1 Introduction and Motivation

The HL-LHC will usher in a new era of precision particle physics unlike any previous experiment. The fact that the LHC will collect a massive data set as a hadron collider creates unique challenges in particular for the software and computing infrastructure of CMS, as the processing time for reconstruction algorithms typically employed to build high-level physics objects often grow non-linearly with the number of overlapping collision events (pileup). A dedicated effort must be initiated to improve the performance of the reconstruction algorithms to avoid increasing the computing budget of CMS far beyond reasonable levels. Research and development into alternative computing architectures, in particular Graphical Processing Units (GPUs), has therefore started in earnest in the last few years. This line of inquiry has so far proven successful in accelerating several algorithms used in CMS High Level Trigger, namely pixel tracking and local calorimeter reconstruction. The effort to adapt offline software to execute on GPUs must now be bolstered as well. In addition, a large portion of the computing power at modern High Performance Computing (HPC) centers in the US is provided by GPUs rather than CPUs. To gain access to these HPCs, the CMS reconstruction algorithms must be significantly restructured and adapted. The proposed project consists of two thrusts: (1) adapting pixel unpacking to execute on GPUs and (2) adapting the vertex reconstruction algorithm to execute on GPUs. These projects are synergistic and leverage existing expertise at Boston University in back-end readout electronics to expand the group's research program into the area of heterogeneous computing for the HL-LHC era. The pixel unpacking project will serve as an educational bridge project, and the vertex reconstruction effort will take advantage of the gained expertise to accelerate a resource intensive portion of the HL-LHC reconstruction.

2 Project Description

Pixel Unpacking

The existing software for unpacking the pixel detector's raw data relies on a sequential algorithm. It starts with the initialization of the cabling map that associates the online local pixel modules to offline global pixel modules. The algorithm then gathers the data for each front-end unit in an event and reads and interprets the control bits that provide information on the data completeness, data size, front-end detector ID. The unpacker then decodes each 32-bit word into a link number, readout chip number, pixel address, and ADC count. The decoded data is then used to begin the pixel reconstruction algorithm.

While the later stages of the pixel reconstruction have been adapted to GPU, the pixel unpacker currently executes on a traditional CPU. Furthermore, since the unpacked data's size is significantly larger than the compressed data, a large GPU memory is needed when copying the unpacked data for later stages of the reconstruction algorithm. Allocation of large GPU memory may further limit the complexity of the algorithm itself. Concurrently, the data format can be optimized such that costly host-device memory transfers can be limited, thereby improving the performance of the pixel reconstruction. Lastly, while there have been significant efforts in revising the GPU-based pixel unpacking for Run 3, HL-LHC studies remain to be done.

Boston University is in a unique position to contribute to the pixel reconstruction development as Prof. Demiragli is currently developing the hardware platform of the pixel back-end readout modules and the necessary firmware and software. Specifically, the group recently proposed a preliminary pixel data format with the following guiding principles in mind: (1) making sure the unpacking can run as much as possible in parallel (i.e., avoid a design that requires sequential decoding) (2) making the data structure as uniform as possible, i.e., no need for checking of detector specific conditions for each fragment and (3) keep the unpacking self-contained, i.e., minimize the use of lookup tables that involve geometry and cable maps. The preliminary data format has also been discussed with the Patatrack team members and GPU experts. The future work will include measuring the performance of the HL-LHC pixel reconstruction with the preliminary data format and unpacker. Addressing all the components of the system will enable rapid optimization.

Vertex reconstruction

Among all high-level physics objects reconstructed in CMS, the vertex reconstruction plays a particularly special role. By efficiently identifying the vertex associated with the hard interaction, the energies of particles from pileup vertices can be subtracted from various quantities such as jet energies or isolation sums. This process improves both the efficiency and resolution of nearly all other physics objects in CMS. However, given the huge increase in pileup, and therefore the number of tracks, the vertex finding stage of the reconstruction becomes much slower compared to Phase 1. To cope with the large number of pileup interactions at the HL-LHC, CMS will install a detector just outside of the tracker volume that will provide timing resolution of approximately 30 picoseconds for all minimum ionizing particles (MIP). This detector, referred to as the MTD (MIP Timing Detector), will allow pileup vertices to be separated from the hard interaction vertex in time as well as space. By adding the time dimension as another input, the problem of maintaining high throughput for the algorithm becomes even more challenging.

The proposed project will consist of adapting the vertex reconstruction algorithm to execute on GPUs. Vertex reconstruction in CMS consists of two stages. The first stage is known as “clustering” and consists of determining which tracks will be assigned to common vertices. The second stage is known as “fitting”, where the clustered tracks are used to determine a single space-time point to each vertex. The proposed project will focus initially on the initial clustering stage. The algorithm currently used by CMS to perform vertex clustering is known as “Deterministic Annealing” (DA). To demonstrate the strong dependence on the number of tracks, a heuristic description of the algorithm is warranted. The DA algorithm consists of several core functions, two of which are the most important: the “split” and “update” functions. The algorithm starts by assigning all tracks to a common vertex, and a “Free Energy” like function (in analogy with statistical physics) is defined. A temperature parameter is slowly lowered allowing the system to relax, and a phase change condition is checked. If satisfied, the vertex is “split” into two new vertices. The phase change condition is calculated from the full list of tracks associated to each subvertex. The positions of the new vertices are then “updated” after each splitting, a calculation that requires for each vertex a loop over all of the tracks in the event. As can be seen, each update or split step requires a loop over all tracks, which for pileup of 200 is expected to be more than 1,000. The

temperature is then lowered again and the process is iterated until the temperature reaches a predefined minimum. Once the system reaches this temperature, splitting is stopped and only track assignments are changed. An additional outlier rejection step is added to lessen the impact of tracks that are not near any vertex.

The DA algorithm involves a large number of parallel calculations, and it is therefore a clear candidate to be adapted for processing using GPUs. Recently, a group from São Paulo Research and Analysis Center (SPRACE) have presented a proof of principle CUDA implementation of portions of the DA algorithm. This development clearly demonstrates that the DA algorithm is well suited to parallelization. However, much additional work remains to be done and the investment by USCMS into this project is still warranted. This initial implementation consists of a CUDA version for a few of the most important calculations (vertex update probability, vertex updated positions, critical temperature), but not the full DA algorithm. Furthermore, the proof of principle uses toy data and operates as a standalone code. A complete implementation will need to be integrated into CMSSW and be consistent with existing CMS data structures. Additionally, quantitative performance measurements have yet been obtained and the host-device memory transfers have not been optimized. Our group will work collaboratively with the SPRACE group and work towards these high-level goals.

3 Project Importance

The pixel unpacking for HL-LHC is a critical task, without it the pixel reconstruction cannot proceed. Optimizing the performance of the pixel unpacking will be especially critical for HLT, where it is a significant fraction of the reconstruction time. In addition, this project will serve to build experience in measuring the performance of GPU-based algorithms. Our second project, efficient vertex reconstruction, is crucial for the success of the CMS physics program at the HL-LHC. The current estimate of its contribution to the total Phase 2 reconstruction time is around 9%, when excluding the I/O time, as shown in Figure 3. The large resource fraction for the I/O is caused by extra event content being saved during the development stage, and therefore is expected to be much smaller during production. Furthermore, significant development on other computationally intensive algorithms is already underway and expected to be completed in time for the HL-LHC. In particular, endcap calorimeter clustering and track reconstruction are both currently being developed by other teams to enable

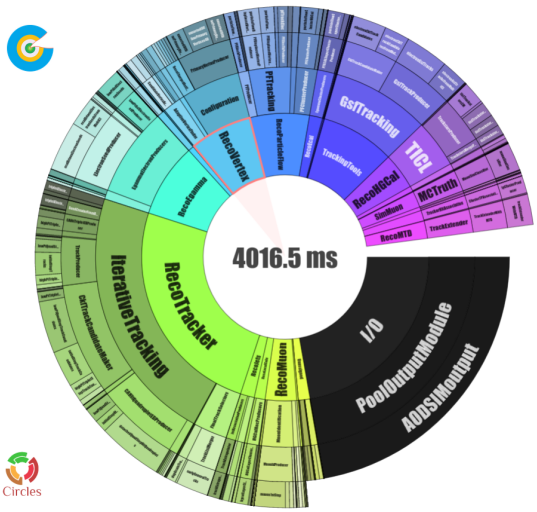

Figure 1: CPU utilization of vertex reconstruction stage in the context of the overall Phase 2 reconsruction. Vertex reconstruction accounts for about 9% of the total time, excluding I/O.

execution on GPUs. Without development, the vertex finding steps will become a bottleneck for CMS reconstruction. Rewriting the algorithm to execute on GPUs will provide a sizeable reduction to the overall event reconstruction time. In addition, once the algorithm is finished it may also be considered for use in the High Level Trigger, where faster but less accurate algorithms are currently utilized. Therefore the project has the further potential to significantly improve the CMS trigger system. Both of the two projects together will help establish Boston University as a source of expertise in GPU-based programming within the US, enabling efficient future algorithm developments upon completion of the proposed project.

4 Plan of Work and Timelines

The project will begin with the postdoc quantifying the performance of the draft HL-LHC pixel unpacking algorithm. This will help the postdoc to quickly gain experience in GPU-based workflows and profiling tools, and to integrate with ongoing GPU-based algorithm development efforts. It is expected that this will take 1-2 months of full-time effort. The data format and unpacking algorithm will then be optimized, if necessary, following the results of these performance measurements. The project will then continue with the prospective postdoc learning in full detail of the CMS implementation of the DA algorithm. To become a complete expert on the algorithm and its parameters is expected to take about one month's full time effort. Concurrently, the postdoc will learn the fundamentals of GPU programming by considering the existing GPU implementations of other algorithms such as pixel tracking, calorimeter reconstruction, and the proof of principle DA algorithm. The time needed to become proficient will depend on whether the postdoc has previous experience, which will be a top desired qualification in the search. It is conservatively estimated that the time needed for this stage will also be 1-2 month's full time effort. Therefore, after four months of preliminary training the postdoc will be well-prepared to begin developing a full implementation of the DA algorithm. This task will account for the remainder of the project period. With a baseline to start from, a full GPU implementation of the DA algorithm can reasonably be completed within the first year of the project. The new postdoc will devote 100% of his or her time to these projects during their first year, with the portion of funding not provided by USCMS coming from a combination of seed and grant funding.

Should the project be renewed for additional time, additional fine tuning of the configurable parameters would be performed. The next step of the overall project would consist of rewriting the subsequent vertex fitting algorithm for execution on GPUs. Alternatively, with the experience gained from the current project we may consider investigating other resource intensive stages of the HL-LHC reconstruction. One possibility would be the Gaussian Sum Filter (GSF) tracking used for electron track reconstruction.

5 Mentorship Plan

The postdoc will be directly mentored by Profs. David Sperka and Zeynep Demiragli on a daily basis. Sperka has acquired using research funds provided by Boston University a

workstation with an Intel Xeon Processor W-2155 10-core CPU and an NVIDIA TITAN V GPU. Exclusive access to this workstation will be given to the postdoc preventing any possible problems from sharing a development machine with other researchers. Sperka has spent part of the last year commissioning this workstation, and CMSSW has been fully integrated as well as the independent CMSSW versions that are currently used for GPU-based pixel track workflows. Sperka has spent the last year gaining experience in GPU based programming and has contributed bug-fixes to the pixel tracking code. In addition, Boston University researchers have access to the Massachusetts Green High Performance Computing Center (MGHPCC), which consists of over over 18,000 CPU cores and a combined 240,000 GPU cores.

In addition to direct mentoring, the postdoc will benefit from the fact that Sperka will for at least the next two years be convening the newly formed "Code Modernization and Performance Working Group" within CMS at large. This group is tasked with, among other responsibilities, facilitating and supporting the overall adaptation of the CMS software to make efficient use of GPUs. This working group will include a core team of experts in CMS reconstruction algorithms and particularly in GPU programming techniques. The postdoc would be a member of this group's core team and profit immensely from the close collaboration with other top experts within CMS.