Iterative MapReduce for Azure Cloud

Thilina Gunarathne, Judy Qiu, Geoffrey Fox

School of Informatics and Computing / Pervasive Technology Institute
Indiana University, Bloomington.
{tgunarat, xqiu, gcf}@indiana.edu

ABSTRACT

MapReduce distributed data processing architecture has become the de-facto data-intensive analysis mechanism in compute clouds and commodity clusters, mainly due to its excellent fault tolerance features, scalability, ease of use and the clean programming model. MapReduce for Azure (MR4Azure) is a decentralized, dynamically scalable MapReduce runtime for Azure Cloud infrastructure built using Microsoft Azure cloud infrastructure services. This paper also present AzureTwister, which adds support for optimized iterative MapReduce computations to MR4Azure, based on the concepts of Twister. MR4Azure and AzureTwister take advantage of the scalability, high availability and the distributed nature of cloud infrastructure services to avoid single point of failures, bandwidth bottlenecks and management overheads.

Keywords

MapReduce, Azure, Iterative

1. INTRODUCTION

Cloud computing platforms offer more accessible and horizontally scalable compute power, which can be utilized for large scale computational analysis. While cloud platforms can be used with very low overhead and very low startup costs relative to the traditional clusters, they also present unique challenges for the existing computational frameworks. MapReduce distributed computing framework, introduced by Google, is an emerging data intensive analysis architecture which allows the users to effectively harness the power of cloud computing platforms with ease.

The Microsoft Azure platform is a cloud computing platform which offers on demand computing services such as Windows Azure Compute, Azure Storage BLOB service, Azure queue service, Azure table service, etc. Azure Compute is a platform as a service infrastructure allowing the users to lease hourly charged virtual machine instances in the form of various Roles (eg: Worker Role, Web Role, VM Role). The Azure storage queue is an eventual consistent, reliable, scalable and distributed message queue service, ideal for small, short-lived, transient messages. The Azure Storage BLOB service provides a distributed storage service where users can store and retrieve any type data as a BLOB through a web services interface. Azure Storage Table service provides scalable non-relational structured data storage in a highly available manner. However Azure platform currently do not offer a distributed computing framework, other than the simple queue based model. Goal of MR4Azure and AzureTwister is to facilitate efficient MapReduce & iterative MapReduce computations in the Azure cloud infrastructure.

2. MAPREDUCE FOR AZURE

MR4Azure is a distributed decentralized MapReduce runtime for Windows Azure cloud platform that utilizes Azure cloud infrastructure services. The usage of the cloud infrastructure services allows the MR4Azure implementation to take advantage of the scalability, high availability and the distributed nature of such services to avoid single point of failures, bandwidth bottlenecks and management overheads. MR4Azure overcomes the latencies of cloud services by using sufficiently coarser grained map and reduce tasks. It overcomes the eventual data availability through retrying and by designing the system to not rely on the immediate availability of data to all the workers. MR4Azure uses Azure Queues for map and reduce task scheduling, Azure tables for metadata & monitoring data storage, Azure blob storage for input, output and intermediate data storage and the Window Azure Compute worker roles to perform the computations.

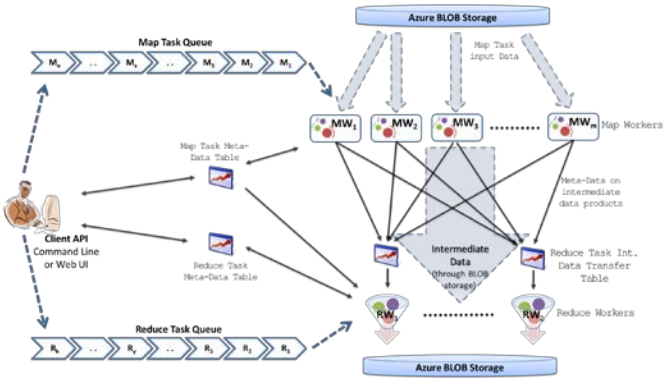

Figure 1 MapReduce for Azure Architecture

In order to withstand the brittleness of cloud infrastructures and to avoid possible single point of failures, MR4Azure was designed as a decentralized control model. MR4Azure also provides users with the capability to dynamically scale up/down the number of computing instances. The map and reduce tasks of the MR4Azure runtime are dynamically scheduled using global queues achieving natural load balance. MR4Azure handles task failures and slow tasks through re-execution and duplicate executions. MapReduce architecture requires the reducers to ensure the receipt of all the intermediate tasks before starting the reduce processing. Since ensuring this is hard in an eventual consistent environment, MR4Azure uses additional data structures for this purpose. In Gunarathne et al.[1] we show that MR4Azure performs comparably to the other MapReduce runtimes.

3. AZURE TWISTER

There exists many data analytics as well as scientific computation algorithms that rely on iterative computations, where each iterative step can be easily specified as a MapReduce computation. Twister4Azure extends the MRRole4Azure to support such iterative MapReduce executions, drawing lessons from Twister [2] iterative MapReduce framework.

Twister4Azure introduces an additional step, Merge, that executes after the reduce step (map->reduce->merge). Merge task receives all the reduce task outputs. Since AzureTwister do not have a

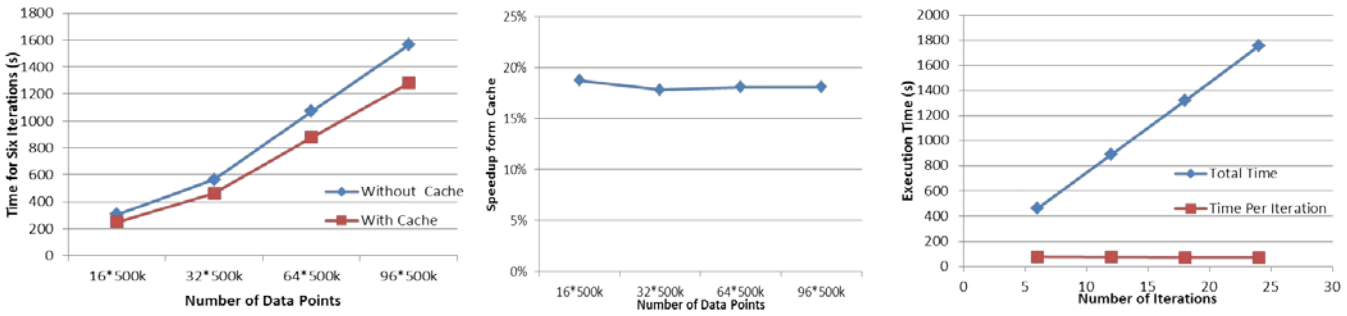

Figure 2. KMeans iterative MapReduce performance using 16 Azure small instances (a) 6 iterations with/without data caching (b) Speedup of using data cache (c) Increasing number of iterations using 32* 500k data points with data caching

always connected client driver due to reliability reasons, the decision to continue with another iteration or finish the job needs to be made in the Merge step. Iterative computations often rely on a set of static data that remain fixed across iteration and a set of transient dynamic data between iterations. Twister4Azure introduces an in memory data cache to store the static data (downloaded from Azure BLOB storage and parsed according to the input format) across the iterations. These data can be reused by subsequent iterations. Each worker role will have one managed DataCache with a given memory size.

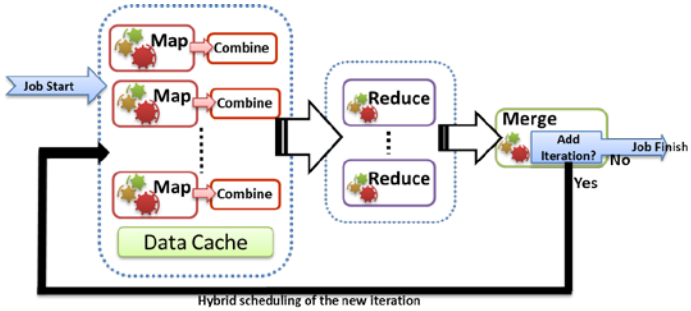

Figure 3 AzureTwister Computation Flow

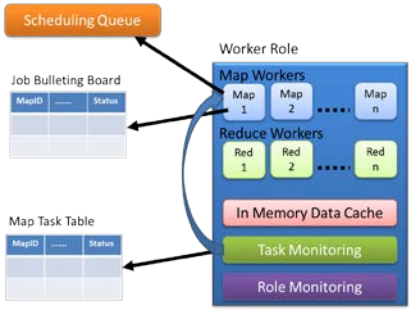

Figure 4 AzureTwister Map Worker Architecture

Since AzureTwister do not have a central coordinator to assign tasks to workers using global knowledge, scheduling of the map tasks to appropriate workers with the cached data products presents a significant challenge. In addition to this, it's desirable to preserve the dynamic scheduling, fault tolerance and other features of MR4Azure in the new scheduling mechanism. In order to address these issues, AzureTwister utilizes a hybrid scheduling approach using a combination of Queues and Azure Tables.

AzureTwister has a special Table called "BulletinBoard", where the tasks will be advertised from second iteration onwards. Map Workers will first check this bulletin board to see any intersection between the data items they have in the DataCache vs the data items needed for the advertised tasks. The first iteration of AzureTwister would be identical to MR4Azure and will get scheduled through Azure queues. From the second iteration onwards, MapWorkers will pick and process the tasks from the bulletin board as mentioned above. In the meantime if a new worker joins, if some worker finishes all its tasks for the cached data, a task or worker fails, then the workers will be able to pick up tasks directly from the queue and will use the AzureTables and the monitoring infrastructure to identify the tasks that are already processed. Task monitoring thread will run in each worker role instance and will monitor the state of the executing map tasks and will get rid of the rare duplicate executions as soon as possible.

Figure 2 presents the performance of KMeans clustering AzureTwister implementation using 16 Azure small instances. Each data set contained 500,000 20-dimensional data points. Cached KMeans computation showed ~18% speedup over non-cached computation, when used with this particular data set. We expect the speedups to be even more significant for more data intensive iterative computations. Figure 2 (c) shows that the AzureTwister was able to sustain the performance with increasing number of iterations.

4. CONCLUSION

AzureTwister and MR4Azure provide MapReduce runtimes for the Windows Azure cloud platform. AzureTwister enables the users to easily and efficiently perform large scale iterative data analysis and scientific computations on Azure. AzureTwister presents a decentralized iterative extension to MapReduce programing model. It also utilizes a novel hybrid scheduling mechanism based on Azure Tables and Queues to take advantage of the caching of static data in iterative computations.

AzureTwister and MR4Azure can also be seen as two examples of architectures that take utilizes cloud infrastructure services effectively to deliver robust and efficient applications.MR4Azure and an alpha version of AzureTwister are available for download at http://salsahpc.indiana.edu/mapreduceroles4azure

5. REFERENCES

[1] Gunarathne, T., Wu,T.L., Qiu, J., and Fox, G.C. 2010. MapReduce in the Clouds for Science. In *Proceedings of CloudCom 2010* Conference (Indianapolis,December 2010)

[2] J.Ekanayake, H.Li, B.Zhang *et al.*, 2010. Twister: A Runtime for iterative MapReduce, in *Proceedings of the First International Workshop on MapReduce and its Applications* of ACM HPDC 2010 conference (Chicago, June 2010)