Dual Direction Big Data Download and Analysis

Jameela Al-Jaroodi
Middleware Technologies Lab.
Bahrain
(971) 50 1380 642
jaljaroodi@gmail.com

Nader Mohamed
College of Information Technology,
UAEU, Al Ain UAE
(971) 50 7533620
Nader.m@uaeu.ac.ae

Abdulla Eid
Department of Mathematics
University of Illinois at Urbana
Champaign, USA
eid1@illinois.edu

ABSTRACT
The term Big Data was recently coined as the amount of generated and stored digital data has grown so rapidly that it has become very hard to store, manage and analyze without coming up with new techniques that can cope with such challenges. Finding innovative approaches to support big data analysis has become a priority as both the research community and the industry are trying to make use of these huge amounts of available data. In this paper we introduce a new approach to enhance the overall big data analysis performance. The approach calls for utilizing data set replication, parallel download, and parallel processing over multiple compute nodes. The main concept calls for simultaneously parallelizing the download of the data (in partitions) from multiple replicated sites to multiple compute nodes that will also perform the analysis in parallel. Then the results are given to the client that requested the analysis.

Categories and Subject Descriptors
C.2.4 **[Computer Systems Organization]**: Computer-Communication Networks, - *Distributed Systems*; D.0 **[Software]**: General; E.0 [Data]: General.

General Terms
Algorithms, Performance, Design, Reliability.

Keywords
Big Data, Parallel Processing, Dual Direction Processing, Data Replication.

1. INTRODUCTION
Big Data refers to the huge data sets obtained from various sources like science experiments, social network activities, telecom data, daily business and financial transactions, and much more [13]. Such data sets pause new technological challenges in terms of storage management and analysis. Although storage capabilities have grown significantly and data stores around the world are available, it is still very hard to capture and store big data efficiently and make it easily accessible. Furthermore, making big data useful requires much more than simple storage. Generally, big data needs to be studied and analyzed to reach some useful conclusions within the particular domain it represents. Computing facilities have also advanced significantly and high performance computing centers are available around the world, many of which may be capable of handling huge amounts of data processing loads. The problem is, in many cases the data is not available where the processing units capable of handling it are. Thus we face two issues: One, how to move the data efficiently to the processing units; and two, are these processing units capable of keeping the full data set?

Considering the strong presence of Cloud Computing, we could assume that we could use available Cloud services to process big data, yet we still have to consider the storage, communication and processing needs. Due to the huge demands imposed by big data processing, in most cases a single Cloud infrastructure will not be able to handle the whole task. One reason for such issue is that the infrastructure is not large enough. Another reason, that is most common, is that such infrastructures are shared among multipl clients' requests and cannot dedicate all its resources to a single task for extended periods of time. Therefore, we must find ways to involve multiple Cloud service providers, concurrent handling of sub tasks and seamless aggregation of generated results. In this paper we propose a model to handle such problems such that we can transfer big data efficiently from multiple replicas to multiple Cloud/Grid sites for processing then aggregate the generated results and send them to the requesting client. Using this approach we achieve two objectives: One, we enhance performance by involving multiple sources of the big data and multiple processing centers. Two, we achieve efficient low overhead load balancing among the different resources in use.

Our technique is based on our earlier work where we introduce the concept of dual direction download and processing [2, 10]. The main idea is to allow the different servers involved in completing a large task to work as independently from each other as possible, while minimizing the controls needed from the client to efficiently complete the task. Therefore, we divide the large task into smaller independent tasks and allow pairs of servers to work on each subtask in opposite directions. Thus each server in a pair will continue working until the client orders it to stop. This way even if the servers in a pair are not performing at the same level, we do not need to know as they both will continue working at their own speeds until the client determines that two consecutive parts of the problem results have arrived. Then the client will ask both servers in the pair to stop. As far as we know, the dual direction method has not been exploited earlier. As we studied the available research results, we could not find any reference to a similar approach in use. Therefore, one of the main contributions of our work is introducing the dual direction technique and using it to combine data transfer and analysis for faster and more efficient processing.

The paper is organized as follows: Section 2 offers some concepts and background information relevant to big data and the proposed approach. Section 3 offers a description of our approach and the types of problems it can be applied to. Section 4 concludes the paper and offers a glimpse of possible future enhancements.

2. BACKGROUND AND RELATED WORK

Big data [3] has emerged as a big concern for the information technology communities as it is becoming harder and harder to process and analyze it efficiently and derive useful results in a timely manner. Over time the amount of data accumulated from various sources has multiplied quickly, yet our capabilities to communicate and process this data has not grown as fast [7]. To further complicate the issue, compute resources are not always available where the data is being stored. In [4] the authors identify the main challenges of big data as: high speed networking, cluster computing programming, extending the reach of cloud computing, machine learning and other data analysis techniques, wide spread deployment, and security and privacy. In industry, various organizations have expressed strong interest in big data and initiated several research and development projects. For example, Intel initiated several research projects through its Intel Science and Technology Center for Big Data [9]. IBM also offers an enterprise class big data platform that allows clients to address the full spectrum of big data business challenges [8]. Several research groups have also investigated the issues of better supporting the processing of big data and improving the performance achievable. For example, in [6] the authors provide a new skyline algorithm SSPL on big data. In [5] the authors offer some models to bridge the gap between big data and compute intensive tools and facilities, while in [1] introduce self adjusting computations to enhance the performance of algorithms such as MapReduce on dynamically changing big data.

In our previous work we investigated several possible approaches to enhance network and computing performance using a new parallelization approach. The use of dual direction processing not only enhances performance, but also provides efficient load balancing among the resources in use. For example, in [2] the approach was introduced to parallelize file downloads (DDFTP) using multiple replicas of the file. In [11] the approach was further refined and adapted to suit the Cloud Computing paradigm and allow for efficient and load balanced parallel download of large data sets. Further, in [10] the same approach was further extended to also offer parallelization of certain types of computational problems (DDPar) using dual direction operations (DDOps). The following section provides our vision of how DDOps can combine data download and computation to achieve better processing of big data across multiple Cloud servers or Grid nodes.

3. THE PROPOSED APPROACH

In this section we offer a description of our proposed approach to combine parallel data download with parallel computations. We foresee that this will allow for better utilization of resources and enhancement in the overall performance. The concept of dual direction operations (DDOps) relies on the idea of pairing resources (file download servers or compute nodes) such that each pair will work in opposite directions until the work assigned to that pair is completed. This way, we do not have to know much about these resources' operational capabilities and conditions before assigning the work, we do not need to interfere with their work during run-time, and the resources need not know anything about each other. Yet, we will still minimize idle time and balance the load across these resources.

To illustrate the concept better, let us take a physical example. Assume we have a very long fence to paint and we have four painters, using this approach we will divide the fence into two sections and for each section, we set two painters at either end of the section and ask them to continue painting until they meet. If the painters all work at the same pace, the work will be completed at the same time and each pair of painters will meet in the middle of their section. However, if each painter works at a different pace, the meeting points will vary (the faster painter will paint more of the sections). In addition, if one pair of painters finishes before the other, we can split the unfinished part of the second section into two and pair one painter to work towards one of the other two painters and the other in the opposite direction. Thus the two painters who are already working will continue normally and the other two will help them until they all meet. This will balance the overall load among the painters without much interference with their work and without any of them having to know what the others are doing (See Figure 1).

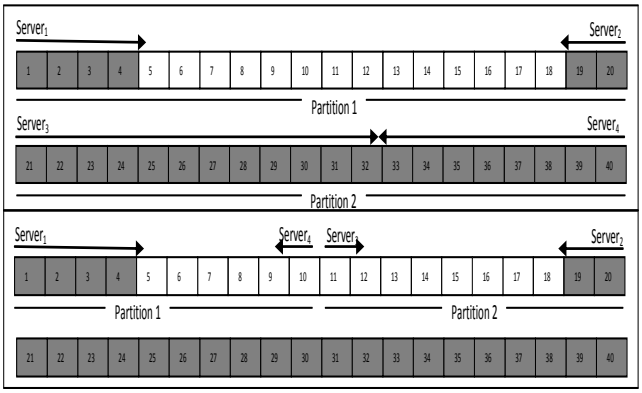

Figure 1. Top: servers 1 and 2 are slow, servers 3 and 4 finish their partition. Bottom: servers 3 and 4 are reassigned to help servers 1 and 2 without disturbing them.

Generally, big data needs to be moved from its data stores to the compute nodes in preparation for processing. As the data grows bigger, it becomes harder to move it fast enough. In addition, some compute nodes may not have the facility or space to hold all the data needed. Therefore, it may be useful to distribute parts of the data to different compute nodes and start the computation as soon as a small portion of that data arrives rather than waiting for the whole set to arrive first. Using the dual direction operations approach (DDOps) we can combine parallel data downloads using DDFTP with dual direction parallel processing using DDPar to achieve better results. Yet there are certain boundaries and conditions that would help make this happen. First, there must be at least two replicas of the data set being used to facilitate the parallel download. Furthermore, having the replicas in different locations will offer better performance as the network conditions at different locations change dynamically, thus we could make use of the best possible network conditions available. Furthermore, the computations that can be supported using this approach must be easily parallelizable with relatively independent subtasks. That is, each compute node should be able to complete its part of the computations without having to consult with or wait for input from other compute nodes. This may limit the scope of application domains that can make use of the approach; however, there is a significantly large set of data intensive computation problems that satisfy these conditions. For example, image processing, pattern recognition, vector/matrix operations, and many more [14, 15].

With these conditions in mind, we can safely assume we can distribute data download and computations across multiple Cloud/Grid nodes. For example, assume that the results of a massive physical experiment were made available as replicas over multiple different data stores distributed around the world. Processing these results at one location will require downloading

the whole data set to the computational cluster or HPC facility such as a computational Grid node. Such facilities usually have limitations in the available storage space that may limit the amount of data that can be kept. In addition, even with the availability of the storage space, downloading the complete data set using conventional transfer models will require a very long time as the network conditions vary and the amount of load and available resources on the data store also vary.

Instead, we can distribute the task across multiple resources that may be commissioned to participate in the task. So, let us assume that we have access to an equal number of HPC facilities that we can use for the analysis (let us refer to such a facility as a compute node, which may include one or more processing units). In this case, we can use the dual direction download to initiate multiple download streams from the replicas of the data set to each of the compute nodes. Depending on the overall capabilities of the data stores and the networks connecting them to the compute nodes, we can open two, four or more parallel streams to download a partition of the data set using DDFTP. At the same time, within the compute nodes, we parallelize the analysis to make use of as many processing units as possible. This way, each compute node will need to hold approximately $1/n^{th}$ of the full size of the data set (n being the number of available replicas) and multiple parallel subtasks will be initiated to handle the analysis (see Figure 2). As the computations take place a controlling process will monitor the operations and forward the generated results to the requesting client. The controller is comprised of a middleware service and a distributed set of software agents on the compute nodes. The middleware service may be residing on one of the compute nodes, the client's site or somewhere else accessible by them. Each agent will monitor the processing progress and the streams that are downloading the data and as soon as two consecutive blocks from the assigned partition are received, it will communicate the information to the middleware service. When the middleware service is updated, it will ask all other agents if they require help. When the responses arrive, the middleware service will select the replica that needs help the most. The agent selected will then initiate a second pair of download streams from the free replica to help speed up the download on its facility. In that case that compute node will be receiving double the download streams and processing is adjusted accordingly. Other agents will continue working normally until another replica becomes free and is assigned to help another agent. Several heuristics can be used to optimize the selection of which agent to help first and how the work is distributed between the original agent and the newly assigned helping agent as explained in [11].

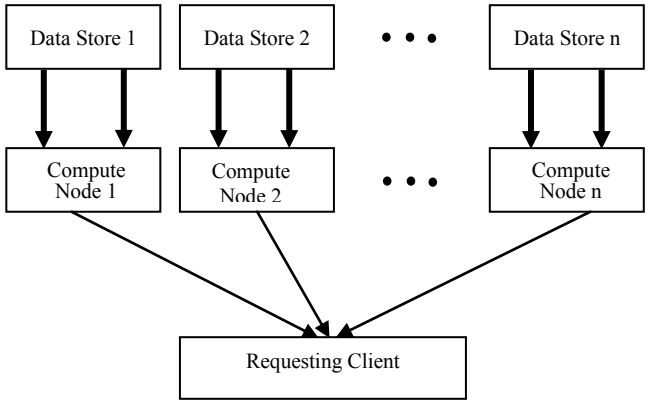

Figure 2. Proposed dual direction operations across multiple data stores and compute nodes.

Based on this set up we can have multiple streams downloading the data from different locations which reduces the download time significantly and overcomes some of the problem that may be encountered on the networks in use. In addition, as DDFTP uses TCP as its base protocol, opening multiple streams within the same data store, will also compensate for the limitations imposed due to the TCP error and flow control mechanisms [12] thus increasing the utilization of the network interfaces. As for the compute nodes, it will be more efficient as the data received is from opposite directions, thus each processing unit will be assigned a different processing direction on the data and parallelization can be enhances as there will be less contention. Overall, the combination of parallel download and parallel processing on the partitions received as they arrive will result in faster processing and more efficient use of storage and compute resources. Furthermore, if the data used is not needed for later access, the compute nodes have the option to only keep small portions at a time throughout the process by discarding the old data after completing the processing on it.

One special case of this approach is when we have one HPC facility with enough compute power to complete the required computations. In this case, we would be extremely slowed down by the download rate if we use conventional download from a single data store. Using the dual direction download from multiple replicas (for example four replicas) we significantly reduce the download time, while allowing the different processing units to use the same dual direction approach to process the data from the different points it is being received. Furthermore, in this case, if the data does not need to be kept after computation, we can store small parts of the data as required by the computations and then remove them to make room for new data blocks that are arriving. Depending on the download to compute rate, the amount of data stored will vary. For example, if computing a block of data requires the same time as it takes to get the block, then a very small number of blocks will be stored (two to 4 per processing unit) just to make sure we do not lose any important blocks. However, if the processing time is longer, then we will need to keep more blocks as the processing continues. The coordination and control in this case is simpler as we only have one compute node to monitor and the middleware service will deal with a single software agent and have better control over the load balancing among the data stores and on the resources used on the compute node.

Using this approach we can achieve efficient load balancing across all available replicas in the data stores such that the computational nodes will always be receiving data until the complete data set is downloaded and processed. Intensive computations will be performed in parallel on the available processors on each site. Therefore, we have an advantage of having multiple data streams downloading the data in parallel thus keeping the processors busy continuously. The requesting client could be residing on one of the computational sites or it could be on a separate site. In either case, the middleware service will control the software agents on the computational sites which in turn will monitor and organize the analysis results and send them to the requesting client's location. To make sure the results are organized and received correctly the middleware service will receive, organize and deliver the final results to the requesting client. Further, if the results are also a relatively large data set, the same two directional data transfer using DDFTP is used to send it to the requesting client's site to speed up the process. Thus, a software agent on the client's site is needed to control the download process and organize the final results.

4. CONCLUSION

With the rapid growth in data generated and digitally stored around the world, it has become important to come up with innovative approaches to help efficiently move and analyze it. Big data comes from multiple sources such as scientific experiments, medical records, social media activities and business and financial transactions. This data could offer a very important insight into possible trends, theories, conclusions, etc. However, doing so requires major changes in how we perform the analysis as conventional methods are not sufficient any more.

In this paper we offer a first look at a possible enhancement that could help speed up big data download and processing. As we developed DDOps in general we used different techniques for different purposes. DDFTP parallelizes downloads from multiple replicas to the client. DDPar parallelizes processing on multiple processors using dual direction as well. However, in this paper we attempt to investigate the benefits of combining the two approaches to help enhance the performance of big data analysis. We assume there exists multiple replicas of the data set needed, which will allow us to use DDFTP to parallelize the download. At the same time, we need an equal number of compute nodes (each of which could have multiple processors to be used). We initiate multiple parallel download streams between data stores and compute nodes such that each compute node will receive and process a partition of the data independently. The results generated at all compute nodes are then collected and delivered to the requesting client.

Using this approach we foresee several benefits. First, we parallelize both the download and processing which speeds up the overall task. We also overcome the dynamic changes in network conditions as we download partitions of the data set from diferent locations. Furthermore, since on each data store we open multiple streams to download the data, we also overcome some of the limitations imposed by TCP. On the compute nodes, as processing is usually fast, it becomes more efficient for the processing units as more data is delivered through the multiple streams instead of waiting for all the data to arrive sequentially. In addition, we now need less storage space on the compute node as each one will only need to store a portion of the data set.

This overview paper offers a starting point for more research and investigation. We intend to carefully and diligently analyze the approach in terms of technical requirements, and design issues. There are several variables that must be studied such as the differences in performance levels among the data stores and the compute nodes. In addition, we will go over the performance, load balancing and fault tolerance issues in more details. Furthermore, we will work on providing a formal evaluation of the approach to show the actual benefits gained when using it.

5. REFERENCES

[1] Acar, U. A. and Y. Chen, "Streaming big data with self-adjusting computation," in proc. workshop on Data driven functional programming (DDFP '13), pp: 15-18, Rome, Italy, January 2013.

[2] Al-Jaroodi, J. and N. Mohamed, "DDFTP: Dual-Direction FTP," in The 11th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid), Newport Beach, California, USA, pp. 504-513, May 23-26, 2011.

[3] Big Data, web link: http://en.wikipedia.org/wiki/Big_data, viewed March 6, 2013.

[4] Bryant, R. E., R. H. Katz and Edward D. Lazowska, "Big-Data Computing: Creating revolutionary breakthroughs in commerce, science, and society," white paper, December 2008, web link: http://cra.org/ccc/docs/init/Big_Data.pdf.

[5] Byun, C., W. Arcand, D. Bestor, B. Bergeron, M. Hubbell, J. Kepner, A. McCabe, P. Michaleas, J. Mullen, D. O'Gwynn, A. Prout, A. Reuther, A. Rosa and C. Yee, "Driving big data with big compute," in proc. IEEE High Performance Extreme Computing Conference (HPEC), Waltham, MA USA, September 2012.

[6] Han, X., J. Li, D. Yang, and J. Wang, "Efficient Skyline Computation on Big Data," in IEEE Transactions on Knowledge and Data Engineering, DoI: 10.1109/TKDE.2012.203, October 2012.

[7] Hilbert, M. and P. López, "The World's Technological Capacity to Store, Communicate, and Compute Information," in Science, Vol. 332, no. 6025, pp:60-65, April 2011.

[8] IBM big data platform, Web Link: http://www-01.ibm.com/software/data/bigdata/enterprise.html, viewed March 2013.

[9] ISTC for Big Data, Web Link: http://istc-bigdata.org/#&panel1-2, viewed March 2013.

[10] Mohamed, N. and J. Al-Jaroodi, "Delay-Tolerant Dynamic Load Balancing," in 13th IEEE International Conference on High Performance Computing and Communications (HPCC-2011), Banff, Canada, pp. 237-245, September 2-4, 2011.

[11] Mohamed, M., J. Al-Jaroodi and A. Eid, "A Dual-Direction Technique for Fast File Downloads with Dynamic Load Balancing in the Cloud," to appear in The Journal of Network and Computer Applications, Elsevier, 2013.

[12] Mohamed, M. J. Al-Jaroodi, H. Jiang, and D. Swanson, "Scalable Bulk Data Transfer in Wide Area Networks," in International Journal of High Performance Computing Applications, Volume 17, No. 3, pp. 237-248, August 2003.

[13] Sources of Big Data, Web Link: http://www.inc.com/ss/best-industries-2012/jj-mccorvey/big-data-5-companies-capitalizing-on-this-business-opportunity#0, viewed March 2013.

[14] Bailey, D.H., E. Barszcz, L. Dagum and H. D. Simon, "NAS parallel benchmark results," in Parallel & Distributed Technology: Systems & Applications, IEEE, Vol. 1, No. 1, pp: 43-51, 1993.

[15] Foster, I., Y. Zhao, I. Raicu and S. Lu, "Cloud Computing and Grid Computing 360-degree compared," in proc. Grid Computing Environments Workshop, pp: 99-106, 2008.