Features of HPC resources for HEP

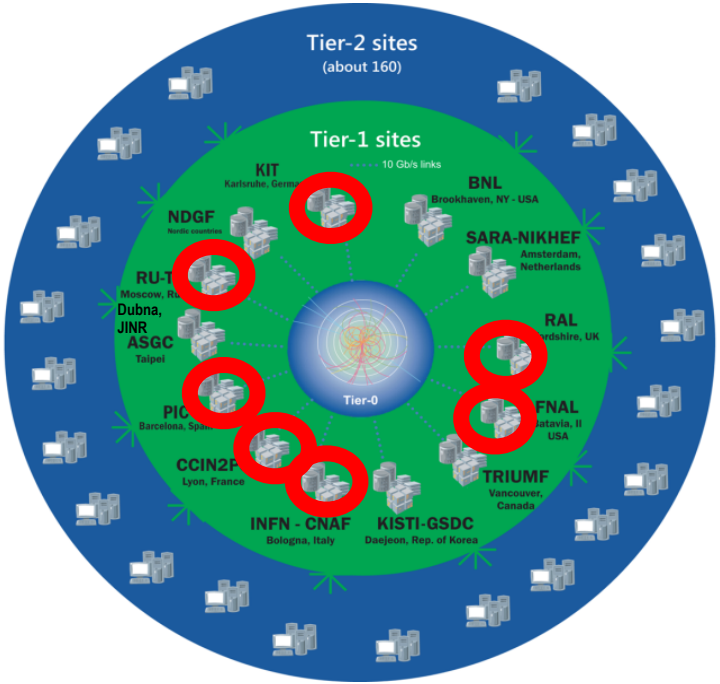
A.Petrosyan, D.Oleynik

Grid 2021, JINR, Dubna
July 9, 2021

Grid: 2 decades of success

The Worldwide LHC Computing Grid (WLCG): integrates computer centres globally to provide computing and storage resources into a single infrastructure accessible by all LHC physicists for data analysis.

42 countries, ~3000000 cores, ~200PB storage, >2 million jobs per days.



Middleware evolution from systems to products

AGIS/CRIC: ATLAS –> AMS, CMS, COMPASS, WLCG, etc.

Dirac: LHCb –> Belle II, BES III, IHEP, ILC, Ibergrid, etc.

PanDA: ATLAS –> LSST, nEDM, SciDAC-4, CHARMM, AMS, IceCube, Blue Brain, COMPASS, etc.

Rucio: ATLAS –> AMS, CMS, BNL, Belle II, etc.

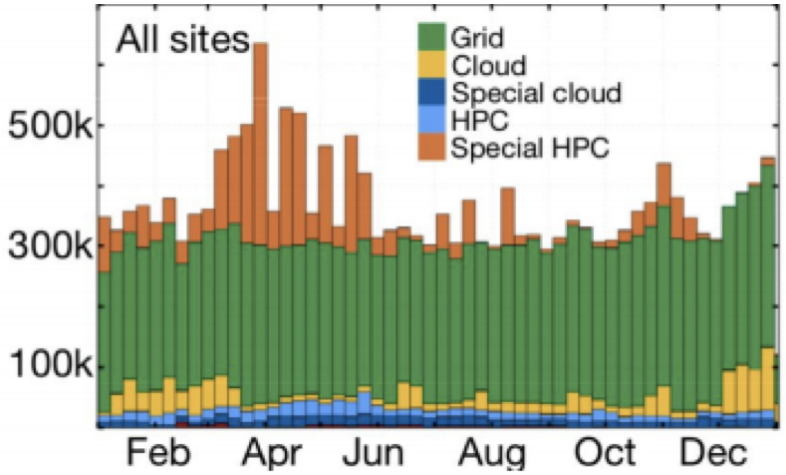
Grid and opportunistic resources

Today we get opportunistic use of many type of compute in particular HPC systems and the experiment online farms presented as Cloud resources.

In the future this heterogeneity will expand and we need to be able to make use of the resources provided to us.

A lot of work going on at the level of policies, infrastructure/services and, very importantly, software.

New Summit HPC @ ORNL:
9.96 Pflops in CPU (non X86_64)
215.7 Pflops in GPUs



HPC basics

Grid site	HPC
1 job per pilot.	N jobs per pilot.
Batch system allows remote jobs submission.	Jobs can be submitted only from the local nodes.
Grid services are available.	Grid services are unavailable.
Data can be delivered directly to the computing nodes by the Pilot application. Various data delivery protocols usually available.	Data delivered to the local storage, then being transferred to the node at the runtime. Only provided by system administrators of the machine data transfer protocols available.
Results can be immediately be transferred to any external storage system.	Results can be transferred to the local storage.
Applied SW distributed via CVMFS.	No CVMFS. Applied SW has to be installed on the local file system.
Applied SW tuning is not needed.	Each step of the data processing must be agreed with the system administrators. The profile of CPU utilization and I/O operations is built and analyzed.
No serious demands to the infrastructure: smooth load grow, loss of one job does not affect the rest.	Makes the highest demands on all infrastructure components: it is necessary to maintain a large number of jobs in the queue, jobs can spend weeks in the queue before being processed, after which jobs on hundreds of nodes can be launched at the same time. If at least one job is lost or suboptimal, damaging the stability of the system, the rest of the user's jobs, including those running, can be removed from the queue.

Large jobs

Usually on a large HPCs local batch manager sets higher priority for larger jobs in the queue.

Example: on Frontera submission size is 50-100 nodes, which is 2800-5600 individual jobs in each. System administrators suggest to define submissions as large as possible.

WMS must be ready to generate enough amount of job definitions to fill out the queue, DDM must be ready to deliver their input data.

Any missed heartbeat leads to the loss of whole submission, jobs re-generation.

Should we consider early workload binding? Data delivery per campaign, jobs reports only on the final statuses, no intermediate heartbeat messages. Applied software multithreading to enable real MPI jobs would help a lot: 1 job instead of 56 per node on Frontera would reduce load on WMS components and would allow to run 56 times more individual jobs and make larger submissions.

Long waiting jobs

Jobs on HPC can spend weeks waiting in the queue before being processed.

Example: on Frontera submission spends 7-10 days in the queue before processing.

In Grid, following an HTC paradigm we work in the high connectivity: each job has to send a heartbeat, if job fails to send the heartbeat it will be considered lost and will be regenerated.

Sending individual job heartbeats while waiting in the queue on an HPC does not look reasonable, we must consider disabling all intermediate messages, because they simply generate load on WMS components. Only final job state must be reported.

With the high load any missed heartbeat can bring to the loss of the whole submission.

Single-core jobs issue

Single-core jobs are the real show stopper on HPCs.

Example: on Frontera we run 56 individual job per node over 56 individual input files. In order to achieve target load we had to install 4 Harvester instances, each of them configured to run 14 000 individual jobs. Each single job sends a heartbeat, generating excessive load. If the job could take 1 input file and split it into 56 parallel jobs, we would reduce the load 56 times (or increase the load 56 times).

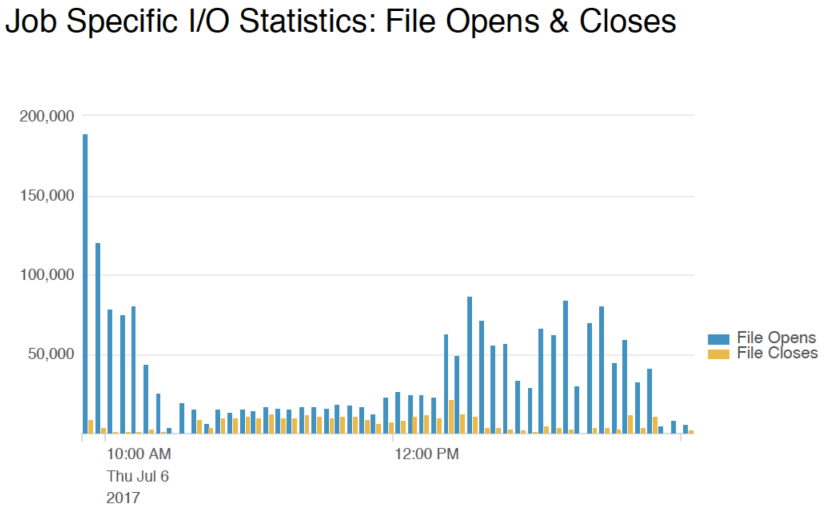
I/O crisis on Titan 1/2

“System administrators have alerted us that your jobs on Titan are currently doing over 2 million metadata operations per job and causing congestion on the metadata server. This is impacting other users' jobs.

To help the current situation, could you stop the jobs you have running on Titan and limit to only 1 running job at a time? We have received several reports of hangs on the atlas2 partition.”

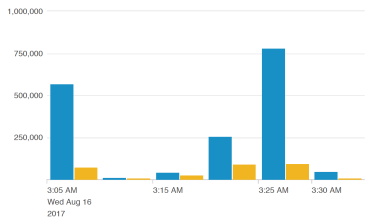
We were allowed to launch up to 20 submissions in a row, so total number of metadata operations from ATLAS production **might reach 40 millions operations**.

Lustre metadata servers were affected, and looks like that NFS some well with loading.

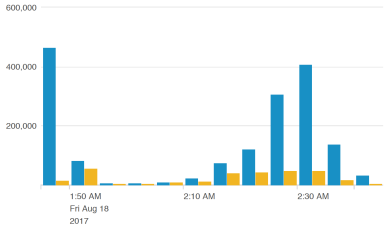


I/O crisis on Titan 2/2

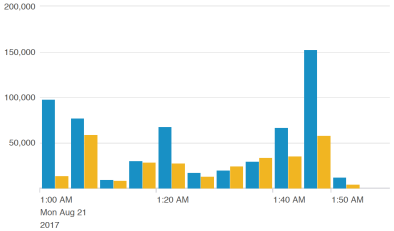
Before optimization



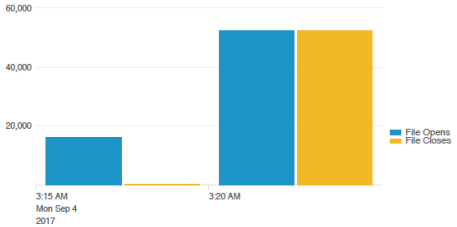
Environment cleanup



ATHENA 'hack'



Working dir in RAM Disk



- Significant reduction of IO. Number of ‘open’ operations almost matches the number of ‘close’ operations.
    - Initial spike came from the MPI wrapper which used to launch ATHENA Job on computing node.
- Current setup of ATLAS production at OLCF
    - ATLAS releases: NFS
    - Job working directories and input data: RAM disk of computing node
    - Output data moved to Lustre at the end of the job

Architecture 1/2

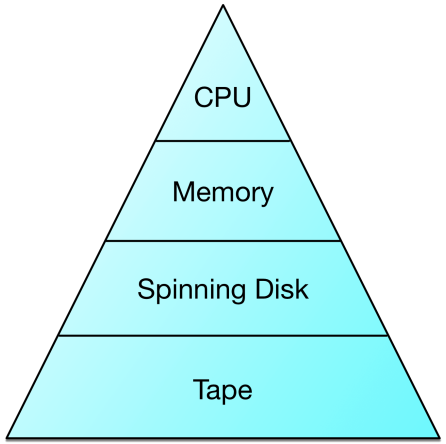
	Name	Processor	Cores per Socket	CPU Cores	Accelerator/Co-Processor
1	Supercomputer Fugaku	A64FX 48C 2.2GHz	48	7 630 848	None
2	Summit	IBM POWER9 22C 3.07GHz	22	202 752	NVIDIA Volta GV100
3	Sierra	IBM POWER9 22C 3.1GHz	22	190 080	NVIDIA Volta GV100
4	Sunway TaihuLight	Sunway SW26010 260C 1.45GHz	260	10 649 600	None
5	Perlmutter	AMD EPYC 7763 64C 2.45GHz	64	91 136	NVIDIA A100 SXM4 40 GB
6	Selene	AMD EPYC 7742 64C 2.25GHz	64	71 680	NVIDIA A100
7	Tianhe-2A	Intel Xeon E5-2692v2 12C 2.2GHz	12	427 008	Matrix-2000
8	JUWELS Booster Module	AMD EPYC 7402 24C 2.8GHz	24	44 928	NVIDIA A100
9	HPC5	Xeon Gold 6252 24C 2.1GHz	24	87 360	NVIDIA Tesla V100
10	Frontera	Xeon Platinum 8280 28C 2.7GHz	28	448 448	None

Nowadays, the most powerful machines in the top 500 were not built on x86 architecture.

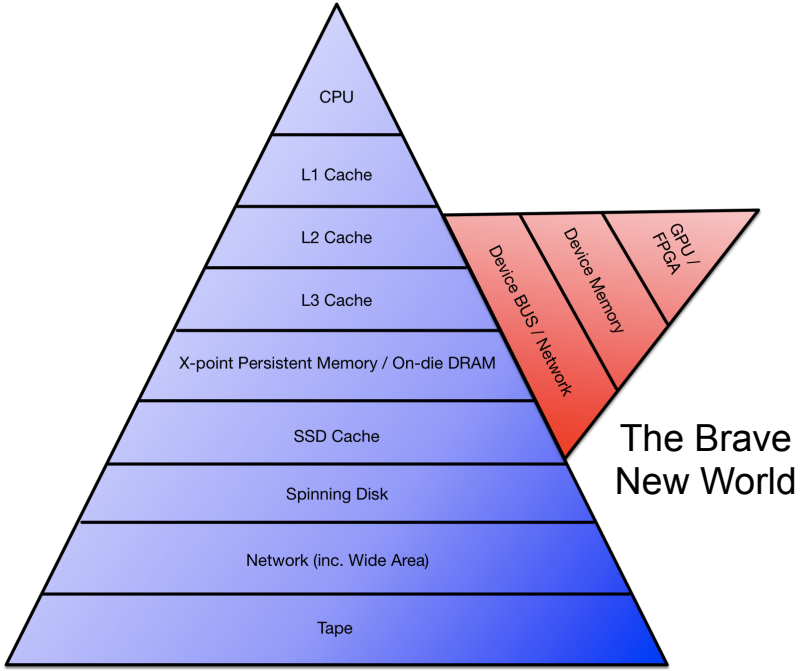
In several years none of top 10 the most powerful machines will be built on x86 architecture.

We must be ready for this. Otherwise, our bright HPC adventure in HEP will come to an end.

Architecture 2/2



The Good Old Days



The Brave New World

Non HPC resources nowadays usually have multiple cores as well.

Conclusions

We have learned to work with HPCs, actually, we are taking the first steps to work on such machines.

While working under special conditions, we can show good results. But when we work on a common basis, we are facing with a lot of problems, which mainly lie in two areas:

- The traditional focus on high connectivity, suitable for grid environment, where we work with individual jobs and need to quickly send the next job to freed up resource, which means it is necessary to monitor its state as often as possible. For HPC this means too many useless messages which generate too high load.
- And applied software, the efficiency of which, when working on small nodes with individual disks, does not affect other jobs and other users. And when running on HPC, it becomes the reason for the difficulty of scaling.

Some of described problems can be solved by the development of middleware, some by the development of the applied SW. We must add even more flexibility to our middleware than we had now and be ready to adapt our SW to the new architectures.