Ad hoc Cloud-based Computing Clusters for Big Data Processing

Abdelmounaam Rezgui
Dept. of Computer Science & Engineering, New Mexico Tech, 801 Leroy Place, Socorro, NM 87801
rezgui@cs.nmt.edu

1 Introduction

Big data processing requires partitioning large data sets into several partitions that are then distributed to several computing nodes that operate on these partitions in parallel. This model assumes that the big data application is to be run on a (generic) *distributed* computing cluster[1] that already exists. Moreover, programming frameworks for big data processing usually assume that the data have already been loaded on the nodes of the computing cluster. As a result, current big data processing frameworks are inherently unable to exploit two important optimization opportunities: (i) forming the computing cluster to best suit the features of the given application, and (ii) interleaving the processes of cluster formation, data loading and data processing. We take an orthogonal approach and ask the following two questions: (i) can a computing cluster be dynamically composed from cloud-based (virtualized) resources to run a *specific* big data workload? and (ii) if the answer is yes, how can we speed up big data workloads by interleaving the processes of cluster formation, data loading and data processing?

The focus of our work is on answering the two previous questions. For this, we model the execution of a big data application as a workflow with three tasks: (i) building a computing cluster to support the execution of the application, (ii) loading the input data to the cluster's nodes, and (iii) running the application on the cluster. The problem then becomes to determine the fastest execution of this workflow. We are currently developing efficient solutions to this problem based on deterministic and probabilistic operations research methods. We are designing several experiments to study the performance of the proposed solutions. To evaluate our solutions, it is crucial to have access to a large cloud computing testbed that can scale to the size of big data applications that we are considering.

2 Research Experiments

Given a very large number of computing cluster configurations (number and power of nodes, their proximity to data source(s), their geographic locations, pairwise inter-node bandwidths, etc.) and given a big data workload, our optimization algorithms will answer the following questions: (i) what is the cluster to form that would yield the best performance and (ii) what is the exact choreography that must take place to achieve that performance level, i. e., how the three tasks of cluster formation, data loading, and processing must be interleaved to achieve the best performance. Once we obtain the (theoretical) answers to these questions, we will need to conduct extensive experiments to evaluate our solutions. In particular, we will focus on the following experiments:

- **Optimal cluster dimensioning**: For a given big data workload, our theoretical work will determine the best cluster size (number of nodes) and topology (locations of nodes). In this set of experiments, we will validate our analytical results by (manually) building a large number of heterogeneous clusters and running several big data workloads on each of them. We will then compare the performance achieved using these clusters with the performance obtained using the cluster configuration computed by our algorithms.
- **Prediction of the cost of the dynamic creation of big data processing clusters**: The overhead of building a computing cluster becomes a crucial factor when clusters are formed on the fly. The purpose of this set of experiments is to assess the accuracy of our model for predicting the time needed to set up a computing cluster for a given big data workload. These experiments will consist of running code that automatically builds a cluster of (possibly geographically distant) computing nodes and measuring the time needed before the cluster is entirely functioning and ready to run the given workload.
- **Determining the best choreography:** The three tasks of cluster formation, data loading, and processing may be interleaved in a large number of ways. For example, one scheme could be to start a first node, start transferring data to that first node while starting the second node, start transferring data to that second node while starting the third node, etc. Each interleaving alternative will yield different performance numbers. The ultimate goal of this research is to determine which of these alternatives is the best. In this set of experiments,

[1] Throughout this document, we will use the term "cluster" to refer to a *distributed cluster*, i. e., a network of computing nodes that may not be geographically co-located.

we will run big data workloads using different interleaving schemes and compare the performance of these schemes.

3 Experimental Requirements

For most researchers, computing, networking, and storage resources available on-premises are often insufficient to conduct large scale experiments. For years, "community" testbeds have been a viable recourse. Several community computing/networking testbeds have been deployed in the past. They provided the research community with invaluable computing and networking resources. Examples include Emulab[2], PlanetLab[3], SURAGrid[4], GENI[5], FutureGrid[6], and Open Science Grid (OSG)[7]. These testbeds have been instrumental in enabling both small and large scale experiments in computing and networking. For example, we recently used GENI resources in a research paper ([1]) where we evaluated the performance of Linux I/O schedulers for big data workloads. Based on our experience with GENI and considering the requirements of our current research, we believe that the following are desirable features in future cloud testbeds such as Chameleon and CloudLab:

- *Application Programming Interface*: We plan to conduct long experiments that will have to be run many times. The testbed should expose an API that can be used to script an entire experiment. Using such an API, researchers would be able to specify an entire experiment once and then run the script on the testbed several times until the desired results are obtained. For example, the testbed's API should provide *programmatic* mechanisms to create a computing cluster with a given size and/or given topology, select specific sites for nodes or entire clusters, *dynamically* resize a cluster, initiate intensive data transfers to and from nodes in the cluster, etc.
- *Automatic Virtual Machine Deployment:* The testbed should make it possible for researchers to configure a virtual machine according to the needs of their experiments and then automatically (possibly concurrently) deploy it on one or more of the testbed's nodes.
- *Location*: In the case of our experiments, it is important to know the topology of the cluster formed as a result of running a given script. Therefore, the testbed must make this information visible, i. e., when a user requests the formation of a cluster, the user must be able to know the geographical location of each node.
- *Transparent Fault Tolerance*: Due to their length, we anticipate that experiments will often run un-attended. While an experiment is running on a cluster, one or more nodes may fail. Failures should be transparent to applications (and experiments).
- *Reproducibility*: The testbed should make it possible to accurately reproduce the same conditions that characterized the execution of a given script. This would make it meaningful to compare the results of two executions of the same script.

4 A Short Bio

I am an assistant professor in the Department of Computer Science and Engineering at New Mexico Tech (NMT), a Hispanic Serving Institution (HSI). I am the director of NMT's Cloud Computing and Big Data (C2BD) Lab. I received my PhD in Computer Science from Virginia Tech, USA. My main research interests are cloud computing and big data. I authored or co-authored over 60 papers in top journals and conferences. I serve on the editorial board of several journals and on the program committee of several conferences. This year, I am the chair of the big data track of the 11th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA'2014).

5 Reference

[1] A. Rezgui, M. White, S. Rezgui, and Z. Malik, Evaluation of Linux I/O Schedulers for Big Data Workloads, 4th IEEE International Conference on Big Data and Cloud Computing (BDCloud), Dec. 3-5, 2014, Sydney, Australia.

---

[2] http://www.emulab.net
[3] http://www.planet-lab.org
[4] http://www.suragrid.org
[5] http://www.geni.net
[6] https://portal.futuregrid.org
[7] http://www.opensciencegrid.org