EPJ Web of Conferences **251**, 02042 (2021) https://doi.org/10.1051/epjconf/202125102042
CHEP 2021

Exploitation of HPC Resources for data intensive sciences

Maria Girone[1,*], *David* Southwick[1,2,**], *Viktor* Khristenko[1,***], *Miguel* F. Medeiros[1,], *Domenico* Giordano[1], *Ingvild* Brevik Høgstøyl[3], and *Luca* Atzori[1]

[1]CERN, Geneva, Switzerland
[2]University of Iowa, USA
[3]Norwegian University of Science and Technology, Norway

Abstract. The Large Hadron Collider (LHC) will enter a new phase beginning in 2027 with the upgrade to the High Luminosity LHC (HL-LHC). The increase in the number of simultaneous collisions coupled with a more complex structure of a single event will result in each LHC experiment collecting, storing, and processing exabytes of data per year. The amount of generated and/or collected data greatly outweighs the expected available computing resources. In this paper, we discuss efficient usage of HPC resources as a prerequisite for data-intensive science at exascale. First, we discuss the experience of porting CMS Hadron and Electromagnetic calorimeters reconstruction code to utilize Nvidia GPUs within the DEEP-EST project; second, we look at the tools and their adoption in order to perform benchmarking of a variety of resources available at HPC centers. Finally, we touch on one of the most important aspects of the future of HEP - how to handle the flow of petabytes of data to and from computing facilities, be it clouds or HPCs, for exascale data processing in a flexible, scalable and performant manner. These investigations are a key contribution to technical work within the HPC collaboration among CERN, SKA, GEANT and PRACE.

1 Introduction

The field of High-Performance Computing (HPC) is undergoing a transition to the next major phase of its development, namely that of exascale computing. The first systems with such capabilities are slated for the early 2020s with numerous development efforts underway in Europe, the United States, China, and Japan. In anticipation of the delivery of these systems, the global HPC community is broadening its horizons: as well as providing a step change in capability for its traditional user base (computational fluid dynamics, quantum chemistry etc.) exascale systems will need to provide the e-Infrastructure required by large experimental facilities that are due to generate unprecedented volumes of data as new capabilities come online in the next decade.

To reach exascale computing, HPC facilities rely heavily on heterogeneous hardware architectures. Accelerated processors like Graphical Processing Units (GPUs) or low-power ARM processors provide the bulk of the computing capacity at the majority of the largest

*e-mail: maria.girone@cern.ch
**e-mail: david.southwick@cern.ch
***e-mail: viktor.khristenko@cern.ch

supercomputers. The use of heterogeneous architectures is both a challenge and an opportunity [1] for the High Energy Physics (HEP) community. Within the context of the DEEP-EST project [2], CERN participated as one of the scientific applications. The experience of porting workflows from the Compact Muon Solenoid (CMS) Experiment to utilize Nvidia GPUs is presented in Section 2. Section 3 further discusses developments in order to enable transparent benchmarking of computing resources available at HPC facilities. Finally, Section 4 looks at challenges that are specific for any data-intensive science that aims to process Exabytes of data.

2 HPC and Heterogeneous Computing

2.1 The DEEP-EST Project

The Dynamic Exascale Entry Platform - Extreme Scale Technologies (DEEP-EST) project is an European HPC initiative to build a prototype of the Modular Supercomputing Architecture (MSA). The DEEP-EST is the latest project in the "DEEP" series, where with each iteration prototypes integrate increasingly diverse hardware architectures. The MSA concept allows integrating compute modules with different performance characteristics into a single heterogeneous system, interconnected using different network fabrics.

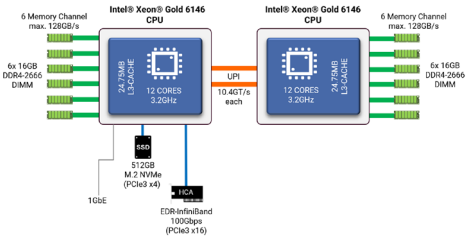


Figure 1: DEEP-EST Prototype CM node specification

The DEEP-EST prototype consists of three types of compute modules: Cluster (CM), Extreme-scale Booster (ESB) and Data Analytics (DAM) modules. Each module is designed in order to provide best performance for a particular type of workload. For instance, CM nodes are targeting applications that require higher single thread performance, whereas ESB nodes are more optimized for scientific codes requiring many-core high degree of parallelism architectures, and therefore draw most of the computational power from Nvidia GPUs.

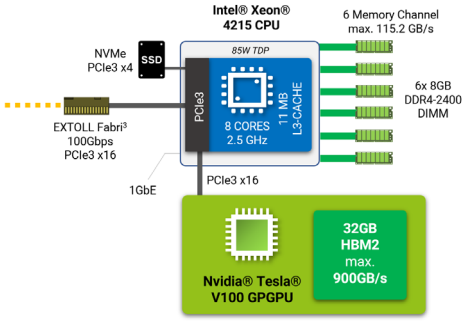


Figure 2: DEEP-EST Prototype ESB node specification

In total, the DEEP-EST prototype contains 50 Cluster Module nodes. Figure 1 provides the exact specification of each node. As can be seen, each CM node is equipped with a dual socket Intel Xeon 'Skylake' Gold 6146 which gives 24 physical cores in total (12 per socket). In terms of main memory, there are 192 GBs RAM, which comfortably gives 8GB per core. The InfiniBand 100Gb/s fabric is used to interconnect nodes, enabling low latency message passing interfaces such as MPI.

For the ESB module there are 75 nodes in total and the exact specification is provided in Figure 2. The Extreme Booster Module is targeted to provide applications requiring extreme scalability. The ESB has been designed to be used together with CM nodes via MPI to offload parts of the code that require a high degree of parallelism. A lower grade CPU has been chosen (Intel Xeon 'Cascade Lake' Silver 4215) where the bulk of the compute capability being the Nvidia V100 GPU, with 32 GB of HBM2.

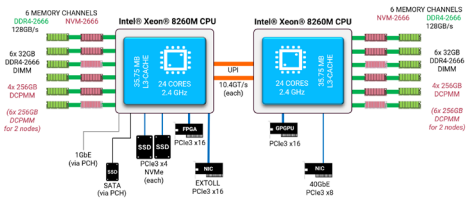


Figure 3: DEEP-EST Prototype DAM node specification

The last and newest computing module in the DEEP series is the DAM. This is the module that exhibits the widest variety of hardware. Figure 3 provides the exact specs of each node. In total there are 16 nodes, each equipped with a dual socket Intel Xeon 'Cascade Lake' Platinum 8260M and 384 GBs of RAM. Additionally, each node contains a single Nvidia V100 GPU and Intel STRATIX10 FPGA, both connected via PCIe Gen3.

2.2 CMS Ecal/Hcal on Nvidia GPUs

The Compact Muon Solenoid (CMS) detector is a general-purpose particle detector which consists of several components: tracker, electromagnetic and hadronic calorimeters, magnet and muon systems. Each component (usually referred to as a sub-detector) accomplishes a different task. For instance, the tracker (both Pixel and Strip parts) is the closest sub-detector to the interaction point and responsible for identifying the trajectories of charged particles. Calorimeters measure energy depositions of the particles passing through them.

As part of the DEEP-EST project, the CMS Electromagnetic (Ecal) and Hadronic (Hcal) calorimeters local reconstruction code was ported to utilize Nvidia GPUs. We targeted the High Level Trigger (HLT) specific parts for porting in this work, however the solution can be further generalized for offline reconstruction. Local reconstruction for both of these sub-detectors consists of a sequence of steps of various transformations to obtain the final event product. The sequence can be divided into the following three parts:

- Unpacking - decoding raw data.
- Multifit/Mahi - iterative routines employed to perform reconstruction of the actual "energy" deposit.
- Corrections - calibration factors can be applied to the values obtained from the previous step.

Out of the three steps above, the most important and time-consuming is the iterative process to reconstruct energy. The procedure is usually referred to as "Multifit"/"Mahi" (Ecal/Hcal), however mathematically it can be formulated in terms of a χ^2 problem with a positivity constraint, as we must require that energies deposited in the calorimeter are always positive. More precisely, we can write this down as:

$$min(\chi^2) = argmin_x((P \cdot x - b)^T \Sigma(x)^{-1}(P \cdot x - b)) \tag{1}$$

$$\forall i : x_i \geq 0 \tag{2}$$

where:

$$x - \text{energy vector} \tag{3}$$

$$P : \text{energy} \rightarrow \text{charge mapping} \tag{4}$$

$$\Sigma - \text{total covariance matrix} \tag{5}$$

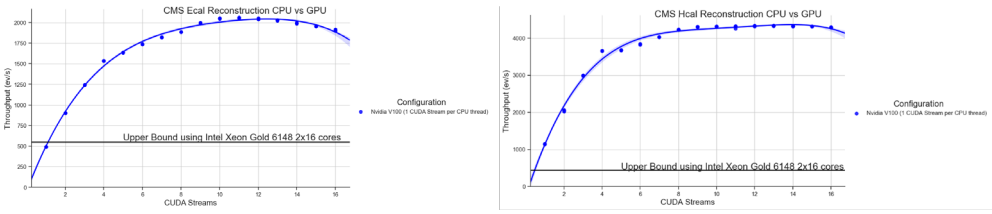
$$n - \text{number of time samples} \tag{6}$$

It is important to point out that the correlation matrix Σ is not a constant but rather depends on the solution vector for which we are minimizing for. Therefore, we do not have a direct solution, but rather employ the following iterative procedure:

1. Compute the Σ matrix using the current iteration of χ
2. Perform minimization of χ^2 for the just computed correlation matrix
3. Compute χ^2 and if below some threshold we have convergence, otherwise go to step 1

Step 2 above constitutes a formulation of a Non Negative Least Squares (NNLS) [7] problem with positivity constraint. To solve we employ something that has been called the Fast NNLS algorithm, which is an active set algorithm that performs iterative minimization preserving the constraint. Although there are notable differences in what goes into the minimization procedure (e.g. different ways to build the Σ matrix), both the Hcal and Ecal algorithm perform essentially an identical mathematical function. We took advantage of this to port a single algorithm for both.

For our performance comparison we chose maximum throughput of events per second that could be achieved either on CPU or GPU. Although the CMS software framework is multi-threaded, the original local reconstruction algorithm implementations are single threaded, therefore they are expected to scale linearly with the number of available cores (provided there are no other limitations). However for the case of a GPU implementation, it is not so straight forward. We try to push as many concurrent events into a single GPU card as possible until we reach the maximum compute capacity. Figure 4 shows the results of testing CMS Ecal and Hcal local reconstruction only workflows using CPU or GPU sequences. Both Ecal and Hcal routines were optimized specifically targeting Nvidia V100 GPU. Results were obtained using CMS Open Data [8]. In summary, substantial speed ups in throughput can be achieved when porting these types of workloads to heterogeneous hardware, like general purpose GPUs.



(a) Ecal Performance (b) Hcal Performance

Figure 4: Ecal (a) and Hcal (b) Performance

Combining the Hcal and Ecal local reconstruction developments with the rest of the Patatrack effort of porting CMS reconstruction algorithms to Nvidia GPUs, we can take a preliminary look at performance of CMS HLT Run3 configuration on the DEEP-EST nodes. Throughout all of these measurements we utilize the same CMS Open Data dataset [8]. Figure 5 shows the results of running CMS HLT Run 3 configuration using each type of compute

modules available at the DEEP-EST prototype. Since the Compute Module contains CPUs only, we have run CPU-only configurations on these types of nodes, however for the ESB and DAM nodes we have tested both of the configurations. For the ESB and DAM nodes we observe approximately 50% speed up when using configurations that make use of the available Nvidia V100 GPU. The reason for a smaller speed up running full CMS HLT Run3 configuration, compared to evaluating Hcal and Ecal ported algorithms independently, is due to performance still being limited by algorithms running on the CPU. Per algorithm there is quite a large speed-up and, as a consequence, significant benefit from offloading reconstruction code to a GPU.

3 Containerized Benchmarking on HPC

The target hardware for benchmarking during the past decades has largely been resources which are owned or administered by the Worldwide LHC Computing Grid (WLCG) and partners. This has allowed a common set of assumptions for users and administrators to operate within, as well as flexibility to change hardware, topologies, and policies when necessary. In contrast, HPC sites are designed in response to market demands from a wide variety of actors. As a result, sites generally feature a diverse array of heterogeneous compute resources, while simultaneously enforcing stricter security policies that appeal to the needs of clients. In order to fully exploit the opportunities presented by HPC sites, a change of assumptions that takes into account this new environment is required.

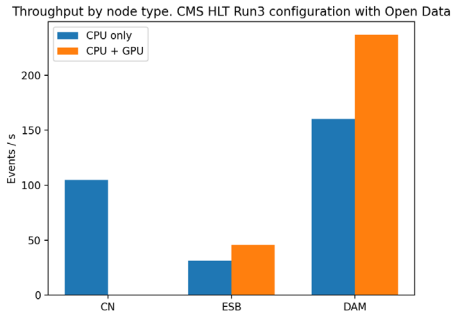


Figure 5: Results running CMS HLT Run 3 configuration using Open Data on different compute modules of the DEEP-EST prototype

Security policies at HPC sites are necessarily more restrictive in order to ensure protection of client data from within and without. In an environment where multiple user's jobs may be simultaneously shared on the same node, restriction of escalated privileges is mandatory. This may include restriction of userspace executables to reduce attack vectors from within the compute cluster, and sometimes, restricted external network connectivity from compute nodes to reduce attack vectors from external locations. Complying with these policies requires that all HEP workloads are executed unprivileged, and frequently only via the binaries exposed by a HPC site. In practice, this means all executables not typically included in a minimal operating system installation must be provided by either binary compilation for the target hardware (in the case of permissive userspace executables) or via container images compatible with a host service (in a restricted userspace).

3.1 Extending HEP benchmarking for HPC

Adapting benchmarks designed under the assumptions of WLCG computing sites necessitated several changes. Docker, despite widespread adoption in IT development and service fields, has not been embraced by the majority of HPC sites. Serious security concerns regarding Docker's privileged container daemon remain unaddressed; potentially allowing unautho-

rized root access to the production network. In addition, resource managers and job schedulers necessary for shared resources are difficult to integrate with Docker. In response to this void, many HPC sites have adopted a container solution designed specifically for the challenges of HPC shared computing environment: Singularity [9].

Singularity is fully compliant with Open Container Initiative (OCI) standards [10] and features native support for translation of Docker images. Therefore, migration to Singularity only requires modification of Dockerfile definitions to account for differences between the two services. Docker, by default, mounts a "thin" read-write partition (squashfs) to containers, enabling file modification at runtime. Singularity does not do this automatically, and the permission to create file systems is not supported on most HPC sites. Docker container users are root by default - for Singularity the container user is the same as the host user. This required new publication of all HEP Dockerfile definitions to eliminate file modifications outside of host-mounted directories, and ensure execution as an unprivileged container user. The migration from using Docker images to Singularity on HPC required an additional step of “flattening” the structure of nested docker containers previously employed for benchmarking. All nested container execution requires privilege escalation, and this approach is not supported regardless of container solution on HPC.

The nested container approach was abandoned in favor of a simplified microservice-style orchestrator, re-written as a python 3 package. Benchmarks are now modular and configurable through declarative YAML language, executing Singularity images by default. At select HPC sites with CVMFS access, pulling and translating docker images can be skipped entirely with the use of the recent CVMFS-unpacked service that hosts singularity unpacked images.

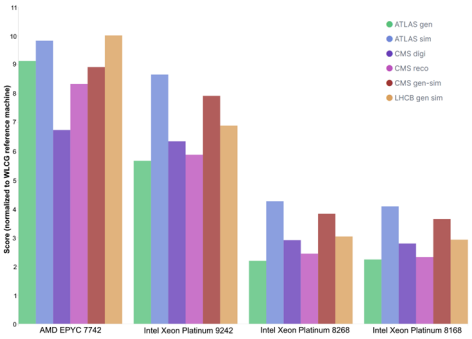


Figure 6: Experiment workload performance (normalized) on select HPC processors, larger is better

Integration with HPC job scheduling tools such as SLRUM [11] have been greatly simplified thanks to the migration to Python. An array of selected benchmarks may be executed across a large batch of heterogeneous resources by a single python argument defining a YAML configuration to use - even remotely hosted YAML definitions. Leveraging this combination of a simplified modular structure, minimal runtime assumptions, and greatly expanded coverage of heterogeneous architectures has enabled the successful benchmarking and collection of over two thousand results covering more than 122,000 cores across numerous HPC sites. Scale deployment up to 200 simultaneous nodes has been run without issue. For further detail on developments of the HEP Benchmark suite, see the proceedings on HEPiX benchmarking solution for WLCG computing resources in this issue[12].

4 HPC and Data Access

The traditional approach to performing data processing has been to employ a distributed computing grid, which has been extremely successful. LHC experiments have been able to utilize computing and storage resources with a common design and specification set forth by the HEP community. The level of control of the design, procurement and access results in

deep implications on the experiments' operational computing model and creates substantial difficulties for the community when trying to integrate external resources like HPC facilities and cloud resources. One of the main difficulties in this integration stems from the fact that the flow of data to/from these HPC facilities is no longer in full control of HEP experiments. Furthermore, each such supercomputing center defines its own interfaces and policies for interaction; each site may have different types of storage systems, network topologies, or interconnect methods. Figure 7 shows an example of a schematic overview of the DEEP-EST prototype network federation [4]. As can be seen, in total there are three types of fabrics used, for each cluster respectively, with many gateways in order to connect different modules. For

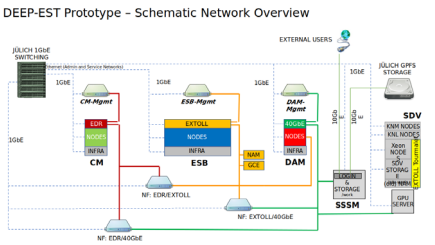


Figure 7: DEEP-EST Prototype Network Federation.

data intensive applications that require large quantities of data ingesting and data production, there are two crucial challenges when approaching heterogeneous compute facilities:

1. It is important to understand the limitations of the internal interconnect and storage systems. In order to obtain the maximum efficiency of the allocated resources, the limiting factors of these systems must be understood. Would network topology between compute nodes and storage systems be a bottleneck or would the storage system itself restrict scaling? Given the variety of storage systems (e.g. shared filesystems, object stores, etc...), each with its own architecture and deployment setup, it is not clear if changes are necessary to the way HEP deploy workflows. Are changes are needed in order to utilize certain types of storage systems to their fullest without reducing the processing performance?

2. Once there is an understanding on a site's internal interconnect and storage capabilities, the second challenge is how to enable data flow from HEP experiments' storage location (e.g. potentially Data Lakes). HPC centers have very different external connectivity bandwidth, and this may prove critical for applications that require sending and receiving large amounts of data to a computing center. Restrictive ingress or egress bandwidth is another potential limiting factor to successful site utilization.

In order to address first point mentioned above, we chose CMS MINIAOD2NANO conversion workflows, as they are one of the more I/O driven data parallel standard workflows. We measured aggregated bandwidth from an HPC shared storage system while scaling the number of compute nodes, with the goal of enabling extrapolations for a hypothetical Exascale HPC storage system. Several tests were performed at San Diego Supercomputer Center (SDSC) using a Ceph storage instance. Using Slurm batch system, hundreds of nodes were allocated with instances reading data using CephFS. Table 4 summarizes the results obtained and extrapolations deduced. Considering that a future Exascale system will bring O(1M) cores, one should expect to be able to draw O(150)GB/s from a storage system, assuming

scaling out to the whole machine. Although this might not seem like a lot (as this is aggregated across all of the nodes), this is on average. Considering that HEP I/O is typically bursty, one could expect going significantly outside of this projection as well, which might impact the overall performance.

N nodes	Throughput/node (Evs/s)	Aggregated B-width
1	284	15 MB/s
200	289	3037 MB/s
300	288	4525 MB/s
O(10K)	—	O(150) GB/s

5 Conclusion

Even with aggressive operational improvements, the computing needs of the experiments during the HL-LHC will exceed the predicted technology evolution. This leads to an expected resource gap between what is needed to complete the science mission and what can be provided by traditional sites. To close this gap new computing resources and techniques will be needed, and HPC supercomputing centres could play a vital role. In this work, we shared three contributions to the effort of the HEP community to exploit High Performance Computing facilities. Utilization of heterogeneous resources is crucial to improve the performance of applications on a per-node basis, particularly so as we move towards using more machine-learning driven techniques. For the purpose of understanding the types of resources and accounting, benchmarking of such facilities is key to properly estimate the amount of resources needed for experiments. Finally, as we are approaching Exabyte scale for moving, processing, storing and analyzing data, an initial study data access methodologies is being performed.

Acknowledgments

This work is funded by the EU Horizon 2020 research and innovation programme under grants no. 754304 (DEEP-EST) and no. 101017567 (EGI-ACE). The European Commission is not liable for any use that might be made of the information contained in this paper. The authors would also like to thank Simons Foundtation and San Diego Supercomputer Center for providing access to the computing resource facilities to allow early testing and development.

References

[1] M. Girone, Common challenges for HPC integration into LHC computing, https://zenodo.org/record/3647548#.YDdh4pNKgWp

[2] The DEEP-EST Project, https://www.deep-projects.eu/

[3] E. Suarez, N. Eicker, T. Lippert, Modular Supercomputing Architecture: from Idea to Production, Contemporary High Performance Computing: From Petascale toward Exascale, Volume 3 FL, USA, CRC Press 3, 3rd, 223-251 (2019)

[4] The DEEP-EST Prototype, https://deeptrac.zam.kfa-juelich.de:8443/trac/wiki/Public/User_Guide/System_overview

[5] ParaStation MPI, https://www.par-tec.com/products/parastation-mpi.html

[6] Sirunyan, A.M. and others, Reconstruction of signal amplitudes in the CMS electromagnetic calorimeter in the presence of overlapping proton-proton interactions, https://cds.cern.ch/record/2721995

[7] Bro, Rasmus and De Jong, Sijmen, A fast non-negativity-constrained least squares algorithm, Journal of Chemometric 11, 5
[8] CMS Open Data, http://opendata.cern.ch/record/12303
[9] Singularity https://sylabs.io/singularity
[10] Linux Foundation, Open Container Initiative (2015), https://opencontainers.org/
[11] Slurm Workload Manager, https://slurm.schedmd.com/overview.html
[12] M. F. Medeiros et al., HEPiX benchmarking solution for WLCG computing resources (2021), submitted to CHEP2021 conference