Scalable, Secure Analysis of Social Sciences Data on the Azure Platform

Yogesh Simmhan, Litao Deng, Alok Kumbhare, Mark Redekopp and Viktor Prasanna
University of Southern California, Los Angeles CA 90089 USA
{simmhan, litaoden, kumbhare, redekopp, prasanna}@usc.edu

Acknowledgement

This material is based upon work supported by the Department of Energy under Award Number DE-OE0000192.

Disclaimer

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, the Los Angeles Department of Water and Power, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Scalable, Secure Analysis of Social Sciences Data on the Azure Platform

Yogesh Simmhan, Litao Deng, Alok Kumbhare, Mark Redekopp and Viktor Prasanna

University of Southern California, Los Angeles CA 90089 USA

{simmhan, litaoden, kumbhare, redekopp, prasanna}@usc.edu

1. Introduction

Human activity and interaction data is beginning to be collected at population scales through the pervasiveness of social media and willingness of people to volunteer information. This can allow social science researchers to understand and model human behavior with better accuracy and prediction power. Political and social scientists are starting to correlate such large scale social media datasets with events that impact society as evidence abound of the virtual and physical public spaces intersecting and influencing each other [1,2]. Managers of Cyber Physical Systems such as Smart Power Grid utilities are investigating the impact of consumer behavior on power consumption, and the possibility of influencing the usage profile [3]. Data collection is also made easier through technology such as mobile apps, social media sites and search engines that directly collect data, and sensors such smart meters and room occupancy sensors that indirectly measure human activity. These technology platforms also provide a convenient framework for "human sensors" to record and broadcast data for behavioral studies, as a form of crowd sourced citizen science. This has the added advantage of engaging the broader public in STEM activities and help influence public policy.

The social sciences community has traditionally been under-served by the cyber infrastructure and computational sciences, with the more recent focus on informatics and eScience attempting to correct this. Despite the overwhelming data that is newly confronting social sciences, there exist a number of software challenges that need to be addressed to support social informatics, some of which are distinctive to it. Two prominent issues have to do with the ability to easily and securely share data collections between researchers, and the need to analyze large scale, network oriented data. Cloud computing technology has a central role to play in social informatics since this research community is not used to acquiring and managing large clusters nor having easy access to supercomputing centers, let alone parallel programming. Cloud platforms, besides democratizing resource access to such sciences that fall in the tail end of computing, are also well suited for data driven applications with a more approachable management and programming interface.

In this abstract, we highlight two research topics that we are investigating on social informatics using the Azure Cloud platform. These can be summarized by the questions:

- *How can social informatics communities securely collect, host and share personally identifiable datasets with social scientists and researchers at large scale?*
- *What programming models and tools can ease the analysis of large scale social network graphs in a timely and reliable manner?*

Specifically, our work is motivated by consumer behavior analysis problems in the Smart Power Grids domain where customer energy consumption and social interaction information is used to predict future power demand and influence their usage profile for energy sustainability efforts. This work is done in the context of the USC Campus microgrid, which is a cyber-physical-social system, and forms a testbed [4] for the Los Angeles Smart Grid Demonstration Project in the largest public utility in the US.

2. Cryptonite: Secure Repository for Sharing Scientific Datasets

Datasets used by social scientists often have sensitive information about human subjects who participate in studies through online surveys and sensors that record activity. Institutional review boards that regulate scientific research in research institutions have strict guidelines for ensuring that the privacy of human subjects is not compromise through improper storage or sharing with unauthorized research personnel. The relative ease with which data about subjects can be collected through pervasive technology does not in any way negate the need to collect, store and share it securely for research use. In fact, the problem is exacerbated as the size of both data and multi-disciplinary research teams grows. In addition, subjects sharing their personal data for research use may restriction data sharing for specific research purposes or to particular organizations within the institution. At the USC campus, students and staff participating in a campus energy sustainability study may share information on their power consumption, perceptions of comfort, and facility usage information through a smart phone mobile app for research. However, they remain the owners of the data and will provide access to a restricted set of research groups or their friends. Similarly, social sciences researchers may wish to share information aggregated from different subjects with electrical engineering researchers for power demand forecasting. Clouds are well suited for collecting and hosting this data from thousands of concurrent users. However, such as Cloud data repository increases the potential for data leakage. Currently, Service Level Agreements from major Cloud providers have limited liability provisions for data loss or leakage, and the Cloud Security Alliance's 2010 guidance recognizes “Malicious Insiders” and “Data Loss or Leakage” among the top five security threats in Clouds [5].

We have designed a scalable storage repository, *Cryptonite* [6], that is backed by the persistent and available storage offered by “untrusted” public Clouds to securely share datasets among a user community. Our design addresses the shortcomings posed by the two common approaches to secure data sharing. In one approach, symmetric shared keys can be used to encrypt data at the client before posting it to the server, and the secret key shared only with authorized users. Here, the key management complexity becomes unsustainable as the number of shared users increases and does not easily allow revocation of rights to a specific user. Other approaches let the Cloud storage provider or a “trusted” third party enforce the authorizations or access control lists based on authenticated users. However, the presumption of such a “trusted” intermediary having access to plain text data may not hold.

Cryptonite achieves several security and data privacy goals – all of this while utilizing management flexibility, scalability and remote accessibility provided by Cloud storage services [6,7]. Cryptonite only allows authorized clients to have access to plain text data. Neither the Cryptonite service running in the Cloud nor the Cloud storage service sees plain text dataset. Encrypted data stored in the Cloud may be accessed by anyone in the community (since the Cloud provider itself is untrusted). However, only authorized users have access to the decryption keys necessary to extract the plain text data. Authorized users need to maintain only a single global private key, and this gives them the ability to access and decrypt data that they are authorized for. While the Cryptonite service is trusted to perform Cloud storage and retrieval operations on encrypted datasets, an audit trail allows both clients and the service to verify and prove operations.

We leverage a combination of well-defined security and cryptographic techniques in innovative ways for implementing Cryptonite. Public-Private key pairs are used for clients and symmetric keys used for encryption and decryption. We introduce the concept of a *strong box* that is used to store the set of symmetric keys used for all files that share a unique set of authorized users. The strong box itself is encrypted using broadcast encryption with the public keys of all the authorized users in this set. Broadcast encryption allows the strong box to be decrypted using any of the private keys of the authorized users, and thus get access to the symmetric key to be used for en/decrypting a specific file with those permissions.

The Cryptonite service runs within an Azure VM instance as a web role. The service offers RESTful services that support operations such as GET, PUT, POST and DELETE of encrypted files. Its web service interface is similar to the Azure BLOB storage service interface to ease the portability of existing applications that use Cloud storage but required added security and privacy. The Cryptonite service also provides interfaces to store and retrieve the strong box files. Both the symmetric key encrypted data files and the broadcast encrypted strong box files are stored in Azure BLOB storage by the Cryptonite service. A Cryptonite client that runs on the user's desktop is responsible for performing the encryption/decryption of files on the client side before transfer to the Cloud. While this client itself has access to the plaintext file, its implementation can be replaced by the user's own implementation that uses the same sequence of cryptographic operations and web service operations. The end result is a Cloud storage framework similar to DropBox, but with the added measure of security.

In out prototype implementation of Cryptonite, we have leveraged task and data parallelism, and pipelined processing on the .NET client to reduce the overhead of the cryptographic techniques imposed on the effective transfer bandwidth of the data files. In fact, leveraging the multi-core support available on most desktop machines and parallel data transfer mechanisms, we are able to match or perform better than the standard Azure BLOB client library for data transfer bandwidth. This does come at the price of limiting the scalability of the Cryptonite service running in the Cloud that reaches its available bandwidth capacity when storing these data files to Azure BLOB storage.

3. Pillcrow: Scalable Graph Analysis on Cloud Platforms

Graphs data structures are often used in modeling different aspects of social sciences data. Social network graphs are a prime example of this. Data analysis and mining over such graphs can be used for a variety of purposes such as detecting communities and clusters, identifying influencers or early adopters of a technology, and searching for people with a specific skill set in an enterprise.

Betweenness centrality (BC) is an important graph problem for social network analysis [8]. It finds out the central vertices in a graph that appear on shortest paths between all pairs of vertices in the graph. BC has a multitude of uses including traffic networks, epidemiology and intelligence analysis. However, computing the BC for a graph can be memory and compute intensive since its computational complexity and memory growth increases with the number of vertices in the graph. While tightly coupled solutions to BC exist on massively parallel shared memory infrastructure, these are not readily accessible to most researchers. A naïve, distributed form of BC exists for loosely coupled execution on Cloud platforms. This approach operates on each vertex in the graph independently, and performs a breadth first traversal (BFS) rooted at each vertex followed by a reverse traversal that aggregates BC scores. This approach, while partitioning the computational complexity across multiple Cloud VMs, continues to suffer from the memory limitations since it needs to retain the entire graph in memory for the BFS traversal.

In our work, we explore the relative merits of various distributed models for computing BC on the Azure Cloud platform. Three candidate programming models we consider are a naïve *tuplespace* model using master-slave configuration, a *Bulk synchronous parallel* (BSP) model recently resurrected by Pregel [8], and an iterative version of the *Map-Reduce* model.

Our initial study is using the tuplespace model, where a loosely coupled BC algorithm is implemented on Azure [8]. A Web Role instance provides a browser based interface for uploading a graph and submitting a BC job to be performed on it. This graph is saved to the Azure Blob Storage Service and a job message placed on an Azure Job Request queue. A *Manager* worker role instance picks jobs from this queue and creates independent tasks from it by partitioning the vertices in the graph. The tasks are placed in an Azure Task queue on which multiple Worker instances are listening. The worker loads the graph from blob storage and for each vertex in the task, it performs a BFS traversal rooted at the vertex and computes the local BC score for it. The BC scores for all vertices in the task are passed back to the Manager either using an Internal Endpoint or by uploading to a blob result file. The manager collects the results for all tasks and computes a global BC score for each vertex.

Our analysis compares the impact of number of worker instances, task size and the use of Internal Endpoint or Blob storage for result exchange on the total runtime of a job. Graphs of up to 100,000 vertices were evaluated on 16 small worker instances. For graphs with large number of vertices, we see a linear speedup as we increase the number of worker instances. Relative to a local machine, the virtualized Azure worker instances deliver only half the performance. However, use of Cloud resources provides ready access to a large number of workers and enables researchers to tackle large sized graph problems.

Our ongoing work is implementing a BSP version of BC on Azure. This partitions different vertices of the graph across different worker VMs without a global, in-memory view of the graph. Each vertex is aware only of the location of other vertices connected to its edges and a BFS traversal occurs by message passing between vertices through their edges. A similar approach is taken in an iterative Map-Reduce approach. For this, we use Microsoft Research's Daytona Project that offers an iterative Map-Reduce implementation on Azure [10]. Both of these trade-off communication cost to conserve memory and allow them to operate on large scale graphs that will not fit in memory. However, given the trivial computational cost for performing BC, the initial results show the communication cost to be prohibitively costly. We are investigating optimizations for efficient message passing in the Cloud and for determining the appropriate number of concurrent BFS traversals to perform to improve the performance.

4. Future Work

Our Cryptonite server offers enhanced performance for transfer of secure data between client and Cloud despite the security overhead compared to the baseline Azure client. This, however, comes at the cost of scalability. We are examining scalability modes of the Cryptonite server across multiple VMs or means to let the client directly use the BLOB service to leverage its scalability. We are also working on a Java client for Cryptonite to complement the C# version. This will offer better interoperability but will continue to use the REST interface exposed by the Cryptonite service.

We are also conducting a comparative analysis – both empirical and analytical – of the loosely coupled, BSP and Map-Reduce versions of the betweenness centrality algorithm on Azure to identify the short comings and advantages of each of these approaches. This will provide insight on the appropriate graph programming model to leverage for other graph based algorithms on the Cloud. Alternatively, this analysis may potentially expose the lack of a suitable graph programming abstraction and execution model that fits the communication, computation and reliability features of Clouds platforms and offer innovative problems for further research.

5. Acknowledgement

This material is based upon work supported by the National Science Foundation under Award Number CCF-1048311 of the *Computing in the Cloud* initiative, and by the United States Department of Energy under Award Number DEOE0000192. The views and opinions of authors expressed herein do not necessarily state or reflect those of the US Government or any agency thereof. We would also like to thank the Daytona Team at Microsoft Research for their technical support.

6. References

[1] Stephanie Alice Baker. The mediated crowd: New social media and new forms of rioting. *Sociological Research Online*, 16(4):21, 2011.

[2] Serajul I. Bhuiyan. Social media and its effectiveness in the political reform movement in egypt. *Middle East Media Educator*, 1(1):14–20, 2011.

[3] M. Yalcintas. Energy-savings predictions for building- equipment retrofits. *Energy and Buildings*, 40(12):2111–2120, 2008.

[4] Yogesh Simmhan, Viktor Prasanna, Saima Aman, Sreedhar Natarajan, Wei Yin and Qunzhi Zhou, Towards Data-driven Demand-Response Optimization in a Campus Microgrid, *ACM Workshop On Embedded Sensing Systems For Energy-Efficiency In Buildings (BuildSys)*, 2011.

[5] Cloud Security Alliance. Security Guidance for Critical Areas of Focus in Cloud Computing V2.1. Technical report, December 2009.

[6] Alok Kumbhare, Yogesh Simmhan and Viktor Prasanna, Designing a Secure Storage Repository for Sharing Scientific Datasets using Public Clouds, *International Workshop on Data Intensive Computing in the Clouds (DataCloud-SC11)*, 2011

[7] Yogesh Simmhan, Alok Kumbhare, Baohua Cao and Viktor K. Prasanna, An Analysis of Security and Privacy Issues in Smart Grid Software Architectures on Clouds, *IEEE International Cloud Computing Conference (CLOUD)*, 2011

[8] Mark Redekopp, Yogesh Simmhan and Viktor K. Prasanna, Performance Analysis of Vertex-centric Graph Algorithms on the Azure Cloud Platform, *Workshop on Parallel Algorithms and Software for Analysis of Massive Graphs (ParGraph)*, 2011

[9] G. Malewicz, M. Austern, A. Bik, J. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, Pregel: A system for large-scale graph processing, in *ACM SIGMOD*, 2010.

[10] Roger S Barga, Jaliya Ekanayake, and Wei Lu, Project Daytona: Data Analytics as a Cloud Service, in *International Conference of Data Engineering (ICDE)*, 2012.