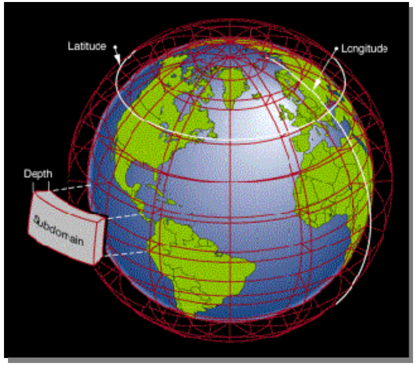
Introduction to Parallel I/O

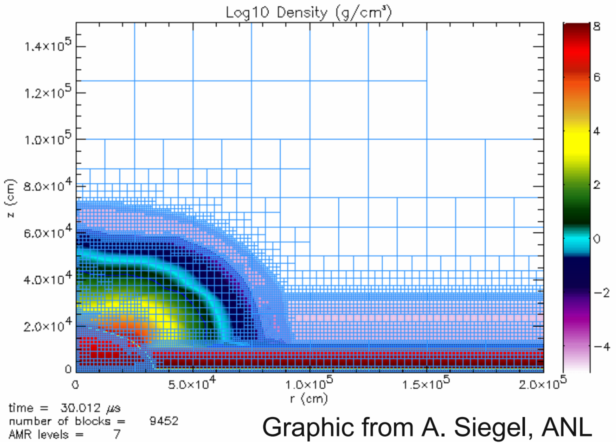
Stefano Markidis
KTH Royal Institute of Technology

Application I/O - I

- Applications have data models appropriate to domain
  - Multidimensional typed arrays, images composed of scan lines, variable length records
  - Headers, attributes on data I/O system as a whole must:
    - **Provide mapping of application data into storage abstractions**
    - **Coordinate access by many processes**
    - **Organize I/O devices into a single space**



Graphic from J. Tannahill, LLNL



Graphic from A. Siegel, ANL

Application I/O - II

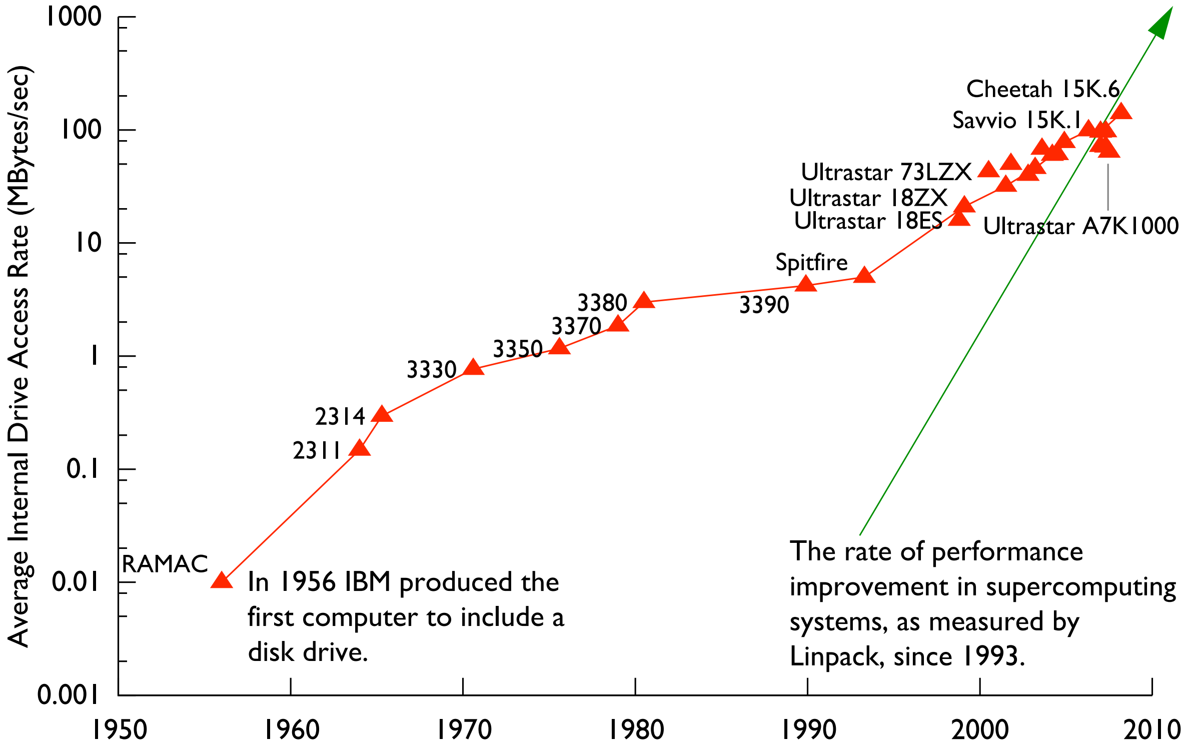
Scientific applications need persistent storage

- Typical to store persistent data in *files*, accessed through input/output (I/O) features of programming language and runtime
- Dominant implementation for persistent storage is **magnetic disks**
    - Tape also used for higher capacity
    - Semiconductor and other technologies used for higher performance/lower power (e.g., FLASH)

The Performance Problem with Single I/O

- Magnetic disks performance
  - Latency 2-10 ms (time it take the disk to spin under the read/write head)
  - 1,000x slower than internode communication
    - 10,000,000x slower than processor core
  - Bandwidth over 100MB/s
    - But only approached for large transfers
- Performance sensitive to exact usage pattern

I/O vs Compute Trend

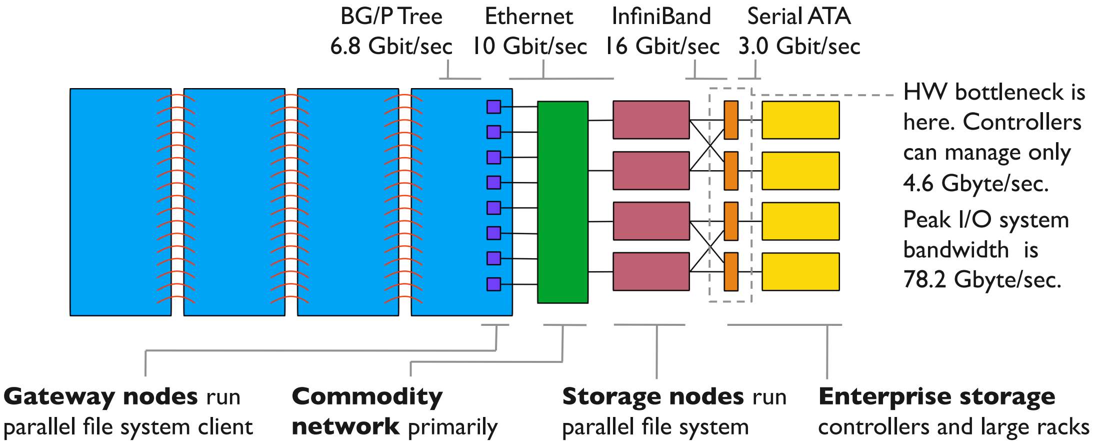


HPC I/O System is Also Rather Complex…

An HPC I/O system is attached to supercomputer

- The HPC I/O system is a supercomputer itself

Architectural diagram of 557 TF Argonne Leadership Computing Facility Blue Gene/P I/O system



Largest I/O Systems

2018

#	site.institution	site.storage system.net capacity	site.supercomputer.compute peak	site.supercomputer.memory capacity
		in PiB	*in PFLOPS*	*in TB*
1	Oak Ridge National Laboratory	250.04	220.64	3511.66
2	National Energy Research Scientific Computing Center	197.65	37.71	857.03
3	Los Alamos National Laboratory	72.83	11.08	2110.00
4	German Climate Computing Center	52.00	3.69	683.60
5	Lawrence Livermore National Laboratory	48.85	20.10	1500.00
6	RIKEN Advanced Institute for Computational Science	39.77	10.62	1250.00
7	National Center for Atmospheric Research	37.00	5.33	202.75
8	National Center for Supercomputing Applications	27.60	13.40	1649.27
9	Global Scientific Information and Computing Center	25.84	17.89	275.98
10	Joint Center for Advanced HPC	24.10	24.91	919.29

IO-500

This is the official *ranked* list from ISC-HPC 2018. The list shows the best result for every given combination of system/institution/filesystem (i.e. multiple submissions from the same system are not shown; only the most recent is shown). The full list is available here.

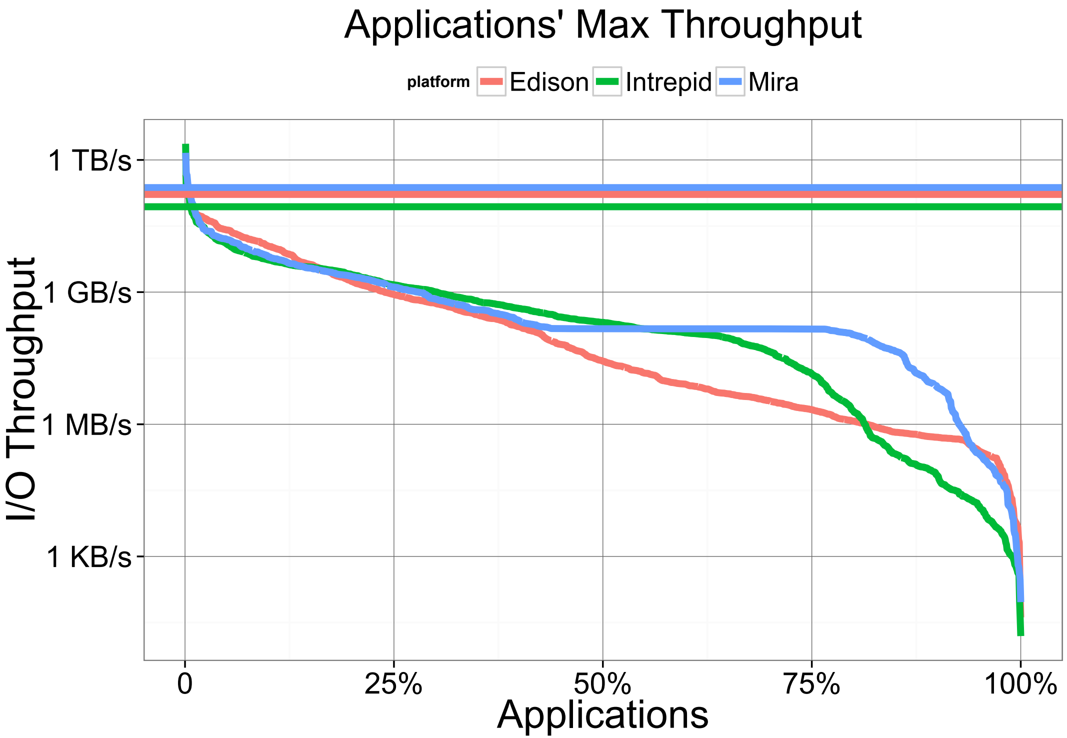
#	information						io500		
	system	institution	filesystem	storage vendor	client nodes	data	score	bw GiB/s	md kIOP/s
1	Oakforest-PACS	JCAHPC	IME	DDN	2048	zip	137.78	560.10	33.89
2	ShaheenII	KAUST	DataWarp	Cray	1024	zip	77.37	496.81	12.05
3	ShaheenII	KAUST	Lustre	Cray	1000		41.00*	54.17	31.03*
4	JURON	JSC	BeeGFS	ThinkparQ	8		35.77*	14.24	89.81*
5	Mistral	DKRZ	Lustre2	Seagate	100		32.15	22.77	45.39
6	Sonasad	IBM	Spectrum Scale	IBM	10	zip	24.24	4.57	128.61
7	Seislab	Fraunhofer	BeeGFS	ThinkparQ	24		16.96	5.13	56.14
8	Mistral	DKRZ	Lustre1	Seagate	100	zip	15.47	12.68	18.88
9	Govorun	Joint Institute for Nuclear Research	Lustre	RSC	24	zip	12.08	3.34	43.65
10	EMSL Cascade	PNNL	Lustre		126		11.12	4.88	25.33

https://www.vi4io.org/hpsl/2018/start

"A supercomputer is a device for converting a CPU-bound problem into an I/O bound problem."

[Ken Batcher]

The Reality …



A Multiplatform Study of I/O Behavior on Petascale Supercomputers, Luu, Winslett, Gropp, Ross, Carns, Harms, Prabhat, Byna, Yao. HPDC'15

I/O Software Stack

Applications (Weather Forecast, CFD, Astrophysics …)

High-Level I/O Level Libraries (HDF5, NetCDF, …)

I/O Middleware (MPI I/O)

POSIX I/O

I/O Parallel File system

I/O Hardware

I/O Parallel File system

I/O Hardware

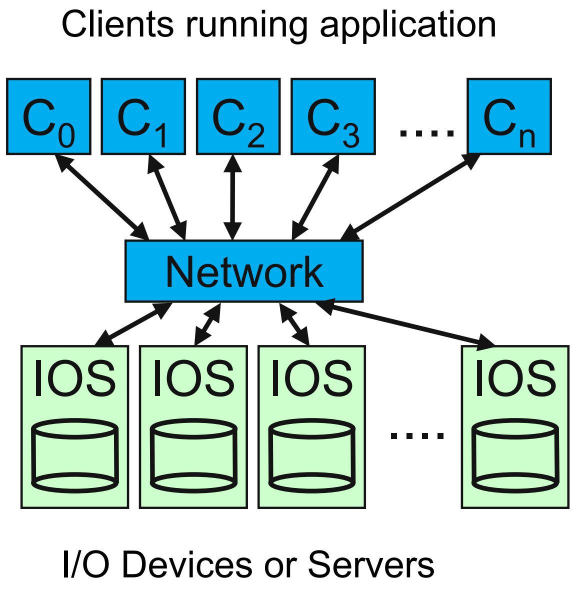
Files and File Systems

- A file is just an **ordered collection of bytes**
- A file system manages collections of files and properties of the files, also called *metadata*:
  - Size
  - Access restrictions
  - Quotas
  - Reading and writing data
- File systems differ
  - services they provide
  - the semantics of data access and update

Parallel File Systems

A parallel file system breaks up a data set and distributes, or stripes, the blocks to **multiple storage drives**, which can be located in **local and/or remote servers**.

- Users do not need to know the physical location of the data blocks to retrieve a file (global namespace)
- Parallel file systems often use a **metadata server** to store information about the data, such as the file name, location and owner.



Example of Parallel File Systems

- Two of the most prominent examples of parallel file systems are
  - **IBM's General Parallel File System** (**GPFS**) is a block-based parallel file system that uses blocks of tunable width and dynamic metadata for information distribution.
  - Open source **Lustre** file system. Lustre is an object-based parallel file system with file regions that can vary in length and static metadata for information distribution.

POSIX I/O

I/O Parallel File system

I/O Hardware

POSIX I/O

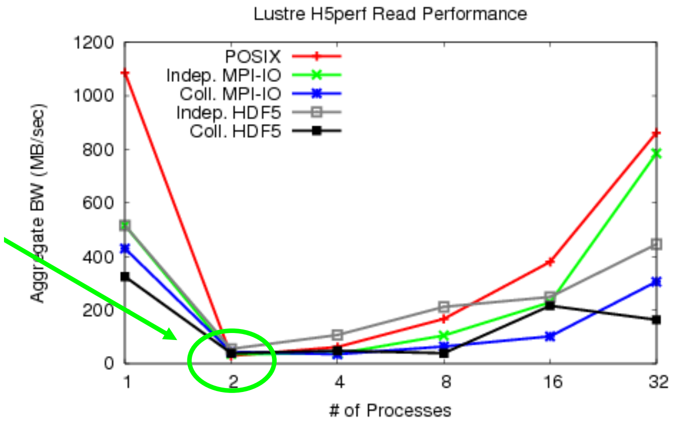
- POSIX is the IEEE Portable Operating System Interface for Computing Environments
  - POSIX defines a standard way for an application program to obtain basic services from the operating system
- Mechanism almost all serial applications use to perform I/O POSIX was created when a single computer owned its own file system
  - No ability to describe collective I/O accesses
- It can be very expensive for a file system to guarantee POSIX semantics for heavily shared files (e.g., from clusters)
  - Once a write completes (on any process), **any read, *from any other process*, must see that write**

POSIX I/O Example

```c
#include <fcntl.h>
#include <unistd.h>
int main(int argc, char **argv)
{
  //Integer “file descriptors” used to refer to open files

  int fd, ret;
  char buf[13] = "Hello World\n"; /* includes NULL */

  fd = open("myfile", O_WRONLY | O_CREAT, 0755);
  if (fd < 0) return 1;

  ret = write(fd, buf, 13);
  if (ret < 13) return 1;

  close(fd);

  return 0;

}
```

Under the covers of POSIX I/O

- POSIX API is a bridge between many tools and the file systems below
- Operating system maps these calls directly into file system operations
- File system performs I/O, using block- or region-oriented accesses depending on implementation
  - "Compliant" file systems will likely perform locking to guarantee atomicity of operations
  - Can incur substantial overhead
    - "Two Process Performance Tank" effect



POSIX Summary

- POSIX interface is a useful, ubiquitous interface for basic I/O
- Lacks any constructs useful for parallel I/O
- **Should not be used in parallel applications if performance is desired**

I/O Middleware (MPI I/O)

POSIX I/O

I/O Parallel File system

I/O Hardware

MPI I/O

- I/O interface **specification** for use in MPI apps Data Model:
  - Stream of bytes in a file
  - Portable data format (external32)
  - Not self-describing - just a well-defined encoding of types
- Features:
  - Collective I/O
  - MPI data types and file views
  - Non-blocking I/O
  - Fortran bindings
- Implementations available on most platforms

MPI I/O Implementations

- Different MPI-IO implementations exist.
- Three better-known ones are:
  - **ROMIO from ANL**
    - *Leverages MPI-1communication*
    - *Supports local file systems, parallel filesystems*
  - **MPI-IO/GPFS from IBM**
    - Data shipping = mechanism for coordinating access to a file to alleviate lock contention
    - Controlled prefetching = using MPI file views and access patterns to predict regions to be accessed in future
  - **MPI from NEC**

High-Level I/O Level Libraries (HDF5, NetCDF, …)

I/O Middleware (MPI I/O)

POSIX I/O

I/O Parallel File system

I/O Hardware

HDF5

- Hierarchical Data Format, from the HDF Group (formerly of NCSA) Data Model:
  - Hierarchical data organization in single file
  - Typed, multidimensional array storage
  - Attributes on dataset, data
- Features:
  - C,C++,and Fortran interfaces
  - Portable data format
  - Optional compression
  - Data reordering(chunking)
  - Noncontiguous I/O (memory and file) with hyperslabs

Parallel NetCDF

- Based on original “Network Common Data Format” (netCDF) work from Unidata
  - Derived from their source code Data Model:
  - Collection of variables in single file
  - Typed, multidimensional array variables
  - Attributes on file and variables
- Features:
  - C and Fortran interfaces
  - Portable data format (identical to netCDF)
  - Noncontiguous I/O in memory using MPI datatypes
  - Non contiguous I/O in file using sub-arrays
    Collective I/O

Conclusion

- I/O is becoming a major performance bottleneck in HPC
- The software for performing I/O on supercomputers consists of different layers
  - File systems
  - POSIX I/O
  - MPI I/O → Next lecture on this
  - Higher level interfaces (HDF5 and parallel NetCDF among the others)