.2. The figure demonstrates that the communication overhead grows dramatically, causing the increasing trend for the total solution time when number of nodes changes from

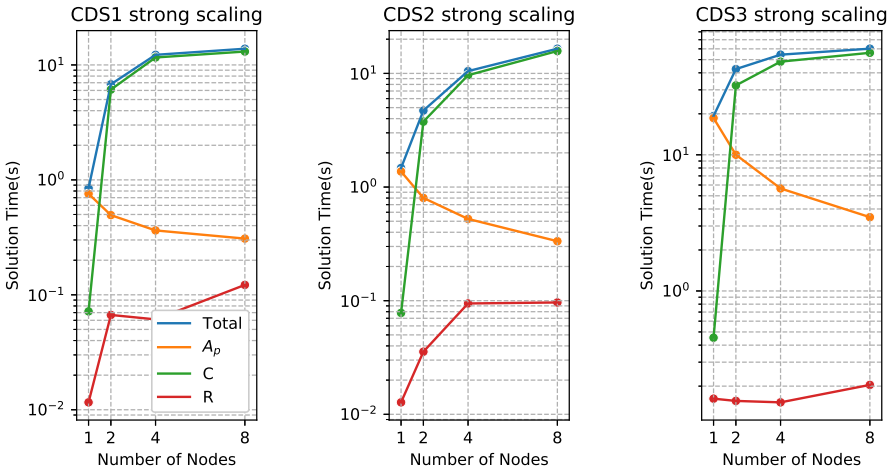

Figure 2 Strong scaling on CDS1, CDS2 and CDS3 datasets, incresing number of nodes

one to eight. Comparing to our strong scaling experiments on the single node, the super-linear speedup for the A_p kernel is more significant, especially in the CDS3 case.

6. Conclusion

In this work, we reproduced both single CPU and single GPU performance in the MemXCT paper. We also validated the scalability by conducting strong scaling experiments on single and multiple nodes, then compared the visualized output on different datasets. Though our results suggested that the MemXCT algorithm has some scaling properties on our architecture, we made explanation and draw consistent conclusions with the data.

Not only did we compare the GFLOPS and effective memory bandwidth between a single CPU and GPU, but also suggested a scaling property for all given datasets in a different environment with a limited computing budget, confirming the scalability of the MemXCT system.

References

[1] M. Hidayetoğlu, T. Biçer, S. G. De Gonzalo, B. Ren, D. Gürsoy, R. Kettimuthu, I. T. Foster and W. M.W. Hwu, "Memxct: Memory-centric x-ray ct reconstruction with massive parallelization," *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis.*, 2019.

[2] Mert Hidayetoglu,"MemXCT-CPU," `https://github.com/merthidayetoglu/MemXCT\protect\discretionary{\char\hyphenchar\font}{}{}CPU`,2020.

[3] Mert Hidayetoglu,"MemXCT-GPU," `https://github.com/merthidayetoglu/MemXCT\protect\discretionary{\char\hyphenchar\font}{}{}GPU`,2020.