Performance Evaluation 101 (2016) 1–19

Contents lists available at ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva

Experience with benchmarking dependability and performance of MapReduce systems

Amit Sangroya [a,*], Sara Bouchenak [b], Damián Serrano [c]

[a] Performance Engineering Research Lab, TCS Innovation Labs, India
[b] INSA de Lyon, France
[c] Universite de Rennes 1, France

ARTICLE INFO

Article history:
Received 21 April 2015
Received in revised form 15 March 2016
Accepted 2 April 2016
Available online 29 April 2016

Keywords:
MapReduce
Benchmarking
Dependability
Performance
Hadoop
Fault tolerance

ABSTRACT

MapReduce provides a convenient means for distributed data processing and automatic parallel execution on clusters of machines. It has various applications and is used by several services featuring fault tolerance and scalability. Many studies investigated the dependability and performance of MapReduce, ranging from job scheduling to data placement and replication, adaptive and on-demand fault tolerance to new fault tolerance models. However, the ad-hoc and overly simplified setting used to evaluate most MapReduce fault tolerance and performance improvement solutions poses significant challenges to the analysis and comparison of the effectiveness of these solutions. The paper precisely addresses this issue and presents MRBS, a comprehensive benchmark suite for evaluating the dependability and performance of MapReduce systems. MRBS includes five benchmarks covering several application domains and a wide range of execution scenarios such as data-intensive vs. compute-intensive applications, or batch applications vs. online interactive applications. MRBS allows to inject various workloads, dataloads and faultloads, and produces extensive reliability, availability and performance statistics. We implemented the MRBS benchmark suite for Hadoop MapReduce, and we illustrate its use with various case studies running on Amazon EC2 and on a private cloud.
© 2016 Elsevier B.V. All rights reserved.

1. Introduction

1.1. Motivations

Cloud computing is increasingly being used, enabling on-demand services hosted on a virtually unlimited set of computing and storage resources. MapReduce has become a popular Big Data cloud service [1], used by a wide range of applications such as log analysis, data mining, scientific computing [2], bioinformatics [3], decision support and business intelligence [4]. MapReduce offers developers means to transparently handle data partitioning, replication, task scheduling and fault tolerance on a cluster of commodity computers. Hadoop, one of the most popular MapReduce frameworks, provides key fault tolerance features such as handling node failures, task failures, hanging tasks [5].

There has been a considerable interest in improving fault tolerance and performance of MapReduce. Several efforts have explored on-demand fault tolerance [6], replication and partitioning policies [7,8], adaptive fault tolerance [9,10], extending MapReduce with other fault tolerance models [11,12], as well as task scheduling policies [13–15], cost-based optimization techniques [16], and resource provisioning [17]. However, there has been very little in the way of empirical evaluation

* Corresponding author.
E-mail addresses: amit.sangroya@tcs.com (A. Sangroya), sara.bouchenak@insa-lyon.fr (S. Bouchenak), damian.serrano@irisa.fr (D. Serrano).

http://dx.doi.org/10.1016/j.peva.2016.04.001
0166-5316/© 2016 Elsevier B.V. All rights reserved.

of MapReduce dependability, and the impact of failures on performance. Evaluations have often been conducted in an ad-hoc manner, such as turning off a node in the MapReduce cluster or killing a task process. These actions are typically dictated by what testers can actually control, but may lead to low coverage testing. Recent tools, like Hadoop fault injection framework [18], offer the ability to emulate non-deterministic exceptions in the HDFS distributed filesystem underlying Hadoop MapReduce. Although they provide a mean to program unit tests for HDFS, such low-level tools are meant to be used by developers who are familiar with the internals of HDFS, and are unlikely to be used by end-users of MapReduce systems. MapReduce fault injection must therefore be generalized and automated for higher-level and easier use. Not only it is necessary to automate the injection of faults, but also the definition and generation of MapReduce faultloads. A faultload will describe *what* fault to inject (e.g. a node crash), *where* to inject it (e.g. which node of the MapReduce cluster), and *when* to inject it (e.g. five minutes after the application started). Furthermore, most evaluations of MapReduce fault tolerance systems relied on microbenchmarks based on simple MapReduce programs and workloads, such as *grep*, *sort* or *word count*. While microbenchmarks may be useful in targeting specific system features, they are not representative of full distributed applications, and they do not provide multi-user realistic workloads. These observations motivate the design of MRBS (MapReduce Benchmark Suite), the first benchmark suite for evaluating the dependability *and* performance of MapReduce systems.

1.2. Contributions

The contributions of the paper are the following:

- We provide automatic faultload generation and injection in MapReduce. This covers different fault types, injected at different rates, which will provide a mean to analyze the effectiveness of fault tolerance in a variety of scenarios.
- We provide a benchmark suite that covers five application domains: recommendation systems, business intelligence, bioinformatics, text processing, and data mining. It supports a variety of workload and dataload characteristics, ranging from compute-oriented to data-oriented applications, batch applications to online interactive applications. Indeed, while MapReduce frameworks were originally limited to offline batch applications, recent works are exploring the extension of MapReduce beyond batch processing [19,20]. The proposed benchmark suite uses various input datasets from real applications, among which an online movie recommender service [21], Wikipedia [22], and real genomes for DNA sequencing [23].
- We describe the design principles of MRBS benchmark suite, and its deployment on Hadoop clusters running on Amazon EC2,[1] and on a private cloud [24].[2] Although the current MRBS prototype is provided for the Hadoop popular MapReduce framework, we believe that the proposed dependability and performance benchmarking solution can be easily applied to other MapReduce frameworks. We, thus, discuss the portability of MRBS to other MapReduce frameworks.
- We illustrate the use of MRBS with six case studies which include, among others, the evaluation of the scalability of Hadoop clusters, and the comparison of performance and dependability levels of different Hadoop framework implementations.
- We describe how MRBS allows automatic deployment of experiments on cloud infrastructures. It does not depend on any particular infrastructure and can run on different private or public clouds. This makes dependability and performance benchmarking easy to adopt by end-users of MapReduce, and by developers of MapReduce fault tolerance and scalability solutions.

MRBS is available as a software prototype to help researchers and practitioners to better analyze and evaluate the dependability and performance of MapReduce systems; it can be downloaded from: http://sardes.inrialpes.fr/research/mrbs/

1.3. Paper roadmap

The remainder of the paper is organized as follows. Section 2 describes the background on MapReduce. Sections 3–8 respectively provide an overview of MRBS, a presentation of dependability benchmarking, a description of performance and dependability analysis in MRBS, and a discussion of its portability. Section 6 describes case studies with MRBS, and Section 7 reviews the related work. Finally, Section 9 draws our conclusions.

2. Background

2.1. MapReduce

MapReduce framework supports distributed computing and large data processing on clusters of commodity machines [1]. The MapReduce functional programming model provides a simple means to write programs that process large input

[1] Amazon EC2 is considered as a platform to test the benchmark suite in a public cloud environment. Public cloud is the traditional mode of cloud computing, where an outside service provider is responsible for dynamic provisioning of data and computation resources over the web.

[2] In a private cloud, data and processes are managed within the organization without the restrictions of network bandwidth, security and other legal issues that public clouds have. Grid 5000 is the chosen platform to test the benchmark suite in a private cloud environment.

Download English Version:

https://daneshyari.com/en/article/464750

Download Persian Version:

https://daneshyari.com/article/464750

Daneshyari.com