A-Brain: Using the Cloud to Understand the Impact of Genetic Variability on the Brain

Radu Tudoran, Alexandru Costan, Benoit da Mota, Gabriel Antoniu, Bertrand Thirion

A-Brain: Using the Cloud to Understand the Impact of Genetic Variability on the Brain

Radu Tudoran [1], Alexandru Costan [1], Benoit Da Mota[2], Gabriel Antoniu[1] and Bertrand Thirion[2]

[1]INRIA Rennes - Bretagne Atlantique
[2]INRIA Saclay - Île de France

{*radu.tudoran, alexandru.costan,benoit.da_mota, gabriel.antoniu, bertrand.thirion*}*@inria.fr*

1 Introduction

Joint *genetic* and *neuroimaging* data analysis on large cohorts of subjects is a new approach used to assess and understand the variability that exists between individuals. This approach has remained poorly understood so far and brings forward very significant challenges, as progress in this field can open pioneering directions in biology and medicine. As both neuroimaging- and genetic-domain observations represent a huge amount of variables (of the order of 10^6), performing statistically rigorous analyses on such Big Data represents a computational challenge that cannot be addressed with conventional computational techniques.

In the A-Brain project, we address this computational problem using cloud computing techniques on Microsoft Azure, relying on our complementary expertise in the area of scalable cloud data management and in the field of neuroimaging and genetics data analysis. We have recently proposed a set of algorithms for data management, combining versioning with decentralized metadata management to support scalable, efficient, fine-grain access to massive, distributed Binary Large OBjects (BLOBs) under heavy concurrency - BlobSeer [2]. Based on these algorithms, we introduce a concurrency-optimized data storage system which federates the virtual disks associated to VMs. The purpose of this paper is to provide an overview of the main results achieved in the A-Brain project.

Besides the efficient storage, a data-intensive application like A-Brain also needs appropriate distributed computing frameworks to harness the power of the cloud easily and effectively. The Map-Reduce programming model has arisen as a very effective approach to develop high-performance applications over very large distributed systems such as grids and clouds. Recently, a MapReduce runtime was proposed, built on top of the Azure BLOBs, for data storage and on the Azure roles model of the VM instances for computations: AzureMapReduce [1]. This framework leverages the distributed execution of computation as well as fault tolerance mechanisms available with the Azure roles. However, the architecture of this system relies on costly accesses from VMs to BLOB storage, which incur a high latency. In contrast, we devised an alternative prototype MapReduce framework specifically leveraging the benefits of our storage system, which aggregates the free virtual disks attached to the VMs into a globally-shared object store and efficiently exploits data locality when storing application's data.

Our initial evaluations demonstrate that this solution brings substantial benefits to data intensive applications compared to approaches relying on state-of-the-art cloud object storage. The A-Brain project further investigates on the benefits of integrating our distributed data management system with Microsoft Azure storage services and aims to evaluate the impact of using it on Azure with large-scale application experiments such as the genetics-neuroimaging data comparisons. The project is supervised by the Joint Inria-Microsoft Research Centre.

2 The A-Brain application

2.1 Initial goals

Several brain diseases have a genetic origin, or their occurrence and severity is related to genetic factors. Genetics plays an important role in understanding and predicting responses to treatment for brain diseases like autism, Huntington's disease and many others. Brain images are now used to understand, model and

quantify various characteristics of the brain. Since they contain useful markers that relate genetics to clinical behavior and diseases, they are used as an intermediate between the two. Currently, large-scale studies assess the relationships between diseases and genes, typically involving several hundreds patients per study.

Imaging genetic studies linking functional MRI data and Single Nucleotide Polyphormisms (SNPs) data may face a dire multiple comparisons issue. In the genome dimension, genotyping DNA chips allow to record several hundred thousands values per subject, while in the imaging dimension an fMRI volume may contain 100k-1M voxels (volumetric picture elements). Finding the brain and genome regions that may be involved in this link entails a huge number of hypotheses. A correction of the statistical significance of pair-wise relationships is often needed, but this may reduce the sensitivity of statistical procedures that aim at detecting the association. It is therefore desirable to set up as sensitive techniques as possible to explore where in the brain and where in the genome a significant link can be detected, while correcting for family-wise multiple comparisons (controlling for false positive rate).

2.2 Application description

Sophisticated techniques need to be used in order to perform sensitive analysis on the targeted datasets. *Univariate* studies find a SNP and a neuroimaging trait that are significantly correlated (e.g. the amount of functional activity in a brain region is related to the presence of a minor allele on a gene). With *regression* studies, some sets of SNPs predict a neuroimaging/behavioral trait (e.g. a set of SNPs altogether predict a given brain characteristic), while with *multivariate* studies, an ensemble of genetic traits predict a certain combination of neuroimaging traits.

Let (X, Y) be a joint neuro-imaging dataset, i.e. a set X of brain images after adequate preprocessing and a set Y of genetic variables (e.g. Single Nucleotid Polymorphisms and/or Copy Number Variants of genetic loci), acquired in the same population of S subjects. X is assumed to comprise n_v variables (e.g., one for each location in the brain image domain), while Y comprises n_g variables, both for n_s subjects, resulting in two matrices of real values; typically $n_v \sim 10^6$, $n_g \sim 10^6$. The dataset may also comprise a set Z of behavioral and demographic observations, such as psychological tests or age. In the tests we performed this set was null. These variables are not independent.

In a first phase, we used univariate analysis [3] to detect the correlations, that is, we tested the statistical significance of the correlation or the equivalent association measure of all (x, y) pairs for $(x, y) \in$X$\times$ Y. After performing the necessary computations, the correlations between the two are obtained, giving a matrix of size $n_v \times n_g$ containing the p-values that represent the statistical significance of the association. Several regressions are performed, each giving a set of correlations, from which the most significant ones are kept.

2.3 Challenges

The initial data sets used for testing contain 50K voxels and 500K SNPs for approximately 2000 subjects. However, they are likely to increase, as higher resolution data sets will be available within the A-Brain project. Following the regression stage, all the intermediate data must be stored in order to compare the values of each correlation and filter the most significant p-values. Taking into account that we use matrices of size $n_v \times n_g$ of doubles (8 bytes), the amount of intermediate data that must be stored can reach 1.77 PB ($8 \times 10^4 \times 50 \times 10^3 \times 500 \times 10^3$).

Computational challenges Additionally, in order to obtain results with a high degree of confidence, a number of 10^4 permutations is required, resulting in a total of 2.5×10^{14} associations to be computed. The initial univariate algorithm was able to perform 1.67×10^4 association per second, but after significantly tuning the algorithm it now reaches 1.5×10^6 associations per second. On a single core machine the time estimation to run the new algorithm is ≈ 5.3 years. However, the algorithm is embarrassingly parallel, leaving room for improvement if appropriate resources are available.

Environment requirements The application needs a special scientific environment to be set up: it requires Python with special scientific libraries such as NumPy, SciPy and H5Py to run all the necessary computations. A complex computing environment is also needed for automatically configuring and scheduling the computing jobs, transferring the data between the nodes and retrieving the results.

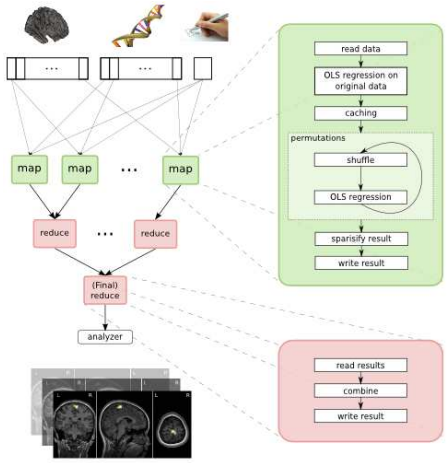

Figure 1: The A-Brain application as a MapReduce workflow

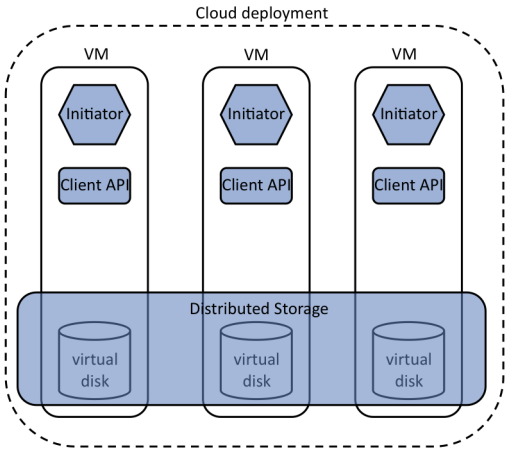

Figure 2: TomusBlobs architecture (the new components in blue).

3 Leveraging Azure clouds for large scale storage and computation

3.1 Motivation

Cloud computing has emerged as a viable alternative to the acquisition and management of physical or software resources. Scientific applications are being ported on clouds to build on their inherent elasticity and scalability. As seen from the previous workload estimations of A-Brain, the application needs to run in parallel on a large set of resources in order to achieve reasonable execution times. Cloud platforms, such as Azure, are an interesting option to tackle this problem.

Our initial estimations show that a core in Azure is able to perform $\approx 1.47 \times 10^6$ associations per second. Our goal is to use the maximum number of cores that can be allocated for a deployment (≈ 350), which would reduce the total computation time of A-Brain from several years to a few days.

From the 1.77 PB of initial data, approximately 10% will be kept as relevant for further processing after the filtering process. As our storage solution relies on federating the virtual disks associated to the VMs, we could use an approximate of 87 TB local storage in Azure VMs and, additionally, some 10 TB of storage in the object based Azure Blobs. These estimations, in conjunction with the role-based programming model provided by Azure, make it an ideal candidate for running A-Brain at large scale.

3.2 Mapping A-Brain onto the Azure cloud

The application can be parallelized using the MapReduce paradigm (Figure 1). We devised a prototype MapReduce framework that exploits the data locality in our storage solution and the computational power of Azure, based on its roles model. In A-Brain, each Mapper takes the same initial data, shuffles it to increase the statistical precision and performs the regression between the 2 sets of data (X, Y) to obtain the correlation between them. This regression represents a series of matrix operations, resulting in a matrix of p-values - the intermediate data. Next, the reduce phase performs a filtering of the intermediate values, generating a unique final result. With this approach, only the correlations with a p-value higher than a specified threshold are considered relevant and kept.

A-Brain is representative of a large class of scientific applications that split an initial large domain into subdomains, each of which is managed by a dedicated process (e.g., image processing, weather simulations, etc). The processes compute different metrics for all the components of their subdomain, and then exchange (parts or all of) these values with each other, performing some associative reductions, that is, binary operations over a set of data in order to produce a single result (e.g., filtering, minimum, selection, etc.).

3.3 Our proposed architecture

We introduce a set of core design principles on which our distributed storage system relies in order to address the previous challenges: *data locality*, by aggregating parts of the storage space from the virtual disks available at no cost, in a shared common pool that is managed in a distributed fashion; *no modication of the cloud middleware*, since the storage system is deployed inside the VMs; *loose coupling between storage and applications*, as we aim at a generic stand-alone storage solution for data intensive processing in Azure.

In [4] we introduced the building blocks of our architecture and showed how to implement them efficiently in such a way that they achieve the previous principles. For convenience, we recall them and give an overview:

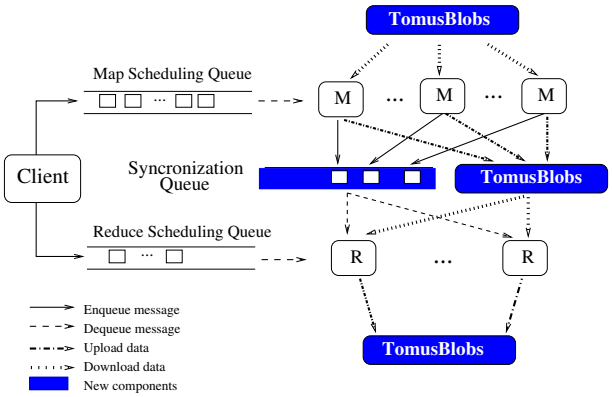

Figure 3: The TomusMapReduce architecture

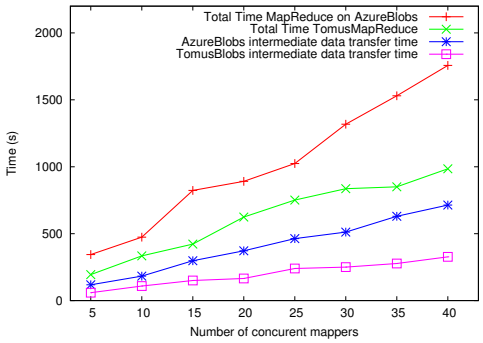

Figure 4: Application execution time and transfer time for intermediate data with AMR and TMR when the number of map jobs is varied

TomusBlobs is a distributed storage system that combines the local storage of the Worker Roles into an uniform storage space, used for sharing and passing data between instances.

TomusMapReduce is a prototype MapReduce framework built on top of TomusBlobs for efficient data processing within scientific applications.

TomusBlobs consists of three loosely-coupled components presented in Figure 2:

- The *Initiator* is a component specific to a particular cloud that has the role to deploy, setup and launch the data management environment. It exposes a generic stub that can be easily implemented and customized for any cloud. The Initiator is able to scale up and down the environment at runtime, by integrating the new added nodes in the system or by seamlessly discarding the deleted ones. Additionally, it is in charge of setting up the environment that is required by the target application (here, install Python and the scientific libraries).
- The *Distributed Storage* has the role of aggregating the virtual disks into a uniform shared storage, which is exposed to applications. It does not depend on a specific solution and hence it can use any distributed file system that could be deployed and executed in a cloud environment. Currently, we are using a file system based on BlobSeer.
- The *Client* encapsulates the API through which the storage is accessed: it implements an interface with a set of primitives, similar to the ones used to access the storage of commercial clouds (Azure BLOBs, Amazon S3).

In order to demonstrate the utility of this storage solution for efficient data processing, we built a prototype MapReduce framework for the Azure cloud - TomusMapReduce (TMR). The proposed framework relies on TomusBlobs to store input, intermediate and final results. Data transfer, protection and confidentiality are enhanced, benefiting from the usage of the local storage instead of the remote Azure Storage. TMR uses a simple scheduling mechanism based on the Azure Queues. The architecture of TMR is presented in Figure 3. Clients are provided with a front-end through which the jobs are submitted. The scheduling of the MapReduce job is performed through small messages that are processed using 3 queues: the *Map Scheduling Queue*, used to submit jobs towards Mappers, the *Reduce Scheduling Queue*, used to submit jobs towards Reducers and the *Synchronization Queue*, used by the Mappers to inform the Reducers about map job completion. The Mappers and the Reducers are implemented as Worker Roles that execute the provided map and reduce functions.

3.4 Practical experience and evaluation

With these tools we were able to execute the neuro-imaging and genetic application in Azure. Additionally, we constructed a demo for offering a visual representation of the experiments performed in this project. The demo and the experiments are submitted through a frontend (Web Role). The web page offers to users a simple interface for setting the parameters of the application, submitting it and retrieving the results. The results are of 2 types:

Scientific - the p-values that give the value (magnitude) of the correlations. Based on the analysis of these values, we aim to determine links between genes and the brain.

Visual - brain images, in which the locations that are involved in a relevant correlation with a certain SNP are highlighted. An example of such a result is provided in Figure 5. The intensity of the color of the

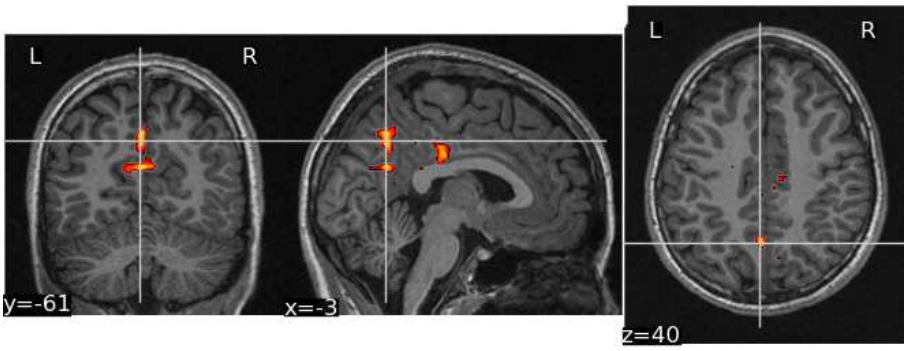

Figure 5: Example of a brain location correlated with a gene, generated by A-Brain

highlighted regions gives the strength of the correlation that was found between that brain location and a certain SNP.

The computation itself is done within the Worker Roles in which TomusMapReduce and TomusBlobs are deployed. So far the experiments scaled fine up to 200 nodes and give promising results. The current simulations used a dataset of thousand of SNPs, analyzed by performing hundreds of permutations. The data used was real genetic and artificial neuro-imaging data, as the neuro-imaging data is not ready yet.

In Figure 4 we present the total execution time of A-Brain with TomusMapReduce and with AzureMapReduce (the existing MapReduce framework for Azure, relying on AzureBlobs for storage). We increased the number of Mappers, keeping the size of the input data constant (1 GB), in an experimental setup with 100 Azure nodes. Data proximity significantly reduces the completion time with TMR, especially for a larger number of Mappers, which can read and write in parallel from the virtual disks; the latencies induced by the remote storage in the case of AMR increase completion time with up to 55%. When isolating the processing times for the intermediate data in the same figure we notice TomusBlobs' support for efficient data handling under heavy concurrency.

4 Final remarks

Porting data intensive applications to the clouds brings forward many challenges in exploiting the benefits of current and upcoming cloud infrastructures. Efficient storage and scalable parallel programming paradigms are some critical examples. We presented our experience with porting A-Brain, a joint genetic and neuro-imaging data analysis application, on the Azure cloud. We developed a cloud storage solution aggregating the virtual disks on the compute nodes, and a prototype MapReduce framework relying on it. Initial evaluations show that it is clearly possible to sustain a high data throughput in the Azure cloud thanks to our low-latency storage.

The next step will consist of designing a performance model for our storage system, which considers the cloud's variability and provides some optimized deployment configurations. We also expect to receive higher resolution data and to scale up the size of the experiments.

References

[1] Thilina Gunarathne, Tak-Lon Wu, Judy Qiu, and Geoffrey Fox. Mapreduce in the clouds for science. In *Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science*, CLOUDCOM '10, pages 565–572, Washington, DC, USA, 2010. IEEE Computer Society.

[2] Bogdan Nicolae, Gabriel Antoniu, Luc Bougé, Diana Moise, and Alexandra Carpen-Amarie. BlobSeer: Next Generation Data Management for Large Scale Infrastructures. *Journal of Parallel and Distributed Computing*, 71(2):168–184, 2011.

[3] Jean-Baptiste Poline, Christophe Lalanne, Arthur Tenenhaus, Edouard Duchesnay, Bertrand Thirion, and Vincent Frouin. Imaging genetics: bio-informatics and bio-statistics challenges. In *19th International Conference on Computational Statistics*, Paris, France, August 2010.

[4] Radu Tudoran, Alexandru Costan, Gabriel Antoniu, and Hakan Soncu. TomusBlobs: Towards Communication-Efficient Storage for MapReduce Applications in Azure. In *12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid'2012)*, Ottawa, Canada, 2012. in press.