Summary of Proposed Research

Bingjing Zhang

Computer Science Department

Indiana University

Bloomington, IN, USA

Many areas of computer science, such as machine learning and computer vision, are being revolutionized by the incredible volume of data available on the Internet. Unfortunately, scaling up algorithms in these fields is difficult because they require iterative computation at unprecedented scale. Many such kinds of applications are implemented in the MapReduce [1] model for big data processing [2]. However, these MapReduce implementations suffer from repeated input data loading from the distributed file system and slow disk–based intermediate data communication (shuffling) through iterations. But later computers have gained more and more memory capacity therefore many new iterative MapReduce tools [3, 4] are designed to utilize memory for data caching and communication instead of using disk-based operations to improve the performance of iterative algorithms.

However, even using in-memory communication, the performance of iterative applications still suffered the overhead from repeated large intermediate data transferring. In my initial research, I proposed a collective communication model as a generalization of the earlier Twister system that is interoperable between HPC and cloud environments. I studied the problem of large-scale clustering, applying it to cluster features from large collections of 7 million social images, with each feature represented as a point in a high dimensional vector space, into 1 million clusters. The clustering application was split to 5 stages in each iteration: Broadcast, Map, Shuffle, Reduce and Combine. I presented new approaches optimized for large data transfers in Broadcast and Shuffle through integrating capabilities developed by the MPI [5] and MapReduce communities. I demonstrated that a topology-aware and pipeline-based broadcasting method gives similar performance as MPI systems and even has better performance than other (Iterative) MapReduce systems [6, 7, 8].

Now big data tools have become a big family with different computation models: not only MapReduce and iterative MapReduce frameworks process data as key-value pairs but also Pregel [9], Giraph [10] and GraphLab [11, 12, 13] abstract data as a graph and process it in iterations. At the same time, to support different tools, big data systems are split to layers in the architecture. A typical layered architecture is seen in Apache Big data Stack [14] in which a system is split into a resource management layer, a file system layer and a big data processing runtime engine layer and so on.

My previous research has proven collective communication is important for efficient big data processing. But in the runtime engine layer, all tools are designed with fixed data abstraction and have limitations in communication support. Therefore I introduce a separate collective communication layer which provides optimized communication operations on several important data abstractions such as arrays, key-values and graphs, and define an improved general Map-Collective model which serves the diverse communication demands in different parallel applications. These enhancements are designed as plug-ins to Hadoop [15] so they can be used with the rich Apache Big Data Stack. Then for example, Hadoop can do in-memory communication between Map tasks without writing intermediate data to HDFS. With improved expressiveness and excellent performance on collective communication, we can simultaneously support various applications from HPC to Cloud systems together with a high performance Apache Big Data Stack. The product of this research work is Harp project [16]. The related paper is still under review in IC2E conference [17]. All these novelties make Harp different from other research work which try to integrate MPI collective communication operations into Hadoop platform [18, 19, 20].

Harp can be improved in several aspects in my future research work. Currently Harp is designed as a plug-in in Hadoop. But the final goal is to let Harp be a separate communication layer which can be easily integrated to different big data tools. So the design and implementation may still need improving. Besides that, there are three potential research directions:

### Provide proper abstractions and models for different computation and communication patterns

Harp has provided hierarchical data abstraction. With this abstraction, any communication operation is described as data moving on a set of tables and partitions. Currently the operations implemented are abstracted from the data transferring operations in MPI, MapReduce and Pregel. It is still uncertain if this kind of abstraction can cover all types of data transfers in big data applications. Besides, the current execution model in Map-Collective model follows the classic BSP [21] model where there is no overlap between computation and communication. But both MapReduce and Pregel allow part of output data sent out first before all the output data being generated. Such kind of asynchronous execution pattern bring efficiency to the computation. So merging it to Map-Collective model is interesting in research. Furthermore, different from batch processing, real-time big data processing is event-driven. Therefore another potential research is to combine event-driven model with Map-Collective model. Events can be distributed through collective communication operations and they can also be used to trigger new collective communication operations.

### Introduce a fault-tolerant model for Map-Collective model

Current Harp only provides job/iteration level fault tolerance. An iterative application is split to several jobs and each job contains several iterations. Once a job fails, the client restarts the job then only the iterations in that job are required to be re-executed. This kind of failure recovery mechanism has huge overhead. A task-level failure recovery mechanism could have much lower overhead but it is more complicated because re-launched tasks have to synchronize with current running tasks. Additionally, proper interfaces have to be provided to guide users to write fault tolerant Map-Collective applications.

### Support a set of real big data applications

Harp has provided implementations of 3 applications: k-means clustering [22, 23], graph layout algorithm [24], and WDAMDS [25, 26, 27, 28]. All these applications have very good performance results. However, these are only a few examples. An end to end solution to a set of real applications has not been presented yet. This could be one possible research direction to apply Harp to big data applications in domains such as computer vision.

# References

1. J. Dean and S. Ghemawat. “Mapreduce: Simplified data processing on large clusters”. OSDI, 2004.
2. Apache Mahout. <https://mahout.apache.org/>
3. J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H, Bae, J. Qiu, G. Fox. “Twister: A Runtime for iterative MapReduce”. Workshop on MapReduce and its Applications, HPDC, 2010.
4. M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. “Spark: Cluster Computing with Working Sets”. HotCloud, 2010.
5. MPI Forum. “MPI: A Message Passing Interface”. In Proceedings of Supercomputing, 1993.
6. J. Qiu, B. Zhang, “Mammoth Data in the Cloud: Clustering Social Images”. In Clouds, Grids and Big Data, IOS Press, 2013.
7. B. Zhang, J. Qiu. “High Performance Clustering of Social Images in a Map-Collective Programming Model”. Poster in proceedings of ACM Symposium On Cloud Computing, 2013.
8. B. Zhang, J. Qiu, S. Lee, D. Crandall. “Large-Scale Image Classification using High Performance Clustering”. Technical report February 21 2014. <http://grids.ucs.indiana.edu/ptliupages/publications/>
9. Grzegorz Malewicz, et al. “Pregel: A System for Large-scale Graph Processing”. SIGMOD. 2010.
10. Apache Giraph. <https://giraph.apache.org/>
11. Y. Low, et al. “GraphLab: A New Parallel Framework for Machine Learning”. Conference on Uncertainty in Artificial Intelligence, 2010.
12. Y. Low, et al. “Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud”. PVLDB, 2012.
13. J. Gonzalez, et al. “PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs”. OSDI, 2012.
14. S. Kamburugamuve. “Survey of Apache Big Data Stack”. Technical report December 19 2013. <http://grids.ucs.indiana.edu/ptliupages/publications/>
15. Apache Hadoop. <http://hadoop.apache.org>
16. Harp. <http://salsaproj.indiana.edu/harp/index.html>
17. B. zhang, Y. Ruan, J. Qiu. “Harp: Collective Communication on Hadoop”. Internal memo June 2 2014. <http://grids.ucs.indiana.edu/ptliupages/publications/>
18. A. Agarwal et al. “A Reliable Effective Terascale Linear Learning System”. Journal of Machine Learning Research. 2014
19. T. Gunarathne, J. Qiu, D. Gannon, “Towards a Collective Layer in the Big Data Stack”. The proceedings of the 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing Conference, 2014.
20. MPICH2-YARN. https://github.com/clarkyzl/mpich2-yarn
21. L. Valiant. “A Bridging Model for Parallel Computation”. Communications of the ACM, Volume 33 Issue 8, Aug. 1990
22. J. MacQueen, “Some Methods for Classification and Analysis of MultiVariate Observations”. Berkeley Symp. On Mathematical Statistics and Probability, 1967.
23. S. Lloyd. “Least Squares Quantization in PCM”. IEEE Transactions on Information Theory 28 (2): 129–137, 1982.
24. T. Fruchterman, and M. Reingold. “Graph Drawing by Force-Directed Placement”, Software – Practice & Experience 21 (11): 1129–1164, 1991.
25. Y. Ruan. “A Robust and Scalable Solution for Interpolative Multidimensional Scaling With Weighting”. E-Science, 2013.
26. Y. Ruan, G. House, S. Ekanayake, U. Schütte, J. Bever, H. Tang, G. Fox. “Integration of Clustering and Multidimensional Scaling to Determine Phylogenetic Trees as Spherical Phylograms Visualized in 3 Dimensions”. Proceedings of C4Bio of IEEE/ACM CCGrid, 2014.
27. Y. Ruan, et al. “DACIDR: Deterministic Annealed Clustering with Interpolative Dimension Reduction using a Large Collection of 16S rRNA Sequences”. Proceedings of ACM-BCB, 2012.
28. Y. Ruan, et al. “HyMR: a Hybrid MapReduce Workflow System”. Proceedings of ECMLS’12 of ACM HPDC, 2012