Simplifying Parallel Graph Processing:
Survey of Existing Platforms

Samuel Pollard (spollard@cs.uoregon.edu)

December 1, 2016

This is a survey of existing graph analytics frameworks.

1 Machine Specifications

Table 1 shows the specifications of the research computer (named Arya).

CPU Model	Intel(R) Xeon(R) CPU E5-2699 v3 @ 2.30GHz
CPU Sockets	2
CPU Cores	72
CPU Clock	3600 MHz
RAM Size	251.794Gib
RAM Freq	2133MHz
GPU Model	ASPEED Graphics Family GM204 [GeForce GTX 980]

Table 1: Machine specifications.

2 Performance

Graphalytics without the use of the Granula plugin produces performance measurement in two forms: runtime in seconds and traversed edges per second.

Table 2 lists performance in millions of traversed edges per second (MTEPS) according to the graphalytics output. While this metric is simple to compute it is invariant across algorithms: for example, computing the local clustering coefficient involves traversing each edge multiple times (proportional to the sparsity of the graph), while breadth first search (BFS) traverses each edge exactly once, and on na ive implementations single-source shortest paths (SSSP) may have $O(|E| + |V|^2)$ traversed edges.

3 Graph Processing Taxonomy

This is in the spirit of [1]. Here, "|" means "or" and "+" means "and." FOSS means Free and Open Source Software. The quotes around "yes" for HPC mean that the product claims to be amenable to high performance computing. Whether these actually achieve their goal is one of the purposes of this project.

	openg	powergraph
CDLP	238	1171
LCC	390	1023
PR	234	937
SSSP	3741	29773
WCC	162	685

Table 2: Performance Results for the `dota-league` dataset with 61,670 vertices and 50,870,313 edges. BFS is breadth-first search, SSSP is single-source shortest paths, LCC is local clustering coefficient.

Name	Type	HPC	Parallelism	Target	FOSS	Source	Notes
PowerGraph	Framework	"yes"	both	CPU	yes	[2]	[a]
GraphBIG	Benchmark	"yes"	shared	CPU|GPU	yes	[3]	[b]

Table 3: Tools used for graph processing

[a] The current version is a closed-source product by Turi though PowerGraph v2.2 is on Github.
[b] Only works on Linux.

4 Conclusion

We have presented an updated survey of parallel graph processing frameworks supplementary to [1]. From this, we have selected a representative subset of frameworks on which performance is analyzed and have stored these results in a database. To facilitate parallel graph processing, hardware information and performance results are automatically populated (as were all the tables in this paper). These performance results are then used to provide simple recommendations of the optimally-performing framework given a particular algorithm and problem size.

References

[1] Doekemeijer, N., and Varbanescu, A. L. A survey of parallel graph processing frameworks. Tech. rep., Delft University of Technology, 2014.

[2] Gonzalez, J. E., Low, Y., Gu, H., Bickson, D., and Guestrin, C. Powergraph: Distributed graph-parallel computation on natural graphs. In *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)* (Hollywood, CA, 2012), USENIX, pp. 17–30.

[3] Nai, L., Xia, Y., Tanase, I. G., Kim, H., and Lin, C.-Y. GraphBIG: Understanding graph computing in the context of industrial solutions. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (New York, NY, USA, 2015), SC '15, ACM, pp. 69:1–69:12.