NEXTGenIO: Moving I/O into the memory system

Adrian Jackson

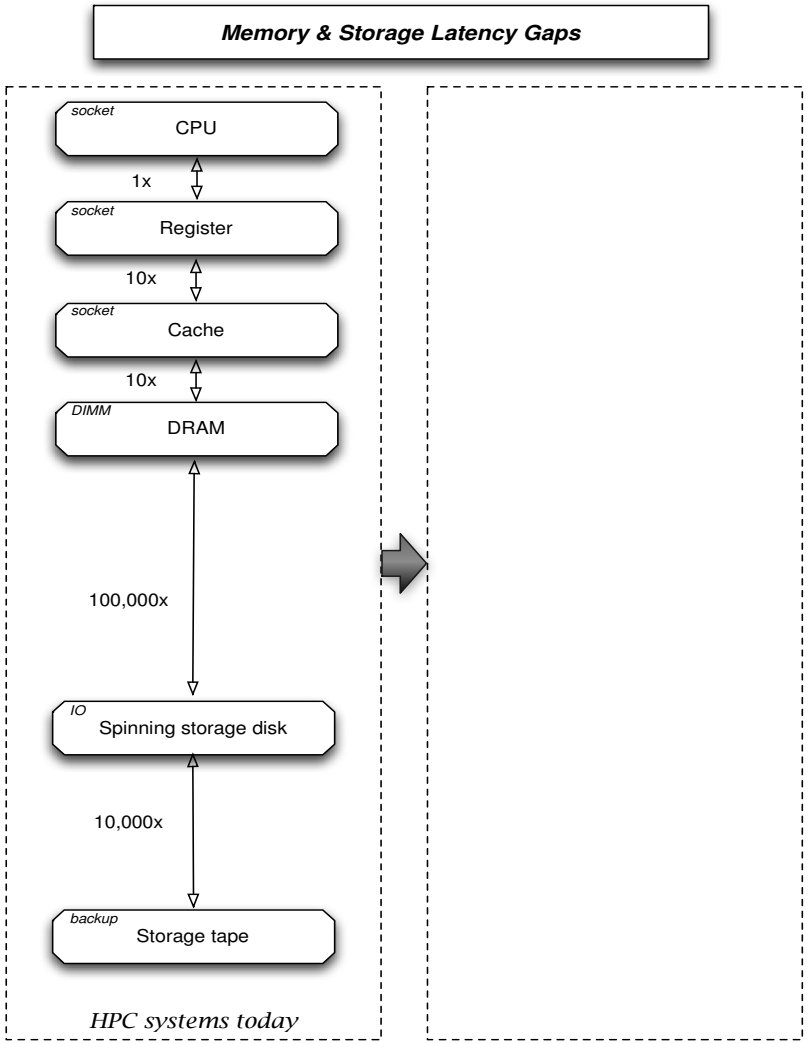
@adrianjhpc

adrianj@epcc.ed.ac.uk

EPCC, The University of Edinburgh

A new hierarchy

- New non-volatile memory technology is going to change the memory hierarchy we have
- What does that mean for applications, particularly scientific simulations?
- I/O performance is one of the critical components for scaling up HPC applications and enabling HPDA applications at scale



NEXTGenIO summary

Project

- Research & Innovation Action
- 36 month duration
- €8.1 million
- Approx. 50% committed to hardware development

Partners

- EPCC
- INTEL
- FUJITSU
- BSC
- TUD
- ALLINEA
- ECMWF
- ARCTUR

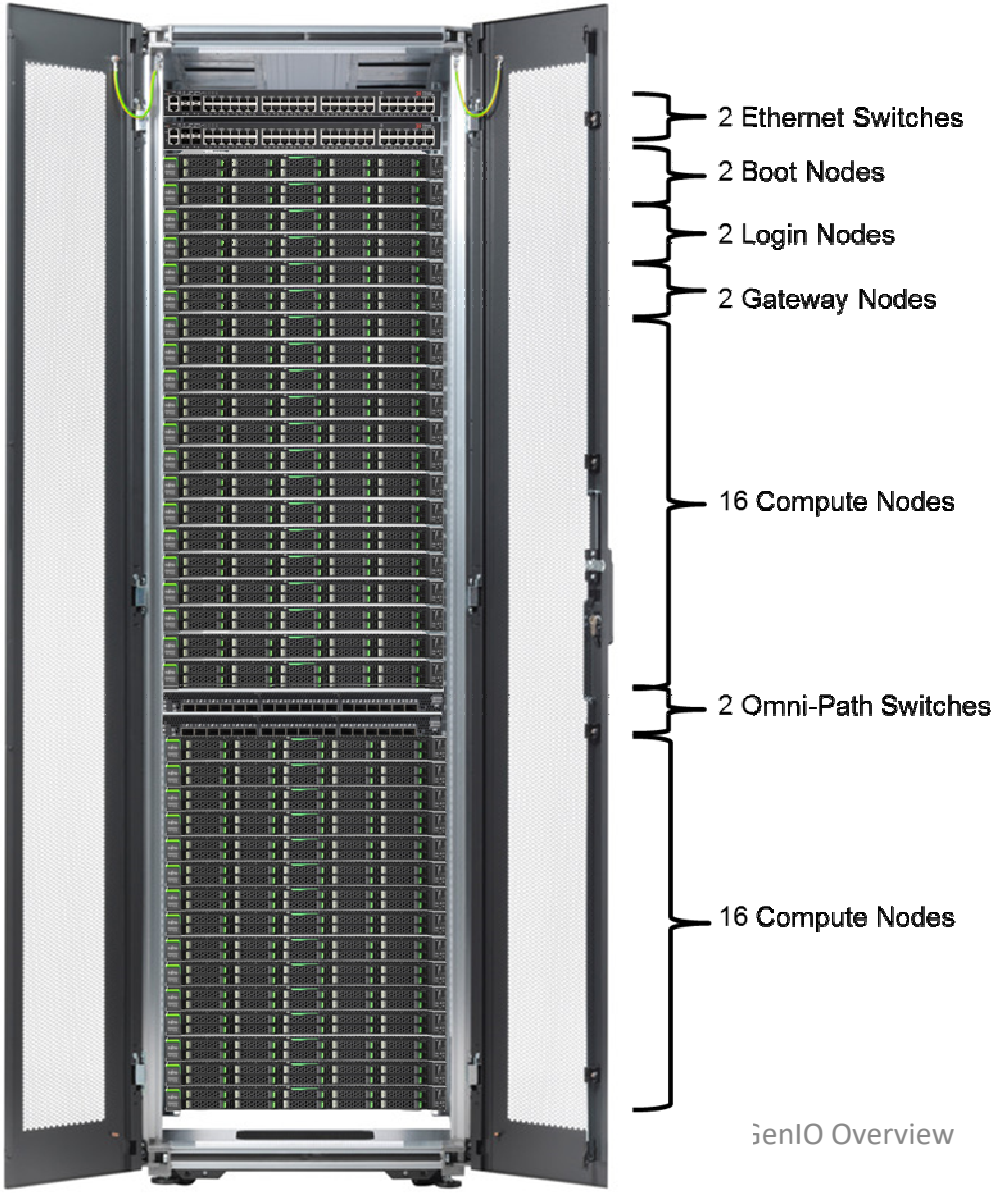
NEXTGenIO objectives

- Develop a new server architecture using next generation processor and memory advances
  - Based on Intel® Xeon and Intel DIMM based on 3D XPoint™ memory technology
- Investigate the best ways of utilising these technologies in HPC
  - Develop the systemware to support their use, particularly at scale
- Model different I/O workloads and use this understanding in a co-design process
  - Representative of real HPC centre workloads

Hardware

- The project is developing a new HPC platform with focus on I/O performance
  - System and motherboard designed & built by Fujitsu
- The *Complete Compute Nodes* are based on
  - Intel™ CPUs - 2 sockets per node
  - Intel™ DIMMs
  - Intel™ OmniPath

Prototype WM [2]1



Note: final configuration may differ

Intel™ DIMMs

- Non-volatile RAM
  - 3D XPoint technology
- Much larger capacity than DRAM
- Slower than DRAM by a small factor, but significantly faster than SSDs ™
- 12 DIMM slots per socket
  - Populated by combination of DDR4 and Intel™ DIMMs

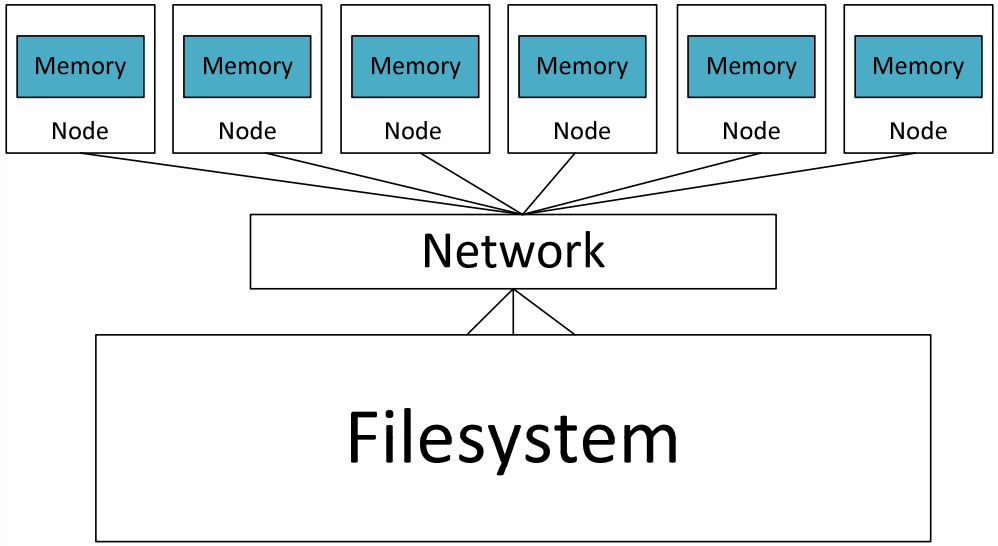
Intel™ DIMMs – Usage Models WM1

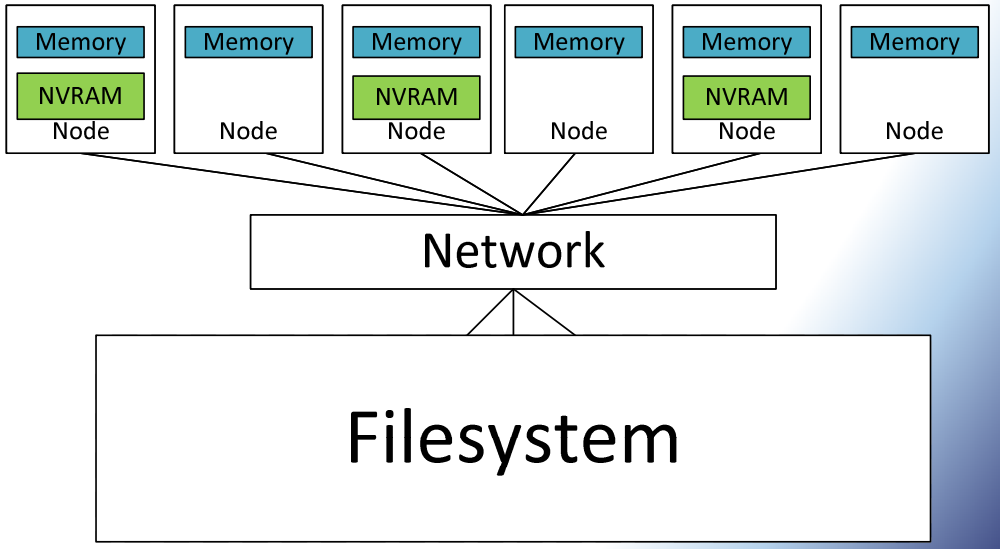
- The “memory” usage model allows for the extension of the main memory with Intel™ DIMMs
  - The data is volatile like normal DRAM based main memory
- The “storage” usage model which supports the use of Intel™ DIMMs like a classic block device
  - E.g. like a very fast SSD
- The “application direct” usage model maps persistent storage from the Intel™ DIMMs directly into the main memory address space
  - Direct CPU load/store instructions for persistent main memory regions

Remote access

- Complete compute nodes and network hardware will support remote access to NVDIMMs from other CCNs
  - Using RDMA between nodes will allow data in NVDIMMs to be shared between CCNs if required by the applications using them
- Systemware will support remote access and use it for data partitioning and replication.

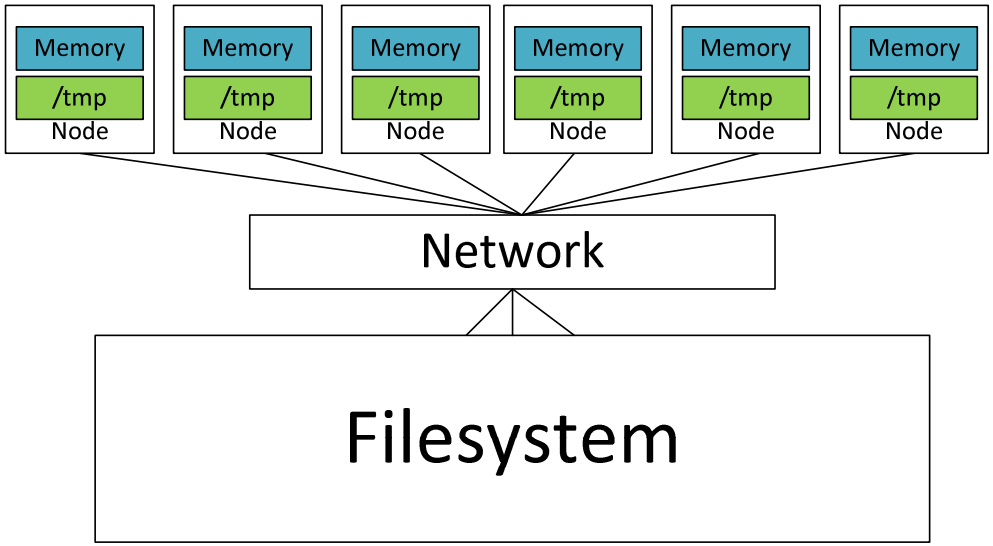
Exploiting distributed storage





Using distributed storage

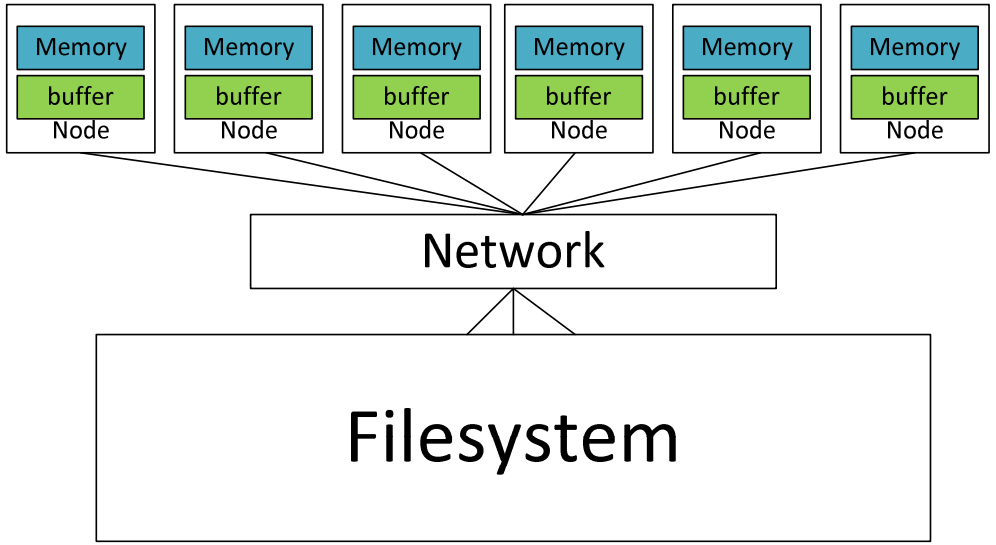
- Without changing applications
  - Large memory space/in-memory database etc…
  - Local filesystem



  - Users manage data themselves
  - No global data access/namespace, large number of files
  - Still require global filesystem for persistence

Using distributed storage

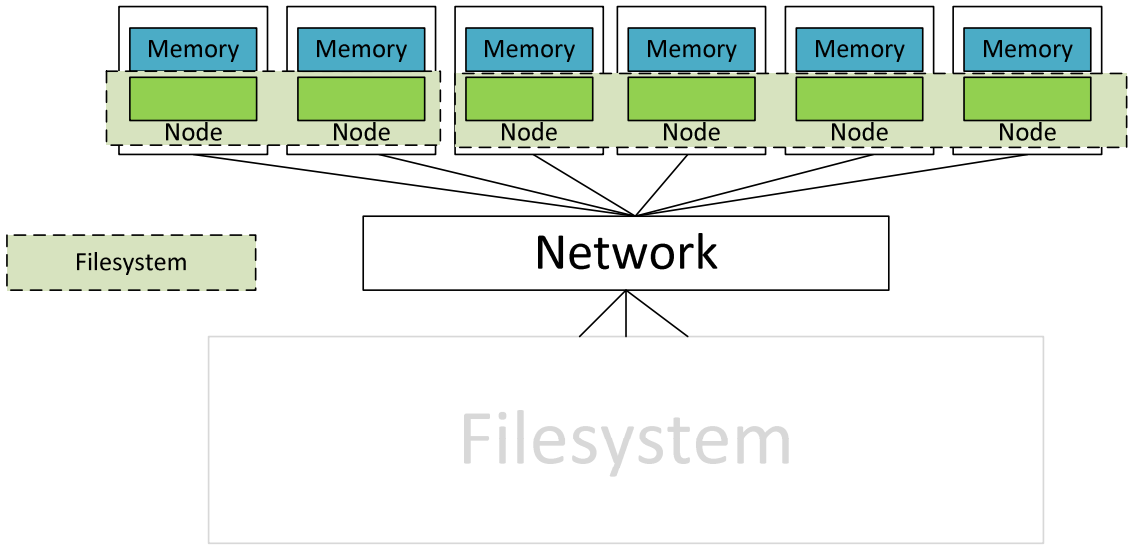
- Without changing applications
  - Filesystem buffer



  - Pre-load data into NVRAM from filesystem
  - Use NVRAM for I/O and write data back to filesystem at the end
  - Requires systemware to preload and postmove data
  - Uses filesystem as namespace manager

Using distributed storage

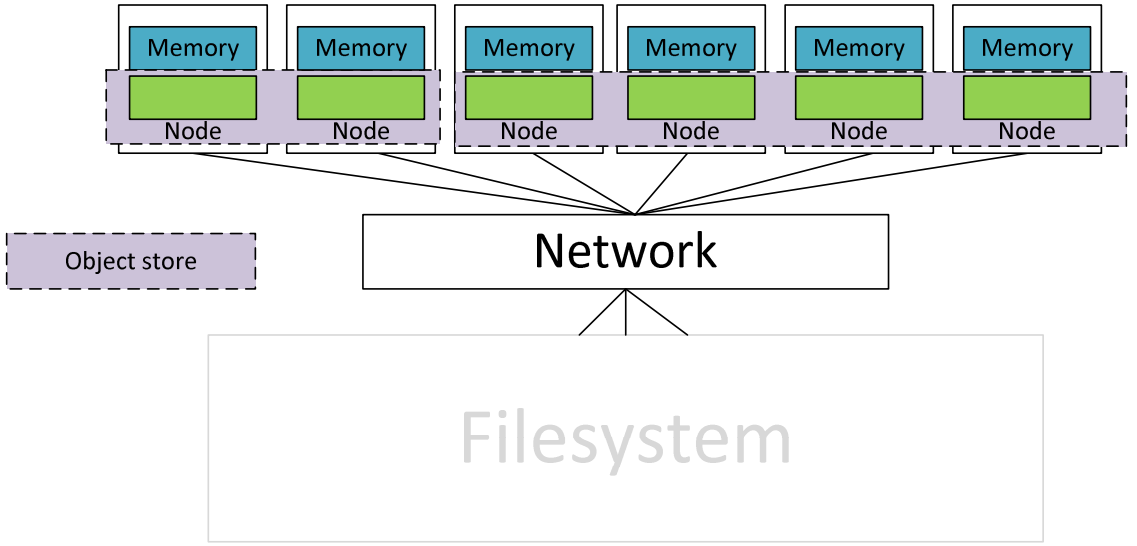
- Without changing applications
  - Global filesystem



- Requires functionality to create and tear down global filesystems for individual jobs
- Requires filesystem that works across nodes
- Requires functionality to preload and postmove filesystems
- Need to be able to support multiple filesystems across system

Using distributed storage

- With changes to applications
  - Object store



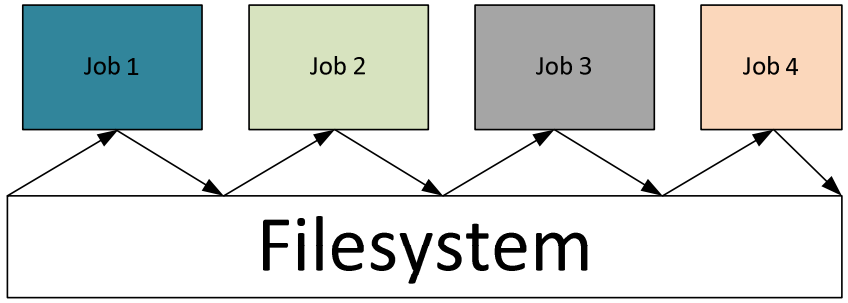
- Needs same functionality as global filesystem
- Removes need for POSIX, or POSIX-like functionality

Using distributed storage

- Without changing applications
  - Automatic checkpointing
    - Resiliency
    - Local checkpointing without hitting the filesystem
  - Pause and restart
    - Just-in-time scheduling/high priority jobs
    - Waiting for something else to happen…

Using distributed storage

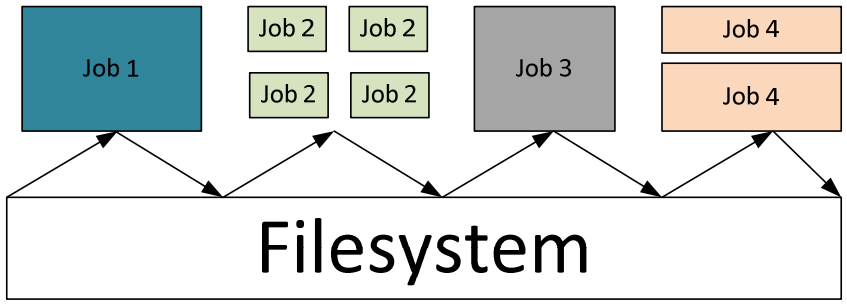
- New usage models
  - Resident data sets
    - Sharing preloaded data across a range of jobs
    - Data analytic workflows
    - How to control access/authorisation/security/etc….?
  - Workflows
    - Producer-consumer model



    - Remove filesystem from intermediate stages

Using distributed storage

- Workflows
  - How to enable different sized applications?



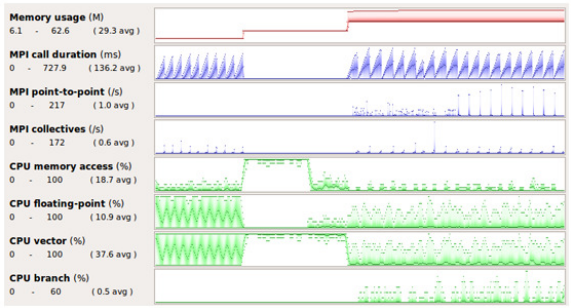
  - How to schedule these jobs fairly?
  - How to enable secure access?

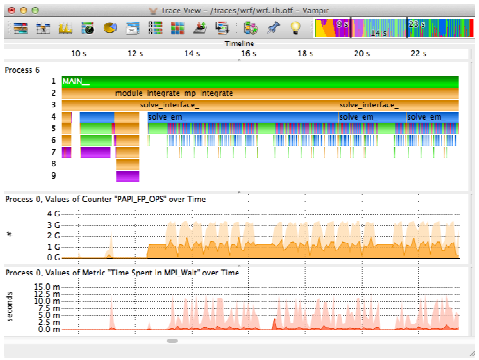
The Challenge of distributed storage

- Enabling all the use cases in multi-user, multi-job environment is the real challenge
  - Heterogeneous scheduling mix
  - Different requirements on the NVRAM
  - Scheduling across these resources
  - Enabling sharing of nodes
  - etc….
- Enabling applications to do more I/O
  - Large numbers of our applications don't heavily use I/O at the moment
  - What can we enable if I/O is significantly cheaper
- NEXTGenIO is tackling these
  - Job scheduler
  - Data scheduler
  - Data movers

Tools

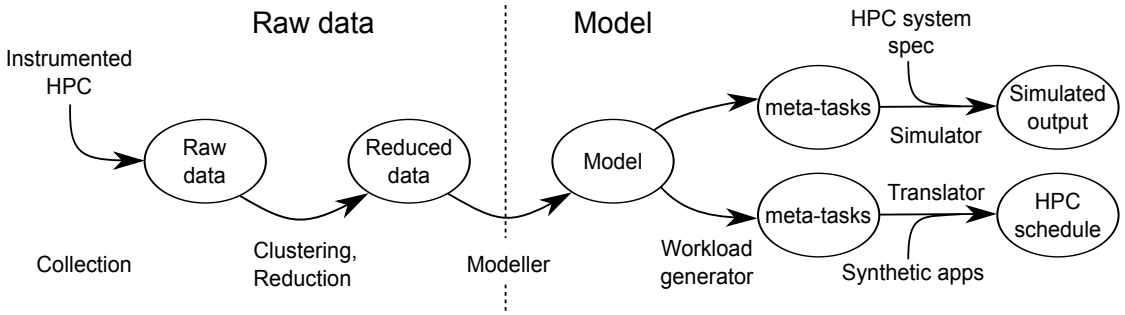
- Performance analysis tools need to understand new memory hierarchy and its impact on applications
  - TUD's Vampir & Allinea's MAP
- At the same time, tools themselves can exploit NVRAM to rapidly store sampling/tracing data

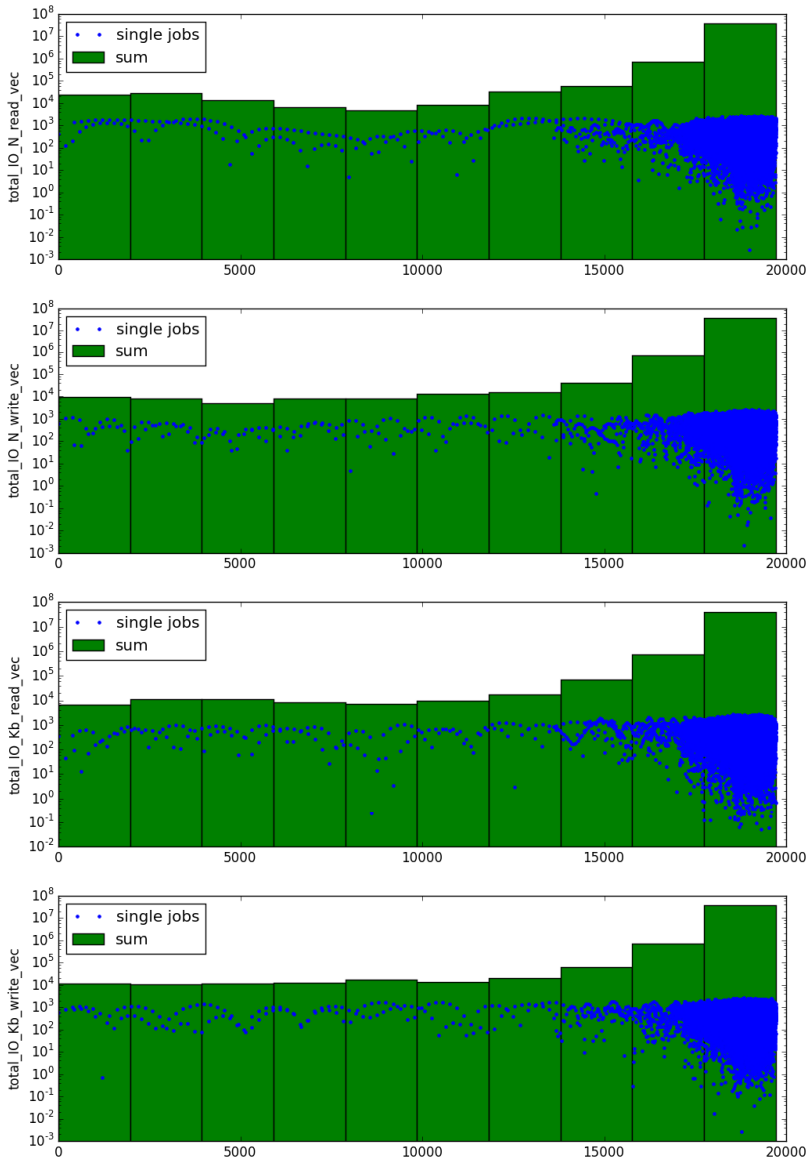




IO workload simulation

- Need to quantify improvements in job runtime and throughput
  - Measure and understand current bottlenecks
- Create a workload simulator and generator
  - Simulator can be used to derive system configuration options
  - Generator can be used to create scaled down version of data centre workload




single jobs
sum
total_IO_N_read_vec
total_IO_N_write_vec
total_IO_Kb_read_vec
total_IO_Kb_write_vec
0
5000
10000
15000
20000

Summary

- NEXTGenIO developing a ***full*** hardware and software solution
- Requirements capture and first architectural designs completed
  - Hardware under development
  - Systemware under development
- Potential to both significantly reduce I/O costs and enable new usage models for HPC and HPDA
  - Proper convergence between HPC and HPDA