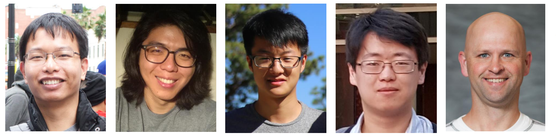
Multi-GPU Graph Analytics

Yuechao Pan, Yangzihao Wang, Yuduo Wu, Carl Yang and John D. Owens, University of California, Davis
{ychpan, yzhwang, yudwu, ctcyang, jowens}@ucdavis.edu



UC DAVIS
UNIVERSITY OF CALIFORNIA

Introduction - about Gunrock

Gunrock is a multi-GPU graph processing library, which targets at:

- **High performance** analytics of large graphs
- **Low programming complexity** in implementing parallel graph algorithms on GPUs

Homepage: **http://gunrock.github.io**

Programming Model

Graph algorithm as a data-centric process

Frontier: compact queue of nodes or edges

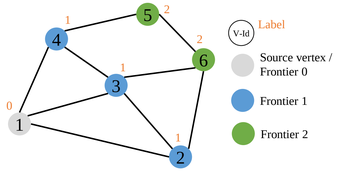
Advance: visit neighbor lists

Filter: select and reorganize

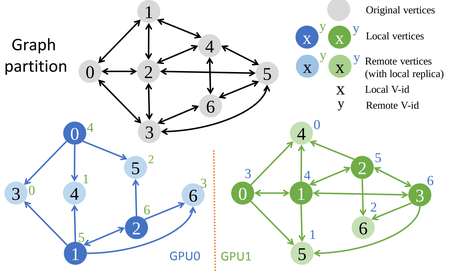
Compute: per-element computation kernels in parallel can be combined with advance or filter

Samples

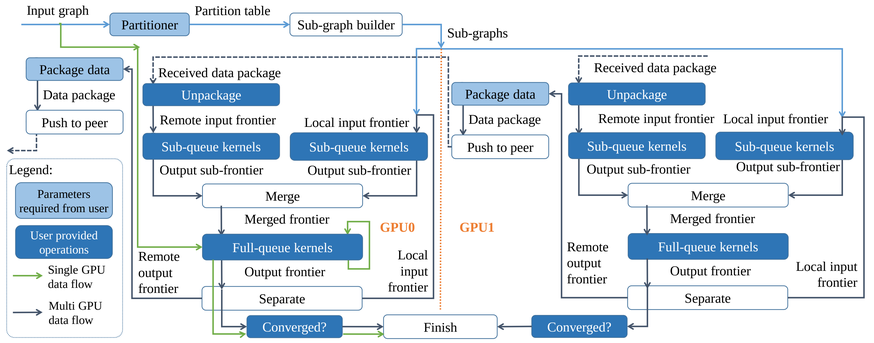
Graph traversal



Graph partition



Multi-GPU Framework



Gunrock's multi-GPU framework aims at:

- **Programmability**: easy to develop graph primitives to support multiple GPUs
  -> hides most implementation details in the framework, and only requires little inputs (what data to exchange, how to combine data, when to stop)
- **Algorithm generality**: support a wide range of graph algorithms
  -> isolates from the actual algorithm implementations
- **Hardware compatibility**: usable on most single node GPU systems
  -> works on any number of GPUs, with or w/o peer GPU memory access
- **Performance**: low runtime, and leverages the underlying hardware well
  -> uses multiple CPU control threads and GPU streams to overlap computations on different portions of frontier, as well as communication
- **Scalability**: scalable in terms of both performance and memory usage
  -> Performs just enough GPU memory (re)allocation to keep usage small

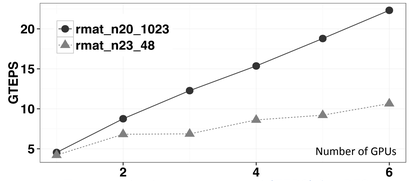
Future Work

- performance analysis and optimization
- extending Gunrock onto **multiple nodes**
- **asynchronized** graph algorithms
- 2D partitioning
- Fixed partitioning
- more algorithms

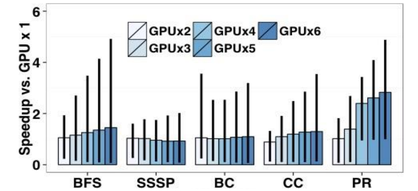
Results

	ref.	ref. hardware	ref. performance	our hardware	our performance
rmat_n20_128	Merrill et al. [3]	4x Tesla C2050	8.3 GTEPS	4x Tesla K40	11.2 GTEPS
rmat_n20_16	Zhong et al. [4]	4x Tesla C2050	15.4 ms	4x Tesla K40	9.29 ms
peak GTEPS	Fu et al. [5]	16x Tesla K20	15 GTEPS	6x Tesla K40	22.3 GTEPS
peak GTEPS	Fu et al. [5]	64x Tesla K20	29.1 GTEPS	6x Tesla K40	22.3 GTEPS

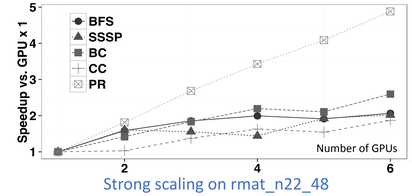
Comparison with previous work on GPU BFS



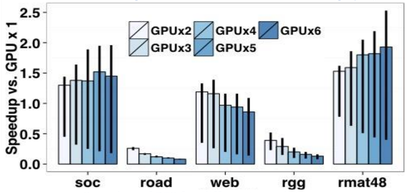
Traversed edges per second (TEPS) for BFS



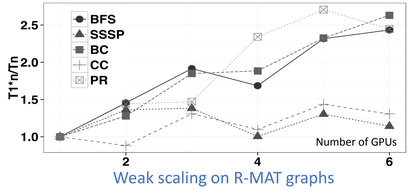
Speedup of different graph algorithms
(error bars showing minima and maxima from different graphs)



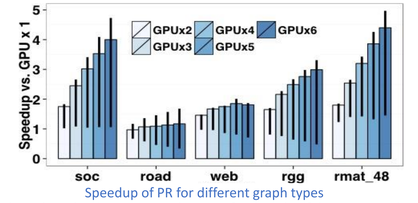
Strong scaling on rmat_n22_48



Speedup of BFS for different graph types
(error bars showing minima and maxima from graphs of the same types)



Weak scaling on R-MAT graphs
(scale 48, each GPU hosting ~180M edges)



Speedup of PR for different graph types
(error bars showing minima and maxima from graphs of the same types)

Acknowledgement

The GPU hardware and cluster access was provided by NVIDIA.
This work was funded by the DARPA XDATA program under AFRL Contract FA8750-13-C-0002 and by NSF awards CCF-1017399 and OCI-1032859.

References

[1] Yuechao Pan, Yangzihao Wang, Yuduo Wu, Carl Yang, and John D. Owens. Multi-GPU Graph Analytics. CoRR, abs/1504.04804, Apr. 2015.
[2] Yangzihao Wang, Andrew Davidson, Yuechao Pan, Yuduo Wu, Andy Riffel, and John D. Owens. Gunrock: A High-Performance Graph Processing Library on the GPU. CoRR, abs/1501.05387v2, March 2015.
[3] D. Merrill, M. Garland, and A. Grimshaw. Scalable GPU graph traversal. In Proceedings of the 17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP '12, pages 117-128, Feb.2012.
[4] J. Zhong and B. He. Medusa: Simplified graph processing on GPUs. IEEE Transactions on Parallel and Distributed Systems, 25(6):1543-1552, June 2014.
[5] Z. Fu, H. K. Dasari, B. Bebee, M. Berzins, and B. Thompson. Parallel breadth first search on GPU clusters. In IEEE International Conference on Big Data, pages 110-118, Oct. 2014.