Exploring Use-cases for Non-Volatile Memories in support of HPC Resilience

Onkar Patil¹, Saurabh Huikerikar², Frank Mueller¹, Christian Engelmann²

¹Dept. of Computer Science, North Carolina State University
²Computer Science and Mathematics Division, Oak Ridge National Laboratory
• Exaflop Computers $\rightarrow$ compute devices + memory devices + interconnects + cooling and power

• Close Proximity!

• Manufacturing processes not foolproof
  • Lower durability and reliability
  • Frequency of device failures and data corruptions $\uparrow \rightarrow$ effectiveness and utility $\downarrow$

• Future Applications need to be more resilient,
  • Maintain a balance between performance and power consumption
  • Minimize trade-offs
• Non-volatile memory (NVM) technologies maintain state of computation in the primary memory architecture

• More potential as specialized hardware

• Data Retention critical in improving resilience of an application against crashes

• Persistent memory regions to improve HPC resiliency key aspect of this project

PROBLEM STATEMENT
• Experimentation Setup
  • 16-node cluster with Dual socket, Quad-Core AMD Opteron, 128 GB DRAM memory, Intel SSD from 100GB to 256GB
  • DGEMM benchmark of the HPCC benchmark suite
  • Tested for 4, 8 and 16-node configurations for a matrix sizes of 1000, 2000 and 3000 elements
• DRAM only allocation and NVM-based main memory perform better
  • An inefficient lookup algorithm
• All modes perform similar and consistently for node and data scaling
• Execution time increases exponentially for multiple copies of memory
Conclusion:

- Non-volatile memory devices can be used as specialized hardware for improving the resilience of the system

Future Work:

- Memory usage modes to make applications efficient and maintain complete system state
- Minimal overhead
- Support more complex applications
- Lightweight recovery mechanisms to work with the checkpointing schemes
  - Reduce downtime and rollback time
- Intelligent policies that can help build resilient static and dynamic runtime system
Evaluating Performance of Burst Buffer Models for Real-Application Workloads in HPC Systems

Harsh Khetawat
Frank Mueller
Christopher Zimmer
Introduction

- Existing storage systems becoming bottleneck
- Solution: burst buffers
- Use burst buffers for:
  - Checkpoint/Restart I/O
  - Staging
  - Write-through cache for parallel FS
Placement

• Burst buffer placement:
  – Co-located with compute nodes (Summit)
  – Co-located with I/O nodes (Cori)
  – Separate set of nodes

• Trade-offs in choice of placements
  – Capability – I/O models, staging, etc.
  – Predictability – Impact on shared resources, runtime variability
  – Economic – Infrastructure reuse, cost of storage device

• I/O performance dependent on placement
  – Choice of network topology
Idea

• Simulate network and burst buffer architectures
  – CODES simulation suite
  – Real-world I/O traces (Darshan)
  – Full multi-tenant system with mixed workloads (capability/capacity)
  – Supports network topologies
  – Local & external storage models
• Combine network topologies and storage architectures
• Performance under striping/protection schemes
• Reproducible tool for HPC centers
Conclusion

• Determine based on workload characteristics:
  – Burst buffer placement
  – Network topology
  – Performance of striping across burst buffers
  – Overhead of resilience schemes

• Reproducible tool to:
  – Simulate specific workloads
  – Determine best fit
Thank You