Resilient Operating Systems Software for Exascale Systems

Scott Levy1, Kurt Ferreira2, Patrick G. Bridges1, Mark Hoemmen2, Michael Heroux2, Kevin Pedretti2, and Ron Brightwell2
1Scalable Systems Lab, Department of Computer Science, University of New Mexico, Albuquerque, NM, {levy,bridges}@cs.unm.edu
2Sandia National Laboratories, Albuquerque, NM, {kbferre,mhoemme,maherou,ktpedre,rbbrigh}@sandia.gov

Introduction

Next-generation HPC operating systems and applications must support resilience both for applications and the system software itself. This is essential because experts are predicting that failure rates may go from the current state of a handful a day to multiple failures an hour. Specifically, the OS must cooperate with scientific applications to enable them to gracefully recover from failures, transparently mask failures that the application cannot address, and recover from failures in the operating system itself. In the research described below, we seek to address each of these important issues.

Cooperative OS/DRAM Fault Recovery

We have begun exploring the fault recovery interface between applications and the OS by examining how to handle uncorrectable DRAM errors. This is a common failure mode in current systems that hardware is already capable of detecting. This makes them an ideal starting place for examining how the OS can provide useful failure information to applications and how applications can utilize OS-provided information to recover from hardware failures.

At the application level, we have been exploring a partial transient DRAM failure model. In this approach, the application designates which portions of its memory can tolerate failure and which portions must be completely reliable. The application’s numerical algorithms are then designed to assume that the portions of its memory that can fail do so only temporarily - failed memory later (through unspecified means) returns to its original value.

The OS and application collaborate to provide this partial transient failure semantics. In particular, the application notifies the OS of memory regions in which it can tolerate failure and periodically checkpoints this memory locally. The OS notifies the application of failures in these regions of memory, allowing the it to schedule recovery of corrupted data in these regions when appropriate. The OS is responsible for handling failures in portions of memory in which the application cannot, either by masking these failures through OS-level resilience techniques or terminating the application if OS-level recovery is impossible.

To evaluate the power of this approach, we developed FT-GMRES, a fault-tolerant version of the GMRES sparse linear solver. Figure 1 compares the convergence of FT-GMRES on a single node compared to two different non-fault tolerant versions of GMRES. Even in the presence of the artificially high failure rates used in this test, FT-GMRES converges towards the correct answer that the other algorithms cannot. We are currently conducting tests of these techniques at larger scales and a wide range of failure rates.

Similarity-based OS Failure Recovery

Even in cases where the application can handle many failures, the OS will still be responsible for masking out failures that the application cannot handle. One novel technique we are examining is to mask such failures is leveraging similarity between memory pages in the system. Recent work has demonstrated that HPC applications frequently have large amounts of memory that contain similar data, with a number of applications having 25% or more of their memory similar to other application memory. This similarity can potentially be used by the operating system to recover failed regions of memory from other similar regions of memory. In particular, the OS can transparently maintain differences between similar areas of memory, for example using hash-based techniques similar to incremental checkpointing. Then, when a failure occurs in memory for which a similar or page exists, the OS can simply recreate the failed memory using the similar page and the stored difference.

Fault-tolerant OS Services

Finally, we are also examining techniques to harden the OS itself to hardware failures, again focusing initially on DRAM failures. An OS resilient to soft errors in memory is key to the scalability of exascale systems for a number of reasons. First, current operating systems are unable to recover from the vast majority of failures. Second, though the typical operating system only occupies a small portion of a system’s total physical memory footprint, recent studies show substantially more errors in this region than the
remainder of a system’s memory. Lastly, future HPC system software will need to continue running in the presence of memory failures if current application-based, forward error recovery mechanisms are to be successful.

We are currently working on augmenting the Kitten lightweight kernel to make it resilient to DRAM failures. Kitten is a special-purpose, limited-functionality OS developed at Sandia National Laboratories that is designed for use on the compute nodes of massively parallel supercomputers. Kitten is an ideal target for this research; its focus on deterministic runtime behavior and simple data structures allow us to more easily reconstruct lost system state in many cases. As an example, consider errors in page-table memory. Like many OSs, errors in Kitten’s page table memory are fatal, either killing the affected application or the entire node. However, Kitten’s deterministic mapping of virtual to physical addresses would make it straightforward to determine corruption had occurred and simple to recreate the corrupted page table memory contents from the base physical address and length information stored in the address space region object. This could be difficult in a general purpose OS like Linux due to its demand paging scheme, where unpredictable physical addresses are assigned to virtual addresses at runtime.

Acknowledgements
This research has been funded in part by a contract and faculty sabbatical appointment from Sandia National Laboratories. Sandia is a multiprogram laboratory operated by Sandia National Company, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Publications