Linux File System Analysis over NUMA Based Servers

Yong Seob Lee* Sungyong Park*

*Department of Computer Science and Engineering
Sogang University, Seoul, Republic of Korea
E-mail: {misteryos, parksy}@sogang.ac.kr

Abstract

This paper evaluates a Linux file system (e.g., Ext4) both on Ramdisk and SSD, and briefly reports the factors that influence the performance of the file system as well as the applications running over the file system. Since the benchmarking is conducted over a NUMA based server by increasing the number of cores, the relationship between the performance of the file system and the inherent NUMA memory latency is also discussed. The experimental results shows that a large amount of overheads in a file system is caused by the block I/O layer and the performance is largely dominated not by the file system but by the applications.

Keywords: Linux, File system, NUMA

1. Introduction

The rapid advances in semiconductor technology have created an opportunity so that the number of cores both in a single chip (multicore) and across the bus (manycore) keeps increasing dramatically [?]. The recent introduction of Intel's MIC technology (Xeon Phi) [2] has also added a lot of complexity to the software running over these processors. In order to fully utilize this environment, the software (from OS and system software to application software) running over the multicore/manycore processors should be optimized to get the performance benefits provided by this powerful hardware [3] [4].

The main focus of this paper is to evaluate the performance of Ext4 file system over NUMA based servers and see the impacts of NUMA architecture and block I/O layer on the throughput of the whole system. Considering that a large number of servers is currently based on the NUMA architecture with multiple cores (generally 8-10 cores per CPU), it is interesting to observe the impacts of NUMA memory latency and block I/O layer on the performance and scalability of the server.

2. Benchmark and Analysis

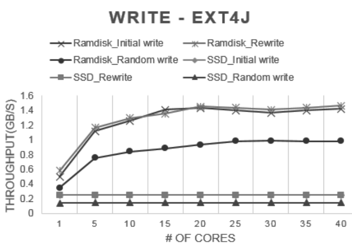

Figure 1. Iozone results.

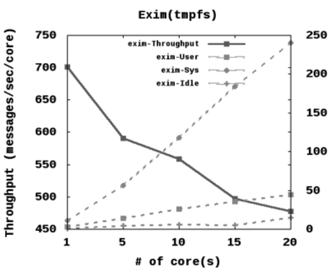

Figure 2. Mosbench result.

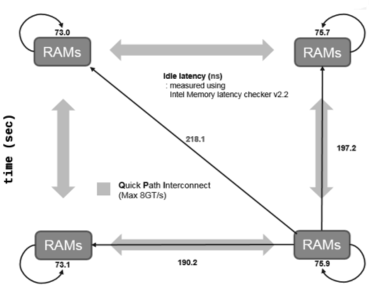

Figure3 NUMA latency

Figure 1 shows the write performance of Ext4 file system running both on Ramdisk and SSD. As we can see from the figure, the throughput of Ext4 over Ramdisk increases as we increase the number of cores, while the throughput of Ext4 over SSD is saturated around 0.1-0.2 GBps. Considering that the same file system (Ext4) runs both on Ramdisk and SSD, the block I/O layer when the real device such as SSD is involved contributes a lot to the total overheads of the file system. This means that the performance of applications running over this environment can be saturated if their workloads are relatively I/O bound.

Figure 2 dipicts the performance of Exim application (Email server) [5] over a memory-based file system, *tmpfs*. It is worthy to note that the throughput degrades as we increase the number of cores although we use the *tmpfs* file system, which is very light weighted and doesn't include the block I/O path when it performs read and write operations. Since it is shown in Figure 1 that Ext4 is scalable over Ramdisk, we can assume that the performance degrade shown in Figure 2 is mainly caused by the behaviors from the applications such as lock contention or memory access. This also means that the optimizations of the applications are sometimes more important than using a low overhead file system.

Figure 3 reports the NUMA memory latency values of the server used for the experiments. As shown in the figure, the access latency of local memory is 2-3 times shorter than that of remote memory. Therefore, it should be noted that the memory access patterns or the locations of memory to be accessed paly an important role to optimize the performance of the file system or the applications. Current Linux version contains an automatic NUMA CPU scheduling and memory migration scheme called AutoNUMA [6] to minimize the impact of NUMA memory latency.

3. Conclusions

In this paper, we have evaluated the performance of Ext4 Linux file system both on Ramdisk and SSD, and showed that a large amount of overheads in a file system is caused by the block I/O layer and the performance is largely dominated not by the file system but by the applications. This means that the optimizations of the applications are sometimes more important than using a low overhead file system.

Acknowledgments

This work was supported by the ICT R&D program of MSIP/IITP. [B0101-15-0644, Research on High Performance and Scalable Manycore Operating System]

References

[1] Geer, David. "Chip makers turn to multicore processors." Computer 38.5 (2005): 11-13.
[2] Jeffers, James, and James Reinders. Intel Xeon Phi coprocessor high-performance programming. Newnes, 2013.
[3] Manferdelli, John L., Naga K. Govindaraju, and Chris Crall. "Challenges and opportunities in many-core computing." Proceedings of the IEEE 96.5 (2008): 808-815.
[4] Silas Boyd-Wickizer, Austin T. Clements, Yandong Mao, Aleksey Pesterev, M. Frans Kaashoek, Robert Morris, and Nickolai Zeldovich. 2010. An Analysis of Linux Scalability to Many Cores, In the Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI '10), Vancouver, Canada, October 2010.
[5] MOSBENCH, http://pdos.csail.mit.edu/mosbench/
[6] Lameter, Christoph. "Local and remote memory: Memory in a Linux/NUMA system." Linux Symposium.