Testing and Programmability of a Graph Framework

November 3, 2022

1 Introduction

2 Testing of a Graph Framework

2.1 Metrics

Discuss different metrics and their importance:

- Runtime
- Nodes / Edges Visited
- **MTEPS**
- Search depth

2.2 Methods

How we went about collecting these metrics:

- Ability to turn performance evaluation on and off
  - Conditional outside CUDA code
- Collect runtime with performance evaluation turned off
- How to track nodes/edges visited varies by primitive
  - BFS/SSSP - increment a counter each time an edge is visited
  - BC - use frontier
  - PR - constant
- Multiple runs / sources
- Throw out anomalous results

- Other items we track for reproducibility
  - GPU info
  - System info
  - Time
  - Git commit
  - Etc.

2.3 Results

Show and discuss results:

- Charts: Runtime - Essentials vs Gunrock 1.0+ - V100 - various datasets - all primitives
  - Gunrock previously shown to be among the best-in-class [1]
- Charts: MTEPS vs Number of Edges - Essentials - A100 - various datasets - all primitives
- Charts: Search Depth vs Number of Edges - Essentials - A100 - various datasets - all primitives
- Discuss characteristics of datasets where we underperform / outperform

3 Programmability of a Graph Framework

4 Conclusion

References

[1] Yangzihao Wang et al. "Gunrock: GPU Graph Analytics". In: *ACM Transactions on Parallel Computing* 4.1 (Aug. 2017), 3:1–3:49. DOI: 10.1145/3108140. URL: http://escholarship.org/uc/item/9gj6r1dj.