Chapter 5

Performance Analysis of a Distributed Execution Environment for JUnit Test Cases on a Small Cluster

Eric Bower
ENSCO, Inc., USA

Tauhida Parveen
Independent Consultant, USA

Scott Tilley
Florida Institute of Technology, USA

ABSTRACT

HadoopUnit is a software testing framework that integrates Hadoop and JUnit to facilitate the distribution of unit tests to the nodes of a cluster for concurrent execution. It was conceived of the need to reduce test execution time for large collections of tests and has been shown to successfully accomplish this in a medium-size cluster of 150 nodes. This work considers its effectiveness on a small cluster (4 nodes) to determine any inefficiency present in the system and attempts to overcome them with the purpose of increasing performance. It was found that naïve use of HadoopUnit on a small cluster was slower than the best times achieved on a single machine, but by properly configuring the number of tests executed by each Hadoop Map task, a 75% reduction in execution time was achieved. These results show that, on a small scale, the original implementation of HadoopUnit may not be the best solution, but with the addition of a new feature, a significant increase in performance can be achieved.

DOI: 10.4018/978-1-4666-2536-5.ch005
INTRODUCTION

Testing is a process that includes evaluation of a software product to assess its quality. Proper testing increases the level of confidence in the quality of a product. Testing is a challenging activity for many software engineering projects and is one of the five main technical activity areas of the software engineering lifecycle (Sommerville, 2010) that still poses substantial challenges, especially for large-scale systems. Because testing can be such a difficult, expensive, and labor-intensive process, there is always high demand for better testing support.

For large, complex systems, the number of test cases can range from a few hundred to several thousand. Even computers with substantial processing capacity may require an excessive amount of time to execute such a large quantity of tests. Furthermore, test suites may need to be executed several times over the course of a day. This adds even more challenge to the problem of properly testing a system and identifying errors early in the development lifecycle. In addition, regression testing is often performed as a part of the software maintenance process, and the inability to provide test results in a timely manner to the software engineers involved can have a negative impact on system maintenance.

Testing also requires considerable resources that are often not readily available, contributing to an inefficient testing process. Imagine a scenario where an application needs to be tested for multiple configurations: different operating systems, multiple browsers, several database clients, and complex server interactions. A tester running thousands of test cases each day is required to manage the configuration for these machines, incurring considerable cost associated with setting up and tearing down the test configurations every time testing needs to be done.

One way to add efficiency to the testing process is through faster test execution, leading to better and faster feedback to programmers and testers.

Our solution to this problem is HadoopUnit, which integrates JUnit with Hadoop and has proven to be an effective means of reducing unit test runtime (Parveen, Tilley, Daley, & Morales, 2009). The initial work on HadoopUnit looked at testing the benefit of using Hadoop as a platform for distributed JUnit test case execution. It looked specifically at using Hadoop to reduce the runtime of JUnit tests of the Hadoop project, which was successfully accomplished. Using a medium-size cluster of 150 nodes, the total runtime was reduced 30 times.

As impressive as the results are, not everyone has access to even a medium-sized cluster such as this. To increase the adoption rate of HadoopUnit for software engineers and testers who may lack IT support for larger clusters, but who have access to a limited set of personal computers that can be formed into a small-scale cluster with relatively little effort, this work considers the performance of HadoopUnit on a small cluster of four nodes.

Execution runtimes of a single machine with varying amounts of processes are compared to those of HadoopUnit on the cluster, using test suites of processor-intensive and I/O-intensive tests at counts of 100 and 1000. Through experimentation, an adjustment to HadoopUnit was made providing a significant reduction of the total execution time, resulting in a benefit that can still be realized by engineers and testers working in highly resource-constrained environments.

The remainder of the chapter is structured as follows. The next section provides background information on software testing and the components used to create HadoopUnit. Section 3 describes the high-level design of HadoopUnit. Section 4 details the experiments conducted with a modified version of HadoopUnit on a 4-node cluster. Section 5 analyzes the results of the experiments and discusses the broader impact of this form of distributed execution for regression testing during software maintenance. Finally, Section 6 summarizes the work and outlines possible avenues for future investigation.
Related Content

Adaptive Threshold Based Clustering: A Deterministic Partitioning Approach
[www.igi-global.com/article/adaptive-threshold-based-clustering/226235?camid=4v1a](www.igi-global.com/article/adaptive-threshold-based-clustering/226235?camid=4v1a)

Resistance Factors in Software Processes Improvement: A Study of the Brazilian Industry
[www.igi-global.com/chapter/resistance-factors-software-processes-improvement/61212?camid=4v1a](www.igi-global.com/chapter/resistance-factors-software-processes-improvement/61212?camid=4v1a)

Benefits of CMM and CMMI-Based Software Process Improvement
[www.igi-global.com/chapter/benefits-cmm-cmmi-based-software/61217?camid=4v1a](www.igi-global.com/chapter/benefits-cmm-cmmi-based-software/61217?camid=4v1a)

An Empirical Bandwidth Analysis of Interrupt-Related Covert Channels
[www.igi-global.com/article/an-empirical-bandwidth-analysis-of-interrupt-related-covert-channels/136464?camid=4v1a](www.igi-global.com/article/an-empirical-bandwidth-analysis-of-interrupt-related-covert-channels/136464?camid=4v1a)