Geospatial Filter and Refine Computations on NVidia Bluefield Data Processing Units (DPU)

Derda Kaymak
Department of Computer Science
Marquette University
derda.kaymak@marquette.edu

Satish Puri
Department of Computer Science
Missouri University of Science and Technology
satish.puri@mst.edu

ABSTRACT

In this poster, we will show how to leverage NVidia's Bluefield Data Processing Unit (DPU) in geospatial systems. Existing work in literature has explored DPUs in the context of machine learning, compression and MPI acceleration. We show our designs on how to integrate DPUs into existing high performance geospatial systems like MPI-GIS. The workflow of a typical spatial computing workload consists of two phases - filter and refine. First we used DPU as a target to offload spatial computations from the host CPU. We show the performance improvements due to offload. Next we used DPU for network I/O processing. In network I/O case, the query data first comes to DPU for filtering and then the query goes to CPU for refinement. DPU-based filter and refine system can be useful in other domains like Physics where an FPGA is used to perform the filter to handle Big Data. We used Bluefield-2 and Bluefield-3 in our experiments. For scalability study, we have used up to 16 DPUs.

Keywords

GIS; Polygon Overlay; MPI IO; GPU; CUDA

1. INTRODUCTION

In geospatial processing systems, polygons and polylines are used to represent shapes in the real-world. Geospatial applications need to combine two (or more) datasets based on some spatial relationship between shapes in the two datasets. In map overlay, superimposing one dataset containing hurricane swath (polygonal area) and another dataset with county boundaries, will help in determining nearby rescue shelters. To combine shape datasets, we have to find overlapping shapes from the two datasets. For efficiency, these query operations are carried out in two phases. The first phase is a filter phase where complex geometries are approximated by their minimum bounding rectangle (MBR). Geometries that could not possibly satisfy the query condition are removed from further processing. The output of filter phase goes to the refinement phase to remove false hits generated by the filter. Filter and refine computations are used in many domains like geographic information systems (GIS) and Physics to handle the data deluge.

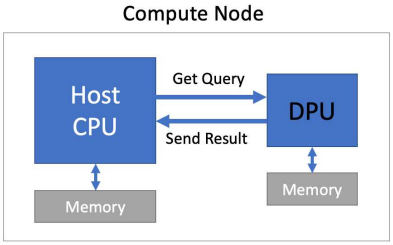

Figure 1: Query processing offload from CPU to DPU.

In the poster, we revisit spatial data analytics on heterogeneous systems comprised of data processing units (DPU) as shown in Figure 1. Bluefield DPUs are a new class of programmable processors made by NVidia (and other manufacturers). Bluefield DPUs contain a multi-core ARM processor and has its own memory. Similar to a modern smart network interface card, DPUs can be used to filter unnecessary data from overwhelming the CPU and memory bandwidth. SmartNICs and DPUs are geared towards offloading computations from a host CPU in order to free up the compute cycles in the CPU. The data can be transferred from the CPU to DPU in offload mode to execute a compute-intensive query. However, in streaming mode, the query comes to the DPU first and so DPU can execute one or more filters on the streaming data to reduce the query and output data from being processed by the CPU. After preprocessing the queries, a DPU then passes the queries to the host CPU. Nvidia's BlueField-2 and BlueField-3 DPUs are recent hardware solution from NVidia. Performance evaluation and benchmarking for an application domain like GIS is crucial due to the hardware's recent release and computational limitations in processing and memory. Therefore, we created a benchmarking tool to measure the performance BlueField-2 and BlueField-3 and compare it with an Intel x86 processor.

Bluefield DPUs have been used in deep neural network training [6], molecular dynamics [7], data compression [11], MPI implementations [5, 12] and databases [13]. Current GPU-based [9, 8] and MPI-based geospatial data processing systems [2, 10, 3] do not leverage the heterogeneity presented by DPUs. We show experimental results using a single filter based on Rectangle tree (R-tree) data structure. We are planning to experiment with further hierarchical filtering in

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

our future work.

2. EXPERIMENTAL RESULTS

Experimental Setup: The benchmark was tested on the Thor cluster of the HPC Advisory Council [4], which has Intel Broadwell E5-2697A CPU node (32 cores) with 2.6 GHz clock frequency. Thor cluster also has 32 BlueField-2 and 16 BleuField-3 devices as DPU nodes. Bluefield-2 DPU has Arm Cortex-A72 processor with 8 cores (2.4 GHz) and 16 GB memory. Bluefield-3 DPU has Armv8.2+ A78 Hercules processor with 16 cores (2.1 GHz) and 16 GB memory. For internode communication, OpenMPI library was employed. On the other hand, communication between CPU and DPU was facilitated using gRPC library. GEOS library was used for parsing shape data and for computational geometry algorithms. Source code is available on GitHub [1].

In the benchmarking tool, the input consists of two files containing geospatial data, Base and Query layers. Spatial join queries are then executed on these data, and the time taken for these operations is measured. The process includes creating an R-tree using the Base Layer, performing queries for each geometry in the Query Layer using the Minimum Bounding Rectangles (MBRs) from the R-tree, and executing specific operations (e.g., intersection, overlap, touch, equality, covering) on the candidate geometries obtained after filtering. The elapsed time encompasses all these steps.

Table 1: Attributes of the geospatial datasets.

Name	Type	#Geometries	File size
cemetery	Polygons	193 K	56 MB
sports	Polygons	1.8 M	590 MB
lakes	Polygons	8.4 M	9 GB

For benchmarking experiments, we utilized cemetery, sports, and lakes data from the UCR-STAR dataset to run spatial join queries. Information about these data is given in Table 1. In a series of experiments, different approaches were tested and compared.

The base layer data was divided into 128 equal partitions, and the round robin method was used to test the spatial join performance. The CPU consistently outperformed the BlueField-2 DPU by approximately 2.7 times for the intersection operation. However, as the number of processes increased, BlueField-3 showed greater performance improvement, indicating better scalability for the DPU. The results are shown in Figure 2. Also, using different dataset pairs for the base and query layers, the CPU spatial join exhibited 1.9 times better performance than BlueField-3 for the Sports-Cemetery file pair. This performance advantage slightly increased with larger data pairs like *lakes*. In a similar experiment, the base layer data was divided according to the number of processes. The CPU-DPU performance ratio remained the same, but the performance increase was limited to 1.7 when doubling the number of processes. This limitation was due to the larger R-tree generated with less data splitting, leading to reduced query time. The results are shown in Figure 3.

When the CPU and DPU were used together with dynamic load balancing, the processing time significantly improved. While spatial join took about 76 seconds on

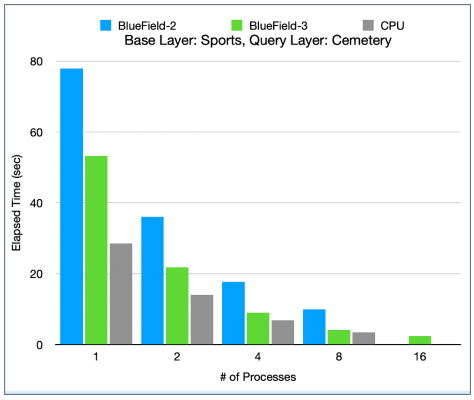

Figure 2: Spatial join query performance of single node using partitioned data (128 partitions).

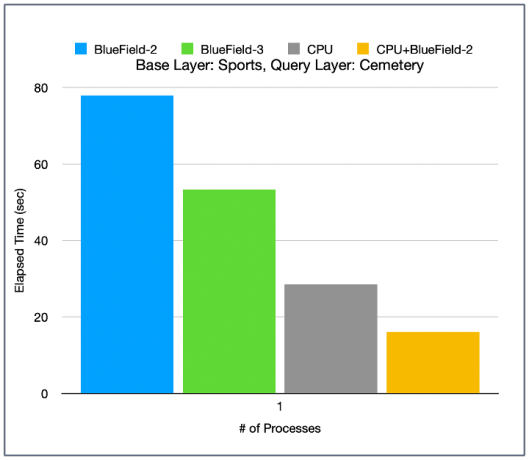

Figure 3: Spatial join query performance on different systems.

BlueField-2 and approximately 28 seconds on the CPU, using both together reduced the processing time to around 16 seconds. So, in the offload mode, we get 1.75X relative speedup by using a DPU.

3. CONCLUSIONS

We show DPU benchmarking results from the domain of geospatial data analytics. For spatial join queries, DPUs can enhance performance by offloading filter and refine tasks from CPU. However, it is important to consider workload characteristics, data properties, and the capabilities of the DPU when determining the effectiveness of offloading tasks. Continued research and benchmarking will help optimize the distribution of workloads and maximize the benefits of using DPUs for geospatial operations. For higher performance, we plan to use the network engines for streaming data and the SIMD units on the ARM processor of the DPU.

4. ACKNOWLEDGEMENT

This work is supported in part by the National Science Foundation CAREER Grant No. 2344578. We thank the HPC-AI Advisory Council testbed for BlueField testing and evaluation.

5. REFERENCES

[1] DPU Filter and Refine GitHub Code. https://github.com/drdkymk/DPU_benchmark/tree/main.

[2] MPI-GIS: An MPI System for Big Spatial Data. https://sc17.supercomputing.org/SC17%20Archive/tech_poster/poster_files/post109s2-file3.pdf.

[3] MPI-GIS GitHub Repository. https://github.com/satishphd/mpigis-lb.

[4] Thor Cluster with Bluefield DPUs. https://hpcadvisorycouncil.atlassian.net/wiki/spaces/HPCWORKS/pages/7864401/Thor.

[5] M. Bayatpour, N. Sarkauskas, H. Subramoni, J. Maqbool Hashmi, and D. K. Panda. Bluesmpi: Efficient mpi non-blocking alltoall offloading designs on modern bluefield smart nics. In *International Conference on High Performance Computing*, pages 18–37. Springer, 2021.

[6] A. Jain, N. Alnaasan, A. Shafi, H. Subramoni, and D. K. Panda. Optimizing distributed dnn training using cpus and bluefield-2 dpus. *IEEE Micro*, 42(2):53–60, 2022.

[7] S. Karamati, C. Hughes, K. S. Hemmert, R. E. Grant, W. W. Schonbein, S. Levy, T. M. Conte, J. Young, and R. W. Vuduc. Smarter nics for faster molecular dynamics: a case study. In *2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 583–594. IEEE, 2022.

[8] Y. Liu and S. Puri. Efficient filters for geometric intersection computations using gpu. In *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, pages 487–496, 2020.

[9] Y. Liu, J. Yang, and S. Puri. Hierarchical filter and refinement system over large polygonal datasets on cpu-gpu. In *2019 IEEE 26th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, pages 141–151. IEEE, 2019.

[10] S. Puri, A. Paudel, and S. K. Prasad. Mpi-vector-io: Parallel i/o and partitioning for geospatial vector data. In *Proceedings of the 47th International Conference on Parallel Processing*, pages 1–11, 2018.

[11] J. Ravi, S. Byna, and M. Becchi. Runway: In-transit data compression on heterogeneous hpc systems. In *2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pages 229–239. IEEE, 2023.

[12] K. K. Suresh, B. Michalowicz, B. Ramesh, N. Contini, J. Yao, S. Xu, A. Shafi, H. Subramoni, and D. Panda. A novel framework for efficient offloading of communication operations to bluefield smartnics. In *2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 123–133. IEEE, 2023.

[13] L. Thostrup, D. Failing, T. Ziegler, and C. Binnig. A dbms-centric evaluation of bluefield dpus on fast networks. In *13th International Workshop on Accelerating Analytics and Data Management Systems Using Modern Processor and Storage Architectures*, 2022.