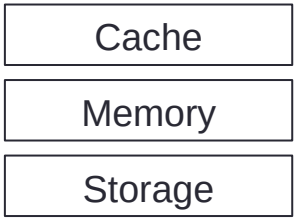
Evaluating the latest Optane memory: A glorious swansong?

Adrian Jackson

a.jackson@epcc.ed.ac.uk

Hierarchical memory potential

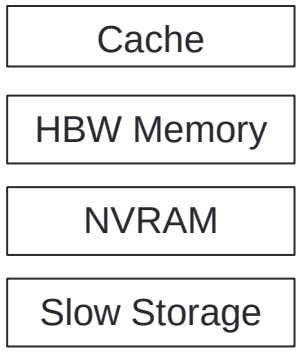
- Application performance key to users and developers
  - Very few systems are application specific
- Multi-purpose, multi-user systems require hardware choices
  - Processor, memory, accelerator, storage
  - Optimising for a range of applications hard



- A64FX one end of the spectrum
  - Small memory footprint for high performance/energy balance
- HPE Superdome flex the other end of the spectrum
  - Very large memory footprint for shared memory/non-scaling applications

Hierarchical memory potential

- Persistent memory provides scope to optimise DRAM usage and I/O performance
  - Support low volume high performance memory
  - Support very high performance I/O
  - Enable application specialisation for memory performance
  - Requires Byte-Addressable Persistent Memory (B-APM)
- Multi-tiered memory configurations
  - 3 tier memory
    - HBM – DRAM – B-APM
  - 2 tier memory
    - HBM – B-APM



I/O Optimisation with persistent memory

- 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

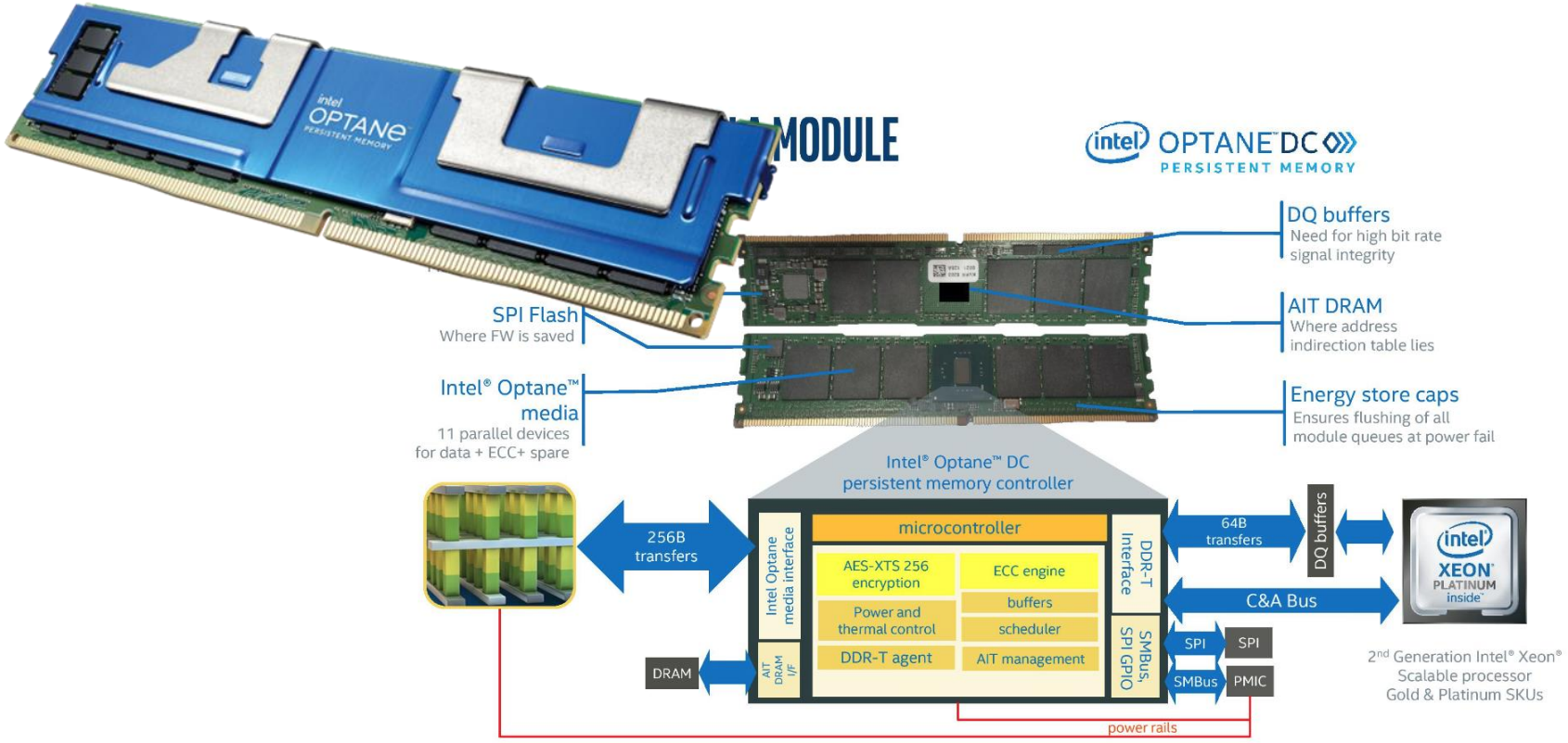
I/O Optimisation with persistent memory

- 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

Memory Persistent Memory

- I/O uses file operations
  - Kernel writing to O/S buffer, operating system writes that back to the file
  - Potential for O/S caching
  - Writes data in large chunks, bad for random access
  - Requires interaction with O/S
  - I/O consistency application responsibility
    - Flush required to ensure actual persistency
- Required because of the nature of previous I/O devices
  - Asynchronous
  - I/O controller
  - Shared
- PMDK/PMEM approach
  - Use as memory
  - No interaction with kernel outside standard malloc/free
  - Byte (cache line) granularity

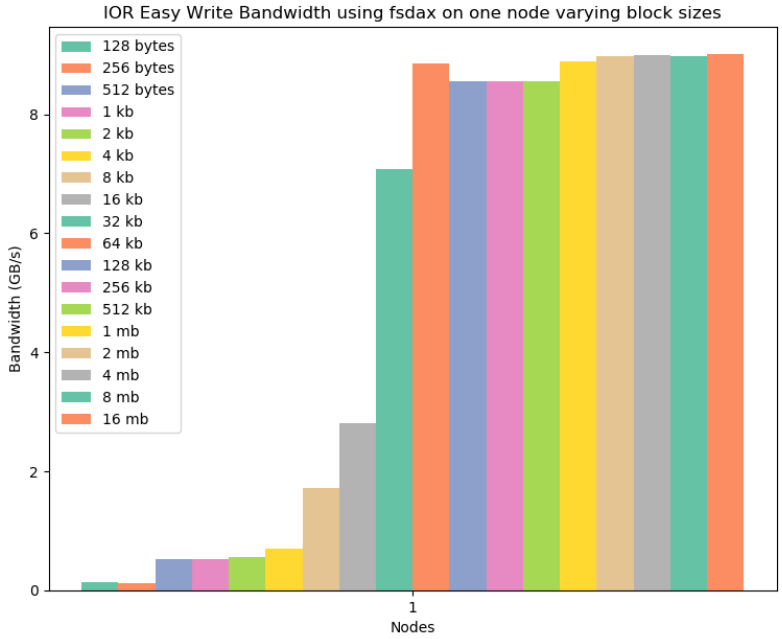
Optane Memory

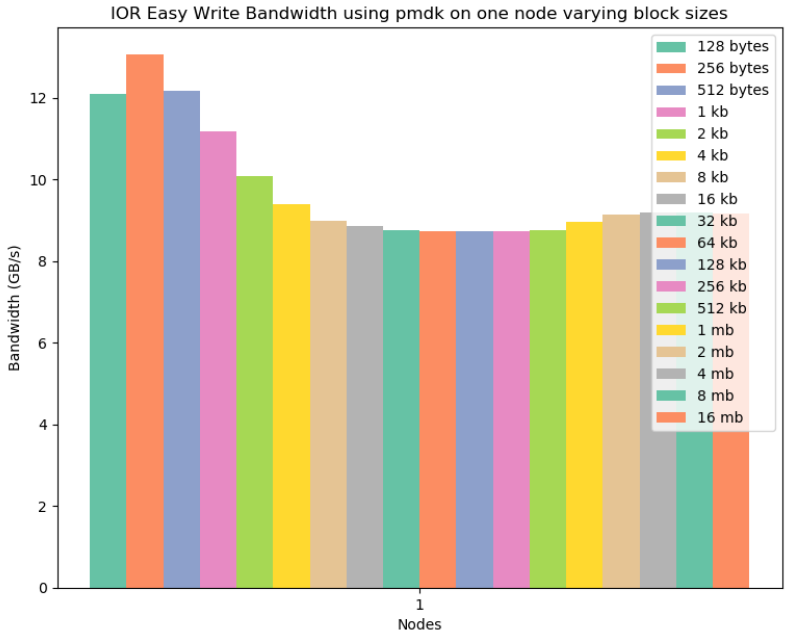


- Higher capacity than DRAM
- Lower performance than DRAM

Optane memory

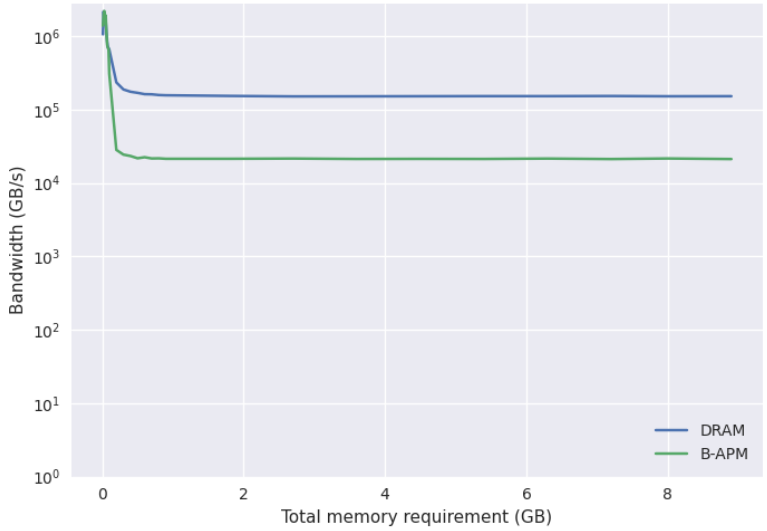
- Using persistent memory to reduce resident set size
  - Shrink architectural memory requirements





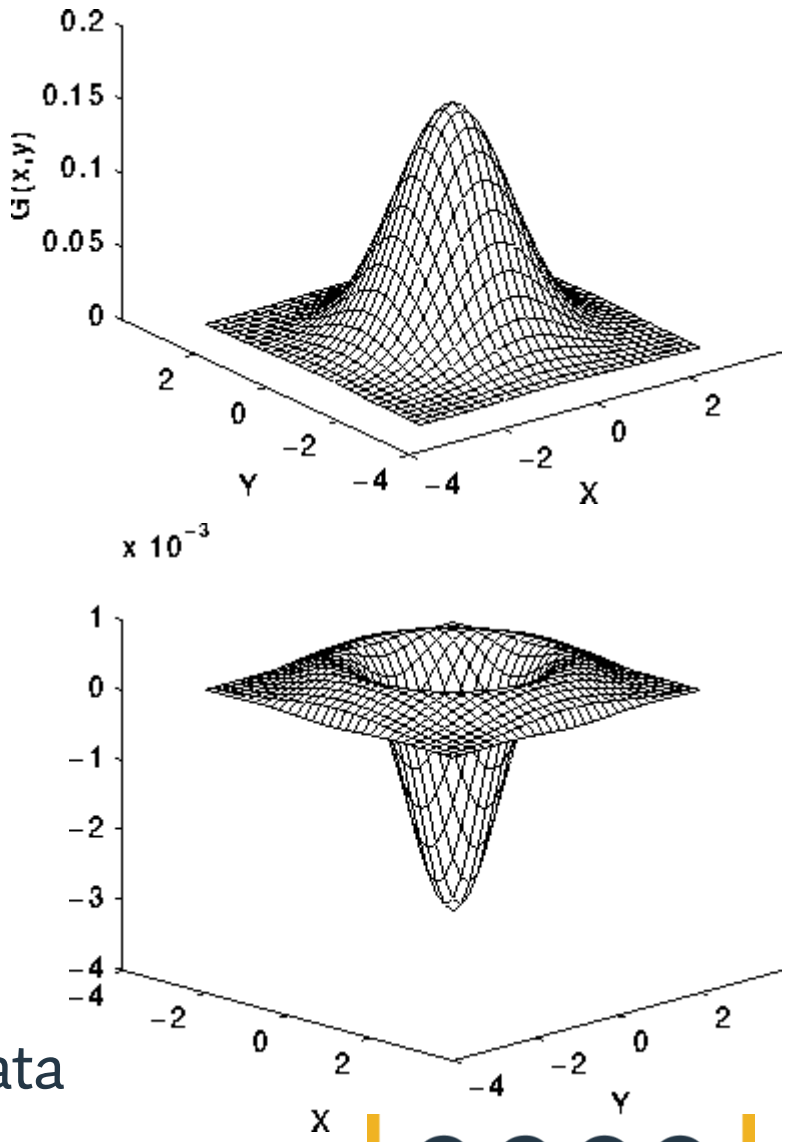
https://github.com/adrianjhpc/DistributedStream.git

Mode	Min BW (GB/s)	Median BW (GB/s)	Max BW (GB/s)
App Direct (DRAM)	142	150	155
App Direct (DCPMM)	32	32	32
Memory mode	144	146	147
Memory mode (large)	12	12	12



Multi-level memory exploitation

- Simple image sharpening stencil
  - Each pixel replaced by a weighted average of its neighbours
  - weighted by a 2D Gaussian
  - averaged over a square region
  - we will use:
    - Gaussian width of 1.4
    - a large square region
  - then apply a Laplacian
    - this detects edges
    - a 2D second-derivative ∇^2
- Combine both operations
  - produces a single convolution filter
- 4 similar sized arrays, two that are updated and two that are source data



```
address = (int **) malloc(nx*sizeof(int *) + nx*ny*sizeof(int));
fuzzy = int2D(nx, ny, address);
```

```
pmemaddr1 = pmem_map_file(filename, array_size,PMEM_FILE_CREATE|PMEM_FILE_EXCL,
                          0666, &mapped_len1, &is_pmem)
fuzzy =  int2D(nx, ny, pmemaddr1);


int **int2D(int nx, int ny, int **idata){
  int i;
  idata[0] = (int *) (idata + nx);

  for(i=1; i < nx; i++){
      idata[i] = idata[i-1] + ny;
    }

  return idata;
}
```

- Read-only data in DRAM

```
Calculation time was 1.426152 seconds

DRAM required 22GB

Calculation time was 25.436569 seconds

DRAM required 280GB
```

- Read-only data in Persistent Memory

```
Calculation time was 1.431711 seconds

DRAM required 15GB

Calculation time was 28.709221 seconds

DRAM required 160GB
```

Comparing performance

- Looking at Optane performance between 1st and 3rd generation
  - 1st generation: NEXTGenIO system
    - Intel Cascade Lake (48 cores)
    - Dual socket
    - 3TB Optane memory
    - 192GB DDR4
  - 3rd generation: Pegasus system
    - Intel Sapphire Rapids
    - Single socket
    - 2TB Optane
    - 128GB DDR5

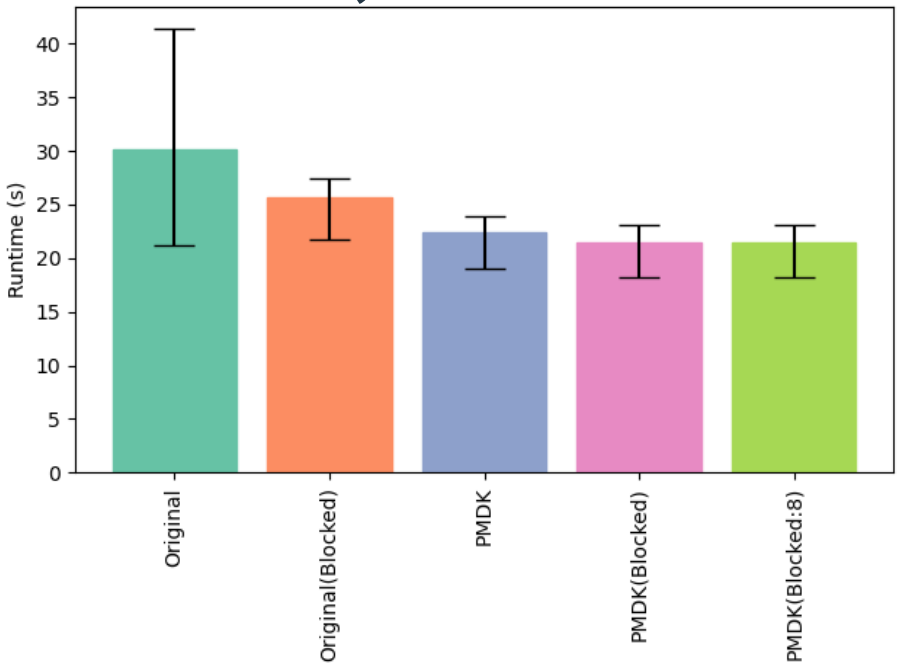
Memory Location	Copy (GB/s)	Scale (GB/s)	Add (GB/s)	Triadd (GB/s)
All in DRAM	136	135	152	154
Read data in Optane Write data in DRAM	80	80	79	80
Read data in DRAM Write data in Optane	16.7	16.7	25.0	24.9
All in Optane	15.5	15.1	21.2	21.4

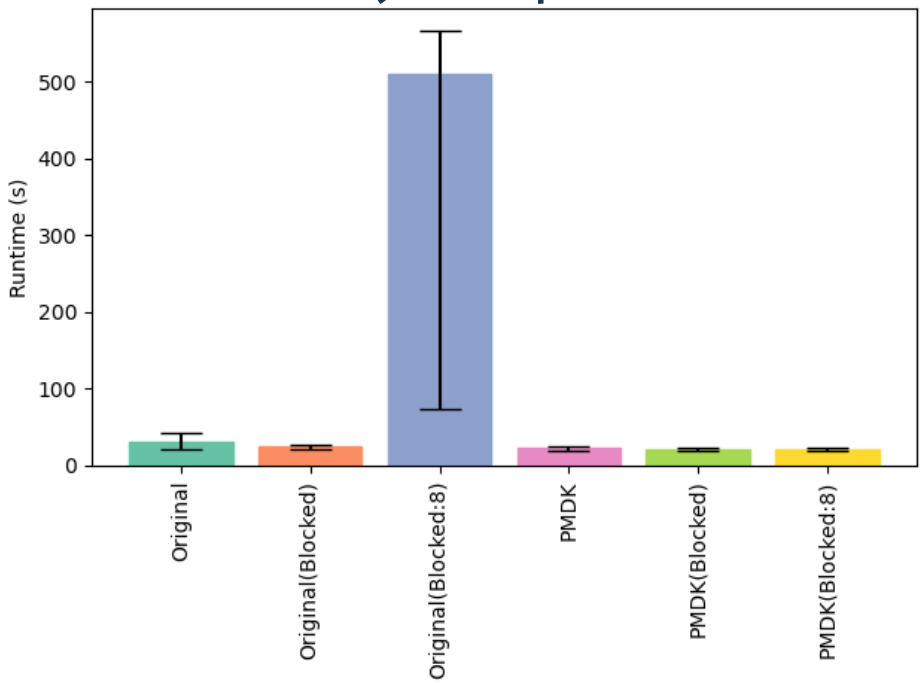
https://github.com/adrianjhpc/DistributedStream

Memory Location (persist horizon)	Copy (GB/s)	Scale (GB/s)	Add (GB/s)	Triadd (GB/s)
All in DRAM	172	172	187	186
Read data in Optane Write data in DRAM	135	135	112	112
Read data in DRAM Write data in Optane	29	29	41	41
All in Optane (no persist)	23	24	28	28
All in Optane (persist at end)	23	23	28	28
All in Optane (persist each write)	3	3	4	4

MAD2Bench

- Port application to use smaller data domain and swap out with persistent memory
  - MAD2Bench was I/O benchmark, extended to memory
  - Heavily blocked to reduce active memory footprint





Application porting

- Using Optane for (simple) applications
  - Custom configuration of which data structures to store in Optane
  - Chose applications that have large read only data structures

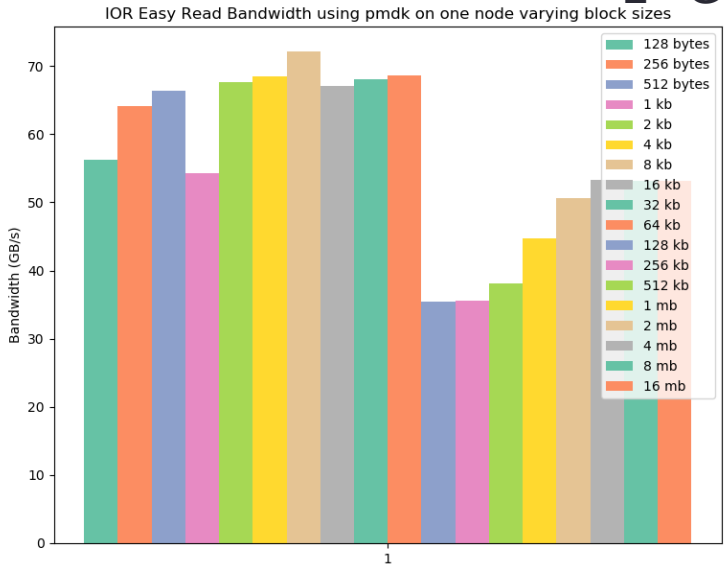
Application	Hardware	Runtime (seconds)	Volatile Memory (GB)
sharpen	DRAM	56	285
	DRAM+B-APM	63	170
cfd	DRAM	7.95	40
	DRAM+B-APM	9.64	25

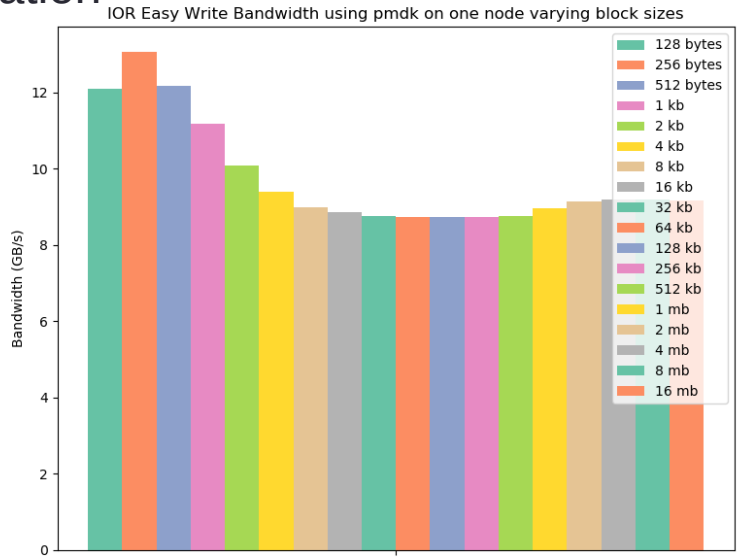
Application porting

- Further memory reductions possible
  - Tuning the persistent horizon
  - Moving more data to the Optane
  - Trade off time for memory

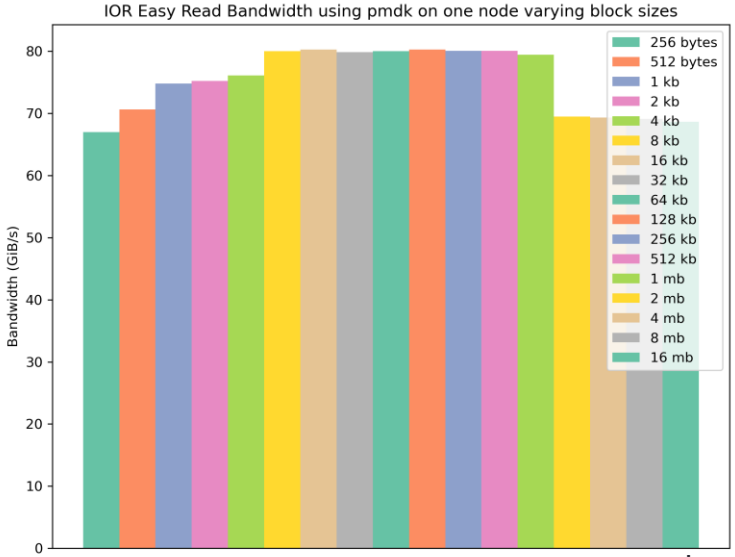
Persistency Horizon	Hardware	Runtime (seconds)	Volatile Memory (GB)
N/A	DRAM	7.95	40
No persist	DRAM+B-APM	9.64	25
Partial persist	DRAM+B-APM	10.67	25
Full persist	DRAM+B-APM	41.84	2

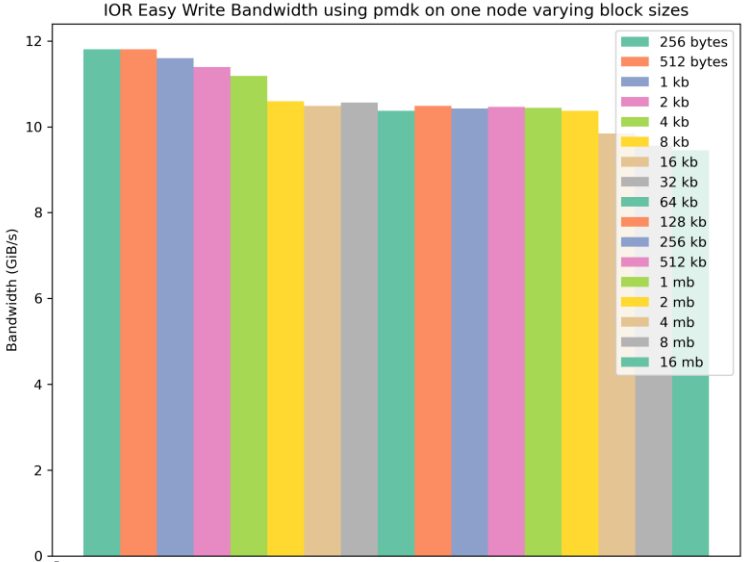
1st Generation





3rd Generation





3rd generation Optane memory

- Strong performance improvement on Optane
  - Especially for sympathetic access patterns
- Architectural opportunities still apparent
  - No replacement for this low latency/small I/O size memory functionality
  - cxl memory won't provide equivalent
- Memory controller lock-in was a big issue
  - What Intel thought was an advantage probably killed the product
- Out-of-(volatile)memory algorithms/implementation still interesting to consider
- Allowing small data movements (I/O or memory) could let applications be redesigned
  - Object store allows this on the I/O side
  - Maybe HBM-X + DRAM allows this for future architectures
    - At an energy cost