# Exploring the Efficiency & Cost-Effectiveness of Distributed Computing for Test Execution

Mrudini Patel and Deshawn Knight

University of Guelph, Guelph ON N1G 2W1, CA

**Abstract.** As modern software systems evolve in complexity, the demand for efficient and rapid software testing becomes increasingly more critical. Distributed computing offers a promising solution to improve software test execution as it enables organizations to deliver high-quality software products at an accelerated rate. This paper examines the extent to which test execution can leverage distributed computing to improve efficiency and speed, while focusing on factors such as memory and CPU usage, execution time, and the trade-offs between vertical and horizontal scaling. By distributing tests across multiple machines, memory and CPU loads can be better balanced, potentially reducing the bottlenecks that often delay testing on single-machine environments. Distributed testing introduces concerns of scalability and cost-effectiveness, as larger infrastructures may increase overhead and require additional resources. Horizontal scaling can parallelize testing processes across multiple machines, promising faster test execution but often at a higher cost. Whereas vertical scaling may limit costs but fail to achieve the same efficient performance. As a result, this paper aims to compare the benefits and limitations of vertical and horizontal scaling in distributed environments, analyzing which approach is more efficient under certain conditions.

**Keywords:** Automated Testing · Distributed Computing.

## 1 Introduction

In the modern era, software has become ubiquitous in human life. It can be found in cell phones, computers, vehicles, etc. Due to technological advancements such as the internet, it has become more important for companies to rapidly iterate on and update their software products to keep up with changing requirements from customers, stay ahead of competitors and maintain security. The concept of continuous integration and continuous delivery (CI/CD) was created to aid in solving these issues by emphasizing rapid delivery of changes to customers via automation. Automated software testing is one important piece of CI/CD which is used to maintain high quality software and reduce the risk of software failures. However, typically as software increases in scale and complexity, running automated test suites can take several hours. This often hinders software engineers in their development lifecycle especially when errors are discovered. Each fix can take hours to validate before additional changes can be made, resulting in significant delays when releasing updates.

Typically, for CI/CD, companies use systems such as Jenkins, GitHub Actions or GitLab CI. While these technologies support the connection of multiple servers to handle several workloads simultaneously, often, automated test suites are executed on a single computer per pipeline. One potential solution is to increase the number of tests that can be run at the same time and the speed at which these tests are executed using Distributed Computing. Distributed Computing in this context refers to spreading workloads such as automated tests across several computers, potentially in different physical locations, to increase the amount of work that can be done in parallel. To that end, this paper seeks to answer to what extent can test execution leverage distributed computing to improve efficiency and speed, considering factors such as memory and CPU usage, execution time, the trade-offs between vertical and horizontal scaling, and cost-effectiveness at scale.

The remaining sections of this paper are as follows: The next section will provide some background information on distributed computing and using this strategy for testing. In Section 3, we describe the design of our solution and the methods we used to measure and compare our solution to traditional approaches. In the Section 4, we will present the results of our quantitative analysis which will then be further analyzed in Section 5. Finally, Section 6 will conclude the paper and propose potential future paths that can be explored.

## 2 Background

The demand for highly reliable software systems has led to new approaches in software testing. Some of which leverage distributed and cloud computing concepts. One notable system is the D-Cloud developed by Banzai et al. [1]. This system uses a cloud-based environment to test the fault tolerance and dependability of distributed software. D-Cloud allows fault injection through virtual machines (VMs) allowing researchers to simulate hardware failures which are critical for distributed system testing but challenging to replicate in physical environments. Through XML-configured automated testing scenarios, D-Cloud efficiently utilizes cloud resources to manage and parallelize tests. This workflow significantly reduced the time and cost of testing. This design demonstrates the benefits of cloud-based distributed testing, managing resources flexibly, and automating fault injection across multiple nodes [1]. This study further showcases the effectiveness of virtual environments in isolating faults and reducing resource constraints. This aligns with the objective to optimize memory and CPU usage, execution time, and cost-efficiency in this research paper.

As a result, Banzai et al.'s [1] research work serves as a strong foundation for this research paper as it explores how distributed computing can enhance the efficiency of testing projects of various scalability. By contrasting vertical scaling with single machines and horizontal scaling across multiple nodes, this research paper aims to build on Banzai et al.'s [1] findings, providing a broader evaluation of distributed test execution strategies in terms of scalability and resource management.

Replace all this with a relevant comprehensive literaure review in your specific area of interest in Software Testing. The Background section provides context and detailed information needed to understand the research problem. It typically includes a review of existing literature, key concepts, theories, and previous studies relevant to the topic. The goal is to frame the research by outlining what is already known, identifying gaps in the knowledge, and explaining why the current study is necessary. The background helps the reader understand how the research fits into the broader field and what it aims to contribute.

The rapid evolution of Generative AI has profoundly transformed technological capabilities, significantly influencing societal interactions, business processes, and educational methodologies. This section is divided into three main parts. The first part delves into the key technologies that have driven this transformation, highlighting their impact and implications. The second part presents an overview of current applications of Generative AI in the context of education, exploring how these innovations are being integrated into teaching and learning environments. The last section presents an overview of various readability metrics commonly used in the assessment of text.

## 3 Methodology

## 4 Findings

## 5 Discussion

## 6 Conclusion

## References

1. Banzai, T., Koizumi, H., Kanbayashi, R., Imada, T., Hanawa, T., Sato, M.: D-cloud: Design of a software testing environment for reliable distributed systems using cloud computing technology. 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (2010)