NEXTGenIO Performance Tools for In-Memory I/O

holger.brunst@tu-dresden.de
ZIH, Technische Universität Dresden

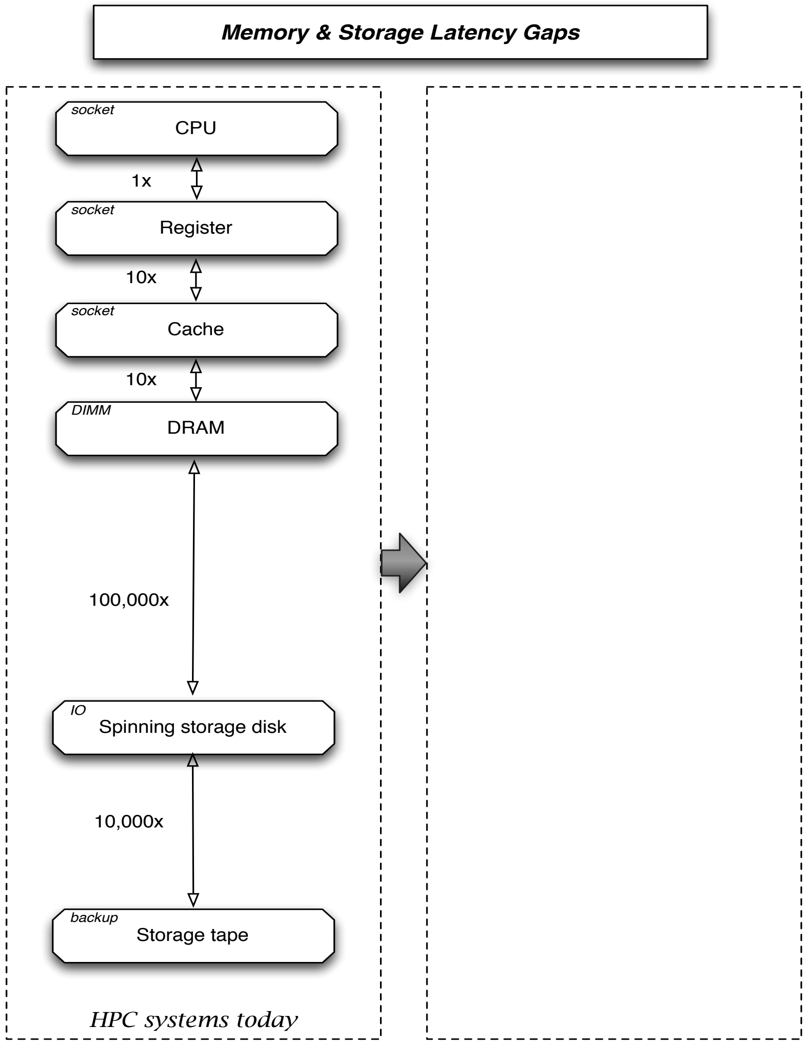
UIOP Workshop, Hamburg 22nd-23rd March 2017

Credits

Intro slides by Adrian Jackson (EPCC)

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

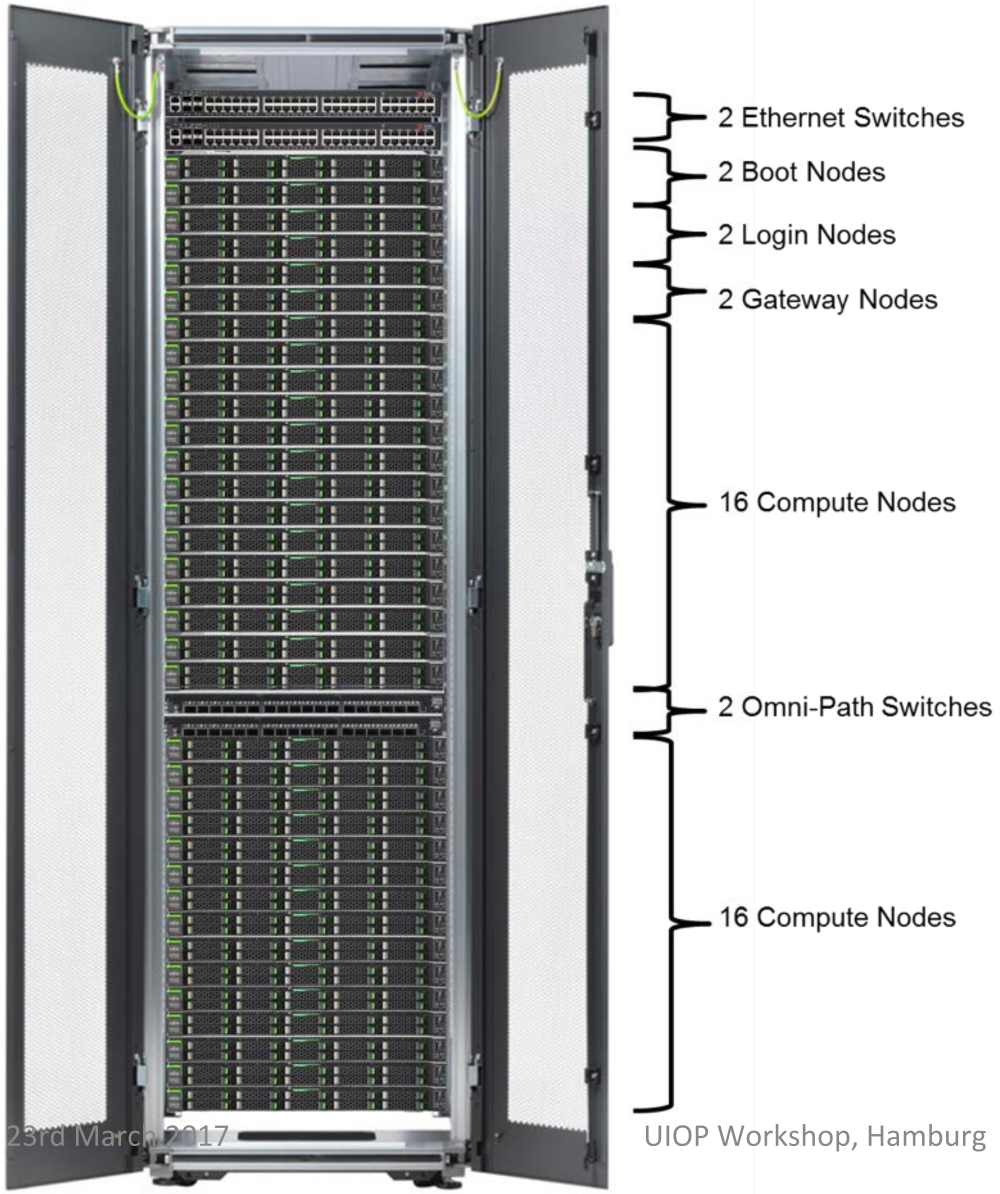
NEXTGenIO objectives

- 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



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

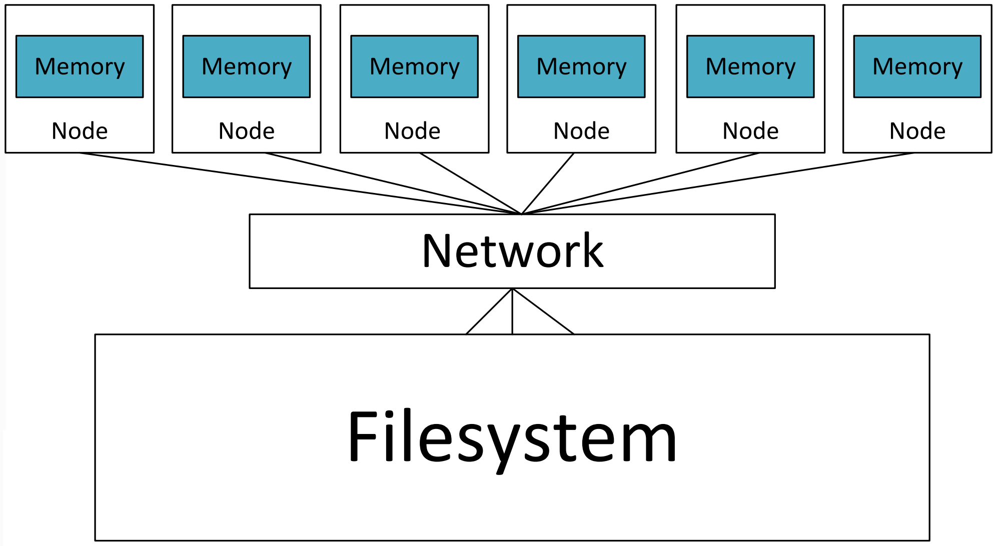
Intel™ DIMMs – Usage Models

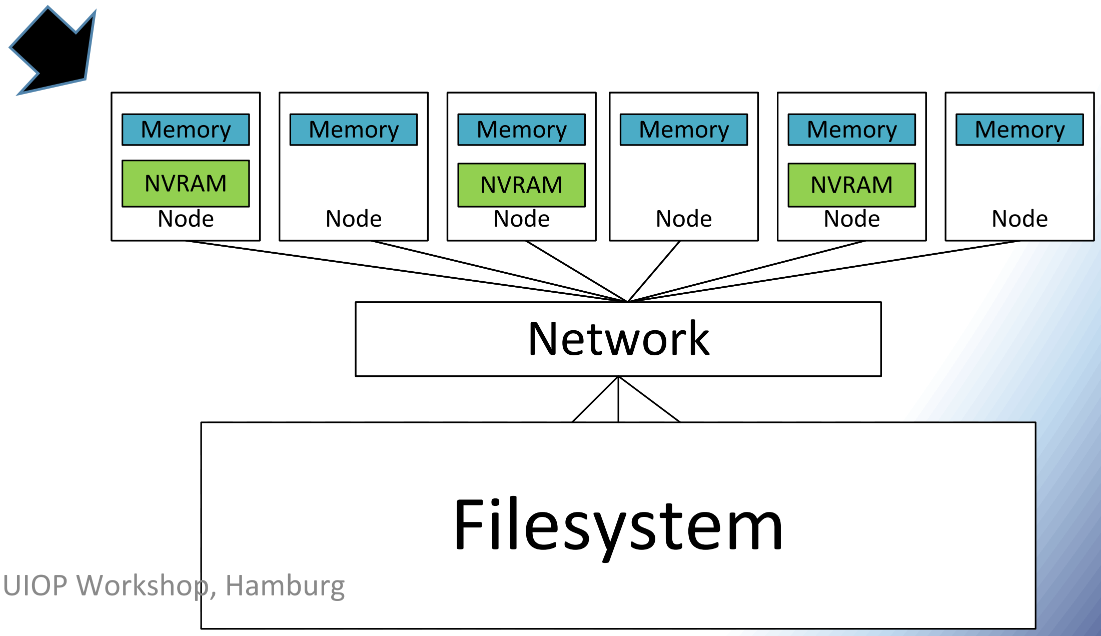
- The "memory" usage model allows for the extension of the main memory with Intel™ DIMMs
  - The data is volatile like normal DRAM based main memory
- The "storage" usage model which supports the use of Intel™ DIMMs like a classic block device
  - E.g. like a very fast SSD
- The "application direct" 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.

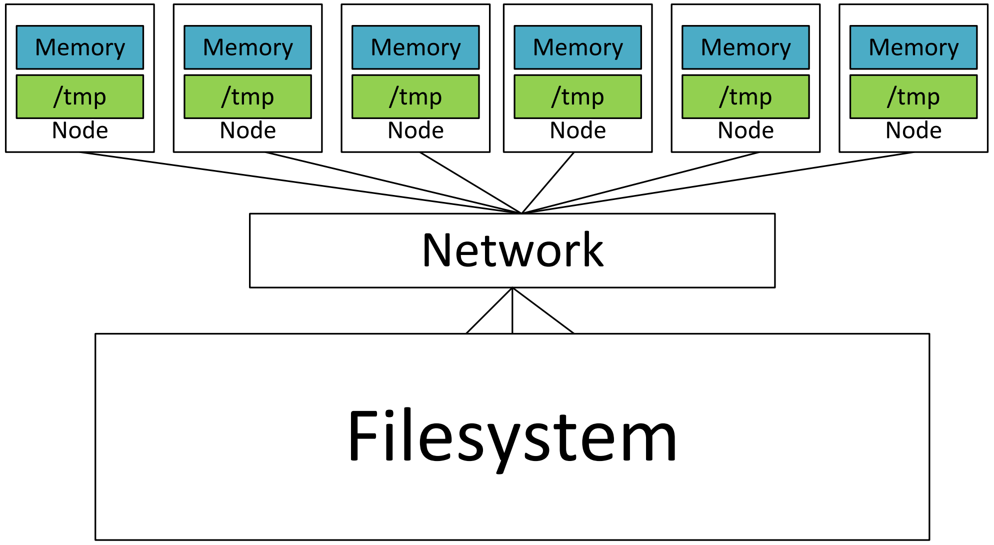
Exploiting distributed storage





Using distributed storage

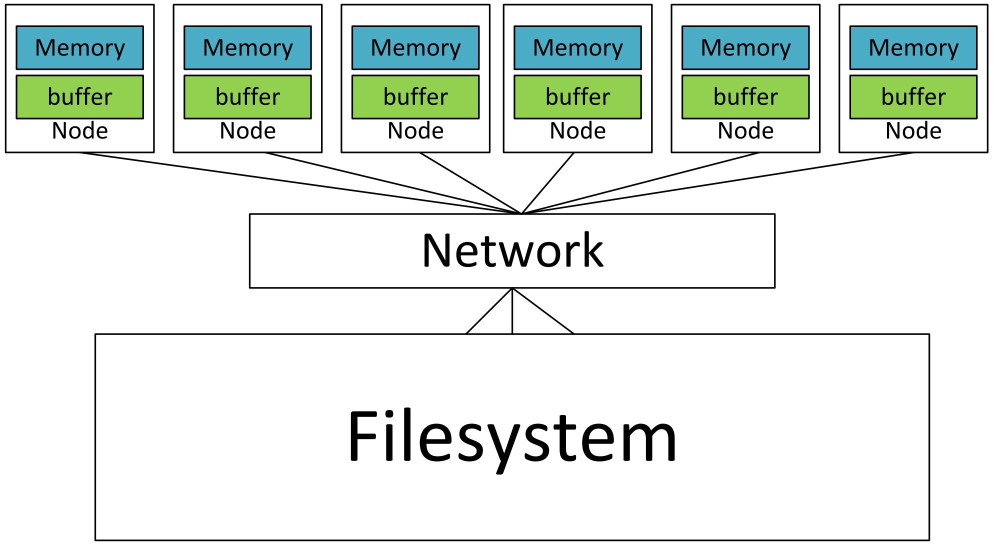
- Without changing applications
  - Large memory space/in-memory database etc…
  - Local filesystem



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

Using distributed storage

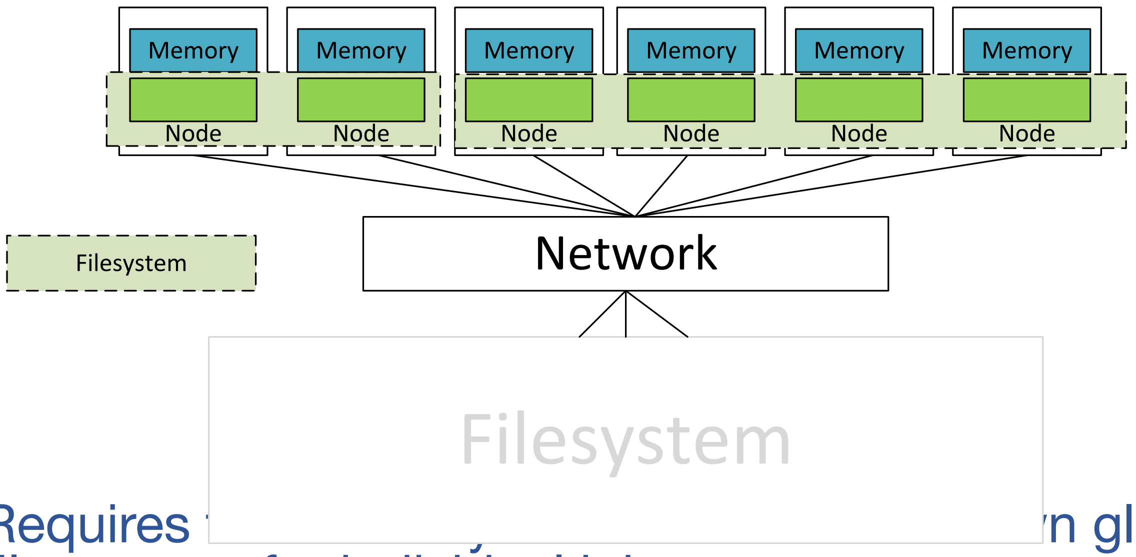
- Without changing applications
  - Filesystem buffer



- 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

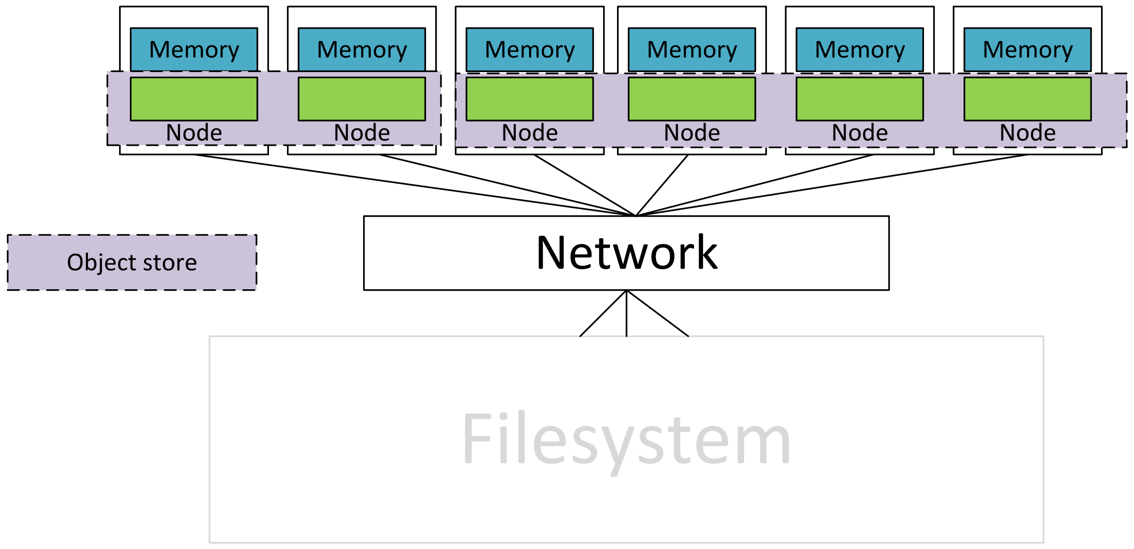
- Without changing applications
  - Global filesystem



- Requires n 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



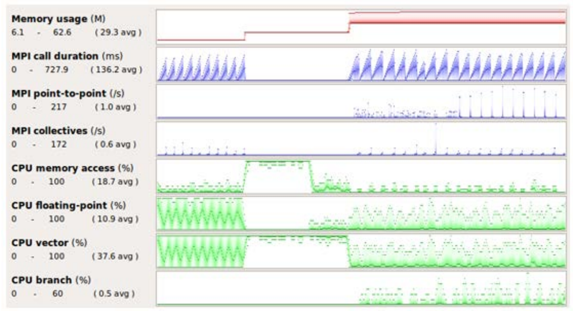
- Needs same functionality as global filesystem
- Removes need for POSIX, or POSIX-like functionality

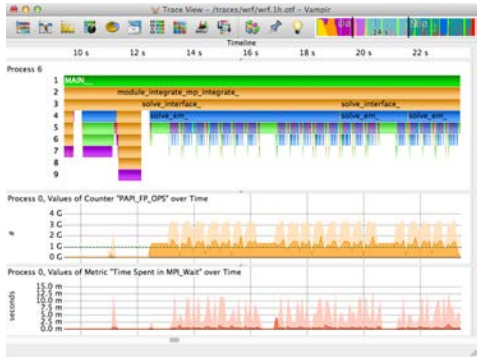
The Challenge of distributed storage

- 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 effort

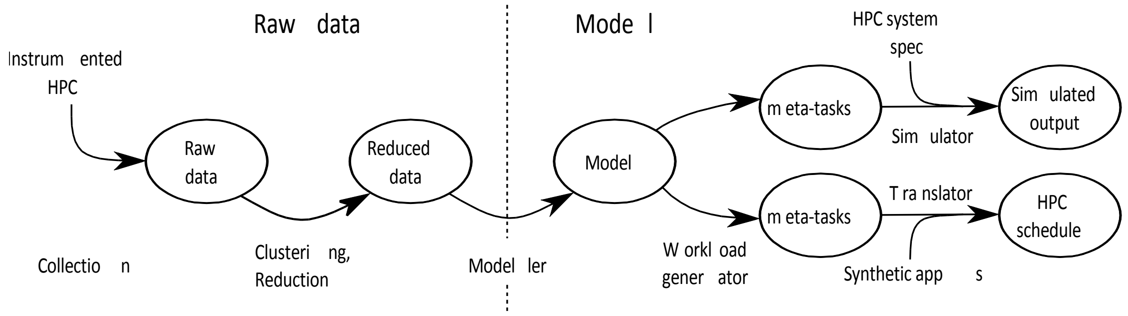
- Performance analysis tools need to understand new memory hierarchy and its impact on applications
  - TUD's Vampir & Allinea's MAP
- At the same time, tools themselves can exploit NVRAM to rapidly store sampling/tracing data




Trace View – /traces/wrf/wrf.1h.otf – Vampir
Timeline
Process 6
MAIN__
module_integrate_mp_integrate_
solve_interface_
solve_em_
Process 0, Values of Counter "PAPI_FP_OPS" over Time
Process 0, Values of Metric "Time Spent in MPI_Wait" over Time
seconds

IO workload simulation by ECMWF

- 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



Performance Features

- Monitoring NVRAM resources
- Recording File I/O operations
- Providing memory access statistics
- Adding system topology information
- Merging perf. data (workflow support)

- Hence, extensions are required in:
  - Data formats
  - Measurement infrastructure
  - Visualization

Functional extensions in measurement infrastructure

Monitoring extensions

1. File I/O operations
2. Hardware/software counters
    1. Non-Volatile Memory Library (NVML) provides information on NVRAM allocation
    2. Counters enable memory statistics on NVRAM
3. System topology information related to NVRAM

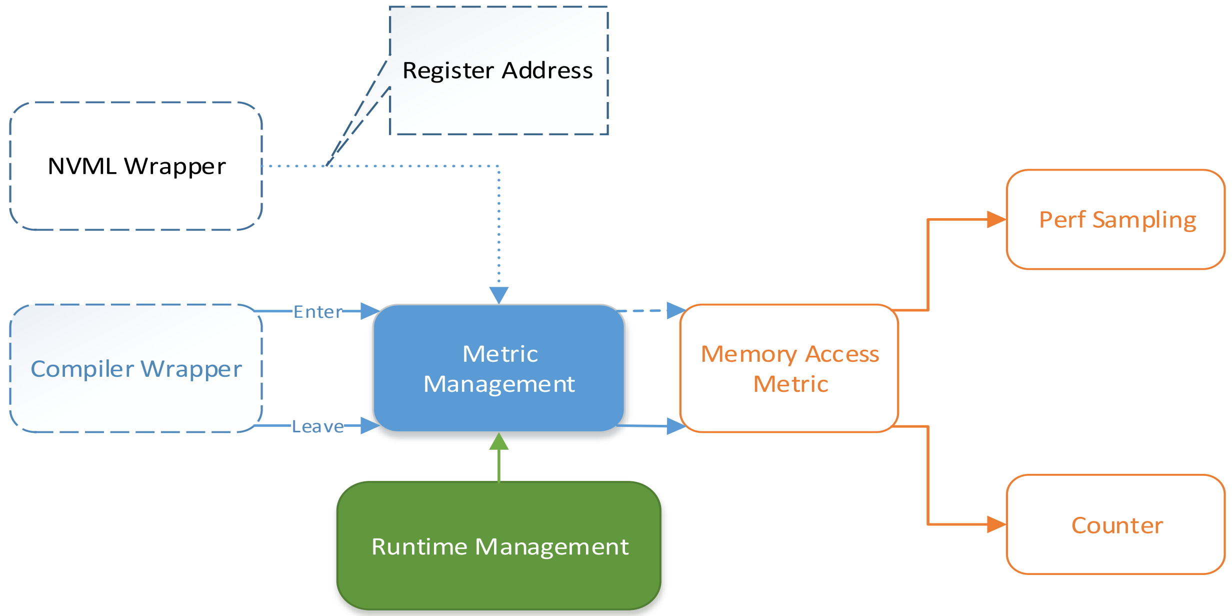


Figure 1 : Metric Architecture with the Memory Access Metric extension. Differences between TUD and Allinea

NVM Library Wrapper

- Enter and leave events
- Additional information being recorded
  - Requested memory size
  - Usable memory size
  - High Water Mark metric for the utilization of memory pool over its entire execution
  - Size and number of elements available in the persistent array
- Byte access activities to/from NVRAM remain out of scope (e.g. memory mapped files)
- NVRAM health status
  - Intel management API access (system configuration)
  - S.M.A.R.T. (on node level)
  - Intel NVDIM_API

Memory Access Statistics

- Memory access hotspots for using DRAM and NVRAM?
  - Where? When? Type of memory?
- Metric collection needs to be extended
  1. DRAM local access
  2. DRAM remote access (on a different socket)
  3. NVRAM local access
  4. NVRAM remote access (on a different socket)

Access to PMU using perf

- Architectural independent counters
  - May introduce some overhead
    - MEM_TRANS_RETIRED.LOAD_LATENCY
    - MEM_TRANS_RETIRED.PRECISE_STORE
    - Guess: It will also work for NVRAM?
- Architectural dependent counters
  - Counter for DRAM
    - MEM_LOAD_UOPS_L3_MISS_RETIRED.REMOTE_DRAM
    - MEM_LOAD_UOPS_L3_MISS_RETIRED.LOCAL_DRAM
    - MEM_LOAD_UOPS_*.REMOTE_NVRAM ?
    - MEM_LOAD_UOPS_*.LOCAL_NVRAM ?

Information of system topology in compute node

- Following information will be incorporated in the system topology of each compute node:
  1. Using procfs:
     - Total size of NVRAM
     - Total size of DRAM
     - Total processors
  2. Using sysfs for:
     - Namespaces for used NVDIMMs
  3. Using libnuma / numactl:
     - Total number of NUMA nodes
     - Memory size of each NUMA node
     - Processing Cores allocated to each NUMA node
     - NUMA distances
  4. SLURM API will be used to provide meta information of following records:
     - Job memory size of NVRAM
     - Job memory size of DRAM
     - Number of compute nodes
     - Number of compute processors

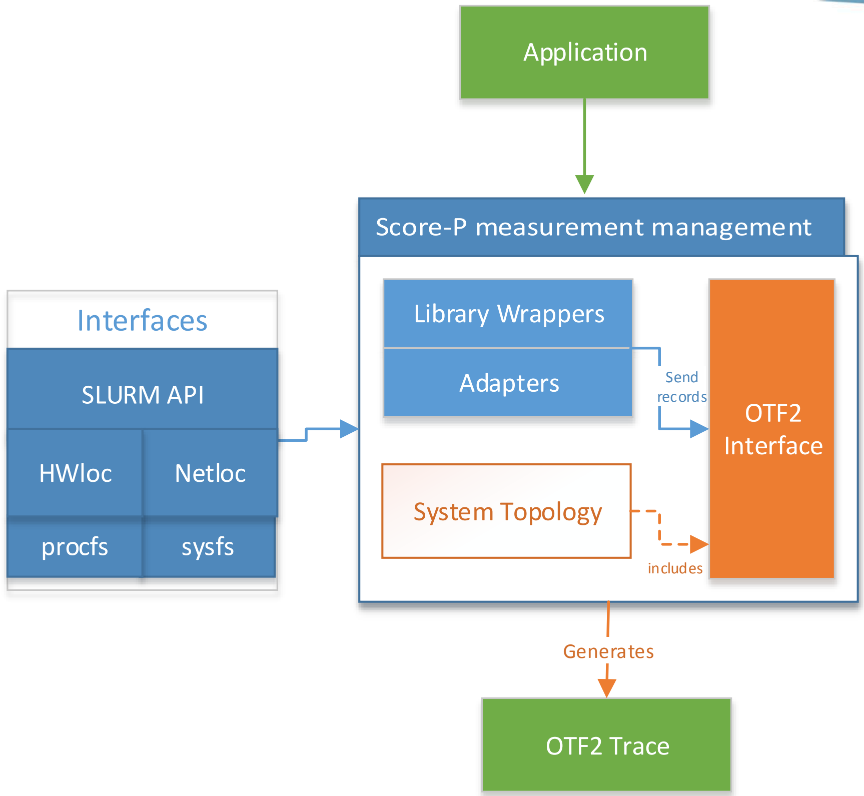


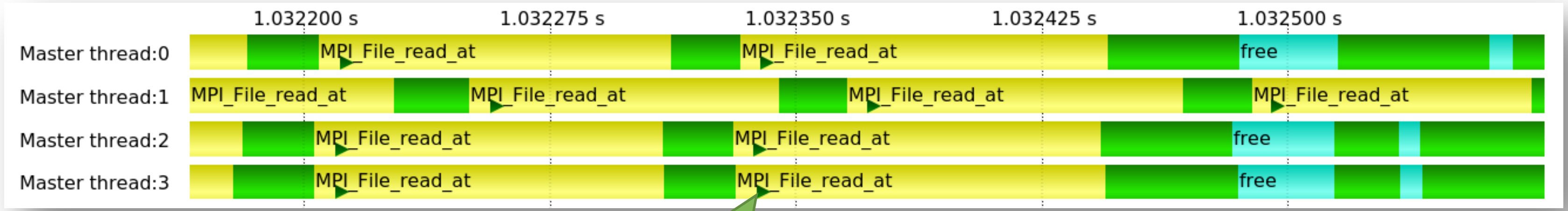
Figure 2 : Extension of Score-P infrastructure to incorporate system topology in OTF2 Trace

Functional Extensions in Visualization (Vampir)

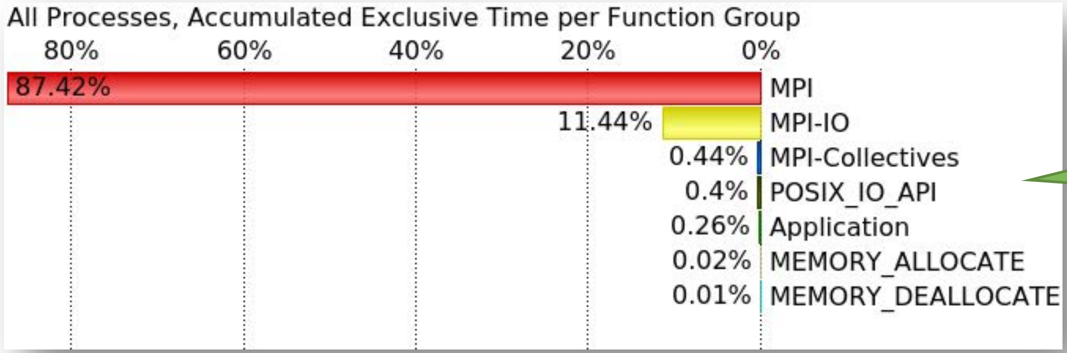
Display stacked I/O layers

- I/O layers
  - POSIX
  - MPI-I/O
  - HDF5
  - NetCDF
  - PNetCDF
  - Adios
  - (Lustre)
- Data of interest
  - File Open/Create Operations
  - File Close Operations
  - Data Transfer operations

I/O operations over time

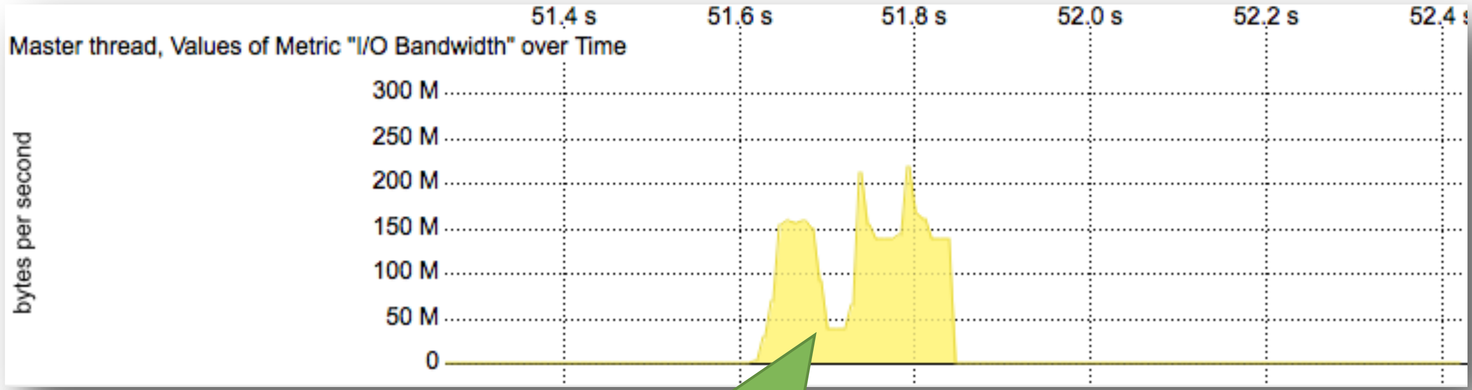


Individual I/O Operation



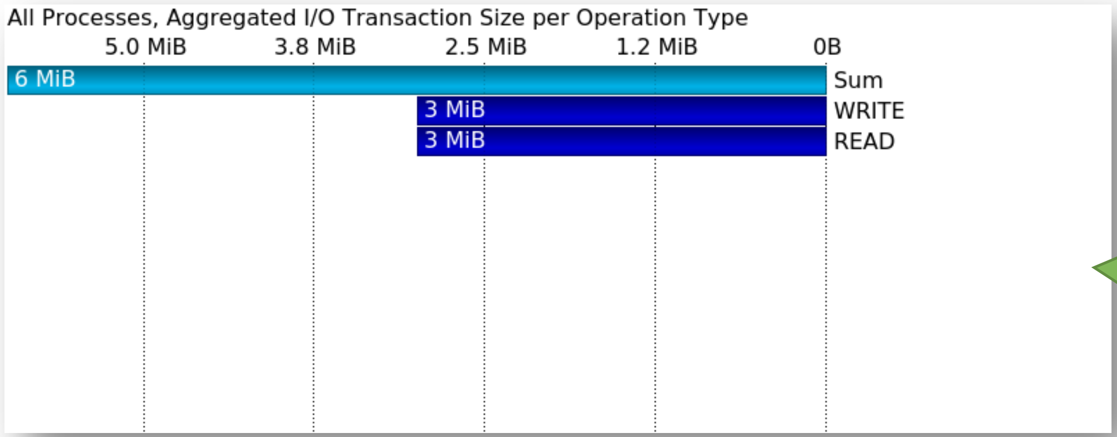
I/O Runtime Contribution

I/O data rate over time



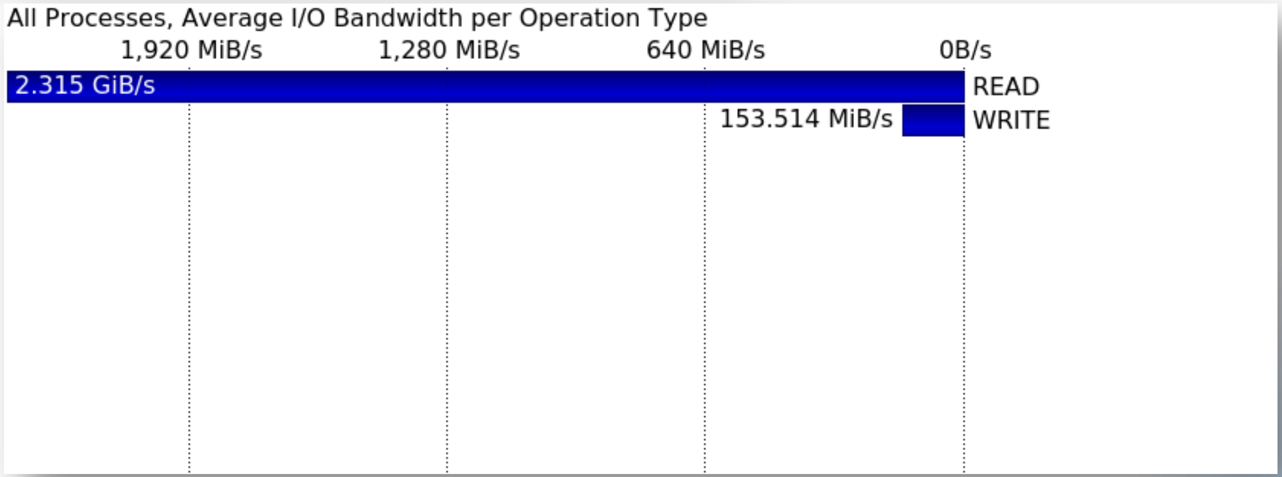
I/O Data Rate of single thread

I/O summaries with totals

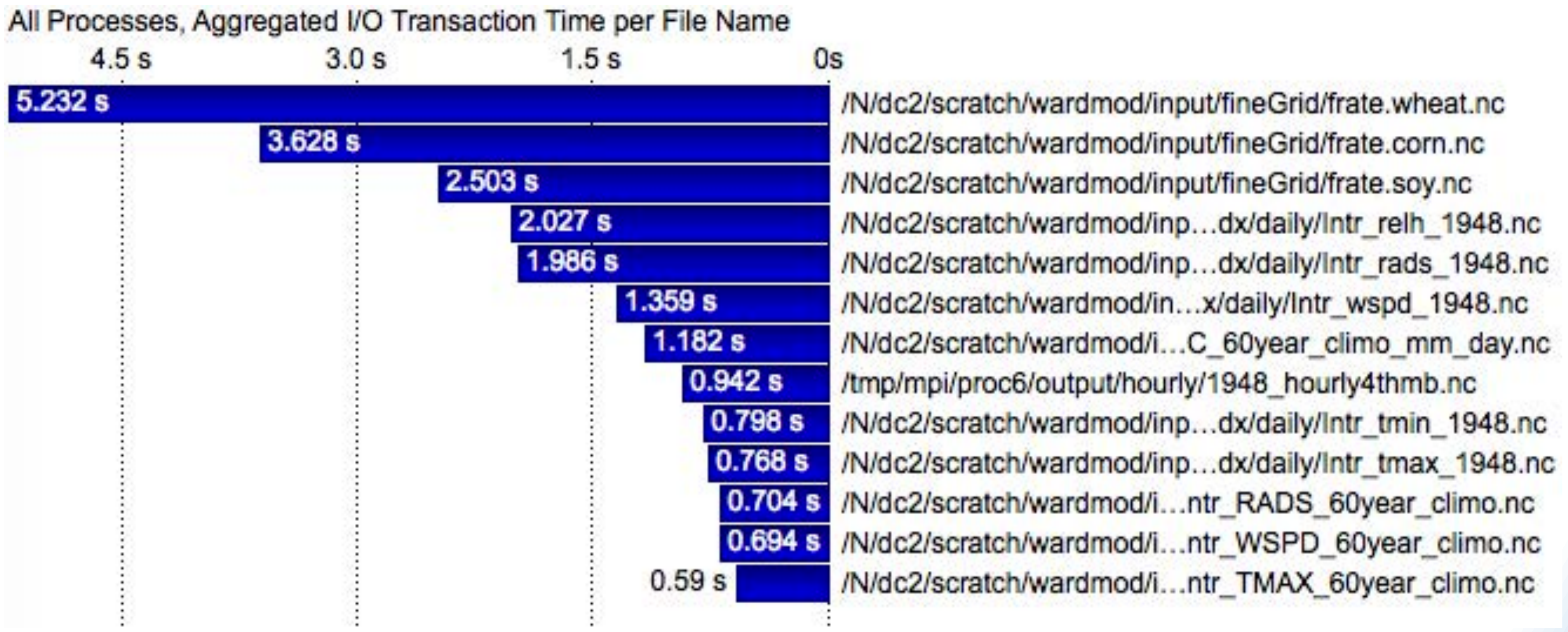


Other Metrics:

- IOPS
- I/O Time
- I/O Size
- I/O Bandwidth

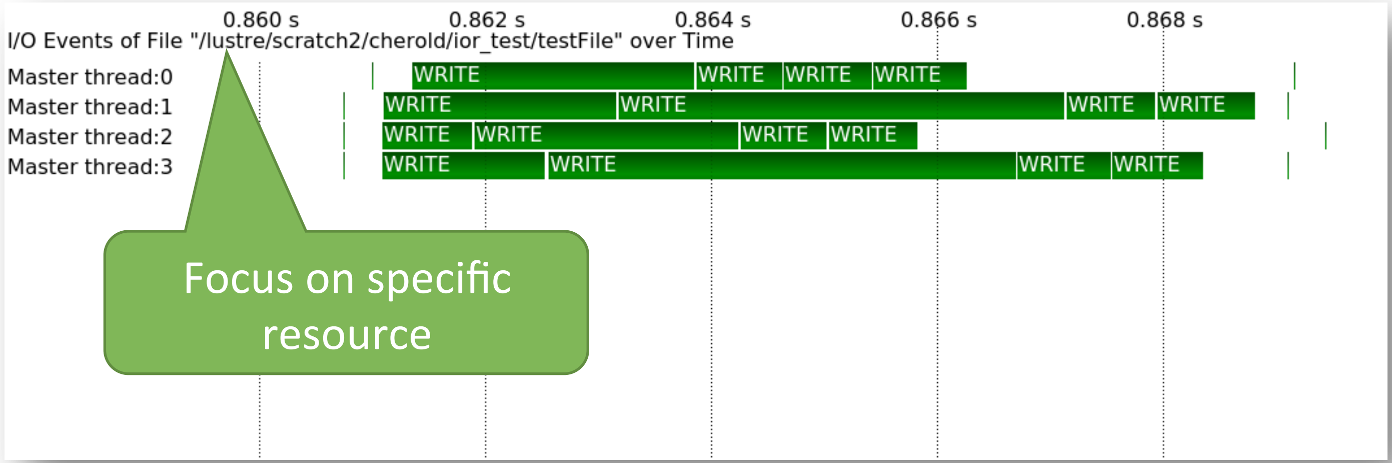


I/O summaries per file

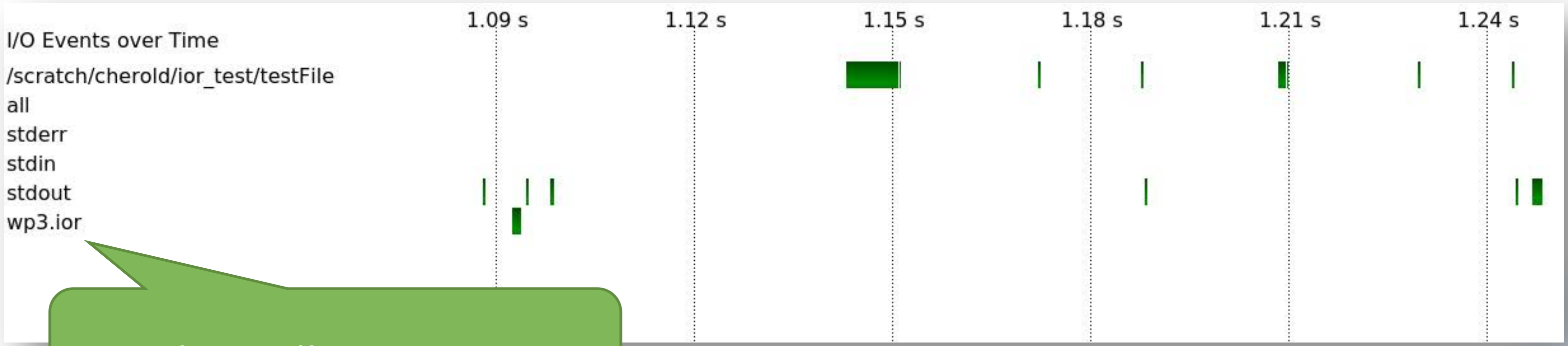


Missing: timeline view per file

I/O operations per thread or per file

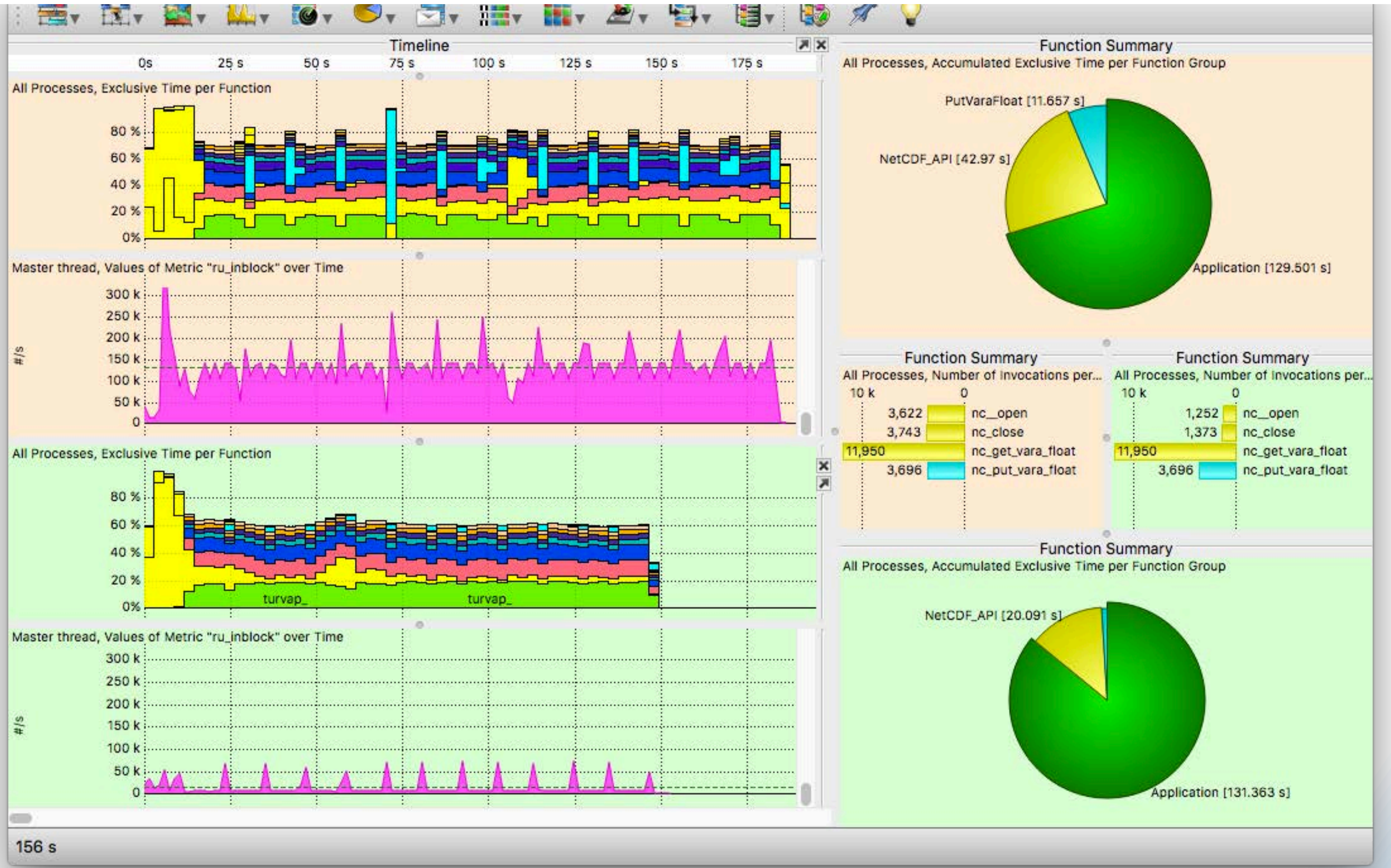


Focus on specific resource



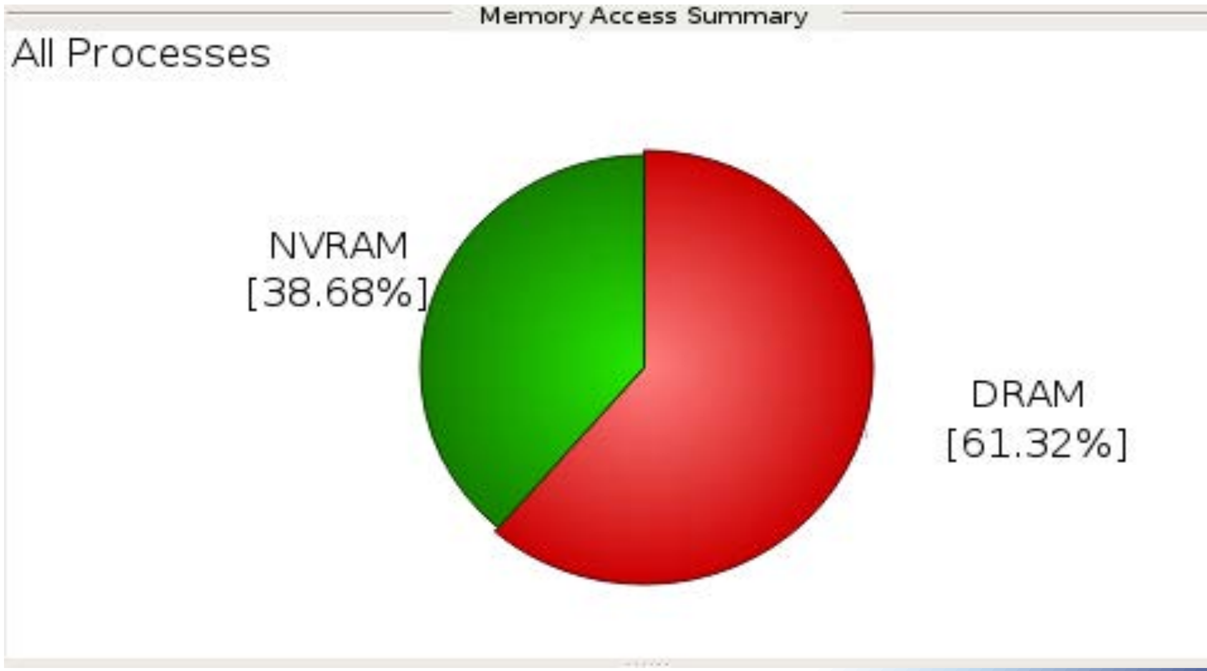
Show all resources

I/O comparison example

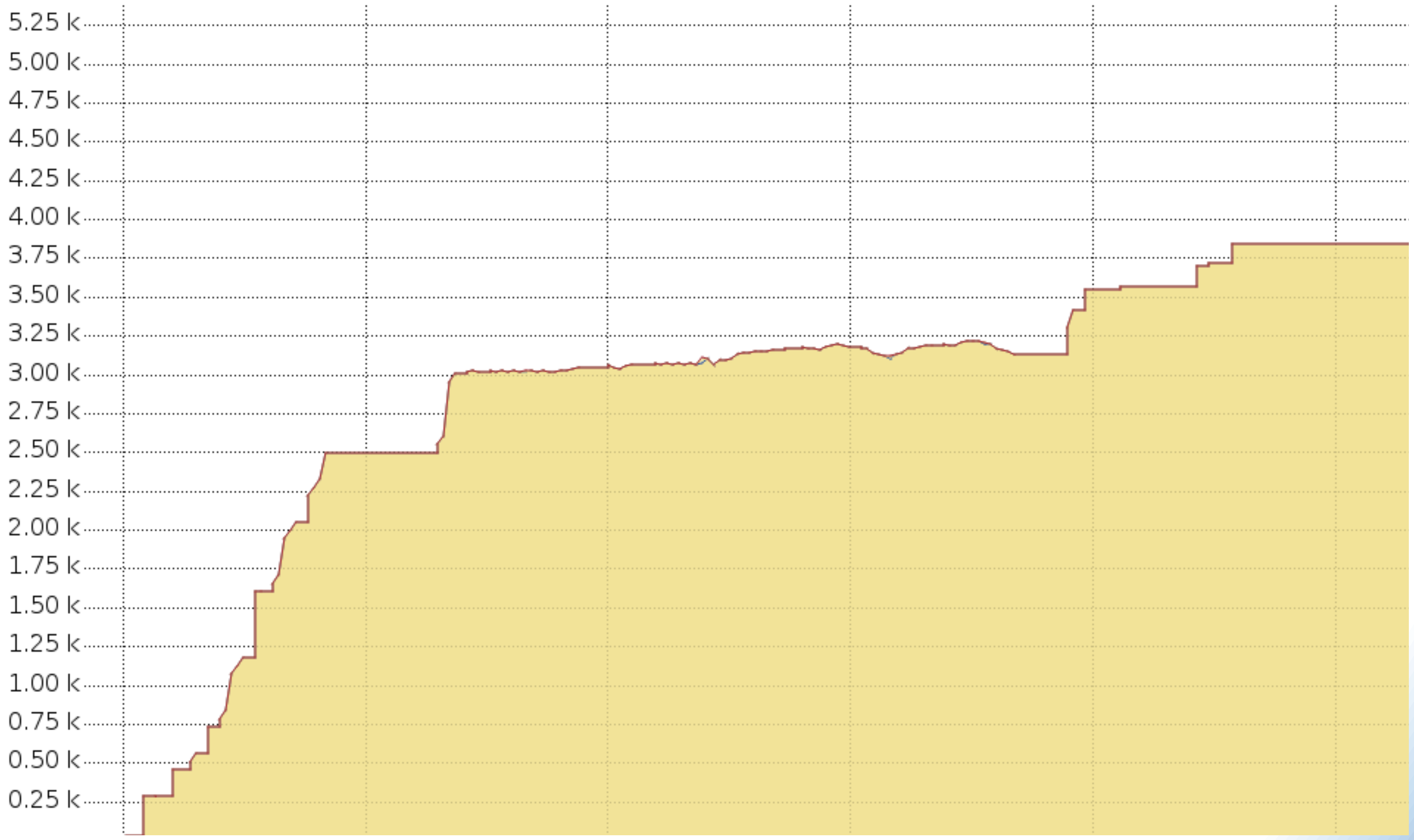


Visualization of memory access statistics

- Currently, Vampir has many different views presenting counter metrics
- Future: Absolute number of memory accesses performed by an application for different types of memory



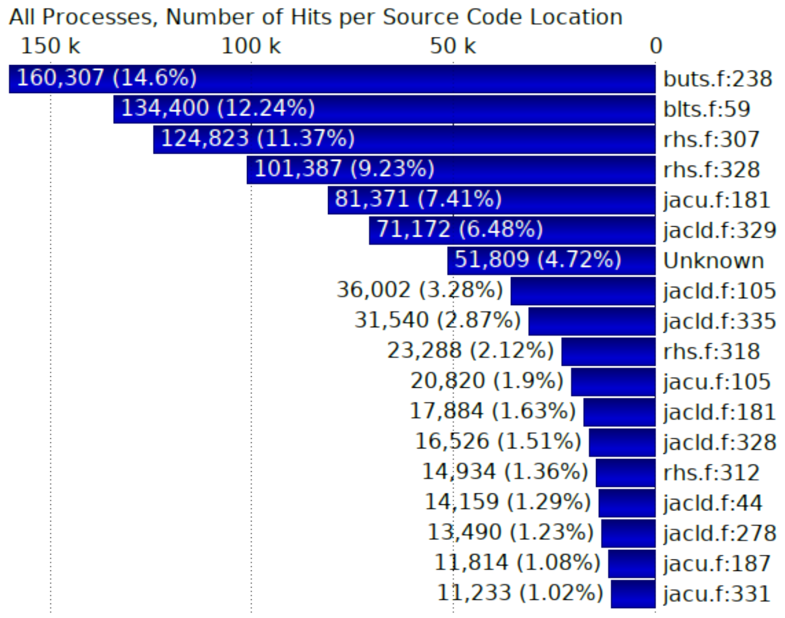
NVRAM allocation over time

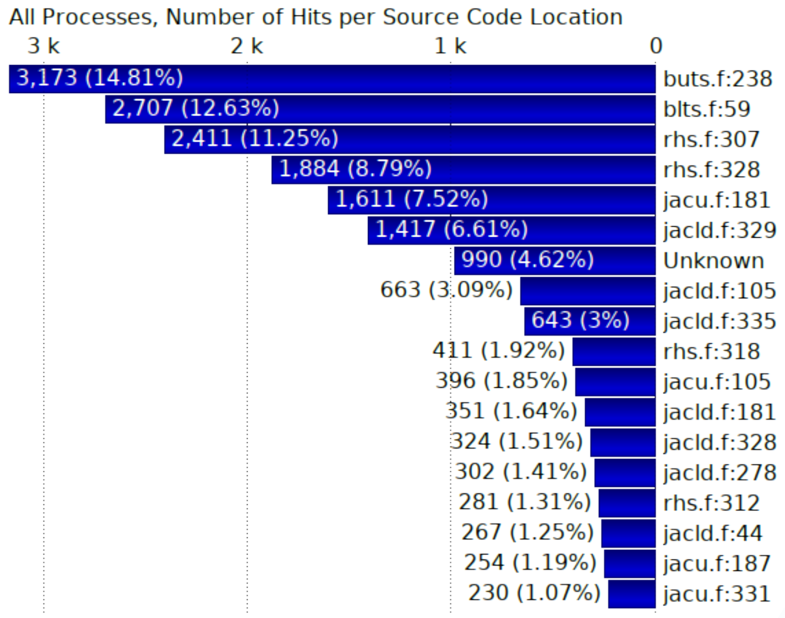


NVRAM accesses over time?

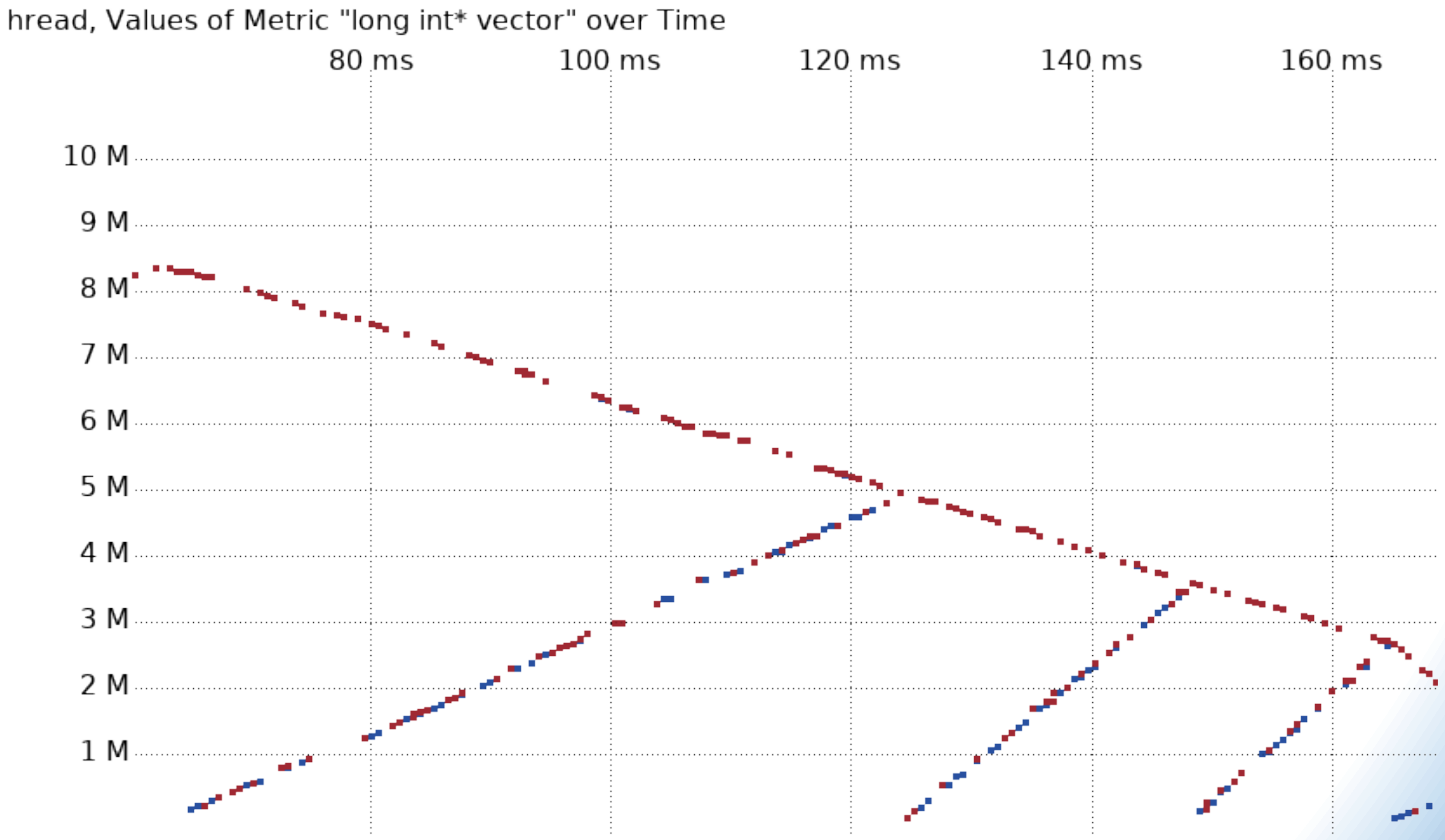
Counters?

Future Stats





Future Stats



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