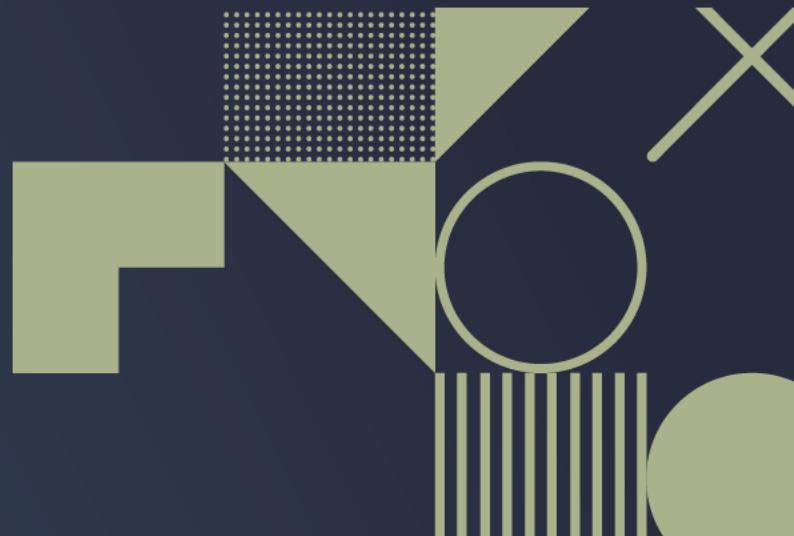
OPTIMISING I/O USING NON-VOLATILE MEMORY

Adrian Jackson

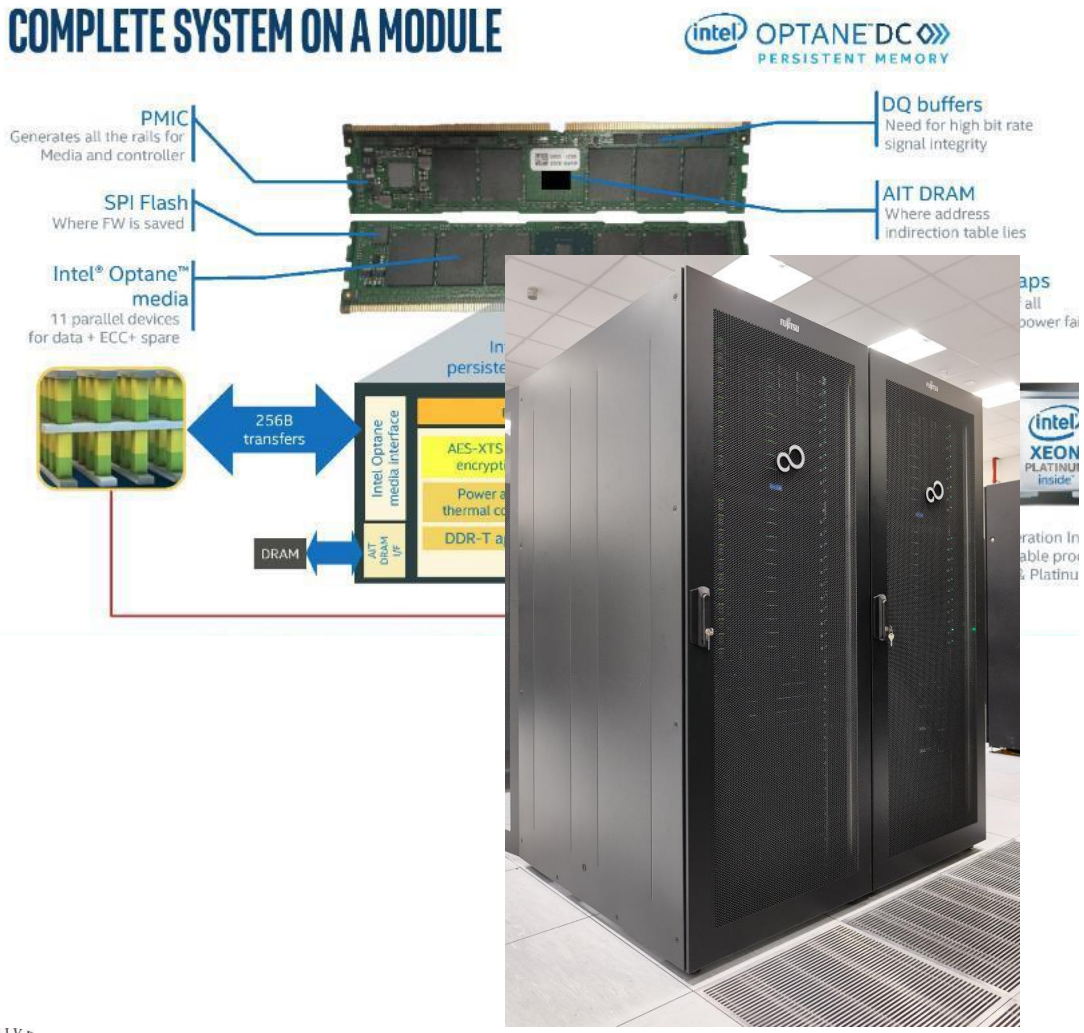
EPCC, The University of Edinburgh

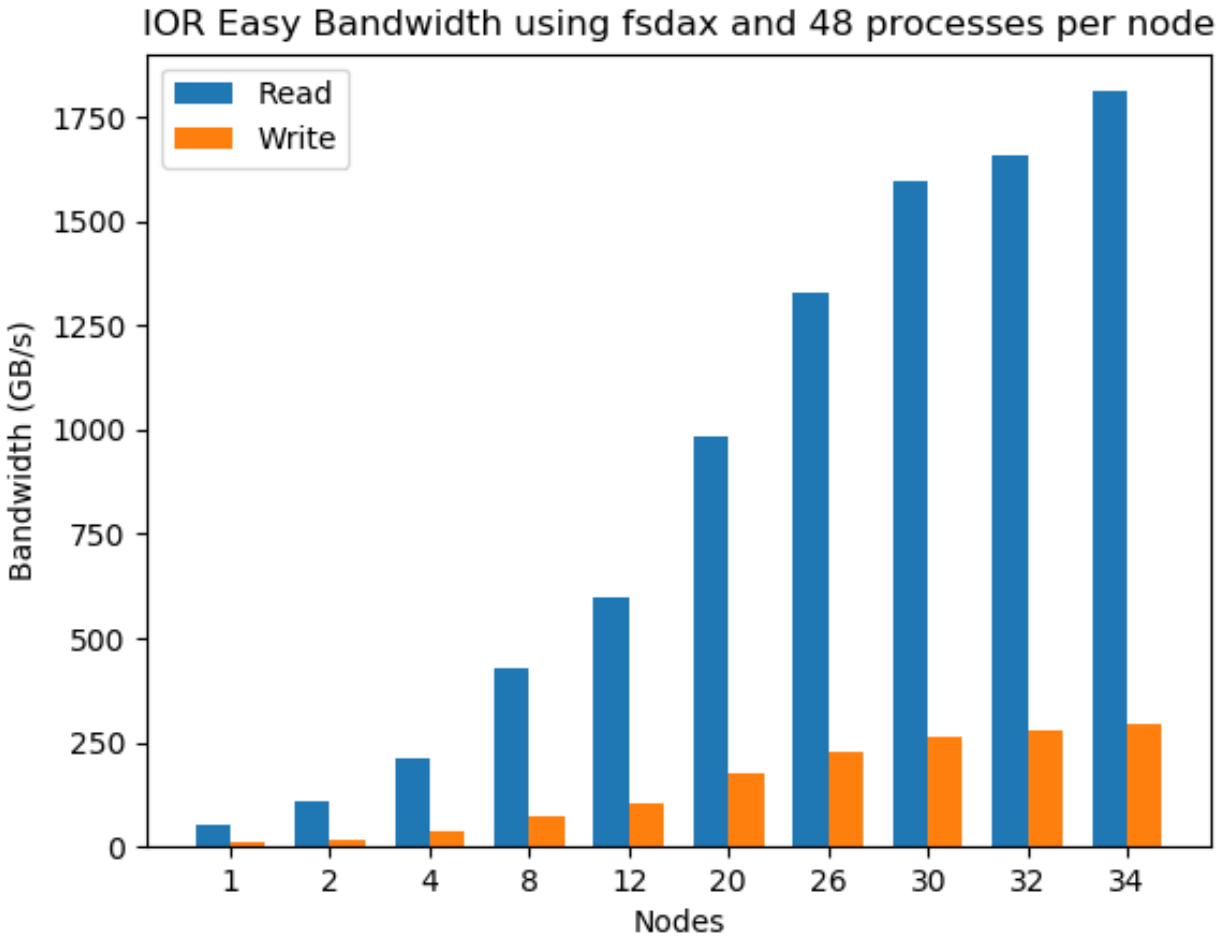
a.jackson@epcc.ed.ac.uk

@adrianjhpc



Non-volatile memory/B-APM



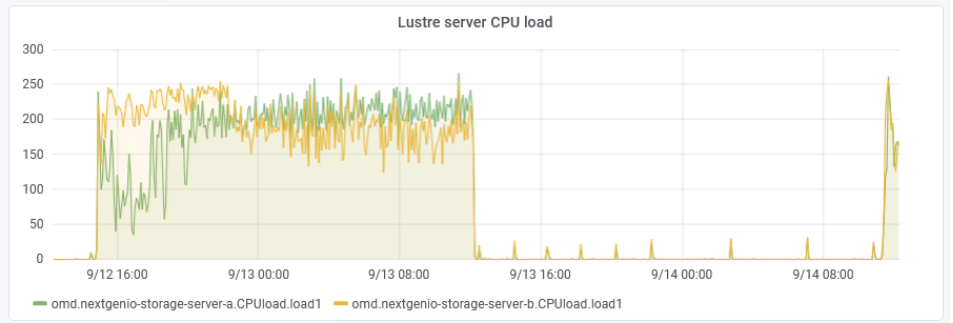


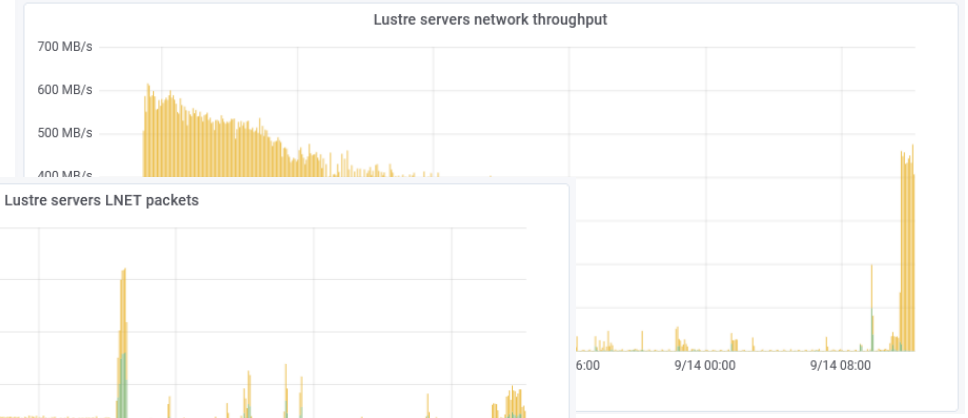
N3D/SEMTEX

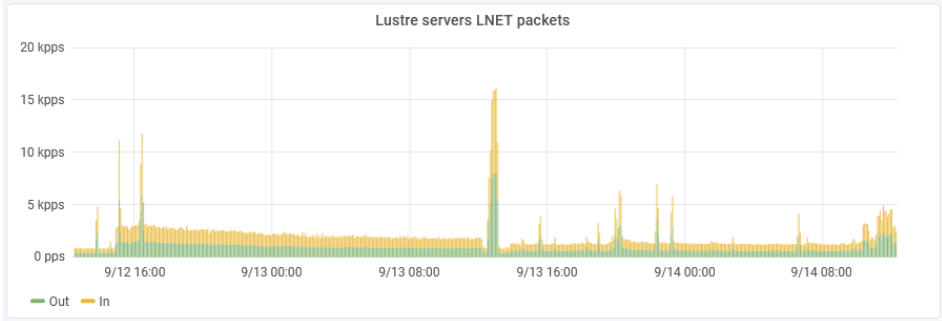
- n3d CFD application that uses combined forward/adjoint method
  - DNS used for Navier Stokes forward approach
  - Adjoint method requires full DNS output
  - DNS state is very large
- Medium simulation
  - 72 processes maximum
  - DNS state requires 4TB for storage
- Large simulation
  - 512 processes maximum
  - DNS state requires 40TB for storage
- Filesystem used to store data for the transition between phases

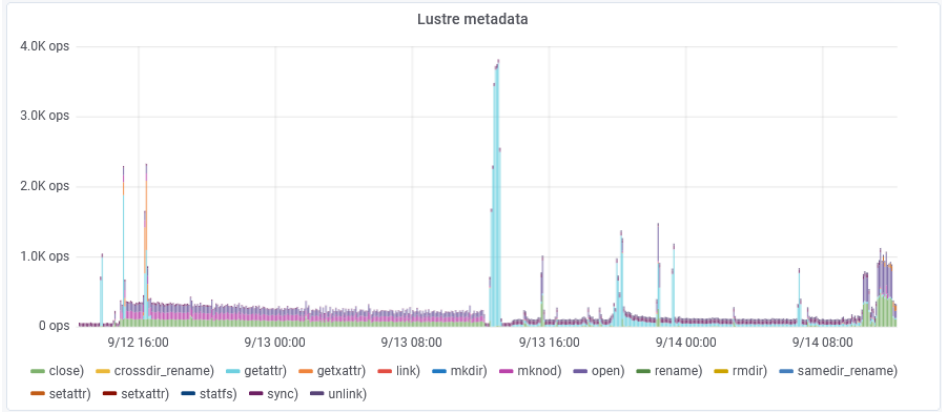
N3D/SEMTEX

- Small test case:
  - 72 processes
  - 900,000 files, 4.5 TBs produced
- Larger test case:
  - 512 processes
  - 6,400,000 files, 30 TBs produced
- Files required to transfer data from the forward phase to the adjoint phase
  - Velocity on each process at each time step

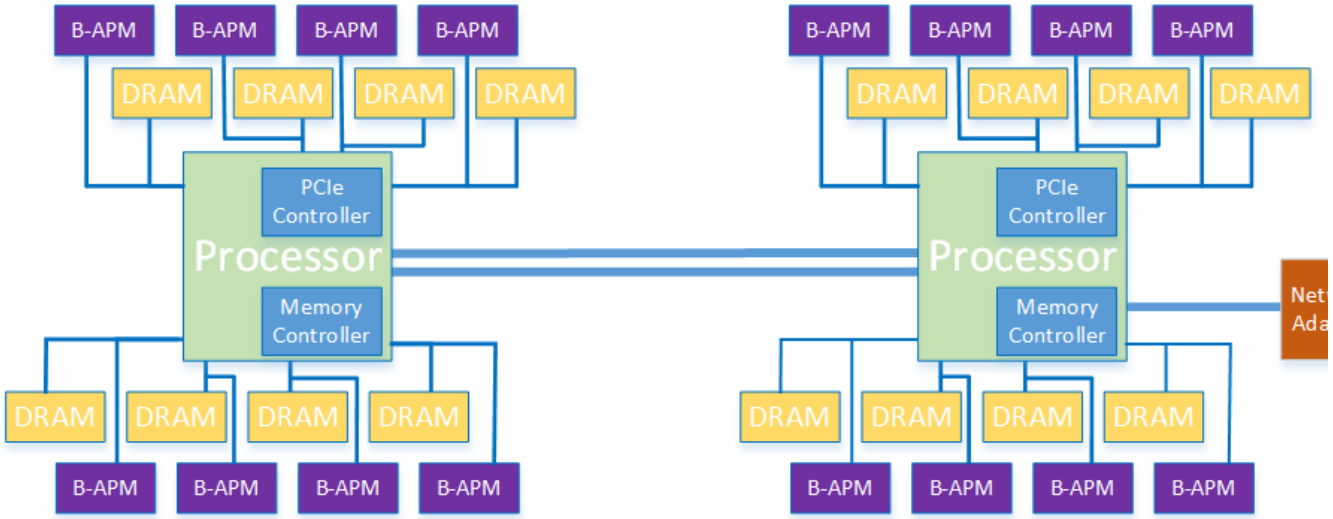




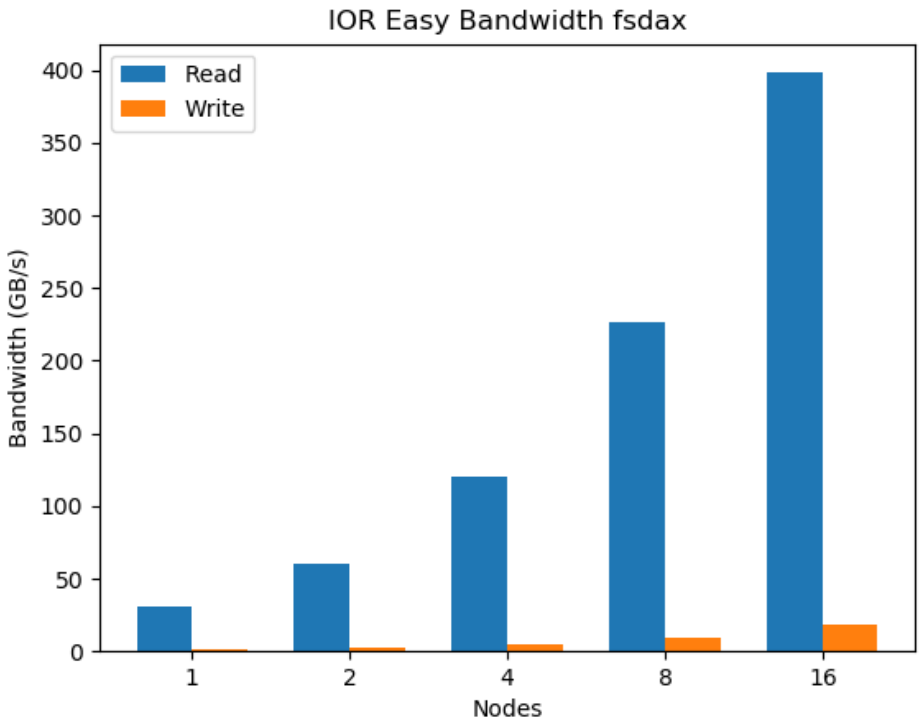




NUMA issues

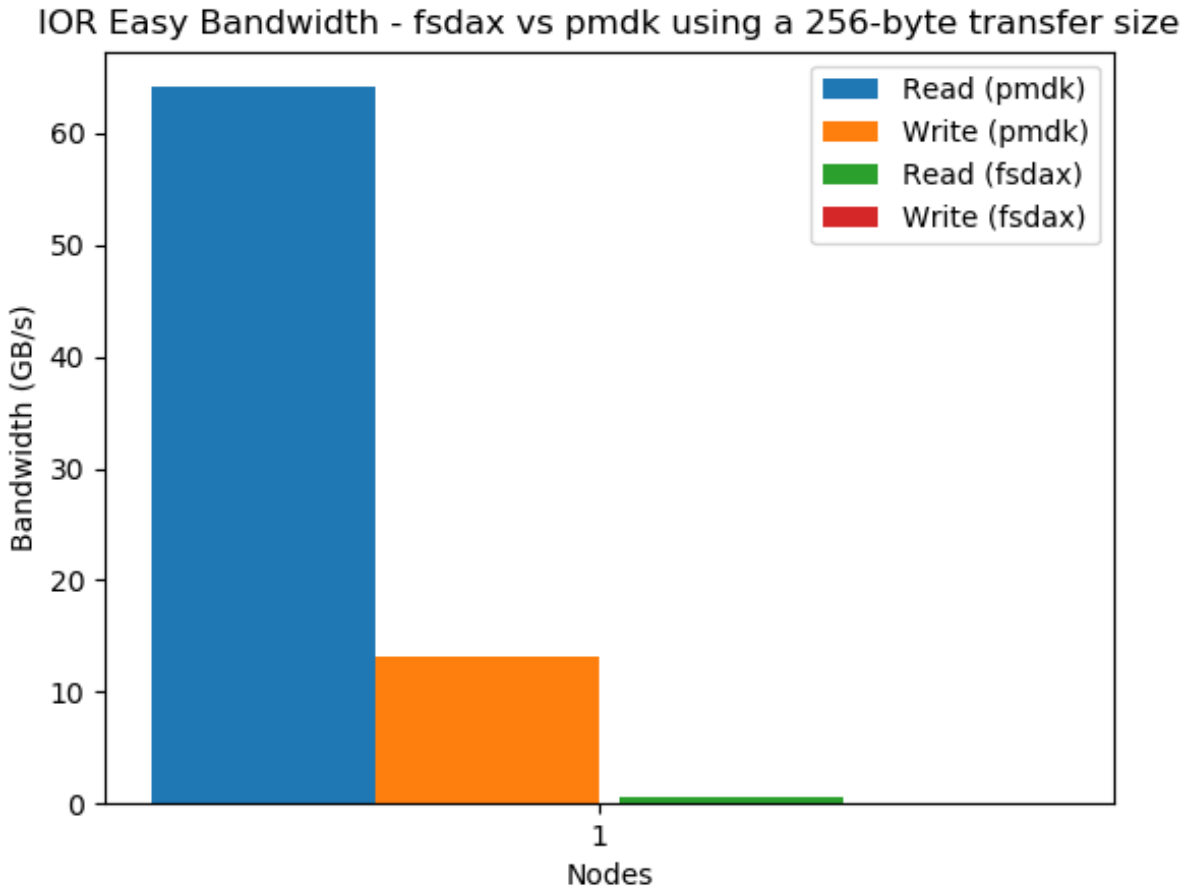


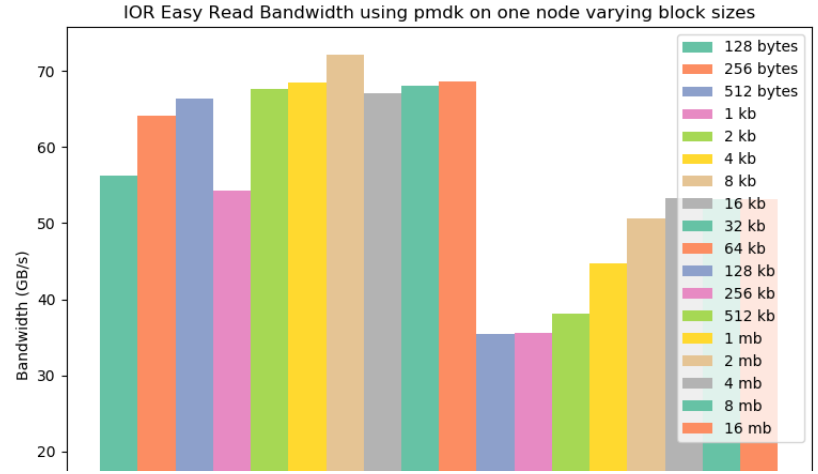

```
unsigned long get_processor_and_core(int *socket
  unsigned long a,d,c;
  __asm__ volatile("rdtscp" : "=a" (a), "=d" (d)
  *socket = (c & 0xFFF000)>>12;
  *core = c & 0xFFF;
  return ((unsigned long)a) | (((unsigned long)d
}
strcpy(path,"/mnt/pmem_fsdax");
sprintf(path+strlen(path), "%d", socket/2);
sprintf(path+strlen(path), "/");
```

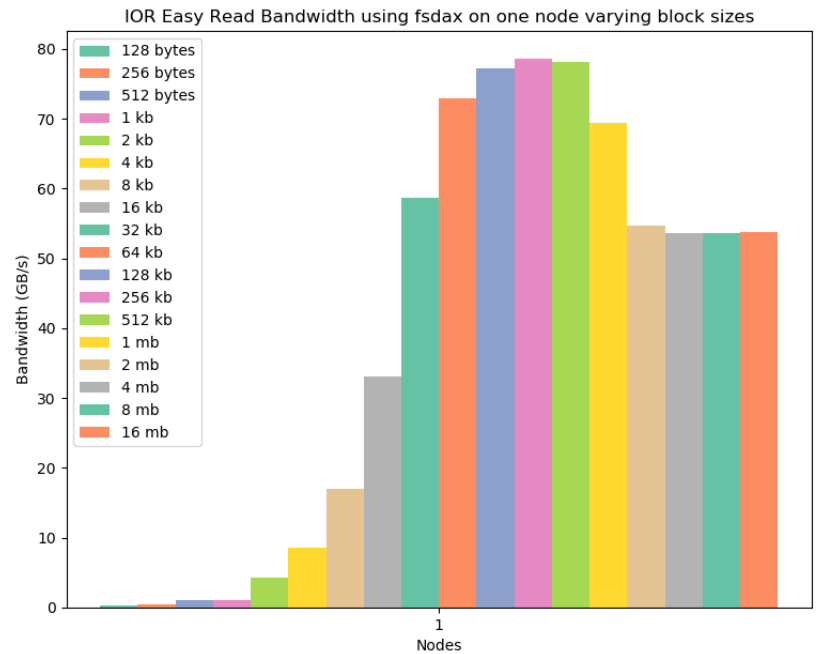


N3D/SEMTEX

- Assuming compute nodes with 256GB DRAM, to fit in DRAM
  - Medium case would require a minimum of 16 nodes
  - Large scale would require a minimum of 160 nodes
- Using filesystem (Lustre) takes:
  - Medium case using 3 nodes: ~9800 seconds
  - Large case using 22 nodes: ~80000 seconds
- Using persistent memory for I/O on the nodes
  - Medium case using 3 nodes: ~8500 seconds (~15% faster)
  - Large case using 22 nodes: ~9200 seconds (~90% faster)
- Using persistent memory as memory on the nodes
  - Medium case using 3 nodes: ~8300 seconds
  - Large case using 22 nodes: ~9000 seconds


IOR Easy Bandwidth - fsdax vs pmdk using a 256-byte transfer size
Read (pmdk)
Write (pmdk)
Read (fsdax)
Write (fsdax)
Bandwidth (GB/s)
0
10
20
30
40
50
60
1
Nodes


IOR Easy Read Bandwidth using pmdk on one node varying block sizes
128 bytes
256 bytes
512 bytes
1 kb
2 kb
4 kb
8 kb
16 kb
32 kb
64 kb
128 kb
256 kb
512 kb
1 mb
2 mb
4 mb
8 mb
16 mb
Bandwidth (GB/s)
20
30
40
50
60
70


IOR Easy Read Bandwidth using fsdax on one node varying block sizes
128 bytes
256 bytes
512 bytes
1 kb
2 kb
4 kb
8 kb
16 kb
32 kb
64 kb
128 kb
256 kb
512 kb
1 mb
2 mb
4 mb
8 mb
16 mb
Bandwidth (GB/s)
0
10
20
30
40
50
60
70
80
1
Nodes