Overlapped HPC Checkpointing with Hardware Assist

Christopher Mitchell, University of Central Florida, mitchell@cs.ucf.edu, Student
Dr. Jun Wang, University of Central Florida, jwang@eecs.ucf.edu
James Nunez and Andrew Nelson, Los Alamos National Laboratory, {jnunez, andy.nelson}@lanl.gov

High Performance Computing (HPC) systems of today and the coming future are grappling with the ever increasing problem of failure recovery due in part to their ever growing complexity. Increased failure rates result in increased time performing system recovery and more time performing failure mitigation processes such as checkpointing; time which is not being utilized towards the system's designated mission. According to Gibson and Schroeder [1], "*in the case where the number of cores per chip doubles every 30 months, the utilization drops to zero by 2013, meaning the system would spend 100% of its time writing checkpoints or recovering lost work, a situation that is clearly unacceptable*". In this context, checkpointing, while necessary, is an I/O operation that needs it's time to completion minimized as much as possible while still providing its intended ability to protect the running application. There are two main methods for reducing the time it takes to checkpoint an application: save less data or increase the speed at which the checkpoint is written to persistent storage. This research focuses on the latter as achieving smaller checkpoints is highly application dependent with varied levels of success per application.

Today's state-of-the-art HPC systems typically rely on a centralized parallel file system running atop a massive array of hard drives. While effective, in transferring these large files which constitute a checkpoint, the throughput of these systems is bound primarily by the number of disk spindles present and the speed of the individual disks in terms of seek time. Adding additional disk spindles could theoretically allow this system to scale enough to keep pace with the needed throughput demanded by the applications. However, it would be accompanied by higher cost, higher failure rates, and increased heat generation. Therefore, a more efficient method would be to introduce a significantly faster, non-volatile storage media between the application and the parallel file system's arrays with the sole purpose of performing a high-speed buffer of the checkpoint. Since this buffer would be non-volatile, it can assume responsibility for the checkpoint data from the application and leisurely write the data off to the parallel file system after the application has resumed computation. Candidates for this buffer include NVRAM, battery-backed DRAM, or high-speed SLC NAND flash memory.

Current progress is being made to prototype this proposed system within a cluster environment and using real HPC scale scientific applications (such as Flash I/O and selected I/O kernels for applications run at LANL) to test effectiveness. This system, once operational will provide an easy to implement API (through MPI-IO) for application developers to harness this faster checkpointing method while leveraging a backend that provides safe transport of data from application to final storage location. Finally, this implementation has the benefit of improving upon existing checkpoint enhancement techniques such as Zest [2] by always having the data available for read, no matter the current location of the data (in buffer or on the parallel file system's arrays) as well as achieving higher throughput by building off of next generation storage technology rather than working around the mechanical limitations of modern hard drives. Future explorations will expand the solution, beyond checkpoints, to improve the performance of all write operations (with no limitations on access type such as random, sequential, strided, and etcetera) to the parallel file system.

[1] Schroeder, B., Gibson, G.A., "Understanding failure in petascale computers." SciDAC 2007., http://www.cs.cmu.edu/~garth/papers/jpconf7_78_012022.pdf
[2] Nowoczynski, P., et al., "Zest: Checkpoint Storage System for Large Supercomputers", http://www.pdsi-scidac.org/events/PDSW08/resources/papers/Nowoczynski_Zest_paper_PDSW08.pdf