Accelerating Large Scale GCN Inference on FPGA

Bingyi Zhang, Hanqing Zeng, Viktor Prasanna
University of Southern California, Los Angeles, California
Email: {bingyizh, zengh, prasanna}@usc.edu

***Abstract*—We propose an algorithm-architecture co-optimization framework to accelerate large-scale GCN inference on FPGA. We first perform data partitioning to fit each partition in the limited on-chip memory of FPGA. Then, we use the two-phase pre-processing algorithm consisting of *sparsification* and *node reordering*. The first phase (*sparsification*) eliminates edge connections of high-degree nodes by merging common neighbor nodes. The second phase (*re-ordering*) effectively groups densely connected neighborhoods to improve on-chip data reuse. Incorporating the above algorithmic optimizations, we propose an FPGA architecture to efficiently execute the two key computational kernels of GCN – feature aggregation and weight transformation. We evaluate our design on a state-of-the-art FPGA device. Compared with multi-core and GPU baselines, our design reduces the inference latency by up to 30× and 2× respectively.**

I. Introduction

Graph convolutional network (GCN) [1] plays an increasingly important role in real-world applications [2], such as e-commerce, recommendation systems, unmanned aerial vehicle (UAV), etc. Many of these applications have strict time constraint, therefore low-latency and high-throughput GCN inference is needed. Accelerating GCN inference on FPGA is challenging due to (1) massive input graph size and irregular memory access (2) intensive tensor operations which need large computational resources. To address these challenges, we develop a two-phase pre-processing algorithm to reduce data communication and design a pipeline hardware architecture to achieve low-latency GCN inference.

II. Approach

For the input graph, we perform data partitioning so that each partition can fit in the on-chip memory of FPGA. Based on the partition-based scheme, we propose a two-phase pre-processing algorithm consisting of graph sparsification and node reordering which leads to lower computational complexity and meanwhile increased data locality. First, the graph sparsification phase using redundancy reduction [3] deletes the redundant edges so that the edge connections of the high-degree nodes are reduced. Second, the node reordering phase reorganizes the data layout using the bandwidth reduction algorithm [4]. This step groups adjacent nodes; This increases the data locality and reduces the overall external memory communication.

We design a pipeline hardware architecture to efficiently perform GCN inference. The two major computational kernels–feature aggregation and feature transformation–communicate through the FPGA on-chip memory. We design hardware modules for these kernels. First, exploring the node-level parallelism and feature-level parallelism, the aggregation module has two vector accumulators for feature aggregation. Then, the feature transformation module uses a 2-D systolic array for efficient multiplication of the feature matrix and the weight matrix.

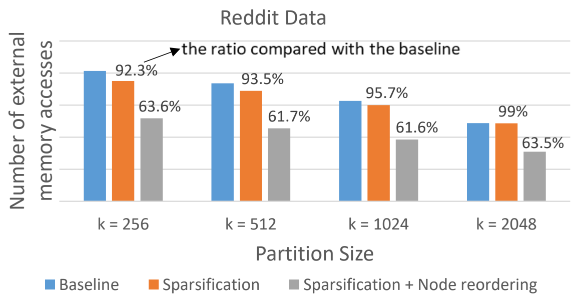

Fig. 1. Number of external memory accesses

III. Evaluation

We evaluate our framework using Flickr, Reddit, and Yelp datasets [3]. We show the effect of two-phase pre-processing algorithm on Reddit in Figure 1. We evaluate the speedup using a three-layer GCN model where each layer has feature length of 128 or 256. We compare our FPGA implementation on Xilinx Alveo U200 with multi-core CPU implementation (Intel Xeon 5120 @2.20 GHz) and GPU implementation (Titan X). The results show that our FPGA implementation is 30× faster than CPU implementation and 2× faster than the GPU implementation.

Acknowledgment

This work was supported by US NSF under grant No. CNS-1643351, Intel Strategic Research Alliance and the Defense Advanced Research Projects Agency under contract No. FA8750-17-C-0086.

References

[1] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[2] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *arXiv preprint arXiv:1901.00596*, 2019.
[3] H. Zeng and V. Prasanna, "Graphact: Accelerating gcn training on cpu-fpga heterogeneous platforms," in *The 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2020, pp. 255–265.
[4] E. Cuthill and J. McKee, "Reducing the bandwidth of sparse symmetric matrices," in *Proceedings of the 1969 24th national conference*. ACM, 1969, pp. 157–172.