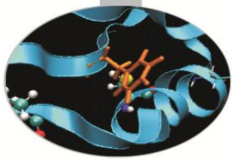
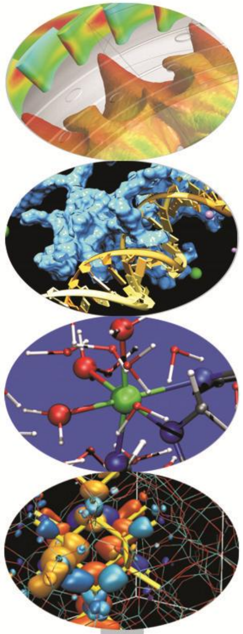


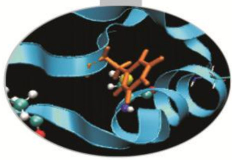
Is I/O still Manageable?



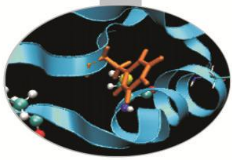
Carlo Cavazzoni, *HPC Department CINECA*

16 May 2013

What is I/O

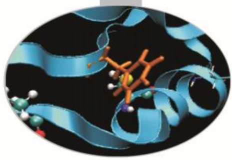


- DATA
- fwritef, fscanf, fopen, fclose, WRITE, READ, OPEN, CLOSE
- Call to an external library: MPI I/O, HDF5, NetCDF, ecc…
- Scalar/parallel/network Filesystems
- I/O nodes and Filesystem cache
- I/O network (IB, SCSI, Fibre, ecc..)
- I/O RAID controllers and Appliance (Lustre, GPFS)
- Disk cache
- FLASH/Disk (one or more Tier)
- Tapes

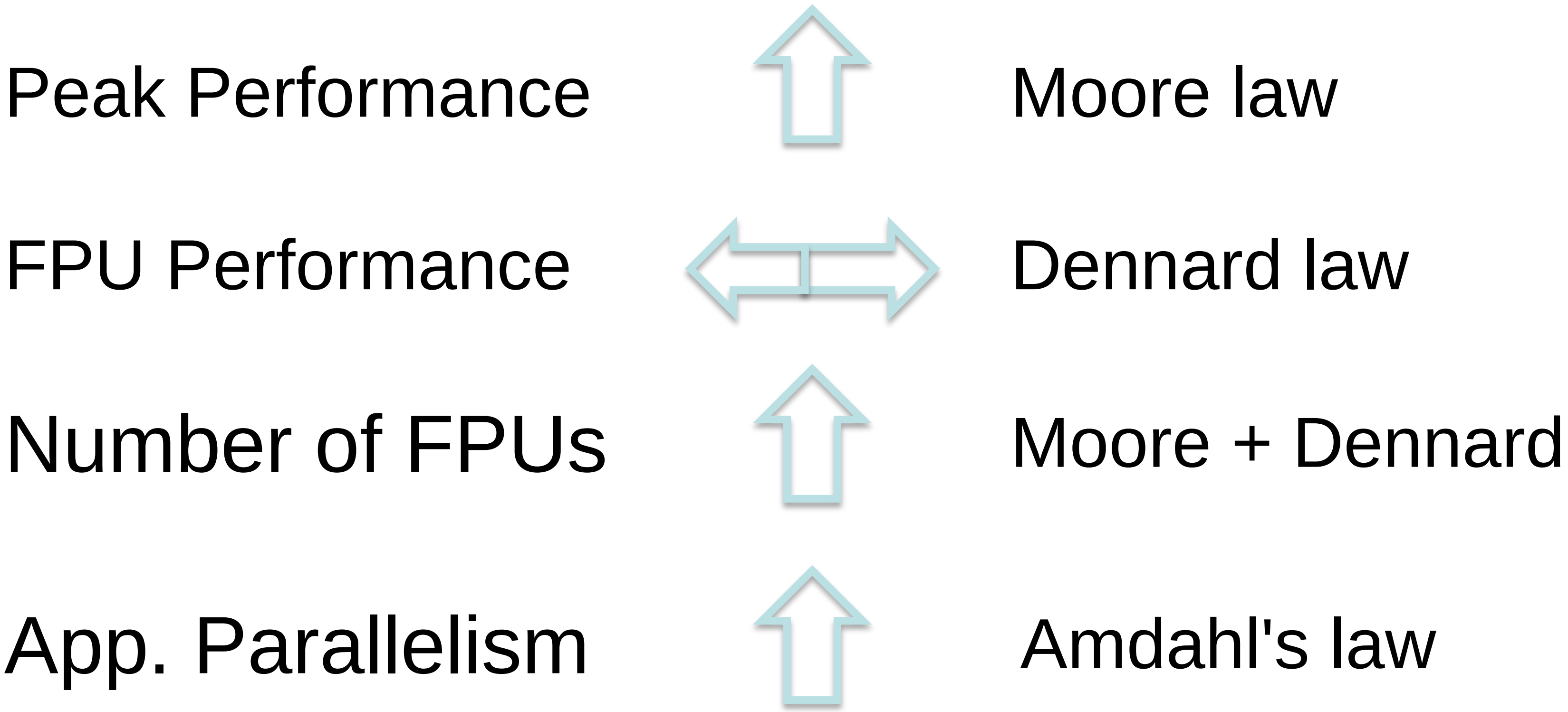


A Strategy

- Understand architectural trends (at all level)
- Evaluate impact on application I/O design
- Plan application refactoring, new I/O algorithms
- Field test on current available machine (anticipating some arch trends), proof of concept.
- Bring stuff into the main trunk for production.



Architectural trends



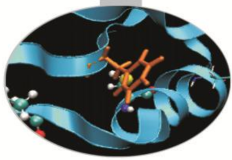
Peak Performance — Moore law

FPU Performance — Dennard law

Number of FPUs — Moore + Dennard

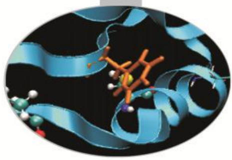
App. Parallelism — Amdahl's law

Architectural trends



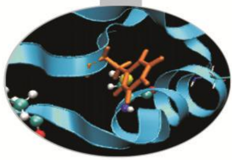
		2020 estimates
Number of cores	⇧	10^9
Memory x core	⇩	100Mbyte or less
Memory BW/core	⇩	500GByte/sec
Memory hierachy	⇧	Reg, L1, L2, L3, …

Architectural trends



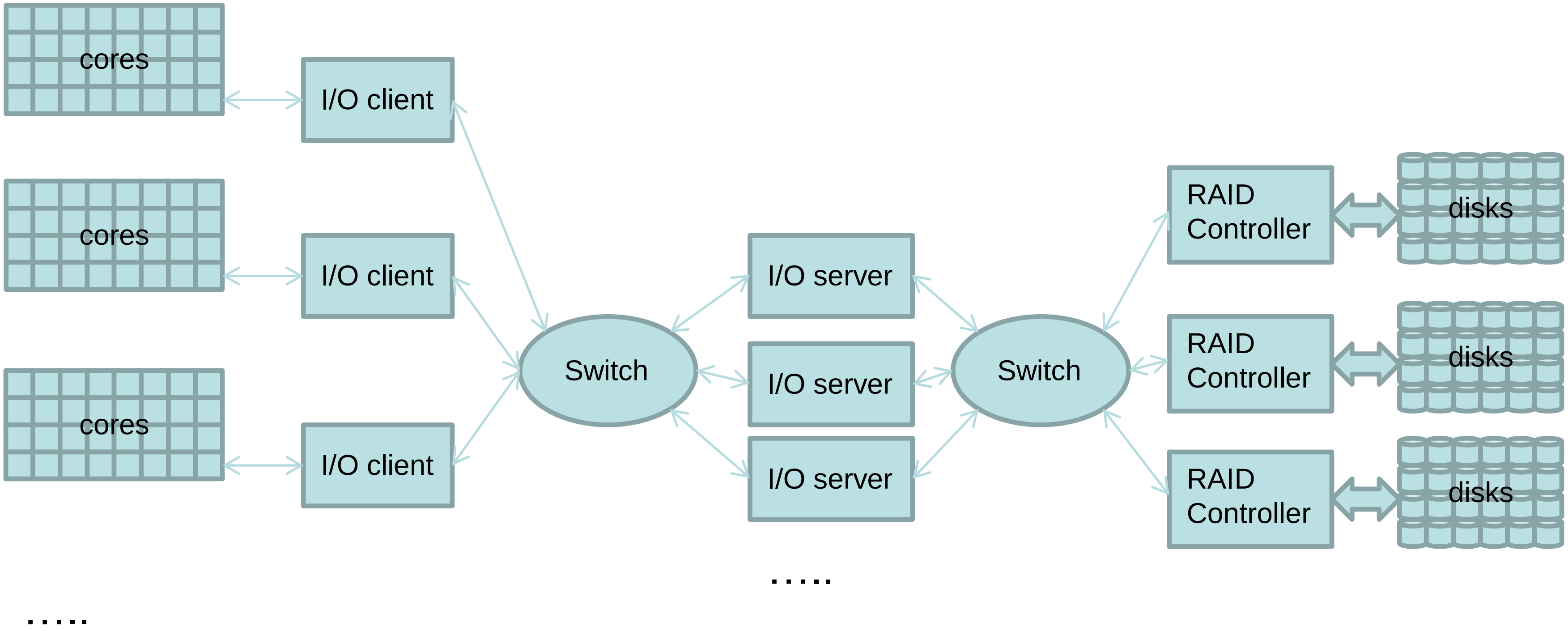
		2020 estimates
Wire BW/core	⇔	1GByte/sec
Network links/node	⇧	100
Disk perf	⇔	100Mbyte/sec Anti revolution – *disks will only be a bit faster than today*
Number of disks	⇧	100K

Challenges



Today (BGQ)	Tomorrow
100 clients	10K clients
1000 core per client	100K core per clients
3PByte	1Exabyte
3K Disks	100K Disks
100 Gbyte/sec	100TByte/sec
8MByte blocks	1Gbyte blocks
Parallel Filesystem	Parallel Filesystem
One Tier architecture	Multi Tier architecture

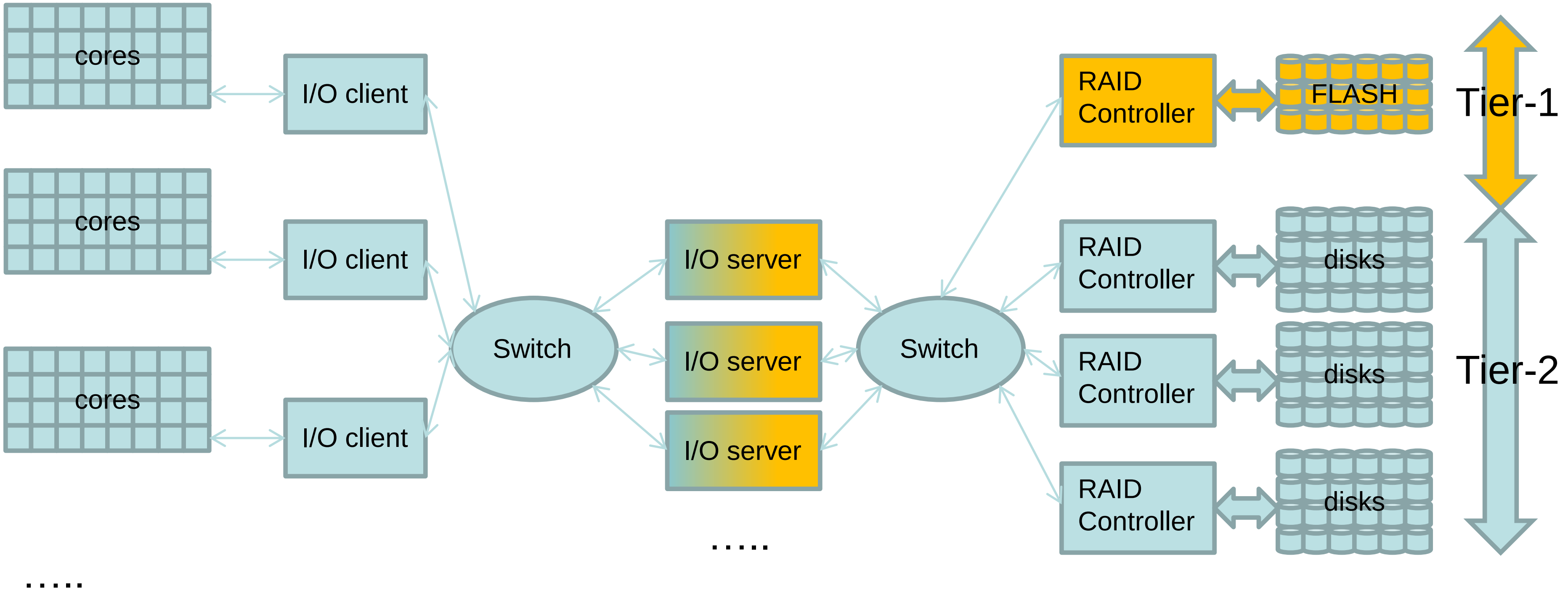
Today



160K cores, 96 I/O clients, 24 I/O servers, 3 RAID controllers

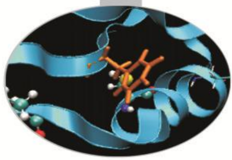
IMPORTANT: I/O subsystem has its own parallelism!

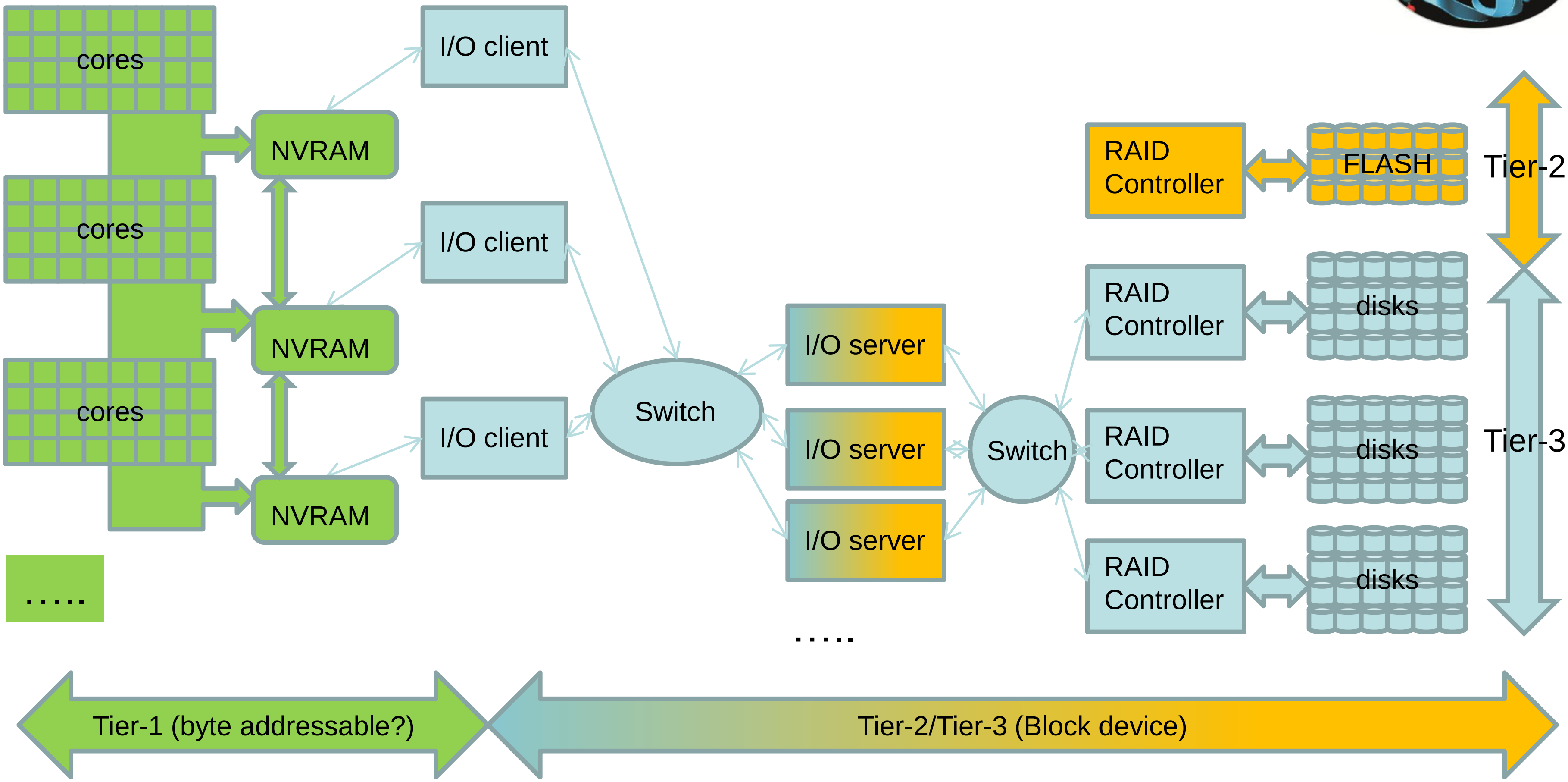
Today-Tomorrow



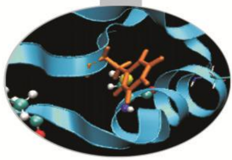
1M cores, 1000 I/O clients, 100 I/O servers, 10 RAID FLASH/DISK controllers

Tomorrow





1G cores, 10K NVRAM nodes, 1000 I/O clients, 100 I/O servers, 10 RAID controllers



Transition

DATA:

- Billion of (application) files
- Large (check-point/restart) file

Posix Filesystem:

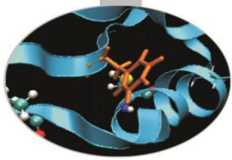
- low level
- lock/syncronization
- low IOPs (I/O operation per second)

Physical supports:

- disk too slow -> archive
- FLASH aging problem
- NVRAM (Non-Volatile RAM), PCM (Phase Change Memory), not ready

Middlewere:

- Library HDF5, NetCDF
- MPI-I/O
- Each layer has its own semantics



Strategy

I/O is the bottleneck -> avoid I/O when possible

I/O subsystem work with locks -> simplify application I/O

I/O C/Fortran APIs are synchronous -> use dedicated I/O tasks

I/O has its own parallelism -> use MPI-I/O

Raw data are not portable -> use library

I/O is slow -> compress reduce output data

Application DATA are too large -> analyze it “on the fly”, re-compute vs write