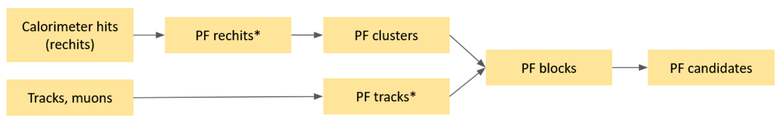
Heterogeneous reconstruction of hadronic Particle Flow clusters with the Alpaka Portability Library

Andrea Bocci[1], Abhishek Das[2], Kenichi Hatakeyama[3], Florian Lorkowski[4], Felice Pantaleo[1], Wahid Redjeb[5], **Jonathan Samudio**[3], Mark Saunders[3], on behalf of the CMS Collaboration

[1]CERN, [2]University of Notre Dame, [3]Baylor University, [4]Deutsches Elektronen-Synchrotron, [5]RWTH Aachen University, III. Physikalisches Institut A

Particle Flow

Particle Flow (PF) [1] is a global event reconstruction algorithm which combines information from all CMS subdetectors to produce a set of final-state particle candidates for each collision event. Inputs for the PF algorithm include reconstructed hits (rechits) from electromagnetic & hadronic calorimeters, used to produce PF clusters; and tracks from the silicon tracker and muon systems to produce PF tracks. These elements are linked to form blocks from which the final-state particle candidates are produced. The formation of hadronic PF clusters is one of the most computationally intensive tasks during PF reconstruction at the High-Level Trigger (HLT) [2]. This served as a good candidate for GPU offloading and the development of a portable algorithm with Alpaka [3].



Alpaka

The alpaka library is a header-only C++17 abstraction library for accelerator development. This library implements an abstraction layer that allows one to develop parallel algorithms with no backend specific considerations. It implements a hierarchical redundant parallelism model which is familiar to CUDA developers, and uses a data agnostic memory model. Alpaka's functionality is being tested at CMS in various reconstruction algorithms used at HLT. The primary goal of porting the reconstruction is to obtain a single maintainable source code that can run on many hardware architectures and backends with performance close to the native ones.

HBHE PF Clustering

PF clusters from the hadronic calorimeter barrel and endcap regions (HBHE) are a crucial component of jet reconstruction as shown below. These clusters are also used in reconstructing tau leptons and determining the missing transverse energy in an event.

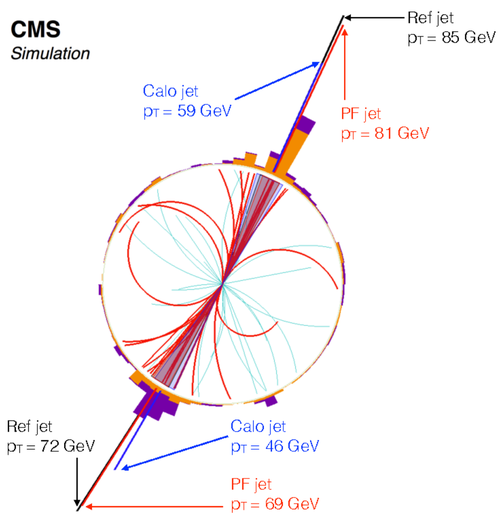


Figure 1. Comparisons showing the reconstruction of a jet from PF with the given transverse momentum (p_T) from simulation.

The clustering algorithm is performed in three stages: seeding, topological clustering, and the formation of PF clusters.

1. Seeding: Local energy maxima among neighboring PF rechits are identified as cluster seeds. Seeding conditions are pulled from a database containing values for each HCAL channel.
2. Topological clustering: Topologically connected PF rechits are linked together to form topological clusters, these can contain multiple seeds where every seed will become a PF cluster.

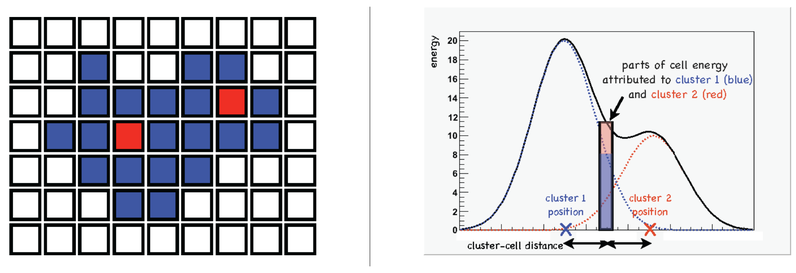


Figure 2. (Left) Topologically connected calorimeter hits are associated to a single cluster. Seeds are shown in red and associated rechit in blue. Each seed will become a PF cluster. (Right) Illustration of energy sharing between two clusters.

3. PF clustering: A Gaussian-mixture model is used to calculate the energy sharing between hits of a given topological cluster. Each seed is assigned a Gaussian energy deposit, and associated rechits contribute a fraction of their energy based on their distance to the cluster. In the case of a single seed, the entirety of the rechit energy is assigned to the cluster.

Porting to Alpaka

Porting HBHE PF clustering to Alpaka was carried out in multiple stages. Previous work had been done to port the PF rechit and PF cluster algorithms to CUDA, and parallel versions of each step were developed. We make the following improvements in the port to Alpaka, with an emphasis on exploiting GPU architecture:

- Data is converted to a Structure of Arrays (SOA) format to take advantage of the parallel architecture of GPUs.
- Multiple PF rechits and topological clusters are processed simultaneously where possible, for example, the application of energy thresholds to PF rechits and seeds. Each rechit is assigned to an individual GPU thread, and all rechits can be checked against the energy thresholds asynchronously.
- We implement a parallel-optimized connected component labeling algorithm, ECL-CC [4], to form the topological clusters. In the initialization of ECL-CC, each "vertex" is labeled by the ID of the first neighbor which has a smaller ID than the vertex. During computation of the representatives of the connected components, "hooking" and "intermediate pointer jumping" operations are used and represent the unique implementation of ECL-CC.

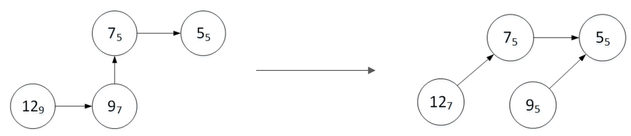


Figure 3. Example of intermediate pointer jumping

In our implementation we only have vertices up to degree 8 and rely solely on thread granularity processing, although ECL-CC has multiple tiers of granularity depending on the vertex size. ECL-CC differs algorithmically from the initial PF implementation, but not logically.

- The core algorithm for energy sharing between seeds remains the same, but is divided into separate tiers of computation and optimized as such. Using tiered computation we can quickly compute single seed clusters, and most clusters can utilize performance improvements from GPU shared memory accesses and parallel processing of rechits. Depending on the size, large topological clusters will use either shared or global memory accesses, and process rechits iteratively.
- Runtime allocation of the PF rechit fraction SoA is used to minimize the memory usage of the clustering algorithm and provide flexibility to cover rare events containing a very high number of rechit fractions.

Performance Results

We validate the physics performance from analyzing cluster properties, and test the event throughput at HLT by switching to the Alpaka version of PF clustering:

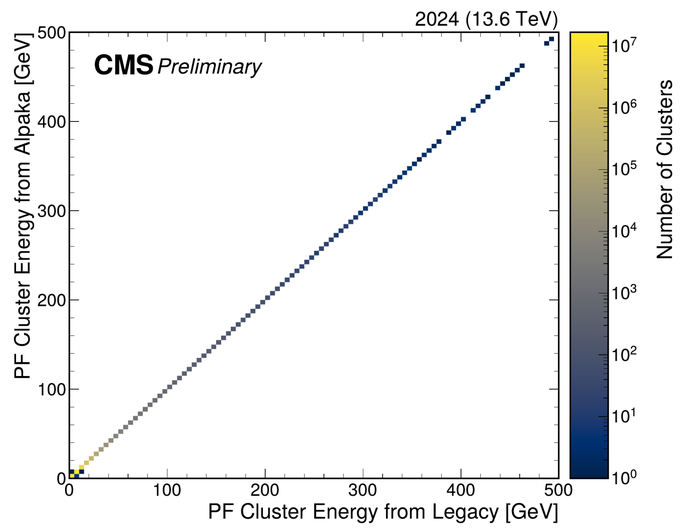


Figure 4. Comparisons of reconstructed HBHE PF cluster energies with Legacy CPU and Alpaka GPU clustering algorithms as measured in Run 3 2024 data. Energy discrepancies greater than 1% occur only in 0.00001% of HBHE PF clusters.

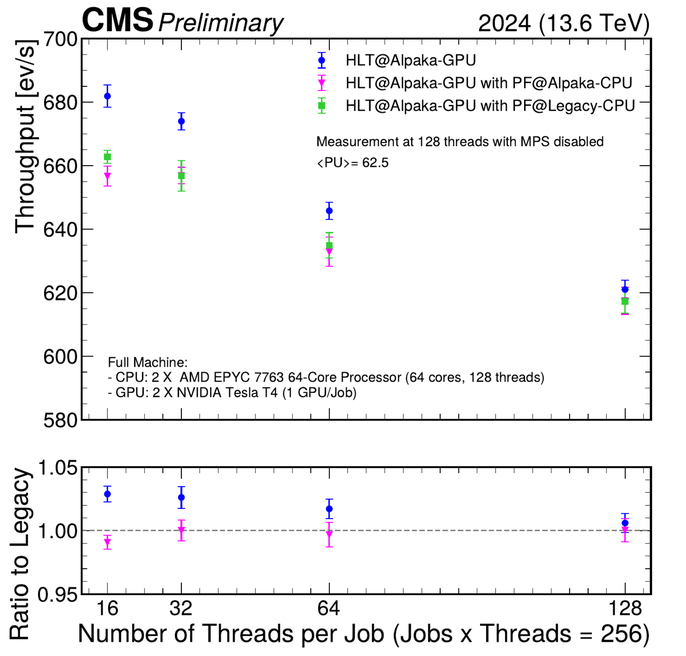


Figure 5. The event throughput of a CMS HLT configuration employed during the 2024 data-taking period. Each measurement runs the configuration on 40,000 events of proton-proton collision data from 2024 at an average pileup of 62.5. The blue points represent the event throughput achieved by executing the HLT with all the available heterogeneous modules on GPU. In contrast, the magenta ones depicts the event throughput when using the Alpaka-CPU version of PFRechit and PFCluster. The green points showcase the event throughput when utilizing the legacy version of PFRechit and PFCluster on CPU. Notably, the plot demonstrates a 2.5% speedup in HLT performance when utilizing 8 jobs with 32 threads each (standard data-taking HLT settings).

References

[1] A.M. Sirunyan et al. (CMS Collaboration), "Particle-flow reconstruction and global event description with the cms detector," JINST 12, P10003 (2017).

[2] V. Khachatryan et al. (CMS Collaboration), "The CMS trigger system," JINST 12, P01020 (2017).

[3] E. Zenker, B. Worpitz, R. Widera, A. Huebl, G. Juckeland, A. Knüpfer, W. E. Nagel, and M. Bussmann, "Alpaka – an abstraction library for parallel kernel acceleration," in *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 631–640, 2016.

[4] J. Jaiganesh and M. Burtscher, "A high-performance connected components implementation for GPUs," in *Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing*, HPDC '18, (New York, NY, USA), p. 92–104, Association for Computing Machinery, 2018.

[5] CMS Collaboration, "Heterogeneous Reconstruction of Hadronic Particle Flow Clusters with Alpaka Portability Library," http://cds.cern.ch/record/2898660.

"This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 1840279. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation."