Scholar Publishing Group
Distributed Processing System
https://doi.org/10.38007/DPS.2022.030107
ISSN 2790-0916 Vol. 3, Issue 1: 54-61

# *Distributed Computing Test Model and Framework Based on Mapreduce*

**Aditinya Kumarun***

*Amman Arab University, Jordan*

**corresponding author*

***Keywords:*** Mapreduce Algorithm, Distributed Computing, Test Model, Test Framework

***Abstract:*** With the rapid development of network and related technologies and the continuous expansion of application fields, distributed systems have become the main choice for building network applications. This paper aims at the research and application of the distributed computing test model and framework based on MapReduce. This paper firstly proposes a performance testing method based on decision tree in cloud environment. The method is based on the decision tree model, and uses the test resources in the cloud environment to describe the test case generation, test script execution and performance bottleneck location in the performance test process in detail; by dividing and selecting the test case set, it can effectively reduce number of performance tests. Secondly, in view of the insufficient scalability of the performance test loop under the traditional test platform, this paper builds a cloud computing-based distribution system based on the open source cloud computing platform CloudStack, the performance testing tool LoadRunner, the Web lightweight development framework SSH, and the JavaScript library JQuery. The prototype system of the system performance test platform realizes the automatic configuration and dynamic expansion of the performance test environment in the cloud environment. The test platform mainly includes functions such as test task submission, virtual machine image matching, test environment automatic configuration, test task distribution and scheduling, cloud platform resource scheduling, and test result return. Experiments show that the algorithm in this paper effectively reduces the execution time by about 20%, and improves the system resource utilization by about 20%.

## 1. Introduction

With the rapid development of network and related technologies and the expansion of application fields, distributed systems have become the first choice for building network applications. At the same time, current distribution systems are mostly written in speech-based languages, and the size of distribution software has increased significantly since the integration of

distribution systems and speech-based technologies. But currently distributed system operations are difficult or sometimes impossible to correctly identify software problems, especially in communication and collaboration. If it is a test operation, concurrent operation, etc., the problem is more obvious. Therefore, it is necessary to study and design a test model and an effective test method suitable for the power distribution system according to the different characteristics of the current power distribution system and combined with the test technology [1-2].

In the research and application of the distributed computing test model and framework based on MapReduce, many scholars have studied it and achieved good results. For example, Yan Q discussed some problems of power distribution system components, and proposed a General distribution system test methods and related test standards. Fixed some issues in component testing [3]. Queiroz W designed a general MapReduce-based ETL process, and carefully studied and optimized the overall structure, size and actual load of the ETL process unit. However, the framework is not flexible enough to handle different data sources and cannot automate relational data automatically [4].

This paper firstly proposes a performance testing method based on decision tree in cloud environment. The method is based on the decision tree model, and uses the test resources in the cloud environment to describe the test case generation, test script execution and performance bottleneck location in the performance test process in detail; by dividing and selecting the test case set, it can effectively reduce number of performance tests. Secondly, in view of the insufficient scalability of the performance test loop under the traditional test platform, this paper builds a cloud computing-based distribution system based on the open source cloud computing platform CloudStack, the performance testing tool LoadRunner, the Web lightweight development framework SSH, and the JavaScript library JQuery. The prototype system of the system performance test platform realizes the automatic configuration and dynamic expansion of the performance test environment in the cloud environment. The test platform mainly includes functions such as test task submission, virtual machine image matching, test environment automatic configuration, test task distribution and scheduling, cloud platform resource scheduling, and test result return.

## 2. Research and Application of Distributed Computing Test Model and Framework Based on MapReduce

### 2.1. MapReduce Job Scheduling

When users use the MapReduce framework to process data, they will write corresponding MapReduce applications, each MapReduce application is regarded as a job, and users can use MapReduce to process different jobs. Common job types are batch jobs, interactive jobs, and productive jobs. Applications such as data mining and machine learning are common batch jobs. Such jobs do not have high requirements for completion time, so they often have a long execution cycle. However, interactive jobs such as instant query jobs require relatively high time, and such jobs need to get response results within a short period of time[5-6]. Productive jobs are jobs that have certain requirements on system resources such as memory or CPU. Statistics jobs are common productive jobs. Job scheduling refers to the rational allocation of tasks according to the information of the job control block. Job scheduling is determined by the scheduling controller, so the choice of job scheduling algorithm has an important impact on the execution efficiency of the job. Hadoop's own schedulers include FIFO scheduler, Fair Scheduler, and Capacity Scheduler [7-8].

The algorithm of the FIFO scheduler cannot make full use of existing resources and is relatively simple, so it cannot meet the diverse needs of enterprises. The fair scheduler and the computing power scheduler are multi-user schedulers. In a Hadoop cluster, there are usually situations where multiple users share a cluster to run various jobs[13-14]. Therefore, the fair scheduler and the computing power scheduler can try to meet the requirements of as many as possible. In the user environment, the resource requirements of various jobs. Later, many scholars have done a lot of research on scheduling algorithms. Many schedulers have been developed to meet the diverse needs of users, such as dynamic priority scheduling, delay scheduling, deadline-constrained scheduling, CHC-based genetic scheduling, resource prefetch-based scheduling and other algorithms. The internal organizational structure of different job schedulers is basically similar, mainly including configuration file loading module (loading configuration file information into memory), job monitoring module (registering job listeners to JobTracker at startup), status update module (updating queue and job information) ) and the scheduling module (choose the appropriate task for the TaskTracker)[9-10].

## 2.2. V Model

The V model is evolved on the basis of the rapid application development model, and can also be regarded as a variant of the waterfall model. Therefore, it has some characteristics of the waterfall model. Its model view is shown in Figure 1.

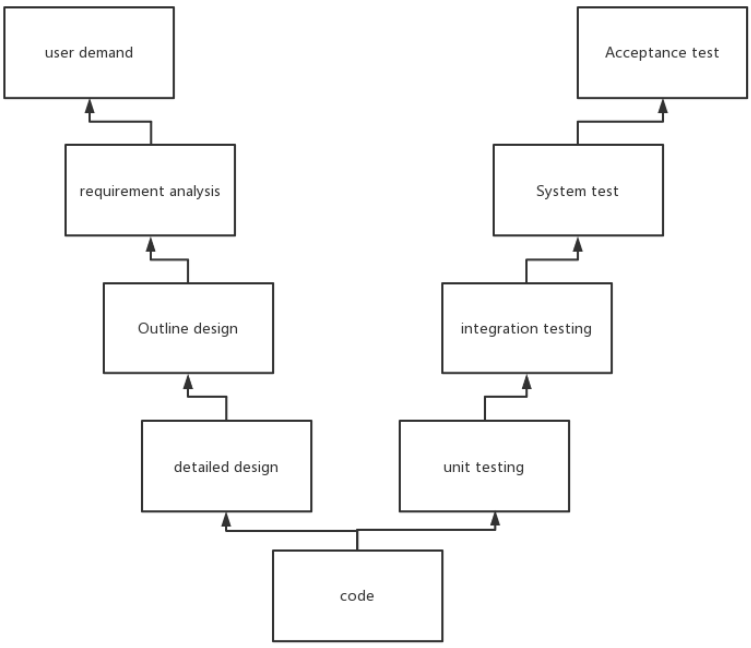


*Figure 1. V model test process*

Viewed from the horizontal direction, the left is the process of software design and implementation; the right is the verification of the results on the left, which is the dynamic testing process. From a vertical perspective, the more you go to the bottom, the more white-box testing

methods, the middle part is the combination of white-box and black-box, that is, gray-box testing, and in the user-facing acceptance stage, it is black-box testing.

In view of the limitations of the V model, the V model can be improved. Static tests can be added to the left part of the V model, that is, in the process of software design and implementation. While doing requirements analysis and product function design, testers can review various analyses. and design results, including a review of coding [11-12].

### 2.3. Algorithm Selection

Based on the ID3 algorithm, this paper introduces a set of pruning conditions in the decision tree construction process, adopts the first pruning strategy, terminates those nodes that do not meet the predetermined threshold for further expansion, and divides the test case set to reduce the final result. The size of the decision tree is generated to effectively reduce the number of test case designs required during performance testing.

Let the data set S be divided into S1, …, Sv, and total v subsets according to v different attribute values of attribute A, and S should divide the information gain calculation formula such as formula (1) [13-14].

$$\text{Gain}(S,A) = Info(S) - Info(S,A) \tag{1}$$

Among them, Info(S) represents the information entropy of the data set S, assuming that there are m categories in S, the calculation formula is as formula (2)[15-16]:

$$Info(S) = -\sum_{i=1}^{m} p_i \times \log_2(p_i) \tag{2}$$

## 3. Research and Application Design Experiment of Distributed Computing Test Model and Framework based on MapReduce

### 3.1. Test Methods in the Distributed System Test Model

The distributed system test model has the characteristics of comprehensiveness, agility and persistence, and the specific manifestations of these characteristics depend on the specific test methods used. Various test methods themselves have not been changed in the model, but are used flexibly according to the needs of distributed system testing. Therefore, the test methods used in the distributed system test model have the characteristics of diversification, close integration, independent coexistence and mutual coexistence. Features such as automation[17-19].

### 3.2. Experimental Design

This paper firstly compares the processing execution time of the system in this paper before and after using MOAP for the same data set, to highlight the superiority of the system in this paper.

## 4. Experimental Analysis of Distributed Computing Test Model and Framework based on MapReduce

### 4.1. Execution Time

This experiment compares the time to perform the ETL process with and without the MOAP

approach for Type-1 SCDs, both for initial loading and incremental loading in a data warehouse. The experimental data are shown in Table 1.

*Table 1. The Type 1 gradient dimension (Type-1 SCDs) compares the time of the execution of ETL procedures when using the MOAP method and without the MOAP method*

| | 40 | 80 | 120 | 160 | 200 | 240 |
|---|---|---|---|---|---|---|
| SCDS increment | 52 | 75 | 89 | 100 | 110 | 120 |
| SCDS Increments (MOAP) | 47 | 51 | 72 | 81 | 90 | 96 |
| SCDS initialise | 40 | 60 | 80 | 90 | 100 | 110 |
| SCDS Initialization (MOAP) | 30 | 40 | 50 | 70 | 80 | 90 |

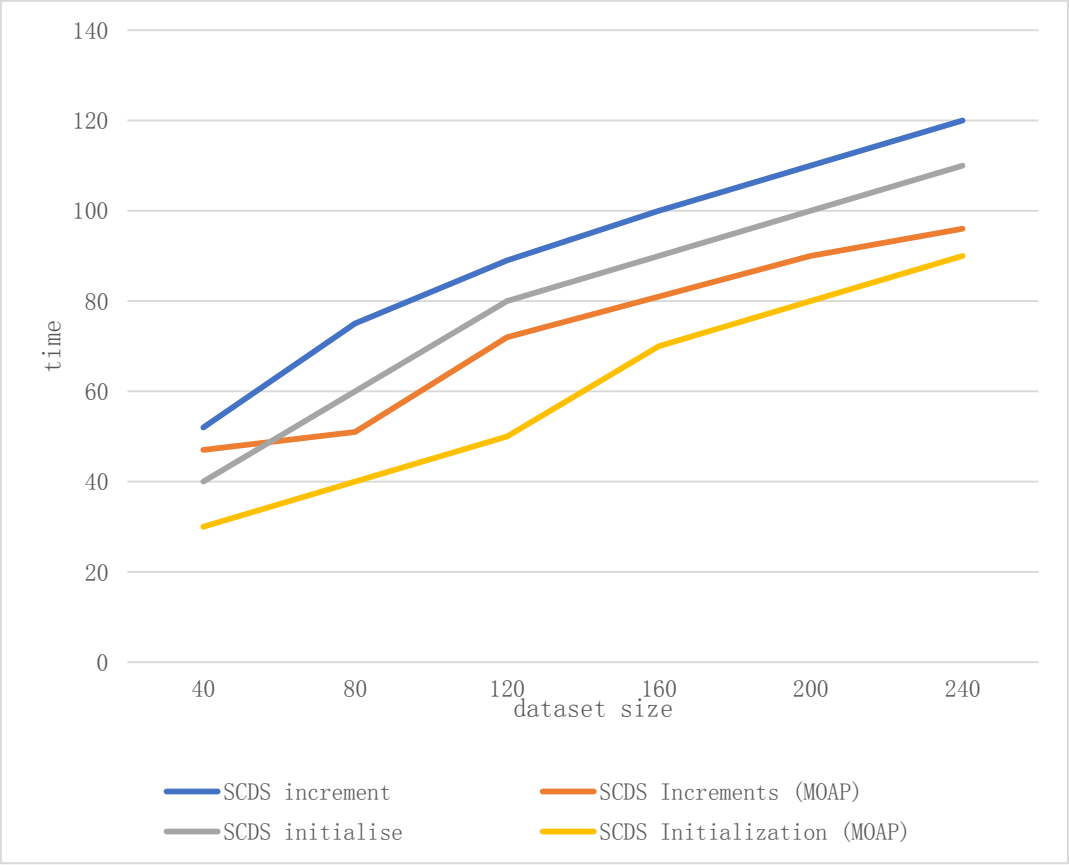


*Figure 2. Time comparison before and after using MOAP*

As can be seen from Figure 2, when processing the gradient dimension data of type 1, using the MOAP algorithm, the execution time is reduced when the data is initially loaded and incrementally loaded. Among them, when initializing data, the time is shortened by about 24%, and when

processing incremental data, the time is shortened by about 20%.

### 4.2. Comparative Analysis of Resource Utilization

Comparative analysis of resource utilization. Through the test experiment, the CPU and memory utilization of the load virtual machine before and after the performance test method based on the MapReduce model was applied to the cloud performance test platform. Expand the scale of the initial test case design, and design a total of 500 test case data as the initial input field of the system under test. Under the cloud test platform, the resource utilization of virtual machine instances is recorded every minute. The experimental data are shown in Table 2.

*Table 2. Resource utilization comparison*

| | 60 | 120 | 180 | 240 | 300 |
|---|---|---|---|---|---|
| Post applicationem | 27 | 37 | 40 | 38 | 35 |
| Before the application | 27 | 42 | 47 | 51 | 46 |

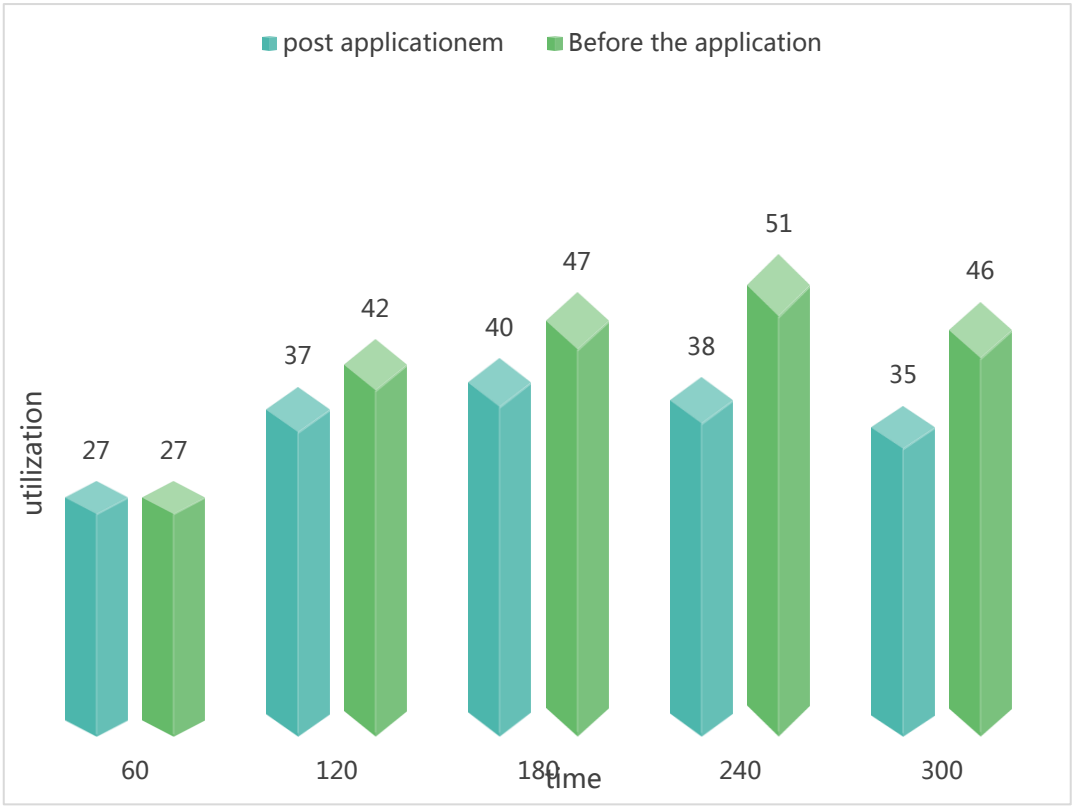


*Figure 3. Comparison of the system CPU resource utilization before and after the system application*

The results in Figure 3 show that after the performance testing method based on the MapReduce model is applied to the cloud test platform, the CPU and memory utilization of virtual machine

instances in the cloud test platform are significantly reduced, and the utilization efficiency of corresponding resources is improved.

## 5. Conclusion

The model is based on theoretical development, including static testing in the requirements analysis stage before the theoretical development and acceptance testing and feature testing after the iterative development stage. Aiming at the weakness of non-distributed unit testing methods of distributed systems, a unit testing method suitable for distributed systems is proposed, which can compare the functional areas of distributed systems and test the input of code units under corresponding commands. Make actionable visualization calls for testing. Coupled with the high implementation cost of distributed system testing, the distributed system testing process is determined, the test case and test code are separated, and the test performance of some distributed systems is improved. This paper discusses automated system testing, and takes the application of MapReduce-based testing tools in distributed systems as an example to analyze and study the characteristics of automated testing tools for distributed systems in order to distribute the most effective tests. System test models and procedures.

## Funding

This article is not supported by any foundation.

## Data Availability

Data sharing is not applicable to this article as no new data were created or analysed in this study.

## Conflict of Interest

The author states that this article has no conflict of interest.

## References

*[1] Ketu S, Mishra P K, Agarwal S. Performance Analysis of Distributed Computing Frameworks for Big Data Analytics: Hadoop Vs Spark. Computacion y Sistemas, 2020, 24(2):669–686. https://doi.org/10.13053/cys-24-2-3401*

*[2] Li F, Chen J, Wang Z. Wireless MapReduce Distributed Computing. IEEE Transactions on Information Theory, 2019, 65(10):610 https://doi.org/10.1109/TIT.2019.2924621*

*[3] Yan Q, Wigger M, Yang S, et al. A Fundamental Storage-Communication Tradeoff for Distributed Computing With Straggling Nodes. IEEE Transactions on Communications, 2020, PP(99):1-1. https://doi.org/10.1109/ISIT.2019.8849615*

*[4] Queiroz W, Capretz M, Dantas M. A MapReduce Approach for Traffic Matrix Estimation in SDN. IEEE Access, 2020, PP(99):1-1.*

*[5] Kavitha C, Lakshmi R S, Devi J A, et al. Evaluation of worker quality in crowdsourcing system on Hadoop platform. International Journal of Reasoning-based Intelligent Systems, 2019, 11(2):181. https://doi.org/10.1504/IJRIS.2019.099856*

*[6] Meraou M A, Al-Kandari N M, Raqab M Z. Univariate and Bivariate Compound Models Based*

*on Random Sum of Variates with Application to the Insurance Losses Data. Journal of Statistical Theory and Practice, 2022, 16(4):1-30.*

*[7] Zhou L, Gai X, Lu Y, et al. Research and Application of Intelligent Learning System for Power Grid All-Element Simulation Based on Microservice. Journal of Physics: Conference Series, 2021, 1802(4):042103 (8pp).*

*[8] Ha H, Kwak Y. Prediction model for discomfort luminance levels of head‐mounted displays. Color Research And Application, 2022, 47(4):1035-1041. https://doi.org/10.1002/col.22783*

*[9] Bartling M, Resch B, Reichenbacher T, et al. Adapting mobile map application designs to map use context: a review and call for action on potential future research themes. Cartography and Geographic Information Science, 2022, 49(3):237-251.*

*[10] Oh S, Kwak Y. A hue and warm‐cool model for warm‐cool based correlated color temperature calculation. Color Research And Application, 2022, 47(4):953-965. https://doi.org/10.1002/col.22764*

*[11] Veluchamy M, Subramani B. Cuckoo search optimization‐based image color and detail enhancement for contrast distorted images. Color Research And Application, 2022, 47(4):1005-1022.*

*[12] Oscar Sánchez, Min A, Mendonca A, et al. Development and application of novel BiFC probes for cell sorting based on epigenetic modification. Cytometry Part A, 2022, 101(4):339-350.*

*[13] Torii R, Yacoub M. CT-based fractional flow reserve: development and expanded application.. Global cardiology science & practice, 2021, 2021(3):e202120.*

*[14] Zjavka L. Power quality statistical predictions based on differential, deep and probabilistic learning using off‐grid and meteo data in 24‐hour horizon. International Journal of Energy Research, 2022, 46(8):10182-10196. https://doi.org/10.1002/er.7431*

*[15] Leibowitz S G, Pennino M J, Beyene M T. Parsing Weather Variability and Wildfire Effects on the Post‐Fire Changes in Daily Stream Flows: A Quantile‐Based Statistical Approach and Its Application. Water R*

*[16] Holsteen K, Hittle M, Barad M, et al. Development and Internal Validation of a Multivariable Prediction Model for Individual Episodic Migraine Attacks Based on Daily Trigger Exposures.. References, 2020, 60(10):2364-2379.*

*[17] Dutta A K, Mandal J J, Bandyopadhyay D. Application of Quintic Displacement Function in Static Analysis of Deep Beams on Elastic Foundation. Architecture, Structures and Construction, 2022, 2(2):257-267.*

*[18] Fernandes C I, Veiga P M, Adro F. The impact of innovation management on the performance of NPOs: Applying the Tidd and Bessant model (2009). Nonprofit Management and Leadership, 2022, 32(4):577-601. https://doi.org/10.1002/nml.21501*

*[19] Billger M, Amborg E, Zboinska M A, et al. Colored skins and vibrant hybrids: Manipulating visual perceptions of depth and form in double‐curved architectural surfaces through informed use of color, transparency and light. Color Research And Application, 2022, 47(4):1042-1064. https://doi.org/10.1002/col.22784*