Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/40/20230645

MapReduce based on serverless platforms

Yuhong Zhang

Aulin College of Northeast Forestry University, Harbin, Heilongjiang province, 150000, China

y.h.zhang224906@nefu.edu.cn

Abstract. This study intends to investigate the application of the MapReduce (MR) framework based on serverless computing in big data processing. By combining the MapReduce model with serverless computing, efficient data processing is achieved. In this framework, the phases of Map task execution, reduce task execution, etc. are accomplished through stateless serverless functions, and data storage is realized with the help of cloud storage platforms (e.g., OSS). In this paper, the author introduces the basic theory of MR, the basic theory of serverless computing, describes the framework implementation process, and discusses the role of OSS in distributed computing. The outcomes of the trial indicate the average execution time of the framework for a WordCount task on 100 pieces of TOEFL English reading data is 6.81 seconds. The discussion analyzes the framework's advantages (high elasticity, resource utilization) and disadvantages (cold start latency, unsuitable for long time tasks). Future research directions encompass performance optimization, long-time task processing, state management, etc. In summary, this study provides valuable insights for the practical application of serverless computing in big data processing.

Keywords: Serverless, MapReduce, Cloud Computing, Functional Programming, Cloud Storage.

1. Introduction

With the rapid advancement of information technology in recent years, big data analysis has penetrated into various fields and become the core of research and application. MapReduce, as a kind of distributed computing model, has demonstrated powerful processing capability and application potential in the big data domain. Traditional deployment methods of MapReduce based on virtual machines or physical machines have limitations in resource utilization, elastic expansion, etc., leading to resource waste and startup delays. Moreover, developers need to pay attention to numerous underlying details, including cluster configuration and scheduling, which undoubtedly increase development and maintenance costs [1]. These limitations constrain the flexibility and efficiency of big data processing. To address these challenges, serverless computing, as an emerging computing paradigm, has rapidly gained widespread attention from academia and industry. Serverless computing abandons the traditional virtual machine deployment model and uses functions as the basic computing units to realize elastic resource allocation and pay-as-you-go. Automated resource management and highly abstracted deployment methods enable developers can focus more on the core logic of data processing. It can be said that serverless computing provides a new solution for big data processing.

DOI: 10.54254/2755-2721/40/20230645

In existing studies, many scholars have begun exploring the potential applications of serverless computing in the field of big data processing and attempting to combine serverless computing with distributed computing. For instance, Erwin et al. explored the evolution of cloud computing and serverless computing, as well as their evolution, state of the art, major challenges and opportunities [2]. Singh et al. explored the rapid growth of cloud computing in different domains, which has had a significant impact on information technology frameworks and service delivery methods [3]. Li et al. provide a comprehensive survey of Serverless computing, focusing on its infrastructure characteristics and analyzing solutions to the challenges facing Serverless computing [4]. Although the aforementioned studies have greatly contributed to the research progress in serverless computing, research on applying serverless computing to the MapReduce framework is still relatively limited, particularly the practical application aspect. In addition, research on efficient file storage management and scheduling in this framework is relatively insufficient.

To cope with this problem, the main goal of this research is to implement the MapReduce framework based on serverless computing in data processing. First, through the existing MapReduce frameworks and serverless computing models, analyze their features, advantages and limitations. Then, design and implement the MapReduce framework based on serverless computing, detailing the architecture of the framework, data transfer and other key parts, especially how to implement the WordCount function in a serverless environment. Finally, the feasibility and validity of this research are verified through experimental evaluation to compare the performance metrics of the MapReduce framework under traditional deployment and serverless computing.

Specifically, this study will attempt to build a MapReduce framework based on serverless computing to implement the classic WordCount functionality. In this process, the author will explore how to efficiently manage data input and output in a serverless computing environment by combining it with Alibaba Cloud's OSS file storage. In summary, this research will focus on exploring the use of the functional programming approach of serverless computing to build MapReduce frameworks to solve specific problems, such as realizing the WordCount function, in order to improve the efficiency and performance of big data processing. Through in-depth research on the cross-application of serverless computing and MapReduce, this study aims to provide valuable guidance for the practical application of serverless computing in the field of distributed data processing.

2. Method

A comprehensive explanation of the implementation strategy will be given in this chapter. of the MapReduce framework based on serverless computing in detail, focusing on the key implementation processes from input to output. It will cover MapReduce's basic theory, and serverless computing's fundamental theory, discuss the role of cloud storage platforms in distributed computing, and provide insights into the principles and technical details behind the framework.

2.1. Basic Theory of MapReduce (MR)

MapReduce (MR), as a distributed computing model, aims to solve large-scale data processing problems. Its fundamental principle includes the Map phase and the Reduce phase as its two primary phases [5].

Map phase. In this phase, the input data, which is cut into several small blocks, is passed to multiple Mapper tasks in parallel. Each Mapper task executes a user-defined Map function that maps each element of the input data into a series of key-value pairs.

Shuffle Phase. Following the Map phase, the Shuffle phase sorts and groups key-value pairs based on their keys, enabling more efficient data processing by subsequent Reduce operations.

Reduce Phase. In this stage, Reduce tasks merge the value lists corresponding to keys according to the grouped keys, generating the final output result.

2.2. Basic Theory of Serverless Computing

The serverless computing model is a computing model that frees users from the underlying resource management. The core idea is to use functions as the basic unit of computation, with cloud service

DOI: 10.54254/2755-2721/40/20230645

providers dynamically allocating and managing computing resources [6]. The Serverless computing model has the following characteristics:

Event-driven. Serverless functions execute upon specific events, ensuring computation occurs when needed and avoiding resource wastage.

Elastic Scalability. The Serverless computing platform automatically scales computing resources based on request loads, enabling applications to handle varying workloads.

Statelessness. Every Serverless function should be stateless, meaning functions do not maintain any persistent state. This statelessness attribute contributes to enhanced scalability [7].

2.3. Key Implementation Process from Input to Output

Combining MR and Serverless theory, we will construct a MapReduce framework based on serverless computing to realize efficient data processing. The following is the key implementation process from input to output:

(1) **Data Input and Trigger**. Upload new data files to an object storage system (e.g., OSS), locally trigger the MR framework, and initiate responsive functions in the function computing service.

(2) **Map Task Execution**. The Map task is decomposed into multiple independent Serverless functions, each function processes a small chunk of the input data and maps it into key-value pairs.

(3) **Shuffle and Grouping**. In a Serverless environment, the Shuffle and Grouping phases are implemented differently from the traditional MR model. The sorted and grouped data can be realized by invoking the cloud storage service to ensure that the grouped data can be processed by the Reduce task.

(4) **Reduce Task Execution.** Similar to Map tasks, Reduce tasks are divided into multiple Serverless functions. Each function handles key-value pairs for a group, ultimately generating the output result of Reduce.

By decomposing the Map and Reduce tasks in the MR model into stateless Serverless functions, the advantages of the Serverless computing can be fully utilized to achieve dynamic resource allocation and highly elastic scaling. The event-driven nature of the Serverless computing platform also ensures real-time and efficient data processing.

2.4. Cloud Storage Platform - Object Storage Service (OSS)

In the MapReduce framework based on serverless computing, the input data, intermediate data and output data of a file involve storage, and the management and storage of these data are crucial. Cloud storage is the product of the integration of distributed storage and virtual technologies. A technique called “cloud storage” enables you to utilize online storage space [8]. Cloud storage platforms, such as Alibaba Cloud OSS (Object Storage Service), play a key role in realizing distributed computing.

OSS Object Storage Principle. The “killer app” of the cloud era is object storage. The service offers users great levels of scalability, elasticity, and dependability for both object persistence and retrieval [9]. OSS is a distributed storage service offered by Alibaba Cloud, providing massive data storage capabilities. In the Serverless-based MapReduce framework, OSS plays a crucial role in data storage, transmission, and management.

Application in workflow. In the MapReduce framework based on serverless computing, we store input data, intermediate data and output data in OSS object storage to realize data persistence and sharing. Specifically, the workflow is as follows:

Input data phase. The user uploads the input data file into the OSS. When the Map task is triggered, the relevant Serverless function can read the input data from the OSS, which helps to realize the efficient distribution and management of data.

Map task execution phase. After processing the input data, the Map task writes the intermediate results to the OSS. This intermediate data is used as input for the Reduce task and can be shared and passed among different Serverless functions.

Reduce task execution phase. The Reduce task reads the intermediate data previously stored in OSS, performs the merging operation, and then writes the final output data back to OSS, which enables the

DOI: 10.54254/2755-2721/40/20230645

data to be stored and managed in a distributed computing environment to ensure the reliability and consistency of the data.

By integrating OSS object storage into the MapReduce framework based on serverless computing, we can fully utilize the high performance and high availability of OSS to achieve large-scale data storage and sharing. At the same time, by using OSS in the whole workflow, we can also reduce the burden of local storage and improve the processing efficiency and scalability of data.

In summary, this study combines the basic theory of MR with the basic theory of Serverless to construct a MapReduce framework based on serverless computing. By splitting Map and Reduce tasks into independent stateless functions and combining them with cloud storage services, we realize an efficient data processing flow from input to output. This integration provides an innovative solution for big data processing and fully utilizes the potential of Serverless computing in the field of data processing.

3. Implementation and Framework Design

This chapter will detail the experimental process and results of the MapReduce framework based on serverless computing in real applications. The experiments will cover the introduction of experimental raw data, description of evaluation metrics, and experimental analysis compared with traditional MapReduce to reveal the advantages and disadvantages of serverless computing in big data processing.

3.1. Introduction of Experimental Raw Data

The raw data for this experiment comes from 100 English articles, aiming to verify the effectiveness of the MapReduce framework based on serverless computing in word frequency statistics. The average size of these 100 input files is about 591.358 KB. These data will be used as the input of the experiment to perform distributed word frequency statistics in the framework to assess the framework's effectiveness and performance.

3.2. Experimental Methods and Evaluation Indicators

In this experiment, to realize the WordCount function, five Mapper functions and four Reducer functions are used, and the combination of serverless computing and object storage is adopted to complete the construction of the whole framework. The framework is started locally by triggering the functions, so that all functions in MR run concurrently according to the workflow, the running time from input to output is recorded, and the average of the running time is obtained from several tests to minimize the error caused by chance. According to the experimental findings, the framework typically executes in 6.811523637006758 seconds (all functions are configured for 16-core vCPU cores with 16GB memory).

4. Discussion

The MapReduce framework based on serverless computing shows some advantages in this experiment, but there are also some disadvantages, which are analyzed as follows compared to the traditional MapReduce:

4.1. Advantages of the Framework

The Serverless-based MapReduce framework offers several advantages, including:

(1) Elastic Scaling. Serverless computing allows computing resources to be automatically adjusted according to changes in workloads, which makes the framework better able to adapt to the needs of data processing of different sizes.

(2) Reduced Resource Management Burden. Serverless computing eliminates the tediousness of underlying resource management, allowing developers to focus more on the implementation of business logic.

Efficient Resource Utilization: Through the Serverless model, computing resources can be allocated on demand, avoiding the waste of idle resources.

DOI: 10.54254/2755-2721/40/20230645

4.2. Limitations of the Framework

The framework also has certain limitations, such as:

(1) Cold-start latency. When a function is invoked for the first time in serverless computing, the runtime environment needs to be initialized, which could cause an execution delay [10].

(2) Not suitable for long tasks. Serverless computing is suitable for short and low-latency tasks but may be less suitable for long-running tasks.

4.3. Future Research Directions

Future research can focus on the following directions to enhance the framework's capabilities:

(1) Performance optimization. Further optimization of function execution time and resource utilization.

(2) Long-term task processing. Exploring mechanisms to address long-running task limitations in the Serverless environment.

(3) State management. Investigating strategies for efficiently managing stateful data in a stateless Serverless context.

In summary, the MapReduce framework based on serverless computing provides a novel solution for big data processing, giving full play to the advantages of cloud computing and distributed computing. Future research will continue to advance this field and address some of the limitations of current frameworks to achieve more efficient and flexible big data processing.

5. Conclusion

This paper summarizes the key findings of the study and provides concluding remarks on the application of Serverless computing in the MapReduce framework. The study successfully implemented a Serverless-based MapReduce framework for efficient large-scale data processing. By integrating Serverless computing and cloud storage platforms, the framework achieved event-driven execution, dynamic resource allocation, and efficient data management. Through this research, we have successfully constructed a MapReduce framework based on serverless computing to realize efficient big data processing. The framework gives full play to the elastic scaling and resource utilization advantages of serverless computing, realizes the distributed processing of Map and Reduce tasks through Serverless functions, and combines with OSS to achieve efficient data management. The experimental results validate the feasibility of the framework and reveal both its advantages and disadvantages. In the future, we will focus on the performance optimization of the framework, investigate more efficient cold-start strategies, explore solutions applicable to long-time tasks, and consider the implementation of state management. In summary, the MapReduce framework built on serverless computing offers novel prospects for distributed computing and provides a wide scope for big data processing.

References

[1] Rahman, M. M., & Hasibul Hasan, M. (2019, October 1). Serverless Architecture for Big Data Analytics. IEEE Xplore. https://doi.org/10.1109/GCAT47503.2019.8978443.

[2] van Eyk, Erwin, et al. "Serverless Is More: From PaaS to Present Cloud Computing." IEEE Internet Computing, vol. 22, no. 5, Sept. 2018, pp. 8–17, https://doi.org/10.1109/mic.2018.053681358.

[3] Rawat, A., & Singh, P. (2021). A Comprehensive Analysis of Cloud Computing Services. Journal of Informatics Electrical and Electronics Engineering (JIEEE), 2(3), 1–9. https://doi.org/10.54060/jieee/002.03.003

[4] Li, Y., Lin, Y., Wang, Y., Ye, K., & Xu, C.-Z. (2022). Serverless Computing: State-of-the-Art, Challenges and Opportunities. IEEE Transactions on Services Computing, 1–1. https://doi.org/10.1109/tsc.2022.3166553.

[5] Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. Communications of the ACM, 51(1), 107–113. https://doi.org/10.1145/1327452.1327492.

DOI: 10.54254/2755-2721/40/20230645

[6] Shafiei, H., Khonsari, A., & Mousavi, P. (2022). Serverless Computing: A Survey of Opportunities, Challenges, and Applications. ACM Computing Surveys. https://doi.org/10.1145/3510611.

[7] Mohanty, S. K., Premsankar, G., & di Francesco, M. (2018). An Evaluation of Open Source Serverless Computing Frameworks. 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom). https://doi.org/10.1109/cloudcom2018.2018.00033.

[8] He, Q., Li, Z., & Zhang, X. (2010, December 1). Study on Cloud Storage System Based on Distributed Storage Systems. IEEE Xplore. https://doi.org/10.1109/ICCIS.2010.351.

[9] Zhou, S., Xu, E., Wu, H., Du, Y., Cui, J., Fu, W., Liu, C., Wang, Y., Wang, W., Sun, S., Wang, X., Feng, B., Zhu, B., Tong, X., Kong, W., Liu, L., Wu, Z., Wu, J., Luo, Q., & Wu, J. (2023). {SMRSTORE}: A Storage Engine for Cloud Object Storage on {HM-SMR} Drives. Www.usenix.org. https://www.usenix.org/conference/fast23/presentation/zhou.

[10] Vahidinia, P., Farahani, B., & Aliee, F. S. (2020). Cold Start in Serverless Computing: Current Trends and Mitigation Strategies. 2020 International Conference on Omni-Layer Intelligent Systems (COINS). https://doi.org/10.1109/coins49042.2020.9191377.