A Simulation-Based Analysis on the Configuration of the Burst Buffer

[Extended Abstract]

Tianqi Xu
Tokyo Institution of Technology
2 Chome-12-1 Ookayama, Meguro
Tokyo, Japan 152-8550
xu.t.aa@m.titech.ac.jp

Kento Sato
Lawrence Livermore National Laboratory
7000 East Ave
Livermore, CA, USA 94550
kento@llnl.gov

Satoshi Matsuoka
Tokyo Institution of Technology
2 Chome-12-1 Ookayama, Meguro
Tokyo, Japan 152-8550
matsu@is.titech.ac.jp

ACM Reference format:
Tianqi Xu, Kento Sato, and Satoshi Matsuoka. 2017. A Simulation-Based Analysis on the Configuration of the Burst Buffer. In *Proceedings of , , ,* 2 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Computational performance in HPC systems has dramatically increased during the recent years. However, the performance of parallel file systems (PFS) in the HPC centers have stagnated, resulting in relative compromising of scientific data-intensive applications. Burst buffer systems have been widely deployed in many supercomputer systems to fill such performance gap [1]. Burst buffer systems are designed to have larger bandwidth and lower latency but lower capacity, which is suitable for buffering intermediate I/O. Previous research and practice have proven that burst buffer systems are effective for alleviating the contentions on PFS and accelerating I/O performance [1].

However, while burst buffers offer a solution to improve performance of PFS, system users and HPC centers today struggle in understanding what burst buffer configurations are suitable for their systems and workloads. As with caches found in general-purpose processors, burst buffer parameters include capacity and bandwidth can be crucial to govern performance of these systems.

We investigate such challenges by estimating the I/O performance by our simulator for burst buffer systems based on I/O traces from real-world applications. Our contributions are: (i) we model burst buffers and implement a simulator to estimate I/O performance of applications; (ii) we show that burst buffer configurations have a significant impact on I/O performance of applications; (iii) we find that, even with burst buffer whose size is half of *access space*[1] (i.e., 0.5 x A_{space}), applications can still achieve comparable I/O performance to ones with A_{space}.

[1]We define *access space* (A_{space}) as a total size of files accessed by an application, i.e., required buffer size to buffer all of files accessed by an application.

,
© 2017 ACM. 978-x-xxxx-xxxx-x/YY/MM…$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

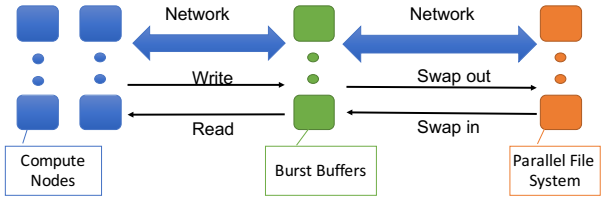

Figure 1: An Overview of Our Target Burst Buffer

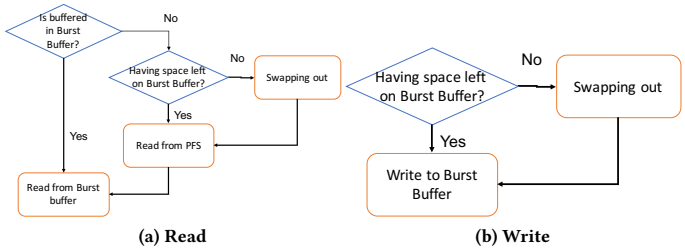

Figure 2: Read/Write Operations on Limited Burst Buffers

2 THE BURST BUFFER MODEL AND THE SIMULATOR

First, we explain our burst buffer model (Figure 1). Our burst buffer model consists of two-level storage: *burst buffer* (level-1) and *PFS* (level-2). Burst buffer is temporal space for buffering and caching files from applications running on *compute nodes*. Burst buffer nodes are independent nodes from compute nodes. PFS is used as persistent data store. When compute nodes need to access to files on the PFS, these I/O requests (i.e., read/write) are handled via burst buffers. If there is no available space in burst buffer, or when reading files from PFS for the first time, our model moves data between burst buffer and PFS (i.e., swap-in/out) based on a particular caching algorithm (LRU in the evaluation). Detailed state transition diagrams are shown in Figure 2. Our model also support *asynchronous write-back*, which means that dirty data is written back to the PFS while concurrently servicing other application I/O requests. Under asynchronous write-back enabled, swap-out overhead is not incurred if the data is already consistent with PFS.

We implement a simulator that estimates I/O performance of applications under a given burst buffer configuration. Our simulator

Table 1: Reedbush system @ University of Tokyo

Name	Reedbush
CPU	Intel Xeon E5-2695v4
Memory	256 GB
Network	InfiniBand EDR 4x (100 Gbps)
Burst Buffer	DDN IME 14K x6 (Capacity: 209 TB) (Latency: 20 usec, bandwidth: 436.2 GB/s)
PFS	Lustre Filesystem DDN SFA14KE x3 (Capacity: 5.04 PB) (Latency: 500 usec, bandwidth: 145.2 GB/s)

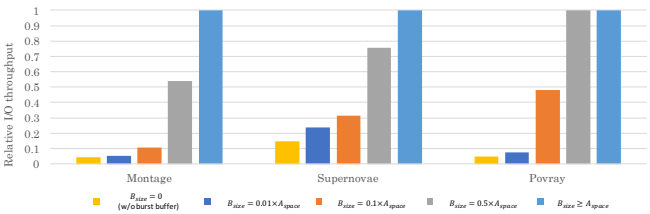

Figure 3: Relative I/O throughput to the ideal case ($B_{size} \geq A_{space}$) under different buffer size

Table 2: Applications details

Application	Input size (MB)	Access space (A_{space}) (MB)	Total I/O (MB)
Montage	1,200	7,500	27,000
Povray	14	25	760
Supernovae	6,600	23,000	55,000

uses I/O traces (which include read/write details including sizes and offsets, with time stamps) from an application as input, simulates the I/O behavior based on our burst buffer model, and then outputs I/O performance numbers (e.g., time to complete all I/O requests, overall average I/O throughput achieved and etc.).

3 SIMULATION WITH REAL APPLICATIONS

To estimate I/O performance of applications under different burst buffer configurations, we first collect I/O traces from three selected workflow applications by using a FUSE-based I/O tracer (MUSE [3]). The I/O traces are collected in each compute node, and we aggregate these I/O traces into a single I/O trace file based on time stamps. Table 2 shows the access space (A_{space}) and the I/O volumes. In our evaluation, performance parameters of a burst buffer and PFS (e.g., bandwidth and latency) are based on a real-world production system: Reedbush [2] at the University of Tokyo (Table 1).

We evaluate I/O performance of applications by overall average I/O throughput (i.e., total I/O size divided by time to complete all I/O requests) under different burst buffer sizes (B_{size}). Figure 3 shows the relative I/O throughput to an ideal case where burst buffer has enough buffer space to cache all of the files (i.e., $B_{size} \geq A_{space}$). We found that varying the buffer size impacted the performance of the I/O throughput. Moreover, one of our key findings is that executions with burst buffers whose size is even 50 % of A_{space} can still achieve comparable I/O throughput to the ideal case for Supernovae and Povray. However, even with 1 % of A_{space}, I/O thoughput of Povray can still keep up half of the ideal case. In workflow applications, intermediate files written by a task is immediately read by the next multiple tasks (i.e., high temporal/spatial I/O locality). Such I/O workload allows smaller buffer size to keep up the I/O performance. Although the temporal and spatial I/O locality of applications differ, our simulator provides the means to determine if and by how much an application can benefit from using a certain burst buffer configuration.

4 CONCLUSION

To investigate the impacts on I/O performance of scientific applications under different burst buffer configurations, we have implemented an I/O simulator. In the evaluation of three selected workflow applications, we have found that applications can still keep up high I/O throughput even with 50 % of A_{space} if access data is properly swapped in and out based on our burst buffer model (LRU in the evaluation). Although current burst buffer systems provide APIs for moving files between burst buffers and PFS, potential of the APIs has not been comprehensively studied. Our finding will also motivate to implement a caching algorithm with the burst buffer APIs.

5 ACKNOWLEDGEMENT

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. (LLNL-CONF-735954). This work was also supported by JST CREST Grant Number JPMJCR1303, and performed under the auspices of Real-World Big-Data Computation Open Innovation Laboratory, Japan.

REFERENCES

[1] N. Liu, J. Cope, P. Carns, C. Carothers, R. Ross, G. Grider, A. Crume, and C. Maltzahn. On the role of burst buffers in leadership-class storage systems. In *Mass Storage Systems and Technologies (MSST), 2012 IEEE 28th Symposium on*, pages 1–11, April 2012.
[2] T. U. of Tokyo. Reedbush. http://www.cc.u-tokyo.ac.jp/system/reedbush/, 2016.
[3] K. Sato. MUSE. https://github.com/kento/MUSE, 2017.