Towards an Optimized Big Data Processing System

Bogdan Ghiţ
Delft University of Technology
b.i.ghit@tudelft.nl

Alexandru Iosup
Delft University of Technology
Supervisor

Dick Epema
Delft University of Technology
Supervisor

***Abstract*—Scalable by design to very large computing systems such as grids and clouds, MapReduce is currently a major big data processing paradigm. Nevertheless, existing performance models for MapReduce only comply with specific workloads that process a small fraction of the entire data set, thus failing to assess the capabilities of the MapReduce paradigm under heavy workloads that process exponentially increasing data volumes. The goal of my PhD is to build and analyze a scalable and dynamic big data processing system, including storage (distributed file system), execution engine (MapReduce), and query language (Pig). My contributions for the first two years of PhD research are the following: 1) the design and implementation of a resource management system part of a MapReduce-based processing system for deploying and resizing MapReduce clusters over multicluster systems, 2) the design and implementation of a benchmarking tool for the MapReduce processing system, and 3) the evaluation and modeling of MapReduce using workloads with very large data sets. Finally, we will optimize the MapReduce processing system to process terabytes of data based on the first two years research.**

I. Problem Statement

To perform fast and inexpensive big data analytics, researchers use an processing system represented by a stack of frameworks for data storage (distributed file system, such as Hadoop Distributed File System [1]), data processing (execution engine, such as MapReduce [2]), and data manipulation (query language, such as Pig [3]) deployed over a large distributed system (such as clusters, grids, and clouds). In the context of the data explosion phenomenon [4], existing performance models for MapReduce are applicable for specific production workloads [5], [6], [7], [8], but are yet to reveal the real capabilities of the processing system under heavy workloads that process tens of terabytes of data. Therefore, the goal of my PhD research is to optimize the MapReduce processing system for processing terabytes of data.

The data is permanently growing in three directions with respect to volume, velocity, and variety. While both LinkedIn and Facebook are processing PB of user generated data, the data size is expected to massively increase over the next decade, reaching unprecedented scales. New York Times processed a 4 TB data set of raw images into 11 million PDFs in about 24 hours; the time to decode the human genome has been reduced from 10 years to 7 days. Computer science (e.g. applications, web, documents, etc.), physics (e.g., particle accelerators), biology (e.g., genome sequencers), or astronomy (e.g., powerful microscopes) generate complex data.

Firstly, we extend the KOALA grid scheduler architecture with support for scheduling, deploying, and dynamically resizing a big data processing system. On one hand, users may require isolated environments to develop their applications and to process their data, which calls for multiple deployments of MapReduce clusters (MR clusters) within the same physical infrastructure. On the other hand, current clusters are not large enough [9] to support processing terabytes of data, which calls for scaling the MapReduce processing system across multiple clusters. During my PhD research we aim to improve the scalability and performance of the MapReduce by dynamically changing the structure of the processing system, using machines leased from other infrastructures. For the future, we aim to incorporate knowledge and prediction based modules in KOALA and enforce scheduling and provisioning decisions compliant with the workload characteristics extracted from previous executions.

Secondly, we evaluate and model the performance of the processing system using different infrastructure configurations (e.g. single cluster, or multicluster system). Given a heavy workload that processes up to tens of TB, the physical resources of the system can easily become saturated: memory, network bandwidth, or disk. We aim to develop a benchmarking tool representative for such heavy workloads and test the boundaries of the processing system under different infrastructure configurations. We design a multi-level approach for building and analyzing a MapReduce performance model, from the infrastructure layer up to the middleware and application layers. For the future, we optimize the MapReduce processing system based on the first two years research.

Thirdly, to move from theory to practice and backwards, we validate our mechanisms and methods by deploying the processing system and **conducting experimental** research on a real multicluster system such as the DAS-4[1].

Furthermore, we understand that MapReduce represents only a part of a larger processing system with many other powerful processing frameworks such as Dryad [10], Nephele and Pact [11], or Hive [12]. Thus, the techniques and the mechanisms we design must be applicable to other frameworks as well, and eventually to paradigms beyond the current processing system. We plan to address this issue in the last year of my PhD.

We identify two main implications of our work related to the *CCGrid community*. From the architectural perspective, we design a new mechanism for dynamic resizing MR clusters using three provisioning policies. In addition, we improve the scalability and the elasticity of the MapReduce processing system in order to process a 14 TB data set.

[1] www.cs.vu.nl/das4/

II. The Big Data Processing System

The big data processing system in Figure 1 has two main components: the KOALA Resource Manager for grids and clouds, and the MapReduce runner (MR-Runner) for Hadoop and Pig deployments. We now describe each, in turn.

A. The Koala Resource Manager

KOALA grid scheduler was originally designed for multi-cluster systems such as the DAS-4 with the goal of designing, implementing, and analyzing scheduling strategies for various application types.

At the core of the system, the scheduler is responsible for scheduling jobs submitted by users by placing and executing them on suitable cluster sites according to its scheduling policies. Jobs are submitted to KOALA through runners, specialized in specific application types (e.g., cycle scavenging jobs [13], workflows [14], and malleable applications [15]). To monitor the status of the resources, KOALA uses a network information service. KOALA interfaces with the local resource managers of the clusters in the multicluster grid system, but it does not fully control the grid resources, as users may submit their jobs directly through the local resource managers deployed on each physical cluster.

The contribution of my PhD research for KOALA is the design and implementation of a runner for dynamic MapReduce clusters (MR-Runner) [16], which we further describe in the reminder of the section. Furthermore, we released a **new version** of KOALA which is now available on the DAS-4 multicluster system [2].

As future work, we are planning to improve KOALA's scheduling policies by adapting its decisions based on the characteristics learnt from the workload's past executions. Therefore, novel workload analysis methods, prediction models and online scheduling mechanisms represent the contributions my research aims to achieve.

B. The MapReduce Runner

We design and implement a resource management system to facilitate the on-demand isolated deployment of dynamic MapReduce clusters in multicluster systems. An MR cluster (e.g., Hadoop) relies on a two-layer architecture: a compute framework to facilitate an execution environment for MapReduce applications, and a storage layer (distributed file system) that manages in a reliable and efficient manner large data volumes. Both layers are distributed across all nodes of an MR cluster, such that each node may execute tasks and also store data. The MR cluster is coupled with a high-level query language such as Pig which provides an easy way to express data analysis programs.

From the infrastructure perspective, the MR-Runner is able to deploy multiple MR clusters within a single physical cluster with the desired isolation properties, but also to span a single MR cluster across a multicluster system.

[2]http://www.pds.ewi.tudelft.nl/koala

Deploying multiple MapReduce clusters enables four types of isolation, with respect to performance, to data management, to fault tolerance, and to versioning. To efficiently manage the underlying physical resources, we propose three provisioning policies for dynamically resizing MapReduce clusters, and we evaluate the performance of our system through experiments on a real multicluster (DAS-4).

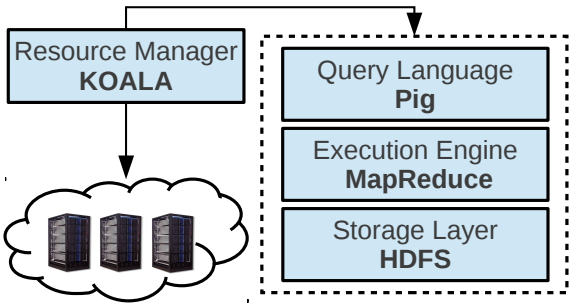

Fig. 1. The Koala-based Big Data Processing System

The contribution of my PhD research is the design and implementation of a novel hybrid architecture for dynamic MR clusters (which may grow or shrink based on the utilization) with core and transient nodes. While the former are used both for computations and storage, the latter are used only for computations. The architecture favours fast reconfiguration of the MR cluster by adding or removing transient nodes, while the structure of the data remains unchanged. Thus, we bypass the burden of redistributing the data.

To take full advantage of the nodes when the MR cluster grows, the storage layer needs to be adjusted as well. While we can easily add more core nodes, removing them from an MR cluster arises issues such as data availability and integrity. For the future, we are going to improve the resizing mechanism to support grow-shrink operations at the storage layer (e.g., HDFS) while the data availability and integrity are preserved. The approach we take relies on a heuristic-based algorithm that ensures the data is not lost when the MR cluster is resized.

III. The Performance Model

In this section we describe the workloads we use in our evaluation and modeling of the big data processing system and we present our approach for building and analyzing a MapReduce performance model compliat with the massively increasing data volumes.

A. The Benchmark Suite

Among the challenges of benchmarking cloud-based systems identified in [17], we have emphasized the importance of the workload suite and the metrics used to test and quantify the performance of the system.

We split the MapReduce processing system benchmarking into two phases: the workload execution, followed by the

result analysis and interpretation. We will design a workload suite for big data benchmarking that covers a broad range of applications and data sets from the academia, industry, and scientific world:

- **Real-world applications.** The most common MapReduce applications implement algorithms for text processing (e.g., Wordcount, Sort, TeraSort), web searching (e.g., Nutch indexing, PageRank), or machine learning (e.g., K-means clustering, Bayesian classification). HiBench [18] is a benchmarking tool for MapReduce that incorporates such workloads with randomly generated data sets.
- **Trace-based workloads.** Although we are not able to reproduce exactly the execution of the original application, the synthetic workloads are not less valuable as the real-world applications. As such workloads reflect the representative set of use cases for MapReduce, we generate such workloads through analysis and modeling of traces gathered from production clusters.
- **BTWorld use case.** The P2P team in our PDS group maintains a data set of BitTorrent logs collected over a period of three years. The data set reached 14 TB which we aim to statistically analyze using the MapReduce processing system. Towards this goal, we have implemented a set of Pig queries which mine through the entire data set to determine the seeder-leecher ratio per tracker, the most popular hashes and swarms, the number of sessions per tracker etc.

Analyzing the results obtained through bechmarking relies on a well-defined set of metrics that quantify the performance of the system. Based on our collaboration with the SPEC Research Group [3], as part of my PhD research, we are going to explore new metrics such as **elasticity** of the processing system (dynamic MR clusters), **performance isolation** of the workloads (multiple MR clusters), **velocity** of the data processing (in future), or **adaptivity** to the data explosion (in future). The benchmark tool we plan to design and implement not only for performance evaluation of the big data processing system, but also in order to enrich KOALA with better knowledge about the workloads, will be certified by the SPEC Reseach Group.

B. The Performance Model

To build and analyze a MapReduce performance model we take a multi-level approach, from the physical system up to the middleware and the application layer with complementary roles:

- **Infrastructure layer.** The structure of the distributed systems on top of which the big data processing system is deployed impacts the performance. The MR-Runner enables MapReduce clusters on demand in a multicluster environment. Thus, there are three deployment scenarios involved in our performance evaluation: single MR cluster over dedicated physical cluster, multiple MR clusters sharing the same physical cluster, and single MR cluster over multiple physical clusters. We aim to develop resource management techniques for scalable provisioning of the processing system over a large multicluster system.
- **Middleware layer.** The big data processing system is controlled by a rich set of parameters with respect to memory, storage, and network. Finding the optimal configuration of the processing system given a certain structure of the distributed system (infrastructure layer) represents an important step towards a reliable performance model for MapReduce.
- **Application layer.** The diverse workloads should reveal both the strenghts and the weaknesses of the big data processing system. With heavy workloads processing very large data sets, physical resources of the system can be easily saturated. Therefore, understanding to what extent the MapReduce processing system is suitable for certain workloads is a prerequisite in building a reliable performance model.

Finally, the methods we will apply to build the performance model are statistical modeling of the workloads and optimization algorithms (e.g., machine learning algorithms).

IV. EXPERIMENTAL RESEARCH

My PhD research is of high importance in the CCGrid community, as we conduct experiments on a **real multicluster system (DAS-4)** and we aim to apply my research work to solve a **real-world application**.

The physical infrastructure is a wide-area computer system dedicated to research in parallel and distributed computing. The Distributed ASCI Supercomputer (DAS-4), currently in its fourth generation, consists of six clusters distributed in institutes and organizations across the Netherlands.

The performance evaluation of the MR-Runner shows that the CPU-bound applications scale on transient nodes as well as on core nodes, while the IO-bound applications suffer a high performance degradation when the number of transient nodes increases. The workloads include two representative MapReduce applications (Sort and WordCount) and process from 1 GB up to 100 GB randomly generated data.

The BTWorld project aims to process a 14 TB data set of BitTorrent logs gathered by the P2P research group at TU Delft over a period of three years. We aim to fire up the processing power of the Koala-based processing system over the DAS-4 infrastructure in order to analyze the data set and extract statistics about the trackers, hashes, seeders and leechers etc.

V. TIMELINE

The timeline of my PhD research for the first two years includes the following four points (Table I):

1) **Koala Big Data Ecosystem.** During the first year of my PhD research I designed and implemented the MR-Runner which enables scheduling MapReduce clusters with KOALA. The MR clusters are enriched with dynamic resizing capabilities. Increasing the elasticity of the processing system and developing more provisioning policies represent further goals of my work.

[3] http://research.spec.org/

TABLE I
MY PHD RESEARCH TIMELINE, 2011-2013

Phase	Duration	Completed	Future Work
Koala Big Data Ecosystem	1 year	Dynamic MapReduce clusters	Elastic HDFS, more policies
Benchmarking Tool	3 months	HiBench applications	Synthetic workloads and BTWorld
BTWorld Use Case	3 months	100 GB data processed	Go up to 10 TB
Performance Evaluation	6 months	Single MR clusters	Multiple MR clusters

2) **Benchmarking Tool.** For the initial performance evaluation we have used applications provided by the HiBench tool. Furthermore, we want to develop our own benchmarking tool based on real-world applications, synthetic workloads and the BTWorld queries.
3) **BTWorld Use Case.** We have used the MR-Runner to process 100 GB of the BTWorld data set and we aim to scale up to 14 TB of data.
4) **Performance Evaluation.** The goal of my second PhD year is to run a complete performance evaluation of the Koala-based big data processing system, with various workloads and different structures and configurations of the system.

ACKNOWLEDGMENT

This publication was supported by the Dutch national program COMMIT and STW/NWO Veni grant 11881. We would like to thank Mihai Capotă, Tim Hegeman, and Lipu Fei.

REFERENCES

[1] T. White, *Hadoop: The Definitive Guide*. Yahoo Press, 2010.
[2] J. Dean and S. Ghemawat, "Mapreduce: Simplified Data Processing on Large Clusters," *Comm. of the ACM*, Vol. 51, no. 1, pp. 107–113, 2008.
[3] C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins, "Pig latin: A Not-So-Foreign Language for Data Processing," *Proc. of the 2008 SIGMOD Int'l Conf. on Management of Data*. ACM, 2008, pp. 1099–1110.
[4] http://www.infoworld.com/d/data-explosion/big-data-get-even-bigger-in-2011-064.
[5] Y. Chen, A. Ganapathi, R. Griffith, and R. Katz, "The Case for Evaluating MapReduce Performance Using Workload Suites," *19th Int'l Symp. on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE, 2011, pp. 390–399.
[6] S. Kavulya, J. Tan, R. Gandhi, and P. Narasimhan, "An Analysis of Traces from a Production Mapreduce Cluster," *10th Int'l Conf. on Cluster, Cloud and Grid Computing (CCGrid)*. IEEE, 2010, pp. 94–103.
[7] D. Jiang, B. Ooi, L. Shi, and S. Wu, "The Performance of MapReduce: An In-depth Study," *Proc. of the VLDB Endowment*, Vol. 3, no. 1-2, pp. 472–483, 2010.
[8] Y. Chen, S. Alspaugh, and R. Katz, "Interactive Analytical Processing in Big Data Systems: A Cross-Industry Study of MapReduce Workloads," *Proc. of the VLDB Endowment*, Vol. 5, no. 12, pp. 1802–1813, 2012.
[9] Y. Kee, H. Casanova, and A. Chien, "Realistic modeling and svnthesis of resources for computational grids," *Proceedings of the 2004 ACM/IEEE conference on Supercomputing*. IEEE Computer Society, 2004, p. 54.
[10] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: Distributed Data-Parallel Programs from Sequential Building Blocks," *SIGOPS Operating Systems Review*, Vol. 41, no. 3, pp. 59–72, 2007.
[11] D. Battré, S. Ewen, F. Hueske, O. Kao, V. Markl, and D. Warneke, "Nephele/Pacts: A Programming Model and Execution Framework for Web-Scale Analytical Processing," *Proc. of the 1st Symp. on Cloud computing*. ACM, 2010, pp. 119–130.
[12] A. Thusoo, J. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy, "Hive: A Warehousing Solution Over a Map-Reduce Framework," *Proc. of the VLDB Endowment*, Vol. 2, no. 2, pp. 1626–1629, 2009.
[13] O. Sonmez, B. Grundeken, H. Mohamed, A. Iosup, and D. Epema, "Scheduling Strategies for Cycle Scavenging in Multicluster Grid Systems," *9th Int'l. Symp. on Cluster Computing and the Grid (CCGrid)*, pp. 12–19, 2009.
[14] O. Sonmez, N. Yigitbasi, S. Abrishami, A. Iosup, and D. Epema, "Performance Analysis of Dynamic Workflow Scheduling in Multicluster Grids," *19th Int'l. Symp. on High-Performance Distributed Computing (HPDC)*, pp. 49–60, 2010.
[15] J. Buisson, O. Sonmez, H. Mohamed, W. Lammers, and D. Epema, "Scheduling Malleable Applications in Multicluster Systems," *9th Int'l. Conference on Cluster Computing*, pp. 372–381, 2007.
[16] B. Ghit, N. Yigitbasi, and D. Epema, "Resource Management for Dynamic MapReduce Clusters in Multicluster Systems," *Proc. of the 5th Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS) co-located with Supercomputing (SC)*. IEEE.
[17] A. Iosup, R. Prodan, and D. Epema, "Iaas Cloud Benchmarking: Approaches, Challenges, and Experience," *Proc. of the Int'l Conf. on High Performance Networking and Computing (SC), MTAGS*. IEEE/ACM, pp. 1–8.
[18] S. Huang, J. Huang, J. Dai, T. Xie, and B. Huang, "The Hibench Benchmark Suite: Characterization of the MapReduce-based Data Analysis," *26th Int'l Conf. on Data Engineering Workshops (ICDEW)*, pp. 41–51, 2010.