Oct 27, 2014

Resiliency Work in CLAMR

Bob Robey
XCP-2 Eulerian Codes Group

Nathan DeBardeleben
Qiang Guan
Ultrascale Systems Research Center
HPC-5
Los Alamos National Laboratory

Brian Atkinson
Clemson University
William Jones
Coastal Carolina University

Ultrascale Systems Research Center
Los Alamos National Laboratory

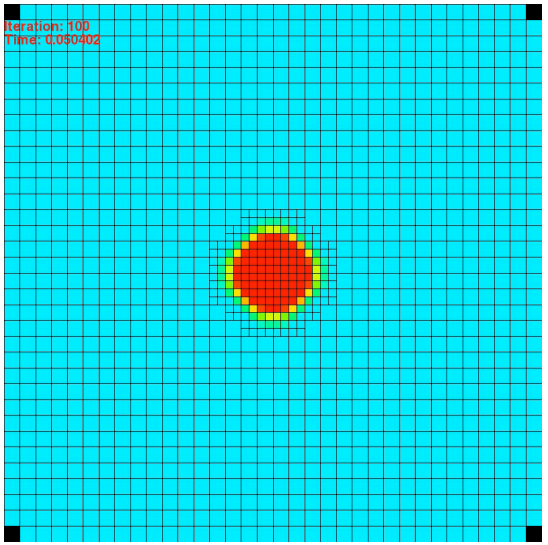
- Bringing Science to the analysis of system failures
  - Mining system logs for failure patterns – DRAM failures
  - Fault injection Studies – F-SEFI fault injection Framework

CLAMR – Cell-based Adaptive Mesh Refinement Hydrodynamics Mini-App

- Reflects the cell-by-cell refinement used in many of LANL codes
- Uses the Shallow Water equations for simplicity
- Open Source at http://github.com/losalamos/CLAMR
- CLAMR runs on GPUs, MICs, multicore, and multi-node with MPI
- Has real-time visualization to accelerate development

Circular Dam Break Simulation

- Cylindrical pulse is created at center of mesh
- Uses conservation of mass and momentum and explicit time-stepping to calculate the wave propagation
  - Challenging problem
    - Sharp rise at dam break
    - Need to maintain spherical symmetry
    - Near zero (or dry) condition at center of problem caused by waves moving outward



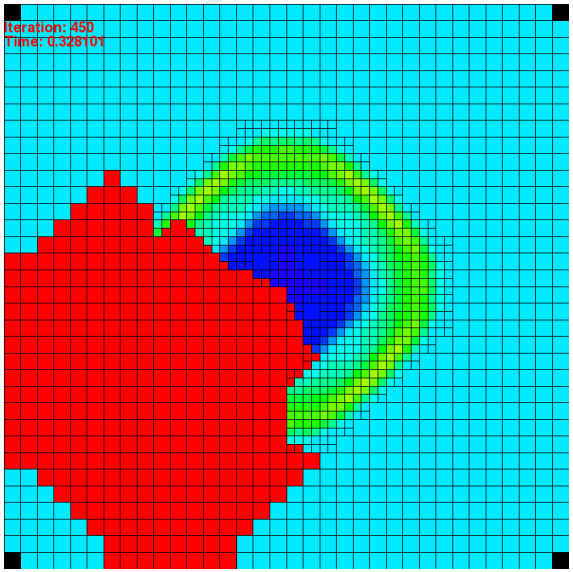
Fault Injection Study

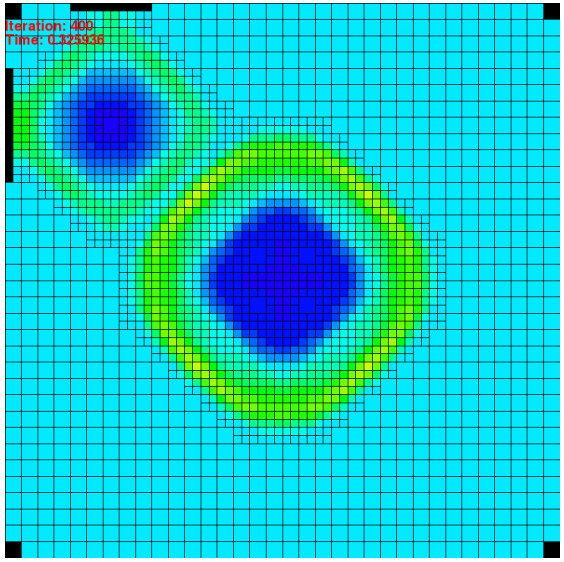
- Target core CLAMR function – calc_finite_difference
- Insert fault randomly using FADD instruction to simulate memory error
- Possible outcomes:
  - Benign – does not manifest in any changes to simulation
  - Crash – simulation becomes unstable, often resulting in negative masses, NaNs and the system aborts
  - Silent Data Corruption (SDC) – most dangerous outcome

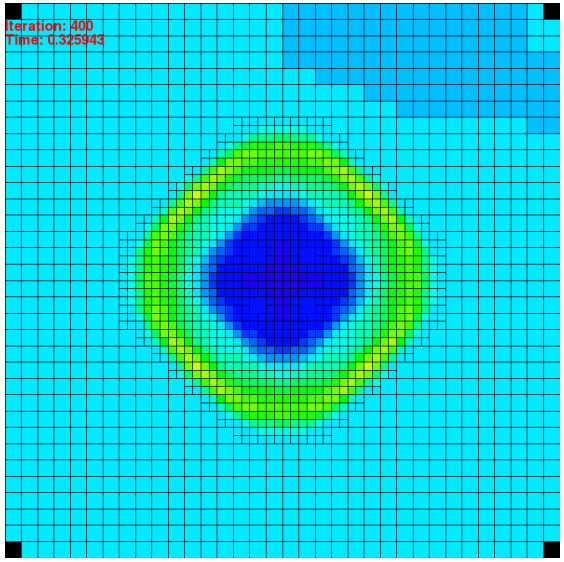
The Fault Injection Experiment

- Initial experiment -- 500 fault injections
  - more recent campaigns have been > 5,000
- 32x32 grid
  - Extremely small grid but recent studies have been on larger grids
  - Larger grid, while improving accuracy, does not impact fault tolerance
- Random iteration
- Random bit pattern
- Single bit flip

Examples of Failure Modes





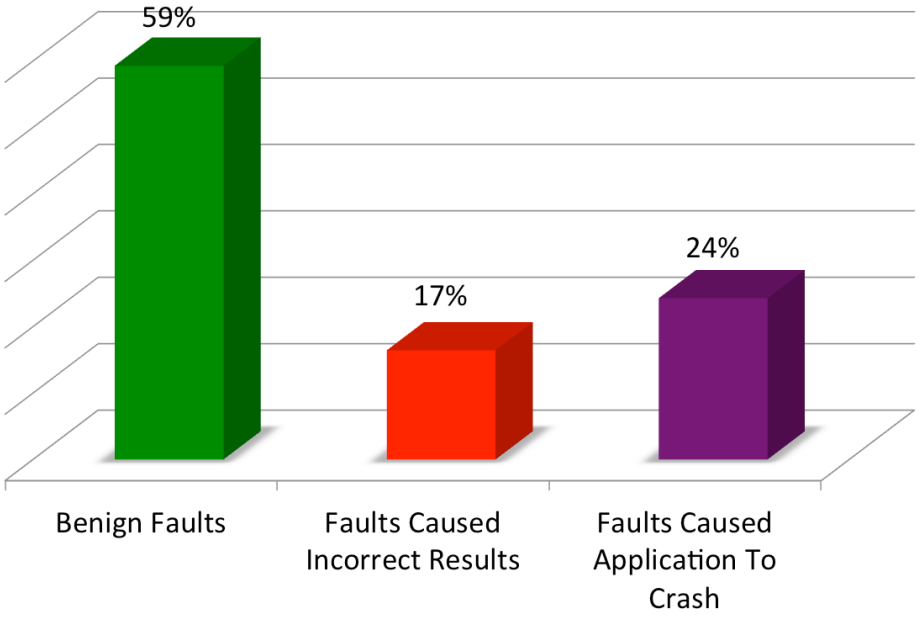


Crash

Two Different
Silent Data Corruptions

Fault Injection Experiments – Results

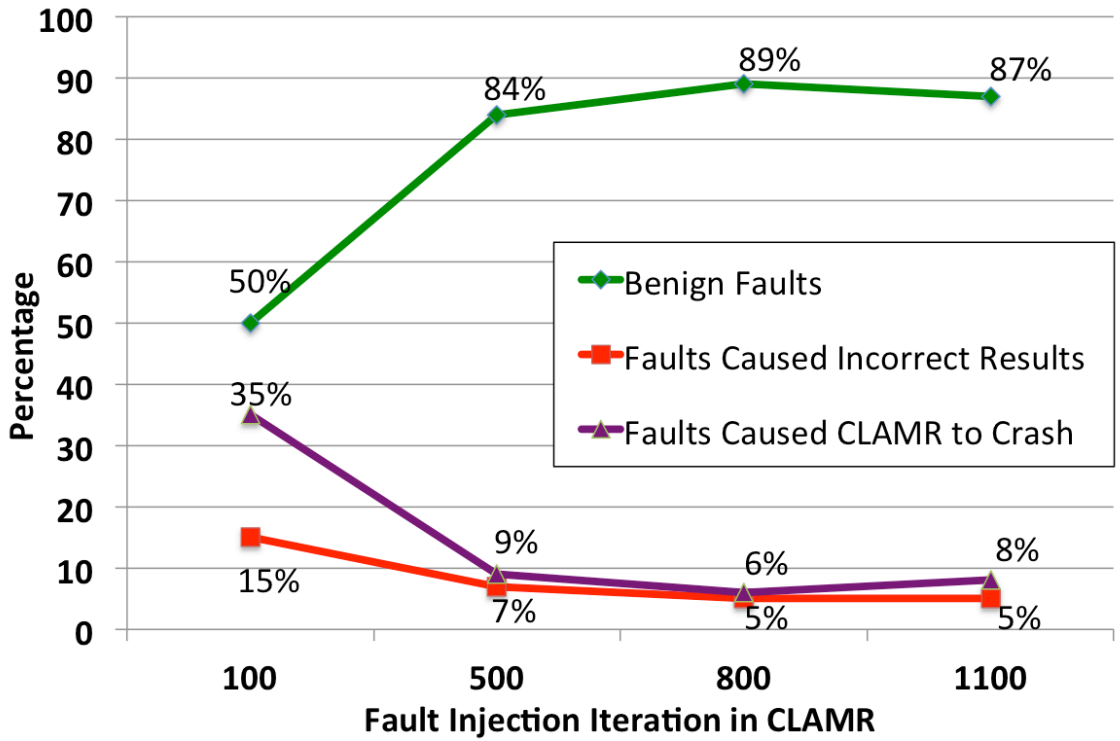
- CLAMR shows surprising resilience – 59% of faults injected at this location have no effect (benign)
- 17% result in SDC
- 24% cause CLAMR crash



More recent studies give slightly different results which will be detailed in a longer talk.

Fault Injection Experiments With Respect to Simulation Time

- Studied 4 injection time steps (100, 500, 800, 1100)
- As the simulation goes on, more of the faults injected result in no effect
- Possible reason is that early on the mesh is smaller (not refined) and changes have large effects. Needs more study.



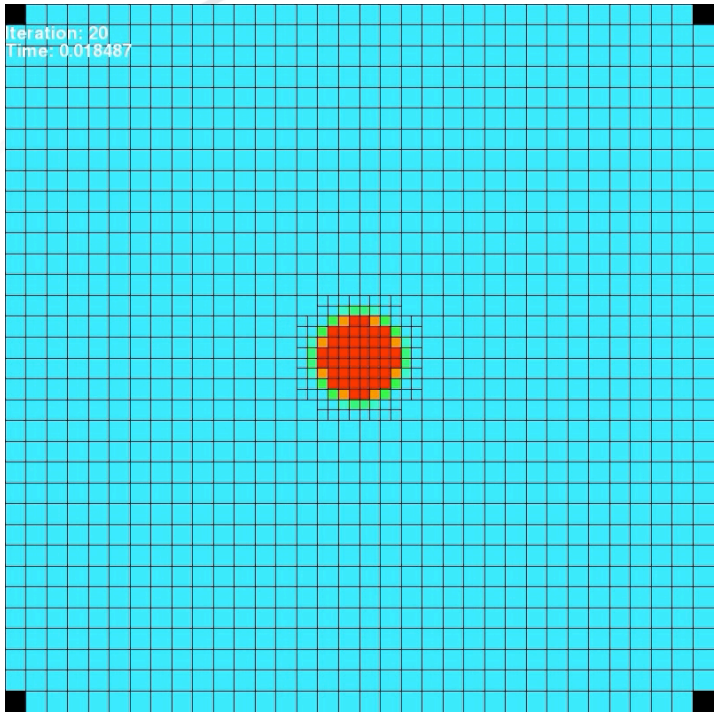
Fault Detection Methods

- Fault Detection Methods
  - Compare graphics
  - Compare checkpoint dumps
  - Check total mass

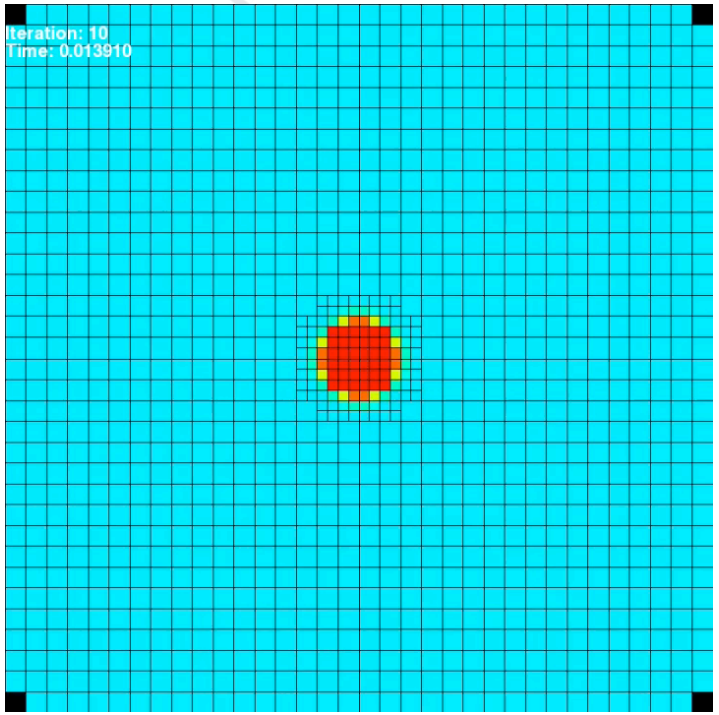
Fault Recovery

- If you can detect fault, why not recover?
  - Use total mass conservation (but need high quality measure)
    - Enhanced precision sums improves sum consistency from around 7 digits to 15 digits
    - Kalman filter
  - Checkpoints
    - File-based
    - In-memory (only difference is file open statement)
    - SSD – future plan

What does a recovery look like?



Fault Recovery from 2nd rollback and another single rollback recovery



Slow Motion

Conclusions and Future Work

- Conclusions
  - Used F-SEFI to inject faults into precise locations of a DOE mini-app, CLAMR
  - ~41% of the faults cause problems (crashes or silent data corruption)
  - Shown that *when* a fault is injected impacts how tolerant CLAMR is to that injection
  - Show visualizations of how faults cause corruption in CLAMR
- Future Work
  - Need more fault injections to establish statistical confidence.
    - This experiment was to demonstrate the usefulness of F-SEFI in performing this type of study
  - Develop fault detection and correction mechanisms

Co-design Impacts … or should it be Cross-group Collaboration

- Added file-based graphics
- Movie Generation
- Checkpointing and Restart (libcrux)
- Automatic Recovery
- In-memory checkpoints with reduction of data having to go to disk

Thanks to – Brian Atkinson, Qiang Guan, and Nathan DeBardeleben