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

Onkar Patil, Saurabh Hukerikar, Frank Mueller, Christian Engelmann

Dept. of Computer Science, North Carolina State University, Computer Science and Mathematics Division, Oak Ridge National Laboratory

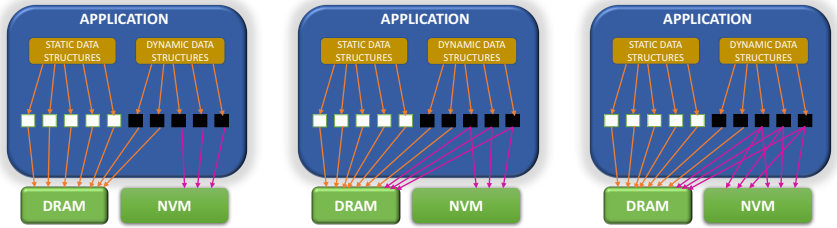
MOTIVATION

- Exaflop Computers → large number of compute + memory devices + different forms of interconnects + cooling and power equipment → Close Proximity
- Manufacturing processes used to make these devices are not foolproof
  - Lower durability and reliability of the devices.
  - Frequency of device failures and data corruptions ↑ → effectiveness and utility ↓
- Future Applications need to be more resilient while they,
  - Maintain a balance between performance and power consumption
  - Minimize trade-offs

PROBLEM STATEMENT

- Non-volatile memory (NVM) technologies → enable memory devices that can maintain state of computation in the primary memory architecture
- More potential in using these memory devices 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

APPROACH



NVM-based Main Memory

```
void main(){
int * a, size;
pmem_cpy_init();
...
a=(int *)pmem_only_alloc(size);
...
pmem_cpy_free(a);
...
}
```

Application-directed Checkpointing

```
void main(){
int * a, size;
pmem_cpy_init();
...
a=(int *)pmem_cpy_alloc(size);
...
pmem_cpy_update(a);
...
pmem_cpy_free(a);
...
}
```

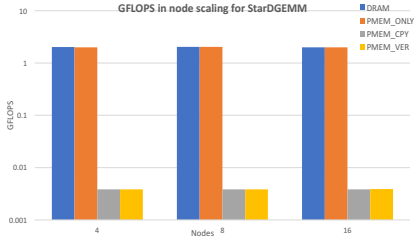
Data Versioning

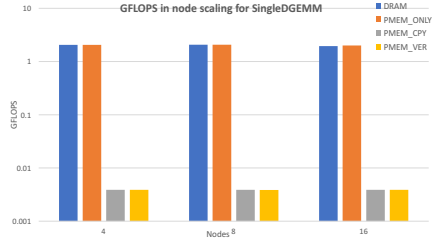
```
void main(){
int * a, size;
pmem_cpy_init();
...
a=(int *)pmem_ver_alloc(size);
...
pmem_cpy_update(a);
...
pmem_cpy_free(a);
...
}
```

- **Design strategy**
  - Enable checkpointing at the data structure level
  - Some data structures are more critical than others at different stages of the application in terms of failure recovery
  - Reduce the space and time overhead considerably in comparison to traditional checkpointing methods
  - Easy to use API with minimal code changes

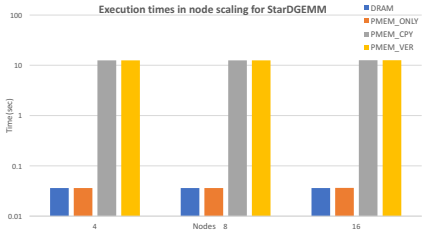
RESULTS

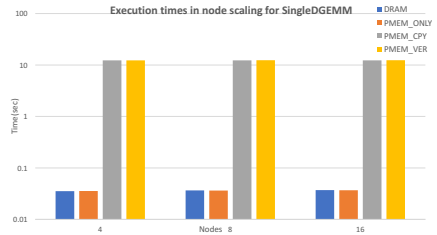
- **Experimentation Setup**
  - 16-node cluster with Dual socket, Quad-Core AMD Opteron, 128 GB DRAM memory, Intel SSD from 100GB to 256GB
  - DGEMM benchmark of the HPCC benchmark suite
  - Tested for 4, 8 and 16-node configurations for a matrix sizes of 1000, 2000 and 3000



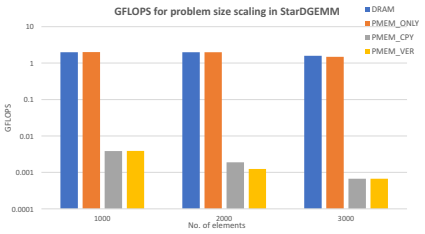


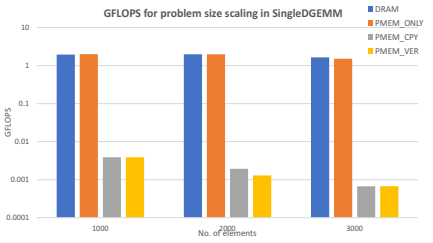
- DRAM only allocation and NVM-based main memory perform better than Application-directed Checkpointing and Data Versioning partly due to an inefficient lookup algorithm



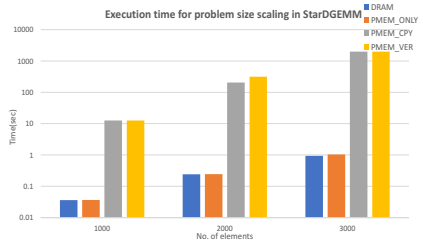


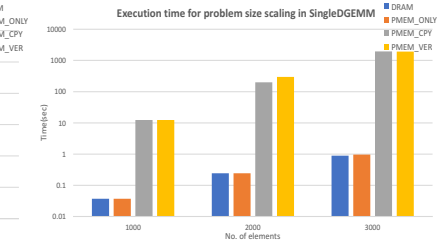
- The performance is consistent for both Single matrix multiplication and multiple matrix multiplication operations





- All modes perform similar and consistently when scaled by node size or problem size





- The execution time increases exponentially when using two types of memory instead of one

FUTURE WORK

- Develop the memory usage modes to make them more efficient and maintain complete system state
  - Minimal overhead
  - Support more complex applications
- Develop lightweight recovery mechanisms to work with the checkpointing schemes
  - Reduce downtime and rollback time
- Combine them with intelligent policies that can help build resilient static and dynamic runtime system

CONCLUSION

- Non-volatile memory devices can be used as specialized hardware for improving the resilience of the system and we demonstrated three potential memory usage models that show consistent performance for compute intensive workloads
- REFERENCES

  - Hukerikar, Saurabh, and Christian Engelmann. "Resilience Design Patterns-A Structured Approach to Resilience at Extreme Scale." *arXiv preprint arXiv:1611.02717* (2016).
  - Mittal, Sparsh, and Jeffrey S. Vetter. "A survey of software techniques for using non-volatile memories for storage and main memory systems." *IEEE Transactions on Parallel and Distributed Systems* 27.5 (2016): 1537-1550.
  - Hsu, Terry Ching-Hsiang, et al. "NVthreads: Practical Persistence for Multi-threaded Applications." *Proceedings of the Twelfth European Conference on Computer Systems*. ACM, 2017.
  - Liu, Qingrui, et al. "Compiler-directed lightweight checkpointing for fine-grained guaranteed soft error recovery." *High Performance Computing, Networking, Storage and Analysis, SC16: International Conference for*. IEEE, 2016.
  - Yang, Shuo, et al. "Algorithm-Directed Crash Consistence in Non-Volatile Memory for HPC." *arXiv preprint arXiv:1705.05541* (2017).
  - Wong, Daniel, G. S. Lloyd, and M. B. Gokhale. *A memory-mapped approach to checkpointing*. No. LLNL-TR-635611. Lawrence Livermore National Laboratory (LLNL), Livermore, CA, 2013.
  - Rezaei, Arash. *Fault Resilience for Next Generation HPC Systems*. North Carolina State University, 2016.

  - This work was sponsored by the U.S. Department of Energy's Office of Advanced Scientific Computing Research. This manuscript has been co-authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).