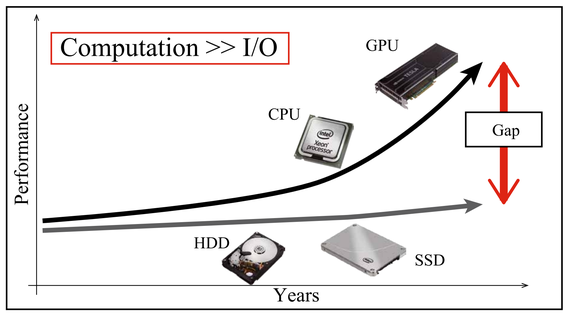
Acceleration of I/O data transfer with RDMA for massively large-scale GPU simulation

Shohei Onishi[1] Jerdvisanop Chakarothai[2] Takuto Ishii[3] Shingo Hashikawa[3] Yukihisa Suzuki[1]

1. Tokyo Metropolitan Univesity
2. National Institute of Information and Communications Technology
3. ELSA Japan Inc.

1. Introduction

Recently, Amount of I/O data for numerical simulations becomes very huge due to rapid increase of simulation scale and accuracy. However, there is a large gap between the computational performance of the processor units; i.e., CPU, GPU, and so on and the I/O performance of persistent/non-volatile storage devices such as HDD (Hard Disk Drive), SSD (Solid State Drive). Therefore, the ratio of latency time in I/O processes has been increasing, in comparison with that of computation time, in massively large-scale GPU simulati



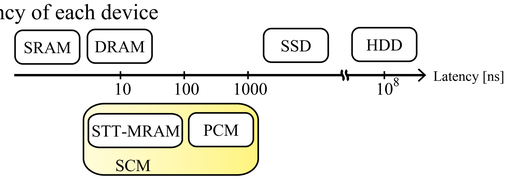
Studies on reduction of the latency time of I/O process

- Acceleration of I/O data transfer with parallel I/O
  - Lustre, Glusterfs etc.
- Using non-volatile memory instead of HDD (Hard Disk Drive)
  - SSD (Solid State Drive)
  - SCM (Storage Class Memory)[1][2]

In our reaserch, we focus on SCM.

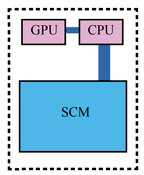
Schematic of SCM (Storage Class Memory)

Access Latency of each device



Features
- Non-volatility
- Low latency
- connectable to a memory bus
- low power consumption

Introduction of SCMs into computer architecture
- as a main mamory
- as a storage device



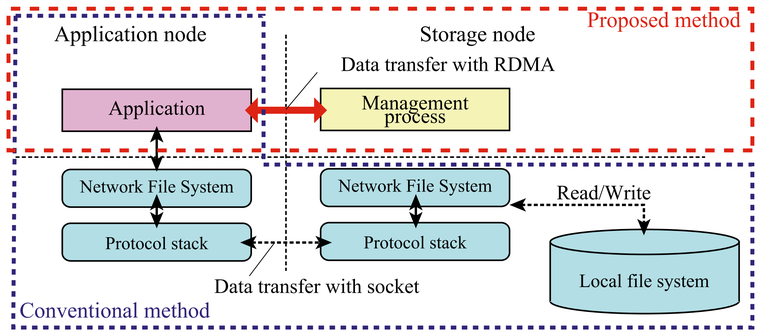
SCM has both features of main memory and storage device.

Purpose

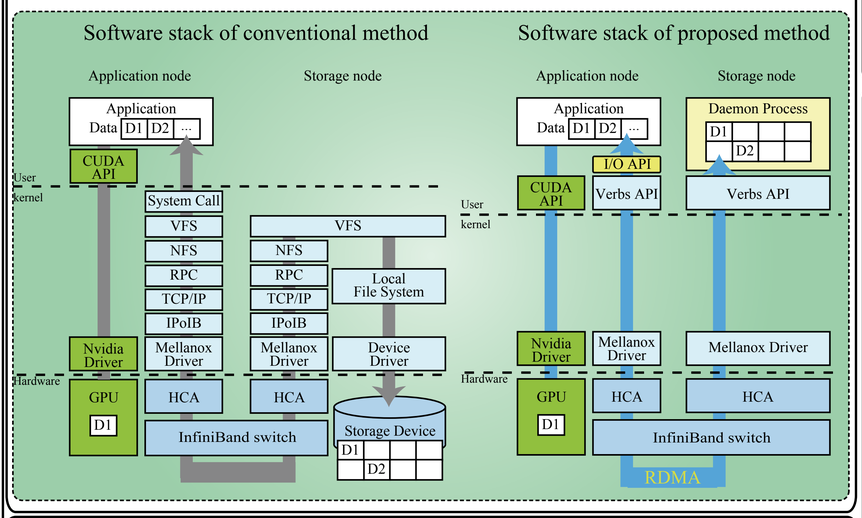
To propose a novel high performance network I/O, in order to achive a reduction of latency time for I/O process in large-scale GPU simulation.

- Using RDMA [3] on InfiniBand [4] interconnection between application node and storage node
- Assuming the introduction of SCMs as a main memory into the computer architecture

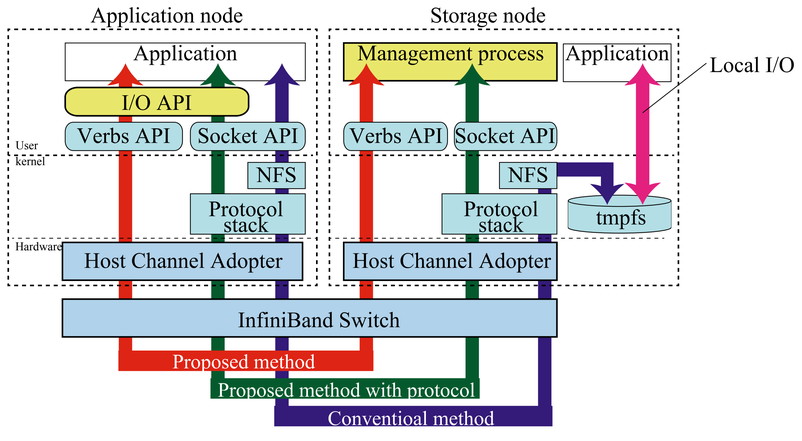
2. Design and implemention proposed method



- Management process is running as a daemon in a storage node which has a huge memory pool for data storage.
- Using RDMA on InfiniBand interconnection between application node and storage node
- Our proposed method bypasses software stacks such as network file system, TCP/IP and IPoIB. Therefore, it eliminates overhead of them.



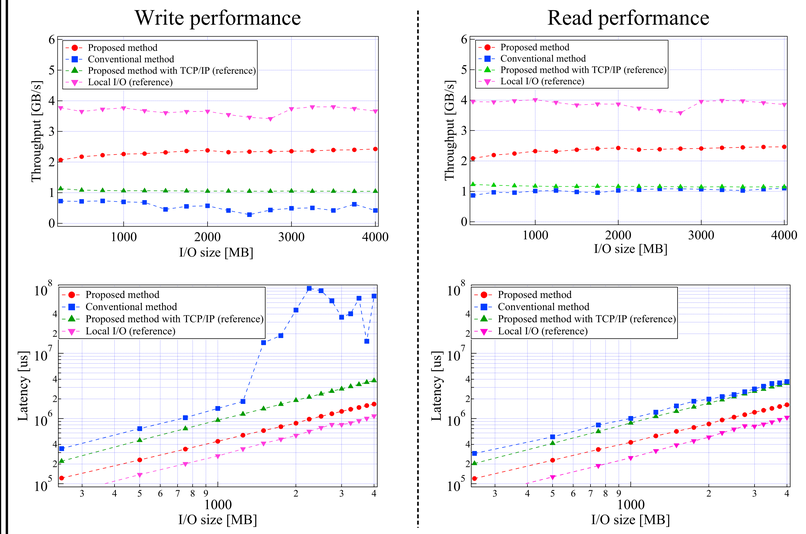
3.Measurement



Procesure of measurement
1) Preparing data array in application
2) following sequences were carried on ten times
   - Copy data from/to storage node using each method
   - Measure I/O latencies during data transfer.

CPU	Intel Xeon X5670
Main memory	16 GB
Network	InfiniBand QDR, 40 Gbps
OS	CentOS 6.3
Network file system	NFS
Local file system	tmpfs [5]

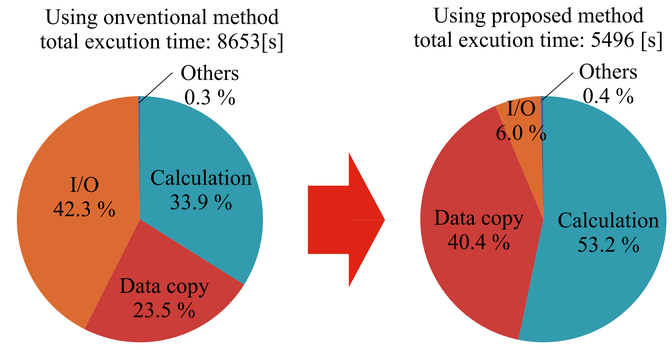
4. Results



- Write performance: approximately 2.4 GB/s, x5.7 compare to conventional method
- Read performance: approximately 2.5 GB/s, x2.7 conpare to conventional method
  * (I/O size: 4 GB)

5. Example of latency time of I/O process

Terget:
- 3 dimensional FDTD (Finite-Difference Time-Domain) method
- parallel computation using 4 GPUs (Tesla M2090 x4) in 1 node
- I/O size: 9.2 GB
- Number of calculation loop: 2000
- Number of I/O process: 200 (at each 10 step)



The latency time of I/O processes was reduced by approximately 6 %.

6. Conclusion

- We proposed the novel high performance network I/O, in order to achive the reduction of the latency time of I/O process in massively large-scale GPU simulation.
- We implemented the orignal I/O API which bypasses the software stacks such as network file system, TCP/IP and IPoIB.
- We found that the latency time of the I/O processes is reduced to approximately 6 % of total execution time by applying our proposed method to large-scale FDTD simulation.

References

[1] K.H. Park, S.K. Park, H. Seok, W. Hwang, D. Shin, J.H. Choi, and K. Park, "Efficient memory management of a hierarchical and a hybrid main memory for MN-MATE platform," Proc. International Workshop on Programming Models and Applications for Multicores and Manycores, pp.83-92, ACM, 2012.

[2] J. Jung, Y. Won, E. Kim, E. Kim, H. Shin, and B. Jeon, "FRASH: Exploiting storage class memory in hybrid system for hierarchical storage," ACM Trans. on Storage, vol.6, no.1. pp.1–25, ACM, 2010.

[3] R. Recio, B. Metzler, P. Culley, J. Hilland, D. Garcia, "A remote direct memory protocol specification," , RFC 5040, Oct. 2007.

[4] RDMA Aware Programming User Manual, http://www.mellanox.com

[5] P. Snyder, "tmpfs: A virtual memory file system," Proc. Autumn 1990 EUUG Conf., pp. 241–248, 1990.