Graduate and Professional Student Research Conference
presentation abstract

Application-Level Fault Tolerance in an Extreme-Scale Parallel Sparse Matrix Eigensolver

Nathan T. Weeks, Pieter Maris, Glenn R. Luecke, James P. Vary

Supercomputers are increasing in size to allow computational scientists to tackle larger scientific problems. Unfortunately, more components means lower mean time to failure across the system. Therefore, science applications that use current and future High-Performance Computing (HPC) platforms must be able to cope with system faults.

The current state-of-the-art for failure recovery in HPC applications is to periodically save important application state to stable storage (checkpoint), and restart the entire application from the last good checkpoint in the event of a failure. While checkpoint/restart can be implemented for a wide class of HPC applications, it suffers drawbacks. Checkpointed application state must not change during the checkpoint operation, possibly adding synchronization points and slowing down application execution. There must be enough (reliable) storage space to receive the checkpoint---which may be impractical if the application uses most of the system memory. Finally, input/output (I/O) performance in the storage system may not be sufficient to avoid creating an additional bottleneck to the application.

Until recently, there was little alternative to checkpoint/restart for fault tolerance in HPC applications. The de facto standard communication library on HPC systems, the Message Passing Interface (MPI), lacked the ability to recover from the failure of even a single MPI process. However, a proposed fault tolerance extension to MPI, User-Level Failure Mitigation (ULFM), shows promise for allowing an application to recover from MPI process crash failures and avoid an entire restart.

Few attempts have been made to validate the proposed ULFM fault tolerance model on real production-scale HPC applications. We describe a case study illustrating how this model can be applied to a large-scale nuclear physics application, Many-Fermion Dynamics (MFDn). Specifically, we show how its communication patterns replicate crucial state the sparse matrix vector multiplication code, which can be used for recovery in the event of an MPI process failure. We also describe practical challenges that remain for this application, lessons that can be applicable to other HPC applications that wish to leverage such a model for efficient fault tolerance at scale.