Procedia Computer Science

Volume 51, 2015, Pages 80–89

ICCS 2015 International Conference On Computational Science

MREv: an Automatic MapReduce Evaluation Tool for Big Data Workloads

Jorge Veiga, Roberto R. Expósito, Guillermo L. Taboada, and Juan Touriño

Computer Architecture Group
University of A Coruña, Spain
{jorge.veiga, rreye, taboada, juan}@udc.es

Abstract

The popularity of Big Data computing models like MapReduce has caused the emergence of many frameworks oriented to High Performance Computing (HPC) systems. The suitability of each one to a particular use case depends on its design and implementation, the underlying system resources and the type of application to be run. Therefore, the appropriate selection of one of these frameworks generally involves the execution of multiple experiments in order to assess their performance, scalability and resource efficiency. This work studies the main issues of this evaluation, proposing a new MapReduce Evaluator (MREv) tool which unifies the configuration of the frameworks, eases the task of collecting results and generates resource utilization statistics. Moreover, a practical use case is described, including examples of the experimental results provided by this tool. MREv is available to download at http://mrev.des.udc.es.

Keywords: High Performance Computing (HPC), Big Data, MapReduce, Performance Evaluation, Resource Efficiency, InfiniBand

1 Introduction

Nowadays, the fast growth of Big Data applications is forcing many organizations to adopt High Performance Computing (HPC) technologies. The use of HPC systems boosts Big Data applications, while HPC developers also benefit from Big Data models. One of them is MapReduce [4], a programming model initially proposed by Google to generate and process large data sets in clusters. Its de-facto standard implementation, Apache Hadoop [1], is being widely used by large organizations like Yahoo!, Facebook or Twitter. Apache Hadoop is a MapReduce framework originally intended for commodity clusters, showing some issues that hinder it from completely leveraging HPC resources, like high performance interconnects (e.g., InfiniBand [10]). This has caused the appearance of many HPC-oriented MapReduce solutions like Mellanox UDA [17], RDMA-Hadoop [8] or DataMPI [3]. The selection of one of them to use in an HPC system depends on factors like performance and resource efficiency, so making a

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2015

doi:10.1016/j.procs.2015.05.202

comparison implies the preparation of a series of experiments, which mainly involves configuring the frameworks, generating the input data sets, executing the workloads and collecting and interpreting the results.

This paper studies the difficulties of evaluating MapReduce solutions, in particular HPC-oriented ones. A new evaluation tool, MapReduce Evaluator (MREv), is presented as a proposal to solve many of them, adapting the configuration of the solutions and easing the task of collecting results. The remainder of this paper is organized as follows: Section 2 presents background information and related work. Section 3 describes the design of MREv and its behaviour. Section 4 shows a practical use case and analyzes the obtained results. Finally, Section 5 extracts the main conclusions and sketches ongoing work.

2 Background and Related Work

This section introduces some basic aspects about MapReduce and the Hadoop implementation. It also includes some details about InfiniBand, which has become the most widely adopted interconnection network in the TOP500 list [19]. Next, it classifies existing HPC-oriented MapReduce solutions. Finally, related work is discussed.

2.1 MapReduce and Hadoop

The MapReduce paradigm [4] consists of two data processing phases: Map and Reduce. In the Map phase the cluster nodes extract significant features of each input data element, representing it by a $< key, value >$ pair. The Reduce phase receives the data elements and operates them to obtain the final result. Currently, the most popular MapReduce implementation is Apache Hadoop [1], an open-source Java-based framework which enables to store and process Big Data workloads. It mainly consists of two components: the Hadoop Distributed File System (HDFS), which distributes the storage of data among the nodes of a cluster, and the Hadoop MapReduce engine. Hadoop 1 has evolved to Hadoop 2 YARN (Yet Another Resource Negotiator) [7], in which the resource management is separated from the MapReduce engine. This separation enables the use of other computing models over Hadoop, such as Apache Spark [2].

2.2 InfiniBand

InfiniBand [10] is a networking technology widely used by supercomputing centers world-wide. It supports Remote Direct Memory Access (RDMA) communications to directly access the memory of the remote nodes without involving CPU usage at the remote side, while also enabling zero-copy and kernel-bypass mechanisms. InfiniBand can also be used as an IP network, which is commonly known as IP over InfiniBand (IPoIB) [11]. IPoIB exposes the InfiniBand device to the system as an available Ethernet card, although it is neither kernel-bypassed nor allows zero-copy communications.

2.3 HPC-oriented MapReduce Solutions

The inability of Hadoop to fully leverage HPC resources has caused the emergence of several HPC-oriented MapReduce frameworks. This work focuses on the experimental evaluation of this kind of solutions, which can be divided into three categories: transparent solutions, modifications over Hadoop and implementations from scratch.

The use of transparent solutions only involves changing the configuration of the Hadoop frameworks. One of these approaches consists in the modification of the network configuration to make use of the IPoIB interface instead of the Ethernet one. Other transparent solution for Hadoop is Mellanox UDA [17], a plugin that accelerates the communications between mappers and reducers. Based on the network levitated merge algorithm [21], it optimizes the original overlapping of Hadoop phases and uses RDMA communications.

Currently, the most popular modification over Hadoop is RDMA-Hadoop, which is framed within the High Performance Big Data (HiBD) project [8] and includes RDMA-based implementations of Hadoop 1 and 2. RDMA-Hadoop adapts different components of Hadoop to use RDMA communications: HDFS [12], MapReduce [22] and Hadoop RPC [15]. The communications between mappers and reducers are redesigned to take full advantage of the RDMA interconnect, while also implementing data prefetching and caching mechanisms.

DataMPI [3] is the most representative framework developed from scratch. It aims to take advantage from the Message-Passing Interface (MPI) [6] and to avoid the use of traditional TCP/IP communications. To do so, it implements the MPI-D specification [16], a proposed extension of the MPI standard which uses the $< key, value >$ semantics taken from MapReduce.

2.4 Related Work

Many works have already addressed the evaluation of Hadoop and Big Data engines like MapReduce. On the one hand, some of them focused on the evaluation of Hadoop when performing specific workloads, like [5], which is oriented to evaluate data-intensive scientific applications. On the other hand, several works presented benchmark and application suites to obtain performance measurements from MapReduce clusters, such as HiBench [9], BigDataBench [20], MRBS [18] and MR-Bench [13]. Some suites (e.g., HiBench) include certain features in order to ease the execution of the experiments, such as user-defined parameters to configure the benchmarks or the gathering of performance results. However, all of them rely on Hadoop installations manually configured and launched in the system, which does not allow the automation of performance measurements to compare different solutions.

Some evaluation works also include resource efficiency comparisons, which is the case of [9], [14] and [23], but they do not provide a tool to perform the evaluations systematically. To the best of our knowledge, this work presents the first evaluation tool to compare MapReduce solutions in terms of performance and resource efficiency metrics, easing the configuration of the frameworks and the task of collecting and displaying results.

3 Design Overview

This section first discusses the need for a tool to carry out comparative performance evaluations of MapReduce frameworks on a certain environment. Next, it describes the main features of the MapReduce Evaluator (MREv) tool.

3.1 Motivation

The selection of a MapReduce framework to solve a certain kind of problem in a specific environment can be affected by several factors. First, the performance of each framework may be limited by its design, in the way that it schedules the MapReduce tasks or the disk and network operations. Second, each solution may adapt differently to the system, making a worse or better use of resources like CPU, memory or network. In some cases, the number of cluster nodes

available to use may be limited by other applications running in the system. The evaluation would have to consider different cluster sizes to determine the best balance between execution time and cluster occupation. Some frameworks could also be adapted to take advantage of specific resources, such as network communications like IPoIB or disk technologies such as Solid State Drive (SSD). Hence, the user would have to install each available MapReduce framework, understand how it works, learn to configure it and check its correct operation. Moreover, the user would have to elaborate a set of workloads, searching for appropriate implementations for each framework and ensuring a fair comparison among them. The evaluation process of each solution would involve its configuration, the launching of its daemons and the execution of the benchmarks. Once the evaluation is finished, the user would have to access the output files to check the successful completion of the benchmarks and to copy the desired values to a separate file to generate the required graphs.

In order to overcome these issues, the MREv tool has been developed. Using different representative workloads, it enables to evaluate several MapReduce frameworks unifying their configuration and helping the user to compare them. It is aimed at two main goals: (1) comparison between frameworks in terms of performance and scalability; and (2) evaluation of resource efficiency.

3.2 MREv Characteristics

The uniform configuration performed by MREv guarantees that every solution has the same conditions, so the results are neither unfair nor biased. The capabilities of the underlying system can be easily leveraged by setting a small set of parameters, which are related to system resources like the number of cores per node or the network bandwidth. Once the parameters are set, the evaluation is started by using a single script, which carries out the necessary actions to launch the frameworks and run the benchmarks. Each time a benchmark is finished, its performance results are automatically recorded and saved to the report files, which are used to generate data graphs.

The user can select the number of times each benchmark is executed. Performing multiple executions of the benchmarks provides two main benefits: the extraction of statistical values from the records and the reuse of the input data sets. MREv also monitors the utilization of the resources during the execution of each benchmark, providing both results per node and average values among the nodes. More details about the main features of MREv are given next.

User-defined parameters. Table 1 shows the configuration parameters currently available in MREv. They are classified depending on whether they affect the configuration of the solutions, the experiments to run or the input data sets of the benchmarks.

Supported frameworks and benchmarks. Tables 2 and 3 show the benchmarks and the frameworks, respectively, currently supported by MREv. All the benchmarks are available for Hadoop-based solutions, although DataMPI only supports WordCount, Sort and TeraSort, provided with the distribution. Benchmarks for Spark are still under development.

Results. All the results from an evaluation are stored in an output directory, which includes the summary report, the log file, the output subdirectories of the benchmarks and the performance graphs. The summary report shows the configuration parameters and a summary of the main performance results. The log file contains the sequence of events in the evaluation, such as the successful configuration and operation of each framework or the end of the execution of each benchmark. The output subdirectories of the benchmarks include the standard output, execution times and resource utilization statistics (CPU, memory, disk and network) for each

Table 1: Main MREv configuration parameters

System dependent	Experiment dependent	Benchmark dependent
Mappers per node	Benchmarks	TestDFSIO: #files & file size
Reducers per node	Cluster sizes	Wordcount: data size
Java heap size	Frameworks	Sort/TeraSort: data size
HDFS block size	Number of executions	PageRank: number of iterations
HDFS replication factor		
SSD optimization[a]		

[a] Available only in RDMA-Hadoop

Table 2: Supported benchmarks

Micro-benchmarks	
TestDFSIO	Tests the read and write throughput of HDFS
Wordcount	Counts the number of times each word appears in the input text data set
Sort	Sorts the input text data set
TeraSort	Sorts 100 byte-sized $< key, value >$ tuples
Real-world Applications	
PageRank	Ranks websites by counting the number and quality of the links to each one
Bayes	Performs a classification algorithm, based on Bayes' Theorem

Table 3: Supported frameworks

Framework	Release Date	Interconnect
Hadoop-2.5.1-GBE	30/06/2014	Gigabit Ethernet
Hadoop-2.5.1-IPoIB	30/06/2014	InfiniBand (IPoIB)
Hadoop-2.5.1-UDA	03/09/2014	InfiniBand (RDMA & IPoIB)
Hadoop-1.2.1-GBE	01/08/2013	Gigabit Ethernet
Hadoop-1.2.1-IPoIB	01/08/2013	InfiniBand (IPoIB)
Hadoop-1.2.1-UDA	07/06/2013	InfiniBand (RDMA & IPoIB)
RDMA-Hadoop-0.9.9	31/03/2014	InfiniBand (RDMA)
RDMA-Hadoop-2-0.9.5	26/11/2014	InfiniBand (RDMA)
DataMPI-0.6.0-HDFS-GBE[a]	16/04/2014	InfiniBand (RDMA) & Gigabit Ethernet
DataMPI-0.6.0-HDFS-IPoIB[a]	16/04/2014	InfiniBand (RDMA & IPoIB)
Spark-1.1.0-YARN-GBE	11/09/2014	Gigabit Ethernet
Spark-1.1.0-YARN-IPoIB	11/09/2014	InfiniBand (IPoIB)

[a] DataMPI uses HDFS 1.2.1, which has its own network configuration (GBE or IPoIB)

framework and cluster size, both in text and graphical format. Finally, the performance graphs allow to compare visually the frameworks in terms of performance and scalability. Section 4 describes some of the graphs produced by MREv.

Operation. Figure 1 shows the control flow of an execution of MREv. At the beginning of the process, MREv is initialized by setting the configuration parameters and creating the output directory. MREv iterates over the selected cluster sizes and frameworks. Once a framework has been configured, its daemons are launched and the benchmarks are initialized, generating the required input data sets. Before running a benchmark, its output subdirectory is created and the processes that collect the resource utilization results are launched. Once the benchmark has finished, MREv automatically generates the resource utilization data files and graphs, saving the information of each cluster node and the average values among them. Once the benchmark has been executed the number of times configured by the user, the most relevant performance results

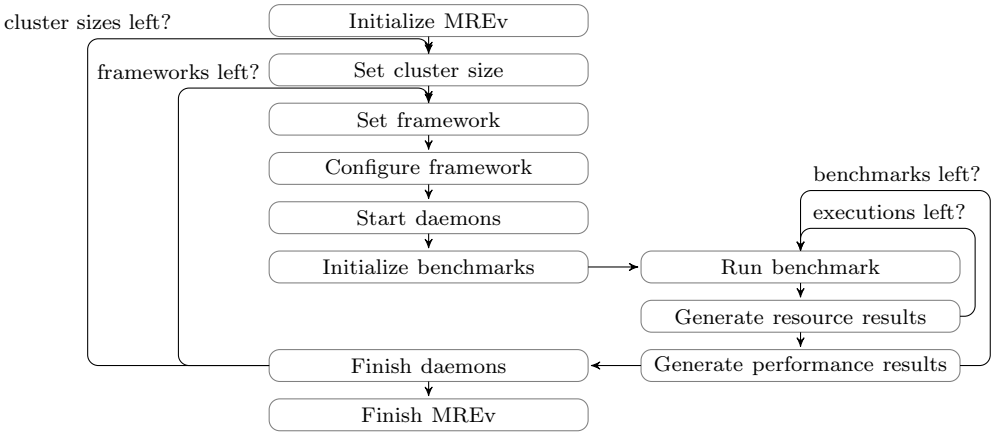


Figure 1: MREv control flow

are appended to the summary report. After executing the selected benchmarks with a particular framework, its daemons are shut down to begin the execution with the next framework.

4 Use Case

This section presents an example of using MREv to run a set of experiments for a specific configuration of the frameworks, including the performance results produced by the tool.

4.1 Experimental Configuration

The experiments have been carried out on a multi-core cluster (Pluton). It consists of 13 nodes interconnected by both Gigabit Ethernet and InfiniBand FDR networks. Table 4 shows more details about the hardware and software configuration. Table 5 shows the frameworks selected in this evaluation and the configuration parameters used to adapt them to the characteristics of Pluton. The TeraSort benchmark with a 128 GB input data set was used in the experiments, testing the scalability of the frameworks for cluster sizes of 5, 9 and 13 nodes.

4.2 Performance Results

MREv eases the extraction of significant conclusions about the suitability of the frameworks to specific system configurations, being also useful to identify the required resources to achieve a certain performance threshold. For instance, Figure 2, generated by MREv, shows the execution times of TeraSort. For each cluster size, the graph includes the average, maximum and minimum times. As can be seen, these values vary significantly over the different cluster sizes.

This graph allows to examine the scalability of each framework, which differs slightly from one to another. Moreover, their relative performance also varies when using different cluster sizes. For example, it can be seen that Hadoop-2.5.1-UDA is the best solution for 5 nodes, but its results are not so good for 13 nodes due to its low scalability. The opposite occurs with RDMA-Hadoop-0.9.9, which is the best for 13 nodes. The variability of the execution times can also be observed, which depends on the framework and the cluster size used; for instance, RDMA-Hadoop-0.9.9 and DataMPI show high variability for 5 nodes.

Table 4: Configuration of the Pluton cluster

Hardware configuration	
CPU	2×Intel Xeon E5-2660 2.20GHz
#Cores per node	16
RAM Memory	64 GB DDR3 1600Mhz
Disk	1 TB HDD

Software versions	
OS	CentOS release 6.4 (2.6.32-358)
Java	Oracle JDK 1.8.0_20
MPI	MVAPICH2 1.9 (for DataMPI)

Table 5: Evaluated frameworks and configuration parameters

Frameworks
Hadoop-2.5.1-GBE
Hadoop-2.5.1-IPoIB
Hadoop-2.5.1-UDA
RDMA-Hadoop-0.9.9
DataMPI-0.6.0-HDFS-GBE

Configuration parameters	
Mappers per node	7
Reducers per node	7
Java heap size	3 GB
HDFS block size	128 MB
HDFS replication factor	3

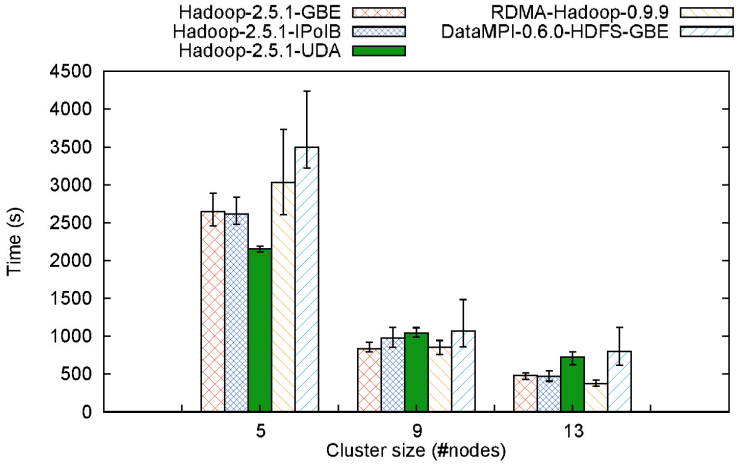
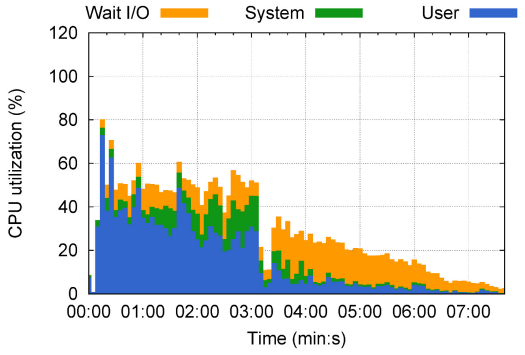


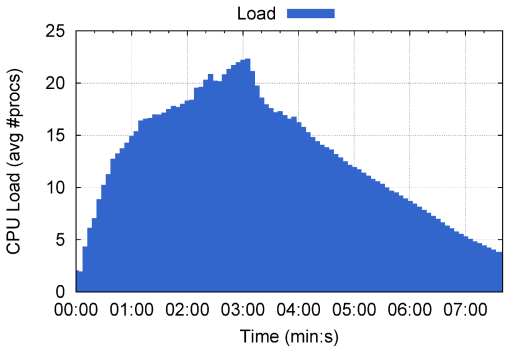
Figure 2: Execution times for the TeraSort benchmark

4.3 Resource Efficiency Results

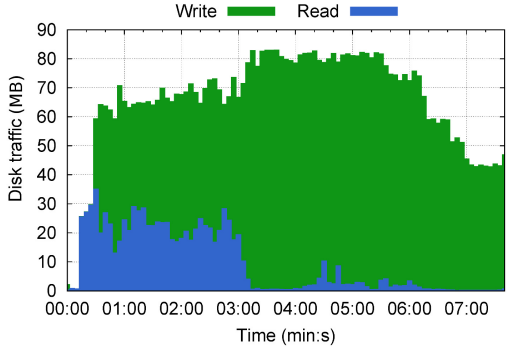
The resource graphs generated by MREv display the use of resources during an evaluation and are useful to detect bottlenecks. Figure 3 includes the graphs taken from an execution of TeraSort using Hadoop-2.5.1-IPoIB on 13 nodes, calculated as the average values for the nodes. Figure 3a shows the CPU utilization during the execution of the workload. It can be seen that in the first three minutes the CPU is mainly used by user code, corresponding to the sorting and partitioning in the map phase. From this point on, the Wait I/O values increase, as the nodes begin to write their outputs to disk in the reduce phase. Figure 3b contains the CPU load results. The value at each point is calculated as the average number of processes during the last minute (note that each node consists of 16 cores). As shown in the graph, these values increase as the benchmark progresses, reaching the top values in the third minute. This point corresponds with the end of the map processes and the beginning of the reduce phase (note that reduce processes were also receiving data during the map phase). The memory usage shows a



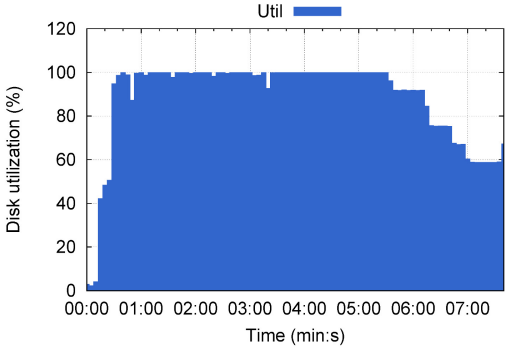
(a) CPU utilization



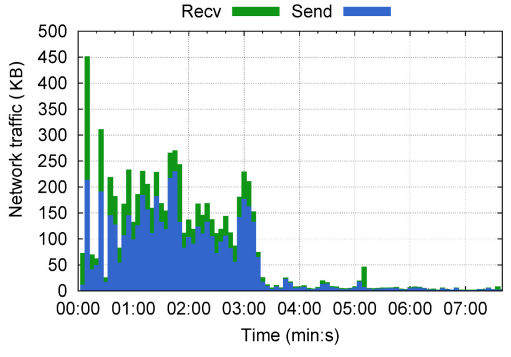
(b) CPU load



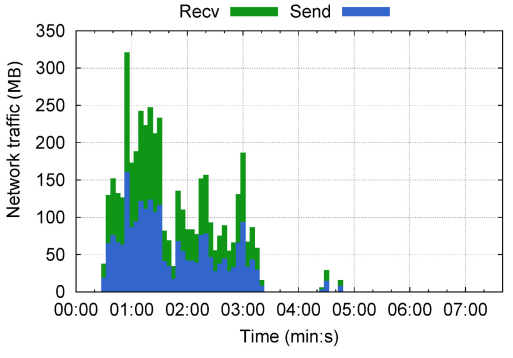
(c) Disk write/read traffic



(d) Disk utilization



(e) GBE network traffic



(f) IPoIB network traffic

Figure 3: Resource utilization results for TeraSort using Hadoop-2.5.1-IPoIB (13 nodes)

similar behaviour to that of the CPU load, and so it is not included due to space constraints.

Disk write/read traffic can be observed in Figure 3c. In the map phase, the read and write traffic correspond to reading the input data set and writing the intermediate results, respectively. In the reduce phase, the write traffic increases significantly due to the writing of the final results, being the read traffic very low as most of the intermediate results are cached in memory. Figure 3d contains the same information in terms of percentage of disk utilization over the available throughput. As can be seen, the utilization is close to 100% for the most

part of the execution, proving that the TeraSort benchmark is mainly disk-bound.

Figures 3e and 3f depict the network traffic over the available network interfaces , GBE and IPoIB. MREv allows to check that the main traffic volume is going through the IPoIB interface. The majority of this traffic is generated in the map phase, in which the mappers send the intermediate results to the reducers. Moreover, a negligible amount of traffic goes through the GBE interface, which is due to monitorization traffic between master and slave nodes.

5 Conclusions and Future Work

This paper has presented some of the issues when evaluating the performance and scalability of MapReduce frameworks on HPC clusters. To solve them, the MapReduce Evaluator (MREv) tool has been developed, enabling the user to unify the configuration of the frameworks, extract statistical values from the execution of different workloads and collect the results in an easy way. MREv includes a set of ready-to-use MapReduce frameworks with several configuration options. These solutions can be evaluated by means of different benchmarks, which check the efficient use of the underlying capabilities. MREv also allows to leverage the high performance resources that might be available by modifying the configuration of the frameworks (e.g., use of SSD disks for RDMA-Hadoop and configuration of IPoIB). The graphs produced by MREv in each evaluation display the performance and resource utilization results for each benchmark, easing the comparison of the frameworks for different system configurations.

MREv was designed in a flexible and modular way, which will ease our future work of including new frameworks and benchmarks, such as iterative and streaming workloads. We will also include new resource results (e.g., native InfiniBand traffic), and we will adapt current MapReduce frameworks to support additional resources that can accelerate the execution of the workloads (e.g., RoCE interconnects). Further details of MREv and its download link are available at `http://mrev.des.udc.es`.

Acknowledgments

This work was funded by the Ministry of Economy and Competitiviness of Spain (Ref. TIN2013-42148-P), and by the Galician Government (Xunta de Galicia) under the Consolidation Program of Competitive Reference Groups (Ref. GRC2013/055), cofunded by FEDER funds of the EU.

References

[1] Apache Hadoop. `http://hadoop.apache.org/`. [Last visited: Mar. 2015].

[2] Apache Spark. `http://spark.apache.org/`. [Last visited: Mar. 2015].

[3] DataMPI. `http://datampi.org/`. [Last visited: Mar. 2015].

[4] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[5] Zacharia Fadika, Madhusudhan Govindaraju, Richard Canon, and Lavanya Ramakrishnan. Evaluating Hadoop for data-intensive scientific operations. In *Proceedings of the 5th IEEE International Conference on Cloud Computing (CLOUD'12)*, pages 67–74. Honolulu, HI, USA, 2012.

[6] William Gropp, Ewing Lusk, Nathan Doss, and Anthony Skjellum. A high-performance, portable implementation of the MPI message passing interface standard. *Parallel Computing*, 22(6):789–828, 1996.

[7] Hadoop YARN. `http://hadoop.apache.org/docs/r2.3.0/hadoop-yarn/hadoop-yarn-site/YARN.html`. [Last visited: Mar. 2015].

[8] High-Performance Big Data. `http://hibd.cse.ohio-state.edu/`. [Last visited: Mar. 2015].

[9] Shengsheng Huang, Jie Huang, Jinquan Dai, Tao Xie, and Bo Huang. The HiBench benchmark suite: Characterization of the MapReduce-based data analysis. In *Proceedings of the 26th IEEE International Conference on Data Engineering Workshops (ICDEW)*, pages 41–51. Long Beach, CA, USA, 2010.

[10] IBTA, InfiniBand Trade Association. `http://www.infinibandta.org`. [Last visited: Mar. 2015].

[11] IP over InfiniBand (IPoIB) Architecture. `http://www.ietf.org/rfc/rfc4392.txt.pdf`. [Last visited: Mar. 2015].

[12] Nusrat S. Islam, Md Wasi-Ur-Rahman, Jithin Jose, Raghunath Rajachandrasekar, Hao Wang, Hari Subramoni, Chet Murthy, and Dhabaleswar K. Panda. High performance RDMA-based design of HDFS over InfiniBand. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC'12)*, pages 35:1–35:12. Salt Lake City, UT, USA, 2012.

[13] Kiyoung Kim, Kyungho Jeon, Hyuck Han, Shin-Gyu Kim, Hyungsoo Jung, and Heon Young Yeom. MRBench: A benchmark for MapReduce framework. In *Proceedings of the 14th IEEE International Conference on Parallel and Distributed Systems (ICPADS'08)*, pages 11–18. Melbourne, Australia, 2008.

[14] Fan Liang, Chen Feng, Xiaoyi Lu, and Zhiwei Xu. Performance benefits of DataMPI: A case study with BigDataBench. In *Proceedings of the 4th Workshop on Big Data Benchmarks, Performance Optimization and Emerging Hardware (BPOE'14)*. Salt Lake City, UT, USA, 2014.

[15] Xiaoyi Lu, Nusrat S. Islam, Md Wasi-Ur-Rahman, Jithin Jose, Hari Subramoni, Hao Wang, and Dhabaleswar K. Panda. High-Performance design of Hadoop RPC with RDMA over InfiniBand. In *Proceedings of the 42nd International Conference on Parallel Processing (ICPP'13)*, pages 641–650. Lyon, France, 2013.

[16] Xiaoyi Lu, Bing Wang, Li Zha, and Zhiwei Xu. Can MPI benefit Hadoop and MapReduce applications? In *Proceedings of the 7th International Workshop on Scheduling and Resource Management for Parallel and Distributed Systems (SRMPDS'11)*, pages 371–379. Taipei, Taiwan, 2011.

[17] Mellanox UDA. `http://www.mellanox.com/page/hadoop`. [Last visited: Mar. 2015].

[18] Amit Sangroya, Damián Serrano, and Sara Bouchenak. MRBS: A Comprehensive MapReduce Benchmark Suite. Technical Report RR-LIG-024, Laboratoire d'Informatique de Grenoble, Grenoble, France, 2012.

[19] TOP 500 List (Nov. 2014). `http://top500.org/lists/2014/11/`. [Last visited: Mar. 2015].

[20] Lei Wang et al. BigDataBench: A Big Data benchmark suite from Internet services. In *Proceedings of the 20th IEEE International Symposium on High-Performance Computer Architecture (HPCA'14)*, pages 488–499. Orlando, FL, USA, 2014.

[21] Yandong Wang, Xinyu Que, Weikuan Yu, Dror Goldenberg, and Dhiraj Sehgal. Hadoop acceleration through network levitated merge. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC'11)*, pages 57:1–57:10. Seattle, WA, USA, 2011.

[22] Md Wasi-Ur-Rahman, Nusrat S. Islam, Xiaoyi Lu, Jithin Jose, Hari Subramoni, Hao Wang, and Dhabaleswar K. Panda. High-Performance RDMA-based design of Hadoop MapReduce over InfiniBand. In *Proceedings of the 27th IEEE International Parallel and Distributed Processing Symposium Workshops and PhD Forum (IPDPSW'13)*, pages 1908–1917. Boston, MA, USA, 2013.

[23] Md Wasi-Ur-Rahman, Xiaoyi Lu, Nusrat S. Islam, Raghunath Rajachandrasekar, and Dhabaleswar K. Panda. MapReduce over Lustre: Can RDMA-Based Approach Benefit? In *Proceedings of the 20th International European Conference on Parallel Processing (Euro-Par '14)*, pages 644–655. Porto, Portugal, 2014.