2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines

NewGraph: Balanced Large-scale Graph Processing on FPGAs with Low Preprocessing Overheads

Guohao Dai[1,2], Tianhao Huang[1], Yu Wang[1], Huazhong Yang[1], John Wawrzynek[2]
[1]Department of Electronic Engineering, BNRist, Tsinghua University, Beijing, China,
{dgh14, hth14}@mails.tsinghua.edu.cn, {yu-wang, yanghz}@tsinghua.edu.cn
[2]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, USA,
{dgh14, johnw}@eecs.berkeley.edu

I. Introduction

Large-scale graph processing has been widely required in various domains, including social network analysis, neural network modeling, database computing, etc. Performance of large-scale graph suffers from random and unpredictable data access pattern, which leads to drastic bandwidth degradation on caches, DRAMs, and disks. The support for high bandwidth random access makes SRAMs the promising solution for graph processing. Many FPGA based large-scale graph processing systems have been proposed in previous works and taken advantage of the SRAM resources [1], [2], [3], [4], [5], [6].

Although FPGAs provide flexible on-chip SRAM resources for random access in graph processing, the total amount of on-chip SRAMs is limited. Thus, previous works divide vertices and edges into several partitions. Moreover, to fully utilize the parallelism and avoid conflicts among different processing units, graphs are divided into smaller partitions and edges are sorted before processing. While fine-grained data partitioning and allocation in ForeGraph do lead to performance improvements to some extent, open problems remain in such design, including: (1) **Heavy preprocessing overheads.** Sorting and shuffling edges in one partition lead to heavy preprocessing overheads [7]. Moreover, dividing graphs into smaller parts will also increase the preprocessing overhead of indexing different partitions. (2) **Unbalanced workloads among small partitions.** ForeGraph uses the hash-based partitioning to balance each partition. However, the unbalance problem of different partitions will deteriorate when the number of partitions increases. These two problems also restrict the flexibility of graph processing on FPGAs, meaning that these systems are not suitable for dynamic graph operations (e.g., inserting edges). For examples, the time complexity of adding an edge into sorted edges in consecutive addresses is O(#edges), leading to heavy re-preprocessing overhead when graph changes.

In order to tackle two problems above, we propose NewGraph, a crossbar based large-scale graph processing accelerator on FPGA chip. To reduce the number of partitions, we use ultra RAMs (URAMs) instead of block RAMs (BRAMs) to provide the larger on-chip memory space and enlarge the size of each partition. Moreover, by introducing two crossbars in NewGraph, edges are dynamically routed during the runtime. A partition does not need to be divided into smaller ones (like ForeGraph). Pipelined crossbars offload heavy preprocessing to runtime without slowing computations. Such design in NewGraph leads to fewer, larger and more balanced partitions, and helps to improve on-chip computation parallelism.

II. Conclusion

In this paper, we present NewGraph, an on-chip crossbar based large-scale graph processing accelerator on FPGAs. Vertex value is stored on FPGA chip using URAMs (ultra RAMs) at hundreds of Megabits. Two FIFO-based crossbars are adopted to shuffle edges from input data to source and destination vertices, without sorting and further partitioning requirements before processing. In this way, edges only need to be divided into a few partitions (up to three orders of magnitudes less than ForeGraph). Experimental results show that, NewGraph lowers up to 3.99x preprocessing overheads and achieves up to 1.57x speedup compared with ForeGraph.

III. Acknowledgement

This work was supported by National Natural Science Foundation of China (No. 61622403, 61621091), National Key R&D Program of China 2017YFA0207600, Joint fund of Equipment pre-Research and Ministry of Education (No. 6141A02022608), and Beijing National Research Center for Information Science and Technology (BNRist). Guohao Dai is also supported by China Scholarship Council for his current visiting at University of California, Berkeley, CA, USA.

References

[1] Eriko Nurvitadhi and *et al.* Graphgen: An fpga framework for vertex-centric graph computation. In *FCCM*, pages 25–28. IEEE, 2014.
[2] Guohao Dai and *et al.* Fpgp: Graph processing framework on fpga a case study of breadth-first search. In *FPGA*, pages 105–110. ACM, 2016.
[3] Tayo Oguntebi and *et al.* Graphops: A dataflow library for graph analytics acceleration. In *FPGA*, pages 111–117. ACM, 2016.
[4] Guohao Dai and *et al.* Foregraph: Exploring large-scale graph processing on multi-fpga architecture. In *FPGA*, pages 217–226. ACM, 2017.
[5] Shijie Zhou and *et al.* High-throughput and energy-efficient graph processing on fpga. In *FCCM*, pages 103–110. IEEE, 2016.
[6] Jialiang Zhang and *et al.* Boosting the performance of fpga-based graph processor using hybrid memory cube: A case for breadth first search. In *FPGA*, pages 207–216. ACM, 2017.
[7] Jasmina Malicevic and *et al.* Everything you always wanted to know about multicore graph processing but were afraid to ask. In *ATC*, pages 631–643. USENIX, 2017.

2576-2621/18/$31.00 ©2018 IEEE
DOI 10.1109/FCCM.2018.00042