Kento Sato

Postdoctoral researcher

Lawrence Livermore National Laboratory
7000 East Ave., Livermore, CA 94550
☎ *925(422)-6918*
✉ *kento@llnl.gov*
http://people.llnl.gov/sato5

April 4, 2017

Research Statement

Summary

My research interests are distributed systems and parallel computing in High Performance Computing (HPC). My major research areas includ user-level filesystem and I/O optimization (Hierarchical and on-demand filesystem, GPU-accelerated I/O interface and other filesystems for burst buffers); HPC tools (MPI reproducibily tool and noise injection tool); Resilience (Lossy checkpoint/restart, resilient burst buffer system, fault tolerant messageing interface, energy-aware checkpoiting technique, asynchronous checkpointing); and Cloud Computing (I/O optimization using VM migration).

Research area 1: User-level filesystem and I/O Optimization

2014-present **HuronFS (Hierarchical, User-level and On-demand File System):** When running data-intensive HPC applications which issue a huge amount of concurrent or parallel I/Os to shared storage, current public clouds cannot provide desirable execution environments for such I/O workloads with respect to performance and data consistency. In order to resolve these problems, our research group proposed a novel fast, scalable and fault tolerant filesystem called CloudBB (Cloud-based Burst Buffer). Unlike conventional filesystems, CloudBB creates an on-demand two-level hierarchical storage system and caches popular files to accelerate I/O performance. CloudBB enables scalable I/O with multiple metadata servers. CloudBB is also resilient to failures by using file replication, failure detection and recovery techniques. We evaluated performance of real data-intensive HPC applications in Amazon EC2/S3 with the CloudBB filesystem. The results showed CloudBB improves performance by up to 28.7 times while reducing cost by up to 94.7% compared to the ones without CloudBB [Xu et al., 2016, Xu et al., 2015b, Xu et al., 2015c, Xu et al., 2015a, Xu et al., 2014]. We also extend CloudBB to exploit low-latency and high-bandwidth interconnects (InfiniBand) for Supercomputers/HPC systems. Our evaluation showed CloudBB can achieve up to 3.5 GB/sec per a single I/O node in I/O throughput, which is comparable to the average throughput of the parallel file systems in Tokyo Tech 's supercomputer, TSUBAME2.5. We eventually relesed the CloudBB filesystem as HuronFS [Xu et al., 2017].

Future work: We will work on I/O performance modeling of CloudBB and automatic allocation so that CloudBB can dynamically tune the number of buffer nodes according to I/O workloads, thereby improving both performance and costs.

2014 **gmfs (User-level GPU-accelerated I/O Interface):** Many supercomputers equip accelerators such as GPUs on each compute node to accelerate computation. However, I/O-bound applications cannot be accelerated in such systems since these applications are not computation-bound but I/O-bound. Therefore, I/O-bound applications typically waste the accelerator resouces. To exploit accelerators and improve I/O-bound applications, our research group developed gmfs (GPU-accelerated I/O interface) that utilizes GPU device memory as buffer cache. Our experimental results showed gmfs can accelerate sequential read/write, and utilize 82% of PCIe-gen2 peak bandwidth, 50% of PCIe-gen3 peak bandwidth [Sato et al., 2014c].

2013-2014 **IBIO (User-level InfiniBand-based filesystem for burst buffers):** To propose resilient HPC systems, we explored use of burst buffers. Burst buffers are dedicated storage resources positioned between the compute nodes and the parallel file system, and this new tier within the storage hierarchy fills the performance gap between node-local storage and parallel file systems. With burst buffers, an application can quickly store checkpoints with increased reliability. In this work, our research group explored how burst buffers can improve efficiency compared to using only traditinal node-local storage. To fully exploit the bandwidth of burst buffers, we developped a user-level InfiniBand-based file system (IBIO). We also developped performance models for coordinated and uncoordinated checkpoint/restart strategies, and we applied those models to investigate the best checkpoint strategy using burst buffers on future large-scale systems, and validated effectiveness of use of burst buffers [Sato et al., 2013, Sato et al., 2014a].

2012-2013 **Energy-aware I/O optimization:** Both energy efficiency and system reliability are significant concerns towards exascale high-performance computing. In such large HPC systems, applications are required to conduct massive I/O operations to local storage devices (e.g. a NAND flash memory) for scalable checkpoint and restart. However, checkpoint/restart can use a large portion of runtime, and consumes enormous energy by non-I/O subsystems, such as CPU and memory. Thus, energy-aware optimization, including I/O operations to storage, is required for checkpoint/restart. In this work, our research group presented an energy-aware I/O optimization technique for NAND flash memory devices based on a Markov model for checkpoint/restart. The experiments showed our profile-based approach improves the energy consumption of write operations by 67.4% and read operations by 40.2% [Saito et al., 2013].

miscellaneous **Other filesystem works:** Also, I have been partly working on an ephemeral burst-buffer file system for CORAL systems [Wang et al., 2015, Wang et al., 2016].

Research area 2: HPC Tools

2016-present **NINJA (Noise injection agent tool):** Debugging intermittently occurring bugs within MPI applications is challenging, and message races, a condition in which two or more sends race to match with a receive, are one of the common root causes. Many debugging tools have been proposed to help programmers resolve them, but their runtime interference perturbs the timing such that subtle races often cannot be reproduced with debugging tools. Our research group proposed novel noise injection techniques to expose message races even under a tool 's control. We first formalized this race problem in the context of non-deterministic parallel applications and used this analysis to determine an effective noise-injection strategy to uncover them. Our evaluations on synthetic cases as well as a real-world bug in Hypre-2.10.1 showed that our noise injection technique significantly helps expose races [Sato et al., 2017c]. We codified these techniques in NINJA (Noise INJection Agent tool) that can expose these races without modification to the application. Then, we released NINJA [Sato et al., 2017a].

Future work: NINJA has a promising future to support programming models beyond MPI, which include tasking models. Because non-deterministic bugs often occurs as a function of specific timings of parallel interaction, we believe that our approach can be extended and generalized for other programming models. Indeed, that is a significant part of our future direction.

2014-present **ReMPI (MPI record-and-replay tool):** The ability to record and replay program execution helps significantly in debugging non-deterministic MPI applications by reproducing message-receive orders. However, the large amount of data that traditional record-and-reply techniques record precludes its practical applicability to massively parallel applications. To reduce record dize, our research group proposed a new compression algorithm, Clock Delta Compression (CDC), for scalable record and replay of non-deterministic MPI applications. CDC defines a reference order of message receives based on a totally ordered relation using Lamport clocks, and only records the differences between this reference logical-clock order and an observed order. Our evaluation showed that CDC significantly reduces the record data size. For example, when we applied CDC to Monte Carlo particle transport Benchmark (MCB), which represents common non-deterministic communication patterns, CDC reduced the record size by approximately two orders of magnitude compared to traditional techniques and incurs between 13.1% and 25.5% of runtime overhead [Sato et al., 2015]. We also developed MPI record-and-replay tool (ReMPI) which uses CDC and relesed the tooll [Sato et al., 2017b].

Future work: We also applyed in other production applications, and found record-and-replay techniques are promising approache for facilitating debug. MPI non-determinism is not an only source of non-determinism. We will extend our approach for other programming models such as OpenMP.

Research area 3: Resilience

2013-2015 **Lossy compression for checkpoit/restart:** The scale of high performance computing (HPC) systems is exponentially growing, potentially causing prohibitive shrinkage of mean time between failures (MTBF) while the overall increase in the I/O performance of parallel filesystems will be far behind the increase in computational performance. As such, there have been various attempts to decrease the checkpoint overhead, one of which is to employ compression techniques to the checkpoint files. While most of the existing techniques focus on lossless compression, their compression rates and thus effectiveness remain rather limited. Instead, our research group proposed a lossy compression technique based on wavelet transformation for checkpoints, and explored its impact to application results. Experimental application of our lossy compression technique to a production climate application, NICAM, showed that the overall checkpoint time including compression is reduced by 81%, while relative error remains fairly constant at approximately 1.2% on overall average of all variables of compressed physical quantities compared to original checkpoint without compression [Sasaki et al., 2015, Sasaki et al., 2014].

2012-2014 **FMI (Fault Tolerant Messaging Interface):** The Message Passing Interface (MPI) is the de-facto HPC programming paradigm, but it employs a fail-stop model. On failure, all processes in the MPI job are terminated, which incur extra overhead for bootstrapping connections. To address this problem, we presented the Fault Tolerant Messaging Interface (FMI), which enables extremely low-latency recovery. FMI accomplishes this using a survivable communication runtime coupled with fast, in-memory C/R, and dynamic node allocation. FMI provides message-passing semantics similar to MPI, but applications written using FMI can run through failures. Our tests showed that FMI incurs only a 28% overhead with a very high mean time between failures of 1 minute [Sato et al., 2014b].

2011-2012 **Design and modeling of asynchronous checkpointing:** In this work, our research group desigined an asynchronous checkpointing system and combined the benefits of asynchronous checkpointing and multi-level checkpointing. We also proposed the asynchronous and multi-level checkpoint/restart model. Our experiments show that our system can improve efficiency by 1.1 to 2.0 $\times$ on future exascale machines. Additionally, applications using our checkpointing system can achieve high efficiency even when using a PFS with lower bandwidth [Sato et al., 2012a, Sato et al., 2012b, Sato et al., 2012c, Sato et al., 2011].

miscellaneous **Others resilience works:** I partly worked on FTA (Fault Tolerance Assistant). FTA is a programming model that provides failure localization and transparent recovery of process failures in MPI applications [Fang et al., 2015]. I also partly worked on a statistical latent fault detection for unbalanced workloads, making it more practical in supercomputers and other large scale systems whose computational workload is not necessarily balanced [Gabel et al., 2015].

Research area 4: Cloud Computing

2007-2011 **VM migration for efficient I/O:** Federated storage resources in geographically distributed environments are becoming viable platforms for data-intensive cloud and grid applications. To improve I/O performance in such environments, our research group proposed a novel model-based I/O performance optimization algorithm for data-intensive applications running on a virtual cluster. This algorithm determines optimal virtual machine (VM) migration strategies, i.e., when and where a VM should be migrated while minimizing the expected value of file access time. We solve this problem as a shortest path problem of a weighted direct acyclic graph (DAG) where weighted vertices represent possible locations of a target VM and expected file access time in the location, and weighted edges represents a migration of a VM and the time. We construct the DAG from our markov model which represents the dependency of files. Our simulation-based studies showed that our proposed algorithm can achieve higher performance than simple techniques, such as ones that never migrate VMs: 38% or always migrate VMs onto the locations that hold target files: 47% [Sato et al., 2009, Sato et al., 2008].

Other partly-worked reseaches

2011-2012 **Efficient all-to-all communication for FFT:** Our research proposed efficent all-to-all communication strategies in order to minimize the overhead. our all-to-all communication optimize the scheduling of the data transfers by dynamically selecting InfiniBand rails, and effectively overlap intra- and inter-node communication in an aggressive fashion [Nukada et al., 2012].

2011 **Physis (A programming framework for stencil computations):** Our research group proposed a compiler-based programming framework that automatically translates user-written structured grid code into scalable parallel implementation code for GPU-equipped clusters [Maruyama et al., 2011].

Reference

[Fang et al., 2015] Fang, A., Laguna, I., Sato, K., Islam, T., and Mohror, K. (2015). Fault Tolerance Assistant (FTA): An Exception Handling Approach for MPI Programs (Hote topic). In *Workshop on Exascale MPI (ExaMPI15) at Supercomputing 2015 (SC15)*.

[Gabel et al., 2015] Gabel, M., Sato, K., Keren, D., Matsuoka, S., and Schuster, A. (2015). Latent Fault Detection With Unbalanced Workloads. In *Proceedings of the Workshops of the EDBT/ICDT 2015 Joint Conference (EDBT/ICDT), Brussels, Belgium, March 27th, 2015.*, pages 118–124.

[Maruyama et al., 2011] Maruyama, N., Sato, K., Nomura, T., and Matsuoka, S. (2011). Physis: An implicitly parallel programming model for stencil computations on large-scale GPU-accelerated supercomputers. In *2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, pages 1–12.

[Nukada et al., 2012] Nukada, A., Sato, K., and Matsuoka, S. (2012). Scalable multi-GPU 3-D FFT for TSUBAME 2.0 Supercomputer. In *High Performance Computing,*

Networking, Storage and Analysis (SC), 2012 International Conference for, pages 1–10.

[Saito et al., 2013] Saito, T., Sato, K., Sato, H., and Matsuoka, S. (2013). Energy-aware I/O Optimization for Checkpoint and Restart on a NAND Flash Memory System. In *Proceedings of the 3rd Workshop on Fault-tolerance for HPC at Extreme Scale*, FTXS '13, pages 41–48, New York, NY, USA. ACM.

[Sasaki et al., 2014] Sasaki, N., Sato, K., Endo, T., and Matsuoka, S. (2014). Exploration of Application-level Lossy Compression for Fast Checkpoint/Restart. In *HPC in Asia Workshop in conjunction with the International Supercomputing Conference (ISC '14)*.

[Sasaki et al., 2015] Sasaki, N., Sato, K., Endo, T., and Matsuoka, S. (2015). Exploration of Lossy Compression for Application-Level Checkpoint/Restart. In *2015 IEEE International Parallel and Distributed Processing Symposium*, pages 914–922.

[Sato et al., 2017a] Sato, K., Ahn, D., Lagnua, I., Lee, G., Schulz, M., and Chambreau, C. (2017a). NINJA: Noise Injection Agent Tool. https://github.com/PRUNERS/NINJA.

[Sato et al., 2017b] Sato, K., Ahn, D., Lagnua, I., Lee, G., Schulz, M., and Chambreau, C. (2017b). ReMPI: MPI Record-and-Replay Tool. https://github.com/PRUNERS/ReMPI.

[Sato et al., 2015] Sato, K., Ahn, D. H., Laguna, I., Lee, G. L., and Schulz, M. (2015). Clock Delta Compression for Scalable Order-replay of Non-deterministic Parallel Applications. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '15, pages 62:1–62:12, New York, NY, USA. ACM.

[Sato et al., 2017c] Sato, K., Ahn, D. H., Laguna, I., Lee, G. L., Schulz, M., and Chambreau, C. M. (2017c). Noise Injection Techniques to Expose Subtle and Unintended Message Races. In *Proceedings of the 22Nd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, PPoPP '17, pages 89–101, New York, NY, USA. ACM.

[Sato et al., 2012a] Sato, K., Maruyama, N., Mohror, K., Moody, A., Gamblin, T., de Supinski, B. R., and Matsuoka, S. (2012a). Design and modeling of a non-blocking checkpointing system. In *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*, pages 1–10.

[Sato et al., 2013] Sato, K., Matsuoka, S., Moody, A., Mohror, K., Gamblin, T., de Supinski, B. R., and Maruyama, N. (2013). Burst SSD Buffer: Checkpoint Strategy at Extreme Scale. In *IPSJ SIG Technical Reports 2013-HPC-141*.

[Sato et al., 2014a] Sato, K., Mohror, K., Moody, A., Gamblin, T., d. Supinski, B. R., Maruyama, N., and Matsuoka, S. (2014a). A User-Level InfiniBand-Based File System and Checkpoint Strategy for Burst Buffers. In *2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 21–30.

[Sato et al., 2012b] Sato, K., Moody, A., Mohror, K., Gamblin, T., d. Supinski, B. R., Maruyama, N., and Matsuoka, S. (2012b). Towards a Light-weight Non-blocking Checkpointing System. In *HPC in Asia Workshop in conjunction with the International Supercomputing Conference (ISC'12)*.

[Sato et al., 2014b] Sato, K., Moody, A., Mohror, K., Gamblin, T., d. Supinski, B. R., Maruyama, N., and Matsuoka, S. (2014b). FMI: Fault Tolerant Messaging Interface for Fast and Transparent Recovery. In *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, pages 1225–1234.

[Sato et al., 2011] Sato, K., Moody, A., Mohror, K., Gamblin, T., de Supinski, B. R., Maruyama, N., and Matsuoka, S. (2011). Towards an Asynchronous Checkpointing System. In *IPSJ SIG Technical Reports 2011-ARC-197 2011-HPC-132 (HOKKE-19)*.

[Sato et al., 2012c] Sato, K., Moody, A., Mohror, K., Gamblin, T., de Supinski, B. R., Maruyama, N., and Matsuoka, S. (2012c). Design and Modeling of an Asynchronous Checkpointing System. In *IPSJ SIG Technical Reports 2012-HPC-135 (SWoPP 2012)*.

[Sato et al., 2014c] Sato, K., Nukada, A., Maruyama, N., and Matsuoka, S. (2014c). I/O acceleration with GPU for I/O-bound Applications. In *GPU Technology Conference 2014*.

[Sato et al., 2008] Sato, K., Sato, H., and Matsuoka, S. (2008). Model-based optimization for data-intensive application on virtual cluster. In *2008 9th IEEE/ACM International Conference on Grid Computing*, pages 367–368.

[Sato et al., 2009] Sato, K., Sato, H., and Matsuoka, S. (2009). A Model-Based Algorithm for Optimizing I/O Intensive Applications in Clouds Using VM-Based Migration. In *2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 466–471.

[Wang et al., 2016] Wang, T., Mohror, K., Moody, A., Sato, K., and Yu, W. (2016). An Ephemeral Burst-buffer File System for Scientific Applications. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '16, pages 69:1–69:12, Piscataway, NJ, USA. IEEE Press.

[Wang et al., 2015] Wang, T., Mohror, K., Moody, A., Yu, W., and Sato, K. (2015). BurstFS: A Distributed Burst Buffer File System for Scientific Applications. In *The International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*.

[Xu et al., 2014] Xu, T., Sato, K., and Matsuoka, S. (2014). Towards Cloud Bursting for Extreme Scale Supercomputers. In *IPSJ SIG Technical Reports 2014-HPC-145*.

[Xu et al., 2015a] Xu, T., Sato, K., and Matsuoka, S. (2015a). Cloud-based Burst Buffers for I/O Acceleration. In *IPSJ SIG Technical Reports 2015-HPC-150*.

[Xu et al., 2015b] Xu, T., Sato, K., and Matsuoka, S. (2015b). Design and Modelling Cloud-based Burst Buffers. In *The International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*.

[Xu et al., 2015c] Xu, T., Sato, K., and Matsuoka, S. (2015c). Towards Cloud-based Burst Buffers for I/O Intensive Computing in Cloud. In *HPC in Asia Workshop in conjunction with the International Supercomputing Conference (ISC '15)*.

[Xu et al., 2016] Xu, T., Sato, K., and Matsuoka, S. (2016). CloudBB: Scalable I/O Accelerator for Shared Cloud Storage. In *2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)*, pages 509–518.

[Xu et al., 2017] Xu, T., Sato, K., and Matsuoka, S. (2017). HuronFS: Hierarchical, User-level and On-demand File System. https://github.com/EBD-CREST/HuronFS.