Twister4Azure: Data Analytics in the Cloud

Thilina Gunarathne, Xiaoming Gao and Judy Qiu, Indiana University

Genome-scale data provided by next generation sequencing (NGS) has made it possible to identify new species and infer the evolutionary relationships between organisms. These techniques have been applied in medicine, such as to screen for recurrent mutations in cancer for use as biomarkers, to classify diseases and suggest treatment. However, developing a complete understanding of a genomic dataset relies heavily on a set of data analytics tools that are tractable to analyze potentially billions of read sequences, where the challenges are both unprecedented at scale and in complexity. Researchers of Informatics and Computing, Biology and Medicine at Indiana University have worked together on a NIH project to investigate emerging scalable computing systems interoperable of Cloud and HPC that meet the common needs across a series of life sciences problems [1]. This effort is exemplified by our bioinformatics pipeline of data storage, analysis, and visualization [2]. At its core, parallel programming paradigms such as Mapreduce and MPI provide powerful large-scale data processing capabilities. Our research is to clarify which applications are best suited for Clouds; which require HPC and which can use both effectively. The long-term goal is enabling cost effective and readily available analysis tool repository that removes the barrier of research in broader community -- making data analytics in the Cloud a reality.

Integrating Scientific Challenge: A Typical Bioinformatics Pipeline

The study of microbial genomes is complicated by the fact that only small number of species can be isolated successfully and the current way forward is metagenomic studies of culture-independent, collective sets of genomes in their natural environments. This requires identification of as many as millions of genes and thousands of species from individual samples. New sequencing technology can provide the required data samples with a throughput of 1 trillion base pairs per day and this rate will increase. A typical observation and data pipeline is shown in Fig. 1 with sequencers producing DNA samples that are assembled and subject to further analysis including BLAST-like comparison with existing datasets as well as clustering, dimension reduction and visualization to identify new gene families. The initial parts of the pipeline fit the Mapreduce (e.g. Hadoop) or many-task Cloud (e.g. Azure) model but the latter stages involve parallel linear algebra for the data mining (e.g. MPI or Twister). It is highly desirable to simplify the construction of distributed sequence analysis pipelines with a unified programming model, which motivated us to design and implement Azure4Twister. Twister and Twister4Azure interpolate between MPI and Mapreduce and, suitably configured, can mimic their characteristics, and, more interestingly, can be positioned as a programming model that has the performance of MPI and the fault tolerance and dynamic flexibility of the original Mapreduce.

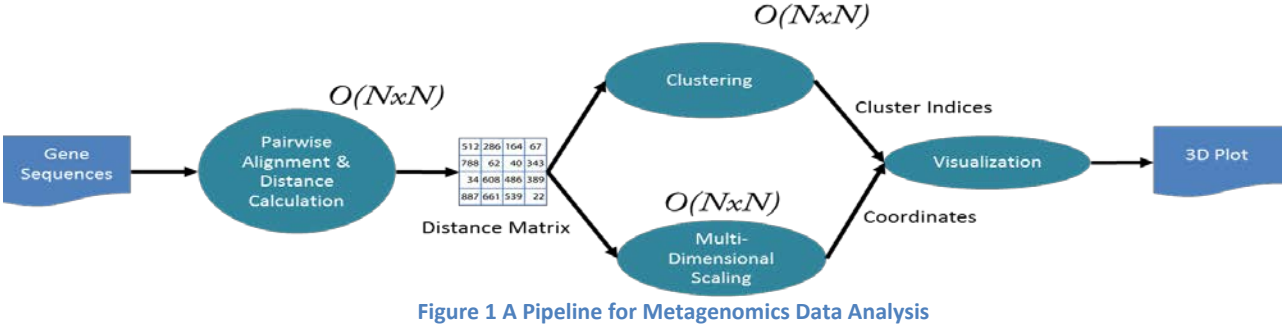

Figure 1 A Pipeline for Metagenomics Data Analysis

Technologies & Applications

Our applications can be classified into three main categories based on their execution pattern, namely pleasingly parallel computations, Mapreduce computations and iterative Mapreduce computations. Twister4Azure [3] distributed decentralized iterative Mapreduce runtime for Windows Azure Cloud, which is the successor to MRRoles4Azure [4] Mapreduce framework and the Classic Cloud pleasingly parallel framework [5], was used as the distributed cloud data processing framework for our scientific computations. Twister4Azure extends the familiar, easy-to-use Mapreduce programming model with iterative extensions, enabling a wide array of large scale iterative as well as non-iterative data analysis and scientific applications to utilize Azure platform easily and efficiently in a fault-tolerant manner, supporting all three categories of applications.

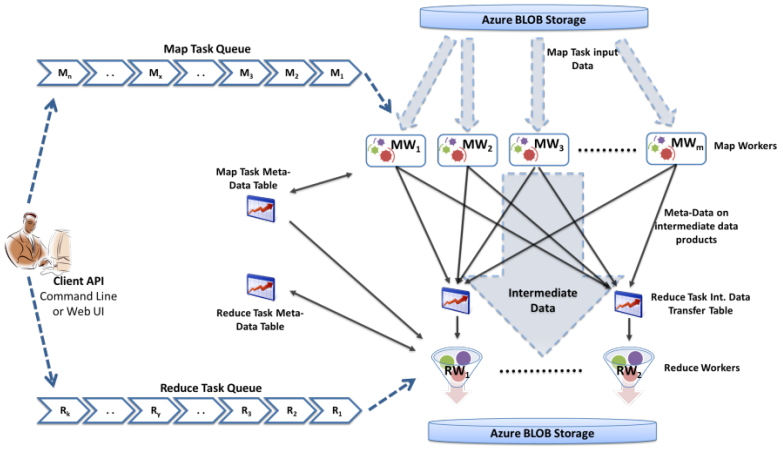

Figure 2 MRRoles4Azure Architecture

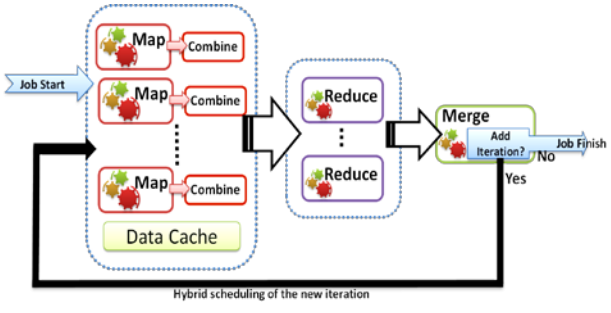

Figure 1 Twister4Azure programming model

Twister4Azure utilize the eventually-consistent, high-latency Azure cloud services effectively to deliver performance comparable to (non-iterative) and outperforming (for iterative computing) traditional Mapreduce runtimes. Twister4Azure supports multi-level caching of data across iterations as well as among workers running on the same compute instance and utilizes a novel hybrid task scheduling mechanism to perform cache aware scheduling with minimal overhead. Twister4Azure also supports data broadcasting, collective communication primitives as well as the invoking of multiple MapRedue applications inside an iteration.

Pleasingly parallel Computations

We performed Cap3 sequence assembly (Fig. 4), BLAST+ sequence search and dimension reduction interpolation computations on Azure using this framework. The performance and scalability are comparable to traditional Mapreduce run times [3]. For Cap3, we assembled up to 4096 FASTA files (each containing 458 reads) in less than one hour using 128 Azure small instances with a cost of around 16$. With BLAST+, the execution of 76800 queries using 16 Azure large instances was less than one hour with a cost of around 12$.

Mapreduce Type Computations

We performed SmithWatermann-GOTOH (SWG) pairwise sequence alignment computations on Azure [3][4] with performance and scalability comparable to the traditional Mapreduce frameworks running on traditional clusters (Fig. 5). We were able to perform up to 123 million sequence alignments using 192 Azure small instances with a cost of around 25$, which was less than the cost it took to run the same computation using Amazon ElasticMapReduce.

Iterative Mapreduce Type Computations

The third and most important category of computation is the iterative Mapreduce type applications. These include majority of data mining, machine learning, dimension reduction, clustering and many more applications. We performed KMeans Clustering (Fig. 6) and Multi-Dimensional Scaling (MDS) (Fig. 7) scientific iterative Mapreduce computations on Azure cloud. MDS consists of two Mapreduce computations (BCCalc and StressCalc) per iteration and contains parallel linear algebra computations as its core.

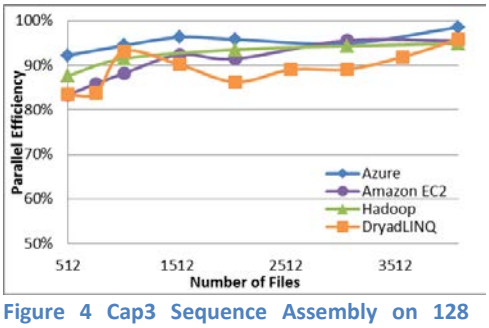

Figure 4 Cap3 Sequence Assembly on 128 instances/cores

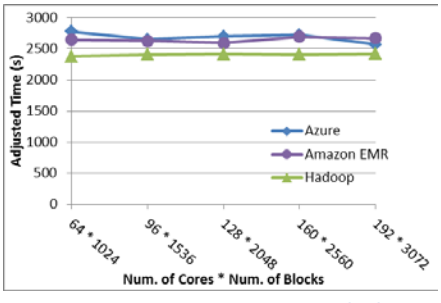

Figure 5 SWG Pairwise Distance Calculation Performance

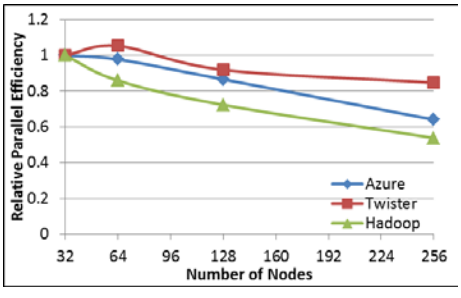

Figure 6 KMeans Clustering Performance

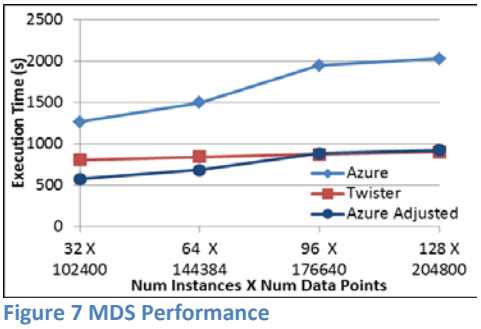

Figure 7 MDS Performance

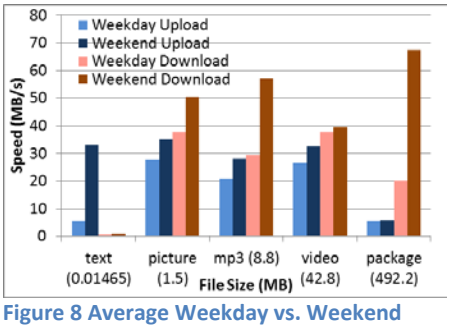

Figure 8 Average Weekday vs. Weekend Transmission Speeds

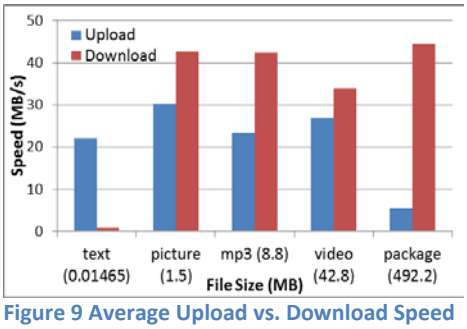

Figure 9 Average Upload vs. Download Speed

Blob storage data transmission performance

We conducted performance tests on the data transmission speed of the Windows Azure Blob service. The tests were done on a virtual machine hosted by the Windows Azure cloud, so the data transmission measured was intra-cloud traffic. Aspects tested include: data transmission speed for objects of different sizes; download vs. upload speed; weekday performance vs. weekend performance, etc. We illustrate our analysis in Fig. 8 and Fig. 9 and summarize three conclusions below.

1. Both upload and download performance fluctuate, but download speed is generally faster than upload, suggesting Azure Blob may be a better option for storing input than saving output. Namely, Azure Blob is not suitable for those scientific applications that generate a huge amount of output.

2. Download speed is generally faster for files of hundreds of MB than for those of tens of MB. So packing scientific input data into larger files may help improve the data transmission efficiency.

3. The top download speed could be high (60+MB/s), but real-time speed fluctuates a lot. Keeping the computation jobs close to where the Azure data blobs will be important for scheduling scientific jobs in Azure.

Some Lessons Learned

The overall major challenge for this research is building a system capable of handling the incredible increases in dataset sizes while solving the technical challenges of portability with scaling performance and fault tolerance using an attractive powerful programming model. Further, these challenges must be met for both computation and storage. Cloud enables persistent storage like Azure Blob. Mapreduce leverages the possibility of collocating data and compute and provides more flexibility to bring data to compute, bring data close to compute or bring compute to data. In concert with virtualization technology, data center model like Azure Cloud is well suited for hosting data analysis for bioinformatics applications as services on demand. Below we give specific comments on Azure.

Azure Programming Model Issues

1. Intermittent performance inconsistency of Azure instances when executing long running memory intensive applications.
   - Fig. 10 shows a sample histogram of computational tasks in a MDS iterative computation. Adjacent blue & red areas represent a single iteration. In ideal conditions, the time taken for tasks in an iteration should be nearly identical across the iteration except for the first iteration. However, after several iterations we started to notice that some tasks randomly take much longer to finish. This result in slowing down of the whole iteration and cause degradation of the computation efficiency as we can see in Fig. 7. We are still investigating the cause for this anomaly and we suspect it to be a behavior of Azure instances after stressing the instance memory for a certain time.

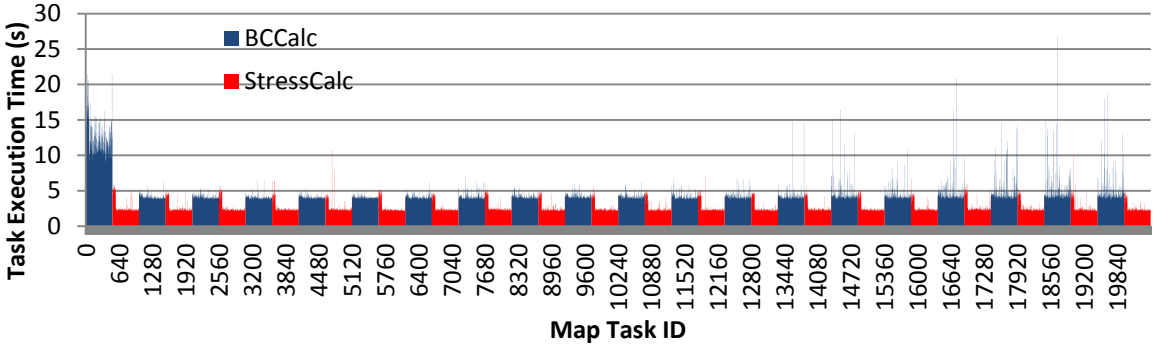

Figure 10 MDS Task Execution Time Histogram for 20 iterations

2. Deployment issues
   - Changing the number of roles in a "running" deployment results in a non-responding deployment. This happened to us several times in the past and lately we avoid this by stopping the deployment before changing the number of roles.
   - Deployment unreliability. Some roles never come to running state or take a very long time to start. This became a major issue for us when running the performance benchmarks. For an example, typically we had to start 132 or more roles to get 128 working roles.
3. API issues
   - More user friendly error messages for the service requests. Most of the error messages we get from Azure services are generic "invalid request" messages, which do not provide information about what really went wrong in the request. This leads to lot of wasted time with trial and error type of development.
   - Better coherent documentation

4. Feature requests
   - Visibility time limit change support for queue messages. This would allow us to support much richer fault tolerance patterns. Amazon SQS supports this through `ChangeMessageVisiblity operation.`
   - Server-side Count or Sum operation for table entries. It's very inefficient to download all the entries in a table to perform a simple sum or a count operation for a single table field.

Acknowledgements

This work was made possible using the computing usage grant provided by Microsoft Azure Cloud. The work reported in this document is partially funded by NIH Grant number RC2HG005806-02. We would also like to appreciate SALSA group for their support.

References

[1] Judy Qiu, Jaliya Ekanayake, Thilina Gunarathne, Jong Youl Choi, Seung-Hee Bae, Hui Li, Bingjing Zhang, Yang Ryan, Saliya Ekanayake, Tak-Lon Wu, Adam Hughes, Geoffrey Fox Hybrid Cloud and Cluster Computing Paradigms for Life Science Applications, *Journal of BMC Bioinformatics*. Open Conferences System BOSC Proceedings (BOSC 2010), VOL.11(Suppl 12): p.S3, August 18, 2010.

[2] Judy Qiu, Jaliya Ekanayake, Thilina Gunarathne, Jong Youl Choi, Seung-Hee Bae, Yang Ruan, Saliya Ekanayake, Stephen Wu, Geoffrey Fox, Mina Rho, Haixu Tang, Data Intensive Computing for Bioinformatics, a book chapter of *Data Intensive Distributed Computing*, ISBN13:978-1-61520-971-2, IGI Publishers, 2012.

[3] T. Gunarathne, B. Zhang, T.-L. Wu, and J. Qiu, "Portable Parallel Programming on Cloud and HPC: Scientific Applications of Twister4Azure," presented at the Portable Parallel Programming on Cloud and HPC: Scientific Applications of Twister4Azure, Melbourne, Australia, 2011.

[4] T. Gunarathne, W. Tak-Lon, J. Qiu, and G. Fox, "MapReduce in the Clouds for Science," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, 2010, pp. 565-572.

[5] T. Gunarathne, T.-L. Wu, J. Y. Choi, S.-H. Bae, and J. Qiu, "Cloud computing paradigms for pleasingly parallel biomedical applications," *Concurrency and Computation: Practice and Experience,* 2011.