Resilience Assessment and Enhancement

Pedro Diniz, Bob Lucas
USC-ISI

George Bosilca, Thomas Herault
University of Tennessee

Chunhua Liao, Bronis R. de Supinski, Todd Gamblin,
Kathryn Mohror, Dan Quinlan
LLNL

Ganesh Gopalakrishnan
University of Utah

We are addressing the problems in software resilience with a holistic multifaceted approach that spans across software levels. One approach emphasizes tracking expected control flows or data invariants, and is aimed at detecting silent data corruption. Another explores language extensions and compiler technology to convey to compilers and run-time system resilience properties of code sections and algorithms. Additionally, we are investigating specific algorithmic properties of applications to develop fault tolerant extensions to dense and sparse methods. At the highest levels, we detect silent-data-corruptions by replicating and comparing values across MPI processes and improve on the state of the art for checkpoint/restart with innovations in files system and checkpoint compression.

**Abstract**

**FUSED Framework**

- Automatically synthesizes and inserts detectors
- Likely invariants are used for soft errors detection
- Uses profilers to generate likely invariants
- Our approach derives likely invariants using predicate transitions
- FUSED is evaluated using SuperLU Library
- Up to 90% of soft errors are detected
- Detectors only inserted into top-level LU factorization routine
- Average execution overhead of 15.7% due to the detectors

**Language Extensions & Compiler Technology for Resilience**

- Annotations that allow user to express fault-tolerant requirements and expectations: when and where errors matter and what to do about them.
- HROSE source-level resilience-oriented and user-guided transformations for array-based pointers and graph-based computations

**Silent Error Detection**

- We have seen unexpected behavior in jobs at scale on the LLNL Sequoia machine.
- Certain high-performance LINPACK runs have high overhead.
- Currently have no way to detect silent memory corruption.
- Conducting a detailed characterization of memory error rate of BG/Q, Cray
- Developing a tool, Dragnet, that finds memory errors through MPI replication
- Replicate MPI processes on-node, do shared-memory comparison of arrays
- Can convert any MPI program into a silent error detector

**Scalable Checkpoint/Restart with SCR**

- The Scalable Checkpoint/Restart Library (SCR) caches checkpoints on compute nodes.
- SCR caches checkpoints 20x faster when using CRUSE than when writing to RAM disk.
- mfiEngine uses semantic information in checkpoint file to increase compression ratios.

**Algorithmic-Based Approaches**

- We demonstrate that is possible to protect scientific applications from many sources of errors combining three different resilience techniques:
  - Algorithmic error checkers
  - Replicas of key data structures
  - Checkpoint restart mechanisms
- Significant improvements can be achieved in terms of reduction of performance slowdown and output ratio.