Exploring FPGA Acceleration of Seed Selection on Influence Maximization

Reece Neff, Marco Minutoli, Antonino Tumeo, Mahantesh Halappanavar, Michela Becchi
{reece.neff, marco.minutoli, antonino.Tumeo, mahantesh.halappanavar}@pnnl.gov, {rwneff, mbecchi}@ncsu.edu

NC STATE UNIVERSITY

Introduction

- The Influence Maximization with Martingales (IMM) algorithm is important for determining the most influential seeds in a network.
  - This includes social media influence, spread of misinformation, and even viral spread in an infectious disease network.
- FPGA Acceleration has been explored previously with the Sampling step[1], making Seed Selection the bottleneck in current implementations.
  - Seed Selection is also an attractive acceleration target for its "almost regular" memory access pattern.

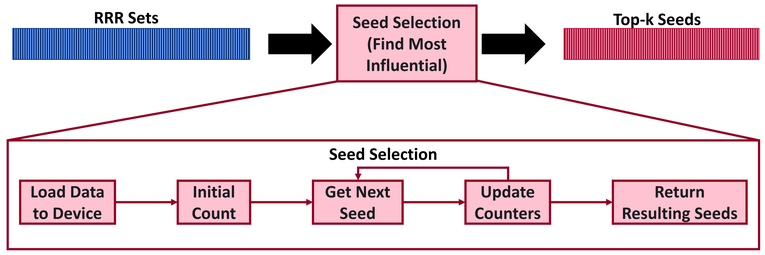
IMM Algorithm Background

- The algorithm is separated into three main parts as shown in the pseudocode

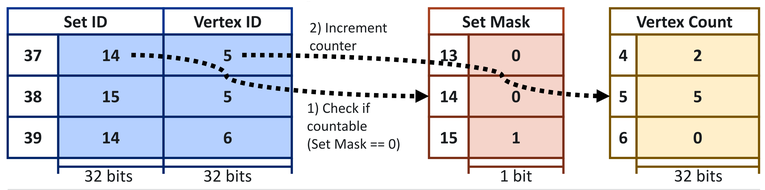
```
Input: G, k, ε
Output: S

begin:
 {R,θ}←EstimateTheta(G,k,ε)
 R←Sample(G,θ-|R|,R)
 S←SelectSeeds(G,k,R)
end
```

- **Estimate Theta**: Estimate the number of sets to be generated by the Sampling stage to achieve a good IMM estimation.
- **Sampling:** Generate several random reverse reachable (RRR) sets determined by Estimate Theta using a given diffusion model. For this work, we use the Linear Threshold Model.
- **Seed Selection:** Find the vertex with the most occurrences in all graphs from the Sampling stage, and record that as the most influential seed. Remove all sets containing this node and repeat the calculation until k vertices have been selected. *This step is the target for FPGA acceleration.*

Seed Selection Overview



Update Counters



Contributions

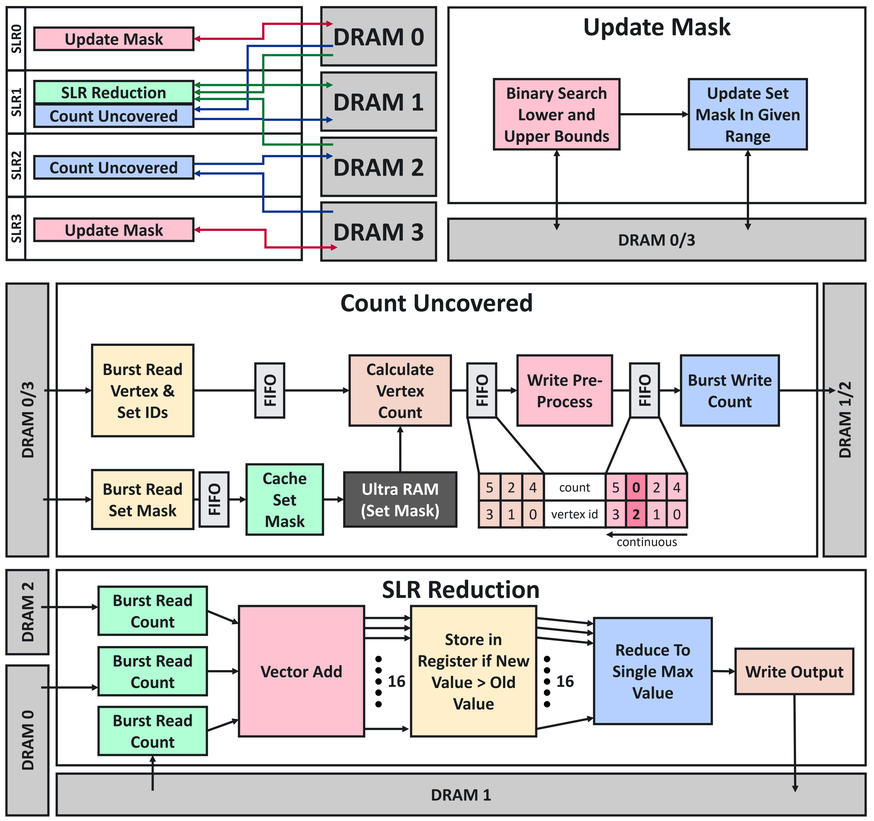
- Implemented FPGA Acceleration for the Seed Selection step
- Outperforms a single CPU core given a favorable workload to vertex ratio, showing promise for acceleration speedup in a multi-core heterogeneous system.

Challenges

- Unlike other histogram computations, an irregular access is required for checking to see if a set has been covered or not.
- The RRR Sets and count outputs are too large to be stored in on-chip memory, so they must be stored in external DRAM.

Proposed Architecture

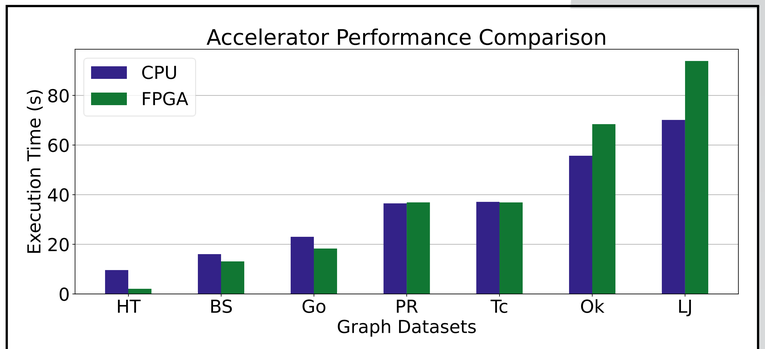
- Utilize port widened bursts for fast and efficient histogram reduction
- Cache the set mask in on-chip Ultra RAMs for fast random accesses
- Burst read RRR sets to be streamed via FIFO into the compute unit and pre-process the output for fast burst writes.

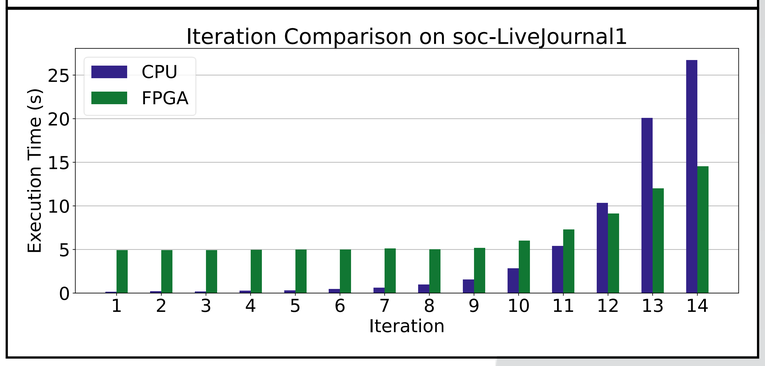


Experimental Setup

- Intel Xeon E5-2637 v4 CPU - 8x32GB DDR4 RAM using OpenMP
- Xilinx Alveo U250 - 4x16GB DDR4 RAM with Vitis Unified Software Platform 2020.2
- We tested our setup on the SNAP data sets containing real-world social networks[2].
  - CPU only with 4 cores using a state-of-the-art parallel IMM algorithm[3].
  - FPGA running in parallel with 3 CPU cores and 1 CPU core managing FPGA data movement. The FPGA is given an equivalent workload as 1 CPU core.

Preliminary Results





Observations

- Up to **4.78x** speedup vs CPU in smaller graphs (cit-HepTh), but only **0.75x** in larger graphs (soc-LiveJournal1).
  - In larger graphs, the overhead for burst writing its entire vertex range in low workloads diminishes the speedup.
- As the workload increases with more seed selection iterations, FPGA overtakes CPU again as the write overhead becomes negligible.

Conclusion and Future Work

- We were able to achieve considerable speedup on smaller sized graphs due to a lower ratio of workload to graph size.
- In the future, we plan to further parallelize the read and calculation units of the Count Uncovered kernel, exposing even better performance.
- Partitioning schemes to lower the number of countable vertices per compute unit can greatly reduce the static overhead on lower workloads.
- Only using the FPGA as an accelerator starting at iterations where it outperforms CPU.

References

[1] R. Neff, M. Minutoli, A. Tumeo, and M. Becchi, "Fpga-accelerated ripples," *in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (SC '21), 2021.
[2] J. Leskovec and A. Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data
[3] M. Minutoli, M. Drocco, M. Halappanavar, A. Tumeo, and A. Kalyanaraman. 2020. CuRipples: Influence Maximization on Multi-GPU Systems. *InProceedings of the 34th ACM International Conference on Super-computing (Barcelona, Spain) (ICS '20)*. Association for Computing Machinery, New York, NY, USA, Article 12, 11 pages. https://doi.org/10.1145/3392717.3392750

9/29/2022 | PNNL-SA-177079

Acknowledgments

The research is supported by the U.S. DOE Exascale Computing Project's (17-SC-20-SC) ExaGraph codesign center at Pacific Northwest National Laboratory (PNNL), by Xilinx, Inc. under Strategic Partnership Project Agreement No. 78023 at PNNL, and by NSF award CNS-1812727 at North Carolina State University.

For additional information, contact:
Reece Neff | (509) 372-6591 | reece.neff@pnnl.gov

U.S. DEPARTMENT OF ENERGY | BATTELLE

PNNL is operated by Battelle for the U.S. Department of Energy

Pacific Northwest NATIONAL LABORATORY
www.pnnl.gov