High Performance Clustering of Social Images in a Map-Collective Programming Model

Bingjing Zhang
Department of Computer Science
Indiana University Bloomington
zhangbj@indiana.edu

Judy Qiu
Department of Computer Science
Indiana University Bloomington
xqiu@indiana.edu

ABSTRACT

Large-scale iterative computations are common in many important data mining and machine learning algorithms. In most of these applications, individual iterations can be specified as MapReduce computations, leading to the Iterative MapReduce programming model for efficient execution of data-intensive iterative computations interoperably between HPC and cloud environments. The initial work of Iterative MapReduce model [1] focuses on optimization of data flow and reducing data transfer between MapReduce iterations by caching invariant data in the local memory of compute nodes. We observe that a systematic approach to collective communication is essential in many iterative algorithms but is missing in the current model. Thus we generalize the iterative MapReduce concept to Map-Collective noting that large collectives are a distinctive feature of data intensive and data mining applications. This abstract shows the value of Map-Collective model applied to large scale social image clustering problems where 10-100 million images represented as points in a high dimensional (up to 2048) vector space are needed to be divided into 1-10 million clusters.

Major Results:

1) Data mining dominated by collectives with large size 512 MB messages require new technology.
2) New broadcast collective is four times faster than best Java MPI and gives 20% better performance than fastest C/C++ MPI methods and a factor of 5 improvement over non-optimized (for topology) pipeline-based method over 150 nodes.
3) Local aggregation in Map stage reduces the size of 20 TB intermediate data by at least 90%.
4) Our new algorithm scales much better than Spark[2]
5) These communication improvement will be combined with triangle inequality improvements [3] (Elkan's algorithm [4] extended for large problem size)
6) Different optimizations for Azure give Twisister4Azure[5] better performance than current MapReduce and Iterative MapReduce Azure platforms

SOCC '13, Oct 01-03 2013, Santa Clara, CA, USA
ACM 978-1-4503-2428-1/13/10.
http://dx.doi.org/10.1145/2523616.2525952

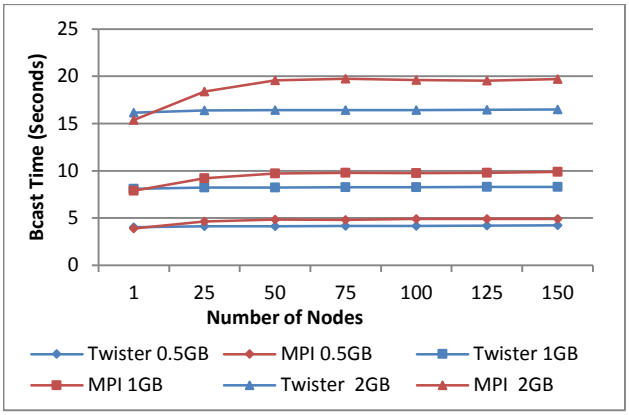

Figure 1. Chain Method/Open MPI MPI_Bcast

Experiment: Clustering problem with 7 million image feature vectors and 1 million clusters. We execute the application using 10000 Map tasks (125 nodes each of which has 8 cores. In broadcasting, the root node broadcasts 512 MB of cluster center data to all nodes therefore the overhead of a sequential broadcasting is substantial.

Novel Collective: We propose chain method, a topology-aware pipeline-based method to accelerate broadcasting by at least a factor of 120 compared with simple algorithm (sequentially sending data from root node to each destination node). Our findings demonstrate that this strategy is outperforms classic C++ MPI implementation by 20% (See Figure 1). More details including Java and Spark comparison can be found at [6].

Aggregation Optimization: Later in Map stage, we use 10000 Map tasks (each node has 80 Map task threads) to process cluster center data and image feature vectors and generates 20 TB of intermediate data (each Map task generates 2GB of data). K-means algorithm requires these intermediate data be aggregated to get the new cluster center data in an iteration. But processing the whole intermediate data on a single node is impossible. We solve this problem by a three-stage aggregation. Firstly, we do local aggregation to the intermediate data generated by all the Map task threads on each node. This can reduce the 20 TB of intermediate data to 250 GB. Secondly, to balance

the workload of the aggregation across nodes, we divide each local aggregated data to 125 partitions and use Shuffle and Reduce to aggregate each partition of the intermediate data on one node. Thirdly, we collect each aggregated data partition from each node back to the root (512 MB in total). Using three-stage aggregation can reduce the size of intermediate data in Shuffle by at least 90%.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems – *Distributed applications.*

General Terms

Algorithms, Measurement, Performance, Design, Experimentation.

Keywords

Social Images, Data Intensive, High Dimension, Iterative MapReduce, Collective Communication

REFERENCES

[1] Jaliya.Ekanayake et. al. Twister: A Runtime for iterative MapReduce, in Proceedings of the First International Workshop on MapReduce and its Applications of ACM HPDC 2010 conference June 20-25, 2010. 2010, ACM: Chicago, Illinois.

[2] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. “Spark: Cluster Computing with Working Sets.” In *HotCloud*, 2010.

[3] J. Qiu, B. Zhang, “Mammoth Data in the Cloud: Clustering Social Images.” In *Clouds, Grids and Big Data*, IOS Press, 2013.

[4] Charles Elkan, “Using the triangle inequality to accelerate k-means.” In *Intl. Conf. on Machine Learning* 2003.

[5] T. Gunarathne, B. Zhang, T.-L. Wu, and J. Qiu. “Scalable Parallel Computing on Clouds Using Twister4Azure Iterative MapReduce.” *Future Generation Computer Systems* (29), pp. 1035-1048, 2013.

[6] Bingying Zhang, Judy Qiu High Performance Clustering of Social Images in a MapCollective Programming Model Technical Report July 2 2013