Threading Opportunities in High-Performance Flash-Memory Storage

Craig Ulmer
cdulmer@sandia.gov
Sandia National Laboratories
Livermore, California

Maya Gokhale
maya@llnl.gov
Lawrence Livermore National Laboratory
Livermore, California

Storage Intensive Supercomputing

The storage-intensive supercomputing (SISC) project [1] at LLNL is a research effort that is currently investigating hardware architectures for improving the performance of large, data-intensive applications. In order to achieve next-generation performance goals for these applications, it is necessary to consider system architectures that employ both computational accelerators (e.g., FPGAs or GPUs) and high-speed, mass-storage devices. In this work we focus on the latter by examining the performance characteristics of emerging flash-memory storage devices in the context of multicore environments.

High-Performance Flash-Memory Storage

Consumer demand for portable music devices with mass storage has driven down the price of flash memory to a point where flash-memory-based storage devices are nearly cost-competitive with enterprise-class hard drives. While a number of vendors have produced flash-memory storage devices that are compatible with existing hard-drive I/O standards, few have demonstrated products that deliver significant performance gains over enterprise disks. Mediocre performance in these initial products can be attributed to a number of factors, including SATA's low data-transfer rates, an inability for flash drives to support a large number of simultaneous transactions, and the use of narrow flash-memory geometries that limit the amount of internal bandwidth available in a drive.

In contrast, Fusion-io's ioDrive[2] is an emerging flash-memory storage device that is optimized for performance instead of integration with existing I/O facilities. The ioDrive's hardware has three characteristics that set it apart from other devices. First, it is a PCIe x4 card that can support high-bandwidth data transfers with the host. Second, the card employs a large number of flash chips that are arranged to exploit parallelism both horizontally (i.e., bus width) and vertically (i.e, die stacks). Finally, the card implements a high-throughput transaction manager in hardware that allows multiple transactions to be processed concurrently. These architecture features enable a single card to deliver up to 700 MB/s of bandwidth and 100k I/O operations per second (IOPS).

Multithreaded I/O Transactions

During the early stages of our investigation into the low-level performance characteristics of the ioDrive, we observed an interesting effect: in several applications, increasing the number of I/O-performing threads increased the overall data-transfer performance of the ioDrive. This effect is opposite of what we have come to expect from traditional hard drives, where concurrent transactions generally degrade performance because they result in high-overhead seek. As such we devised a number of tests to quantify the low-level performance characteristics of the ioDrive and provide examples of how threaded applications on a multicore system can exploit these characteristics.

Multithreaded Microbenchmarks

As a means of observing the impact of multithreaded performance on storage devices, we constructed four multithreaded microbenchmark applications that perform operations commonly found in data-intensive applications. While the applications all perform computations involving vectors of floating-point numbers, we selected operations that are I/O bound instead of compute bound in order to stress the storage subsystem. The microbenchmarks are threaded at a coarse granularity and assume out-of-core operation, where datasets are much larger than the capacity of main memory. While a reasonable amount of effort has been made to maximize performance, we have not taken heroic measures to optimize the applications to a particular system architecture (e.g., application-level caching). The intent is to give an idea of the performance that can be obtained using built-in features of the hardware (e.g., multiple cores) and operating system (e.g., OS file caches). The microbenchmarks are block transfer, k-nearest neighbors (kNN), external sort, and binary search.

The performance numbers reported are for a single server that employs two quad-core CPUs (2.33MHz Intel E5345s), 2GB of memory (PC2-5300f), one 80GB Fusion-io ioDrive, and two SATA drives arranged in a software-based RAID0. The system utilizes the Linux 2.6.23 OS found in Fedora 8. In all tests, we utilized 8GB input files (64 million vectors of 32 single-precision floating-point values) in order to negate caching effects.

Block Transfer

The block transfer microbenchmark measures raw data-transfer performance characteristics, similar to other benchmark programs such as IOzone [3]. The block transfer program invokes multiple threads that issue either reads or writes to sequential or random locations within one or more file. Transfers are intentionally misaligned in order to minimize overlap and reduce caching effects. Performance is reported in terms of the aggregate amount of data transferred per second. For sequential reads and writes, the ioDrive provided 687 MB/s and 662 MB/s respectively, compared to the SATA RAID's 118 MB/s and 99 MB/s.

However, the true benefit of flash memory became apparent in the random I/O tests. As illustrated in Figure 1, the ioDrive provides at worst a 17x improvement over the SATA RAID for random I/O. As this data indicates, the ioDrive can provide better performance when it has more requests in-flight at the same time. The dip in performance

at 256KB can be attributed to the block size that the ioDrive uses internally.

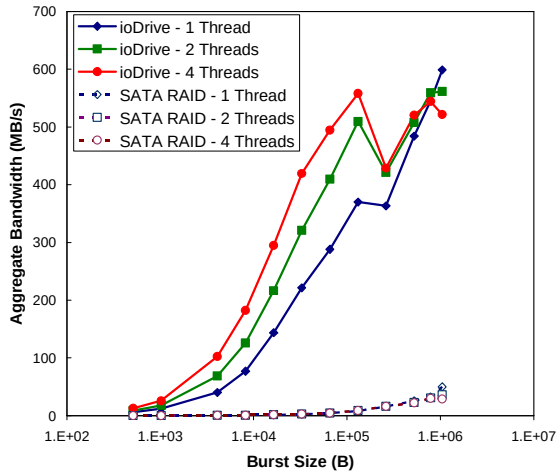

Figure 1: Random Read Performance

k-Nearest Neighbors (kNN)

The second microbenchmark implements the k-Nearest Neighbors (kNN) algorithm for classifying input vectors based on their similarity to labeled, training vectors. Each thread in the program reads its own section of the training data and locates the k training vectors that have the shortest Euclidean distance to one or more input vectors.

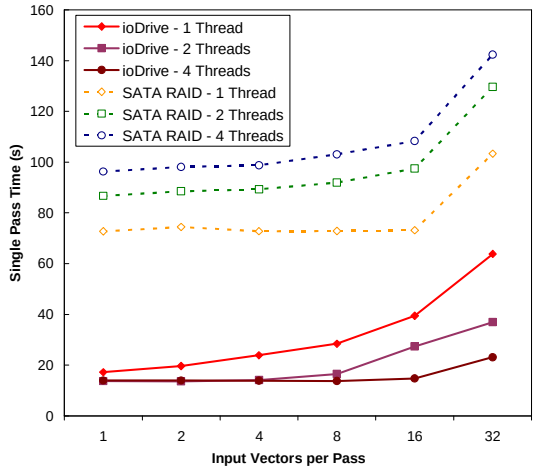

Figure 2: Time Required for Single kNN Pass

The amount of time required to make a full pass of the training data for a given number of input vectors is presented in Figure 2. While the SATA RAID's performance degraded with multiple threads, the ioDrive's improved. Additionally, the upward trend for the ioDrive illustrates the transition from an I/O bound problem to CPU bound.

External Sort

The external sort microbenchmark converts an unsorted file of vectors into a sorted file of vectors. Due to the large size of the input file, this implementation must process data out-of-core in two phases. First, multiple threads read in different sections of the input file, quicksort the individual sections, and then write out the results to intermediate files. Second, a single thread merges all of the intermediate files into an output file in a streaming manner. For fairness the tests use a fixed buffer size of 512 MB that is divided evenly among the threads. As the results indicate in Figure 3, RAID performance degrades as the number of threads increases. In contrast, performance is maximized in the ioDrive when four threads are utilized.

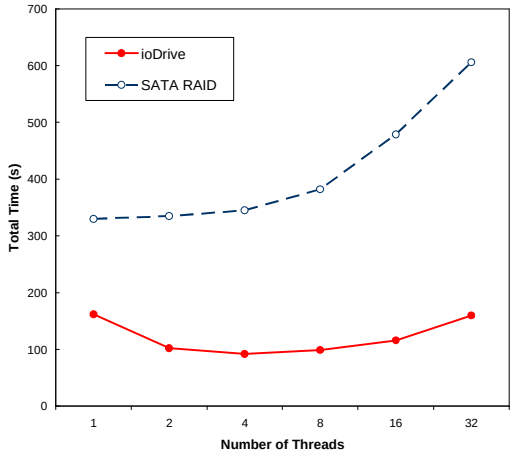

Figure 3: External Sort Time

Binary Search

The final microbenchmark performs a binary search on a sorted file to determine whether it contains one or more input vectors. This search is performed directly on the file and requires log(n) reads per input to different locations in the file. In order to reduce the number of file reads, the program can be configured to build an index of the file in main memory at start time. The average amount of time required to process an input vector when optimal indexing is available is presented in Figure 4. Similar to the previous tests, increasing the number of threads improved performance. The SATA RAID performance numbers are omitted from this figure as they were approximately 10 ms (i.e., 100x slower than the ioDrive) for all thread sizes.

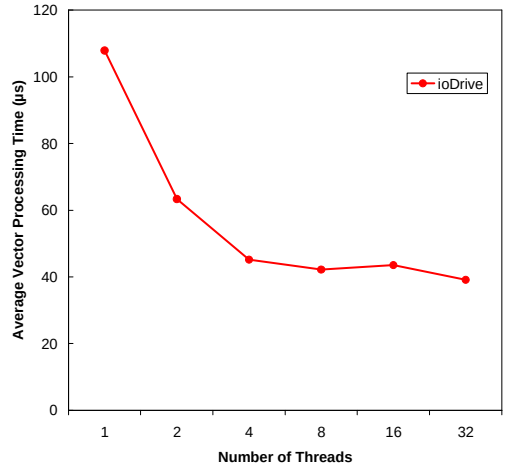

Figure 4: Binary Search Peformance

References

[1] M. Gokhale, J. Cohen, A. Yoo, M. Miller, A. Jacob, C. Ulmer, and R. Pearce, "Hardware Technologies for High-Performance Data-Intensive Computing," in IEEE Computer, vol. 41, No. 4, April 2008

[2] Fusion-io website: http://www.fusionio.com

[3] Iozone Filesystem Benchmark: http://www.iozone.org