Distributed graph computation

Géraud Le Falher

January 5, 2016

In the 90s, Valiant [1] defined a new model of parallel computation called bulk-synchronous parallel.

> Valiant observes that parallel processing has not made the expected progress in displacing sequential processing in computationally intensive domains. He attributes this failure to the lack of an appropriate bridging model for parallel computation, analogous to the von Neumann model for sequential computation. He then proposes the bulk-synchronous parallel (BSP) model as a candidate bridging model. The key claim concerning BSP is that it provides a fixed intermediate-level model through which high-level programs can be efficiently mapped to diverse machine architectures. The paper supports the claim with evidence of varying sorts: certain high-level constructs (such as memory hashing) and algorithms are efficiently implemented in BSP, and BSP is implemented in two low-level models. Most of the arguments are only sketched, with details left to the references. The BSP model consists of (1) components (each performing processing or memory functions or both), (2) a point-to-point routing mechanism for delivering messages between components, and (3) facilities for periodic synchronization of components. The efficiency of BSP as a bridging model is improved by parallel slackness in programs; that is, the number of virtual processes must be sufficiently larger than the number of processors. Another parameter important to BSP efficiency is the ratio g of total local computation rate to total router delivery rate. Valiant states that existing machines have g-values higher than ideal for BSP and investment in communication hardware to reduce g (and keep it small as machine size increases) will result in more programmable machines. The paper is a brief, well-written introduction to a research topic. Whether a bridging model is needed for practical progress in parallel processing, or whether BSP is a suitable bridging model, remains unclear. The author does not discuss evidence against the existence of bridging models; for instance, parallel and distributed processing form a continuum with vastly different properties from one end to the other. Finally, one might consider the analogous failure of nuclear fusion reactors to replace fission reactors–is something lacking in

our understanding of fusion, or is fusion simply a more difficult engineering problem than was first expected?
Reviewer: Ronald J. Watro

Closer to us, the increasing size of available data has pushed companies to innovate and Google introduced the Pregel model [2].

While Pregel is not publicly available, several open implementation have been described, with various modifications

- Apache Giraph[1]
- Mizan[2] [3]
- GPS[3] [4]

Han *et al.* [5] conducted an experimental comparaison of their performance and usability[4] on four tasks: PageRank, Single Source Shortest Paths, Weakly Connected Components and Minimum Spanning Tree.

Another evaluation was conducted recently by Lu, Cheng, and Wu [6]: "We evaluated the performance of Giraph, GraphLab, GPS, and Pregel+. Our results show that Pregel+[5] [7] and GPS have better overall performance than Giraph and GraphLab."

"Four frameworks - GraphLab, Apache Giraph, Giraph++ and Apache Flink - are used to implement algorithms for the representative problems Connected Components, Community Detection, PageRank and Clustering Coefficients." [**Koch2016**]

Besides Graph500 benchmark and its BFS kernel, Beamer, Asanović, and Patterson [8] introduce more primitives and optimized baselines.

Outside the Pregel family (*is it?*), we can cite:

- GraphLab[6][9, 10, 11, 12] and GraphChi[7] [13].
- Spark [14] and its GraphX extension[8]. Yet another execution engine from Apache is Flink, which feature a graph API[9] and is the results of the Stratoshere[10] research project.
- FlashGraph[11] [15]: using SSD disks instead of memory to store large graph without compromising too much performance.

[1] http://giraph.apache.org/
[2] Actually, I can't find its implementation
[3] http://infolab.stanford.edu/gps/
[4] Along with GraphLab
[5] http://www.cse.cuhk.edu.hk/pregelplus
[6] http://graphlab.org/projects
[7] http://graphlab.org/projects/graphchi.html
[8] https://spark.apache.org/graphx
[9] http://flink.apache.org/docs/0.8/spargel_guide.html
[10] http://stratosphere.eu/project/publications/
[11] `http://www.cs.jhu.edu/~zhengda`

- Using GPU like MapGraph[12] [16], Totem[13] [17] or Gunrock[14] [18]
- Asynchronous [19]
- TinkerPop [15]
- Focusing not on vertex but on neighborhood allow some algorithms to have smaller memory footprint and message overhead than the approaches above. See for instance NScale [20].
- Likewise, in Giraph+ [21] "instead of exposing the view of a single vertex to the programmers, this model opens up the entire subgraph of each partition to be programmed against"
- focusing on graph mining and frequent subgraph handling, Arabesque[16][22]

Stinger http://www.stingergraph.com/index.php?id=publications

More system in session *graph processing* of HPDC'14 The 23rd International Symposium on High-Performance Parallel and Distributed Computing

There's been more than 80 systems proposed in the last ten years, as showed by two surveys: [23, 24].

As hinted by a recent paper[25] and another blogpost[17], one must make sure that graph's size require distributed computation, as it comes with a non negligible overhead. Otherwise, it possible to use single machine, parallelized algorithms. Ligra[18] [26] (and its compressed version [27]) illustrates this approach. Another approach based on Stanford SNAP software is called Ringo[19][28].

On the same topic, Satish *et al.* [29] compare GraphLab, CombBLAS, SociaLite, Galois (single node) and Giraph on the following tasks: PageRank, Breadth First Search, Collaborative Filtering, Triangle Counting. They quantify the overhead introduced by these framework by implementing hand tuned native solution. These abstractions make the tasks slower by a factor of 1.1 to 568 in some Giraph cases. They also propose their own framework, based on sparse linear algebra, which alleviate these slowness by taking advantage of processors parallelism, GraphMat [30].

In other cases, plain MapReduce[31], although it's not specialized in graph handling can also offer competitive performances, for instances to find Connected Components [32, 33, 34]

Then talk about related problem like edges partition [35], graph partition [36, 37, 38, 39], approximating the top-k PageRank vertices [40], counting small motifs [41] or

[12] http://mapgraph.io/
[13] http://netsyslab.ece.ubc.ca/wiki/index.php/Totem
[14] http://gunrock.github.io/gunrock/
[15] http://www.tinkerpop.com/docs/3.0.0.M5/
[16] `http://arabesque.io`
[17] http://aadrake.com/command-line-tools-can-be-235x-faster-than-your-hadoop-cluster.html
[18] https://github.com/jshun/ligra
[19] http://snap.stanford.edu/ringo/

algorithms optimization [42, 43] (for instance using algebrisation [44]) and complexity [45, 46].

Also show some recent applications, for instance in Correlation Clustering [47, 48]. [49]

References

[1] L. G. Valiant, "A bridging model for parallel computation", *Communications of the ACM*, vol. 33, no. 8, pp. 103–111, 1990. DOI: 10.1145/79173.79181. [Online]. Available: http://dl.acm.org/citation.cfm?id=79181.

[2] G. Malewicz *et al.*, "Pregel: a system for large-scale graph processing", in *Proceedings of the 2010 international conference on Management of data - SIGMOD '10*, ACM Press, 2010, p. 135. DOI: 10.1145/1807167.1807184. [Online]. Available: http://dl.acm.org/citation.cfm?id=1807184.

[3] Z. Khayyat *et al.*, "Mizan: a system for dynamic load balancing in large-scale graph processing", in *Proceedings of the 8th ACM European Conference on Computer Systems - EuroSys '13*, ACM Press, 2013, p. 169. DOI: 10.1145/2465351.2465369. [Online]. Available: http://dl.acm.org/citation.cfm?id=2465369.

[4] S. Salihoglu and J. Widom, "Gps: a graph processing system", in *Proceedings of the 25th International Conference on Scientific and Statistical Database Management - SSDBM*, ACM Press, 2013, p. 1. DOI: 10.1145/2484838.2484843. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2484838.2484843.

[5] M. Han *et al.*, "An experimental comparison of pregel-like graph processing systems", *Proceedings of the VLDB Endowment*, no. i, pp. 1047–1058, 2014. [Online]. Available: http://www.vldb.org/pvldb/vol7/p1047-han.pdf.

[6] Y. Lu, J. Cheng, and H. Wu, "Large-scale distributed graph computing systems: an experimental evaluation", *Proceedings of the VLDB Endowment*, vol. 8, no. 3, pp. 281–292, 2015.

[7] D. Yan *et al.*, "Effective techniques for message reduction and load balancing in distributed graph computation", in *WWW'15*, 2015. [Online]. Available: http://arxiv.org/abs/1503.00626.

[8] S. Beamer, K. Asanović, and D. Patterson, "The gap benchmark suite", 2015. [Online]. Available: http://arxiv.org/abs/1508.03619.

[9] Y. Low *et al.*, "Graphlab: a new framework for parallel machine learning", in *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010. [Online]. Available: http://arxiv.org/abs/1006.4990%20http://event.cwi.nl/uai2010/papers/UAI2010%5C_0171.pdf.

[10] J. Gonzalez *et al.*, "Powergraph: distributed graph-parallel computation on natural graphs.", *OSDI*, 2012. [Online]. Available: https://www.usenix.org/system/files/conference/osdi12/osdi12-final-167.pdf.

[11] Y. Low *et al.*, "Distributed graphlab: a framework for machine learning and data mining in the cloud", *Proceedings of the VLDB Endowment*, vol. 5, no. 8, pp. 716–727, 2012. DOI: 10.14778/2212351.2212354. [Online]. Available: http://dl.acm.org/citation.cfm?id=2212354.

[12] Y. Low, "Graphlab: a distributed abstraction for large scale machine learning", PhD Thesis, Carnegie Mellon University, 2013. [Online]. Available: http://reports-archive.adm.cs.cmu.edu/anon/ml2013/CMU-ML-13-104.pdf.

[13] A. Kyrola, G. Blelloch, and C. Guestrin, "Graphchi: large-scale graph computation on just a pc.", *OSDI*, 2012. [Online]. Available: https://www.usenix.org/system/files/conference/osdi12/osdi12-final-126.pdf.

[14] M. Zaharia *et al.*, "Spark: cluster computing with working sets", *HotCloud '10*, 2010. [Online]. Available: http://static.usenix.org/legacy/events/hotcloud10/tech/full%5C_papers/Zaharia.pdf.

[15] D. Zheng *et al.*, "Flashgraph: processing billion-node graphs on an array of commodity ssds", 2014. [Online]. Available: http://arxiv.org/abs/1408.0500.

[16] Z. Fu, M. Personick, and B. Thompson, "Mapgraph: a high level api for fast development of high performance graph analytics on gpus", in *Proceedings of Workshop on GRAph Data management Experiences and Systems - GRADES'14*, ACM Press, 2014, pp. 1–6. DOI: 10.1145/2621934.2621936. [Online]. Available: http://dl.acm.org/citation.cfm?id=2621936.

[17] A. Gharaibeh *et al.*, "Efficient large-scale graph processing on hybrid cpu and gpu systems", 2013. [Online]. Available: http://arxiv.org/abs/1312.3018v2.

[18] Y. Wang *et al.*, "Gunrock: a high-performance graph processing library on the gpu", p. 17, 2015. [Online]. Available: http://arxiv.org/abs/1501.05387.

[19] G. Wang *et al.*, "Asynchronous large-scale graph processing made easy", *Biennial Conference on Innovative Data Systems Research*, vol. 6, 2013.

[20] A. Quamar, A. Deshpande, and J. Lin, "Nscale: neighborhood-centric large-scale graph analytics in the cloud", 2014. [Online]. Available: http://arxiv.org/abs/1405.1499v2.

[21] Y. Tian, A. Balmin, and S. Corsten, "From "think like a vertex" to "think like a graph"", *Proceedings of the VLDB Endowment*, vol. 7, pp. 193–204, 2013. [Online]. Available: http://scholar.google.com/scholar?hl=en%5C&btnG=Search%5C&q=intitle:From+?think+like+a+vertex?+to+?think+like+a+graph?%5C#0.

[22] C. H. C. Teixeira *et al.*, "Arabesque: a system for distributed graph mining - extended version", in *Proceedings of the 25th ACM Symp. on Operating Systems Principles*, 2015. [Online]. Available: http://arxiv.org/abs/1510.04233.

[23] N. Doekemeijer and A. L. Varbanescu, "A survey of parallel graph processing frameworks", Delft University of Technology, Tech. Rep., 2014. [Online]. Available: http://www.pds.ewi.tudelft.nl/research-publications/technical-reports/2014/.

[24] R. R. McCune, T. Weninger, and G. Madey, "Thinking like a vertex: a survey of vertex-centric frameworks for distributed graph processing", 2015. [Online]. Available: http://arxiv.org/abs/1507.04405.

[25] F. McSherry, M. Isard, and D. G. Murray, "Scalability! but at what cost?", in *15th Workshop on Hot Topics in Operating Systems (HotOS XV)*, USENIX Association, 2015. [Online]. Available: https://www.usenix.org/conference/hotos15/workshop-program/presentation/mcsherry.

[26] J. Shun and G. E. Blelloch, "Ligra: a lightweight graph processing framework for shared memory", in *PPoPP '13 Proceedings of the 18th ACM SIGPLAN symposium on Principles and practice of parallel programming*, ACM, 2013, p. 135.

DOI: 10.1145/2517327.2442530. [Online]. Available: http://dl.acm.org/citation.cfm?id=2517327.2442530.

[27] J. Shun, L. Dhulipala, and G. E. Blelloch, "Smaller and faster: parallel processing of compressed graphs with ligra +", in *Proceedings of the IEEE Data Compression Conference*, 2015. [Online]. Available: http://www.cs.cmu.edu/~jshun/ligra+.pdf.

[28] Y. Perez *et al.*, "Ringo: interactive graph analytics on big-memory machines", p. 6, 2015. DOI: 10.1145/2723372.2735369. [Online]. Available: http://arxiv.org/abs/1503.07881.

[29] N. Satish *et al.*, "Navigating the maze of graph analytics frameworks using massive graph datasets", in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '14, ACM, 2014, pp. 979–990. DOI: 10.1145/2588555.2610518. [Online]. Available: http://doi.acm.org/10.1145/2588555.2610518.

[30] N. Sundaram *et al.*, "Graphmat: high performance graph analytics made productive", 2015. [Online]. Available: http://arxiv.org/abs/1503.07241.

[31] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters", *Communications of the ACM*, vol. 51, no. 1, p. 107, 2008. DOI: 10.1145/1327452.1327492. [Online]. Available: http://dl.acm.org/citation.cfm?id=1327492.

[32] H. Kardes and S. Agrawal, "Ccf: fast and scalable connected component computation in mapreduce", in *2014 International Conference on Computing, Networking and Communications (ICNC)*, IEEE, 2014, pp. 994–998. DOI: 10.1109/ICCNC.2014.6785473. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6785473.

[33] L. Qin *et al.*, "Scalable big graph processing in mapreduce", in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data - SIGMOD '14*, ACM Press, 2014, pp. 827–838. DOI: 10.1145/2588555.2593661. [Online]. Available: http://dl.acm.org/citation.cfm?id=2588555.2593661.

[34] R. Kiveris *et al.*, "Connected components in mapreduce and beyond", in *Proceedings of the ACM Symposium on Cloud Computing - SOCC '14*, ACM Press, 2014, pp. 1–13. DOI: 10.1145/2670979.2670997. [Online]. Available: http://dl.acm.org/citation.cfm?id=2670979.2670997.

[35] F. Bourse, M. Lelarge, and M. Vojnovic, "Balanced graph edge partition", pp. 1456–1465, 2014. [Online]. Available: http://research.microsoft.com/pubs/209318/MSR-TR-2014-20.pdf.

[36] A. Buluc *et al.*, "Recent advances in graph partitioning", 2013. [Online]. Available: http://arxiv.org/abs/1311.3144.

[37] C. E. Tsourakakis, "Streaming graph partitioning in the planted partition model", p. 22, 2014. [Online]. Available: http://arxiv.org/abs/1406.7570.

[38] H. Meyerhenke, P. Sanders, and C. Schulz, "Parallel graph partitioning for complex networks", 2014. [Online]. Available: http://arxiv.org/abs/1404.4797.

[39] M. Li, D. G. Andersen, and A. J. Smola, "Graph partitioning via parallel submodular approximation to accelerate distributed machine learning", 2015. [Online]. Available: http://arxiv.org/abs/1505.04636.

[40] I. Mitliagkas *et al.*, "Frogwild! – fast pagerank approximations on graph engines", 2015. [Online]. Available: http://arxiv.org/abs/1502.04281.

[41] E. R. Elenberg *et al.*, "Beyond triangles: a distributed framework for estimating 3-profiles of large graphs", in *KDD'15*, 2015. [Online]. Available: http://arxiv.org/abs/1506.06671.

[42] S. Salihoglu and J. Widom, "Optimizing graph algorithms on pregel-like systems", *Proceedings of the VLDB Endowment*, vol. 7, no. 7, pp. 577–588, 2014. [Online]. Available: http://www.vldb.org/pvldb/vol7/p577-salihoglu.pdf.

[43] ——, "Help: high-level primitives for large-scale graph processing", in *Proceedings of Workshop on GRAph Data management Experiences and Systems - GRADES'14*, ACM Press, 2014, pp. 1–6. DOI: 10.1145/2621934.2621938. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2621934.2621938.

[44] K. Censor-Hillel *et al.*, "Algebraic methods in the congested clique", 2015. [Online]. Available: http://arxiv.org/abs/1503.04963.

[45] H. Klauck *et al.*, "Distributed computation of large-scale graph problems", in *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2014, ch. 28, pp. 391–410. DOI: 10.1137/1.9781611973730.28. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9781611973730.28.

[46] G. Pandurangan, P. Robinson, and M. Scquizzato, "Almost optimal distributed algorithms for large-scale graph problems", 2015. [Online]. Available: http://arxiv.org/abs/1503.02353.

[47] F. Bonchi, A. Gionis, and A. Ukkonen, "Overlapping correlation clustering", *Knowledge and Information Systems*, vol. 35, no. 1, pp. 1–32, 2012. DOI: 10.1007/s10115-012-0522-9. [Online]. Available: http://link.springer.com/10.1007/s10115-012-0522-9.

[48] F. Chierichetti, N. Dalvi, and R. Kumar, "Correlation clustering in mapreduce", in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '14*, ACM Press, 2014, pp. 641–650. DOI: 10.1145/2623330.2623743. [Online]. Available: http://dl.acm.org/citation.cfm?id=2623330.2623743.

[49] L. Quick, P. Wilkinson, and D. Hardcastle, "Using pregel-like large scale graph processing frameworks for social network analysis", in *Proceedings of the 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2012*, 2012, pp. 457–463. DOI: 10.1109/ASONAM.2012.254.