Interfaces to Leadership Class Computing

Dirk Hufnagel and Andrew Norman

The Department of Energy has invested heavily in new Exascale computing facilities at Argonne National Lab, Oak Ridge National Lab and at Lawrence Berkeley Lab, which will both provide a significant fraction of and shape the future of high energy physics computing resources. This shift of HEP computing in the United States towards more centralized supercomputing facilities and away from smaller distributed University campus computing clusters and high throughput grid computing sites, will require significant shifts in the ways that the HEP community access resources.

This Letter envisions a future where leadership class facilities provide a significant part of the total compute cycles for HEP Experiments. We look at what interfaces we need to Leadership Computing Facilities (LCFs, meaning ALCF and OLCF, not including NERSC or non-DoE HPC) to integrate them with both traditional HEP computing and next generation AI and ML applications that will be used during and beyond the HL-LHC and DUNE eras.

Computing Models with LCF Computing

We assume that future HEP computing models will heavily leverage the computing cycles which will be available from exa-scale facilities, but that the DoE LCF infrastructure will not provide a fully encapsulated end-to-end computing solution for HEP data processing (meaning input data comes from somewhere else and the output goes to somewhere else, the LCF will only ever handle a fraction of the whole data processing chain). Instead the design of the current and near future LCF centers are aimed at providing heterogeneous compute cycle endpoints. HEP computing models will need to consider how these compute warehouses will access the vast amounts of data that are processed by HEP experiments or are generated in their simulation codes. Data movement, data locality, network connectivity and unique networking topologies of the LCF machine platforms will require specialized “interfacing” with the HEP workflow engines and batch queuing systems. Considerations also need to be made for how multiple LCF can be used in conjunction with each other to fill the burst needs of specific computing challenges or computing campaigns which either exceed the needs of a single facility or require coordination between the different types of computing hardware that are available at different facilities (i.e. coordinating GPU intensive work at one facility with I/O intensive work more suited to a different facility).

Technical Integration

The current LCFs each have their own customized solutions to providing access to the communities they serve. Technical integration with HEP Computing has to find commonality to

be long term maintainable. We aren't starting this process in a vacuum, we have gained experience with HPC integration for years already..

The current LCF have a very restricted network topology, with the worker nodes providing the compute cycles not being able to directly connect to outside HEP Computing resources. We will need to utilize edge services that are running next to the LCF and can facilitate this access. The LCF need to provide resources on which we can deploy such edge services. The exact form is not important, but for long term maintenance reasons it would be preferred if the different LCF could provide a common solution. Technologies that are being explored at some HPC are container orchestration platforms (Kubernetes).

Edge services are currently foreseen to be able to provide workflow management integration (tying the experiments job submission systems together with the LCF batch systems) and condition data delivery to payloads running on LCF resources (through frontier edge squid proxies). Experiment software can also be made available through cvmfs (utilizing the same edge squid proxies).

Access to experiment data would be much more difficult to route through edge services, as the data volumes are much larger. Different to condition data and experiment software, caching would also not help much to reduce the traffic since experiment data is usually only read once for processing. Actively managing LCF storage as part of the distributed storage systems of an experiment and managing the dependencies of data management and workflow management externally to the LCF might be the more promising approach. This requires integration of the LCF into the data transfer matrix of the experiment to be able to transfer data from/to other labs and universities. Many HEP experiments are or will be using Rucio for this.

Non-Technical, how to get allocations

The current process to request compute cycles on the LCF very much ties any allocation grant to a specific purpose. One has to say exactly what the computations are for and what the desired results are. This does not match well with how computing activities are planned for many HEP experiments. Based on new discoveries priorities might shift and having the majority of the compute cycles locked for a specific purpose would be a problem. Also, many HEP experiments are long-running, over years and sometimes decades. Compute activities are planned well in advance, at longer time scales than the allocation approval cycles for the LCF.

For LCF to provide the majority of compute cycles to HEP experiments, they will need to be reliably available. This means an experiment needs to know well in advance how many resources it will receive. In practice this means multi-year allocations. The allocations will also need to be more flexibly usable, with the experiment deciding internally what to use the resources for, at least to some extent.