LUG Panel: 2020 HPC Platform Architectures and Their Impact on Storage

Terri Quinn
Lawrence Livermore National Laboratory

LUG ◆ April 9, 2014

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Power constraints present a significant challenge for HPC

	White	ASCI Q	Purple	Red Storm	Dawn	Cielo	Sequoia	Future
Mem /core	▨ 1	🟩 >2	🟩 4	🟩 2	▨ 1	🟩 2	▨ 1	🟥 <0.2
B:F	🟩 0.58	🟩 1.04	🟩 1.6	▨ 0.44	🟩 1	🟩 0.86	🟨 0.28	🟥 ~0.03

Green – good; striped – suboptimal; yellow- marginal; red – extremely challenging

- Core density *increasing dramatically*
- Memory size and bandwidth per core *decreasing dramatically*
- Data movement is the dominant power consumer
- Hierarchical compute, memory, & storage is dominate in roadmaps
- Innovative technologies are in the works (and are needed)

Performance depends on managing data motion

Memory/storage hierarchies and innovative memory/storage technologies will be introduced to compensate

- Several memory innovations are promising (fast but small, slower but larger, even slower but even larger)
- NVRAM will be introduced “near” to compute to increase memory capacity and in sufficient quantities to eliminate the need for disk (almost) entirely
- Bulk of our I/O may stay on the compute volume in the NVRAM
- RAIT(?) tape archive with a large disk cache in front may be our future site-wide file system

Biggest worry is whether the software will be able to evolve to work with this hardware paradigm

How will the application teams compensate?

- Only a performance crisis (not just the fear of one) will motivate progress - adding capabilities to apps takes precedence over performance
- Teams will continue to use a POSIX interface – though early adopters will lead the way
- Increased use of approaches to avoid saving and storing data
  - In-situ analysis and visualization
  - Checkpointing on the computer
  - Heightened consideration of what data absolutely has to be saved
  - Greater consideration of recompute than store and retrieve – a more systematic approach is needed
- Most applications will need to be n-m (since n-n and n-1 will be too painful since we won't solve the fundamental metadata issues)

Today's lessons will still apply. If you don't have to move data, don't. If you don't have to store data, don't.