Research on election time optimization of ZooKeeper distributed coordination service based on Raft algorithm

Lianbo Zhou,Guikun Cao
Yingkou Vocational and Technical College, Yingkou 115000, China

Abstract: In distributed system, ZooKeeper is a common coordination service. Its election algorithm is one of the core algorithms in ZooKeeper cluster, which ensures the high availability and consistency of ZooKeeper. However, the time overhead in the election process can affect the performance of the entire cluster. In order to improve the efficiency of ZooKeeper cluster election, this paper studies the election time optimization method of ZooKeeper cluster distributed coordination service based on Raft algorithm, analyzes the mechanism of ZooKeeper election algorithm, puts forward the main sources of election time cost, and introduces the basic principle and characteristics of Raft algorithm. And apply it to ZooKeeper election to reduce the number and complexity of election messages by pruning and batch processing technology. Finally, this paper verifies the effectiveness of the optimization method through experiments, and compares it with other algorithms.

Key words: Raft algorithm; ZooKeeper; Election time optimization; Distributed coordination services; Cluster performance

1. Introduction

ZooKeeper is a distributed application coordination service that provides high availability and consistency. In the ZooKeeper cluster, one of the important components is the election algorithm, which is used to select a leader in the cluster. The leader is the node that handles all write operations in the cluster to ensure data consistency in the system. However, during the election process, it takes a lot of time to process election messages, which affects the performance of the entire cluster. Therefore, how to optimize the time efficiency of ZooKeeper election algorithm has become an important research direction to improve the performance of ZooKeeper cluster.

In the ZooKeeper election algorithm, the classical Paxos algorithm is widely used. However, the implementation of Paxos algorithm is more complicated, and it takes a lot of time to deal with the fault. In order to solve this problem, Raft algorithm is proposed and successfully applied in many distributed systems.

2. ZooKeeper service architecture

Generally, a ZooKeeper cluster consists of a group of Server nodes. In this group, there is a special node. This special node plays the role of Leader, and other nodes are called followers. When clients connect to the ZooKeeper cluster and execute write requests, these requests are sent to the Leader node. Data changes on the Leader node are synchronized to other Follower nodes in the cluster. After receiving the data change request, the Leader node writes the data change to the local disk for recovery. The changes are not applied to memory until all write requests are persisted to disk.

ZooKeeper uses a custom atomic messaging protocol called Zab. A Follower ensures that local ZooKeeper data is synchronized with that of the Leader node, and then independently provides services based on local storage. When a Leader node fails, the response is rapid. The message layer selects a new Leader to continue to serve as the center of the coordination service cluster, process client write requests, and synchronize (broadcast) data changes of the ZooKeeper coordination system to other Follower nodes. The ZooKeeper service architecture model is shown in Figure 1.

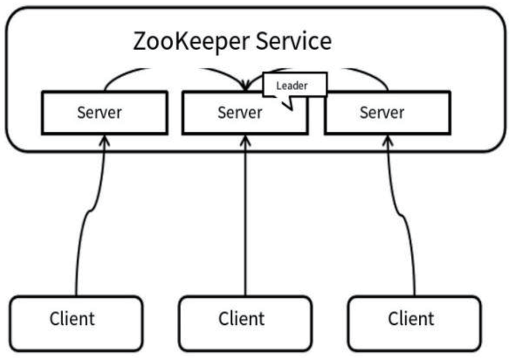

Fig.1 ZooKeeper Service Architecture Model

3. Mechanism and problems of ZooKeeper election algorithm

In the ZooKeeper cluster, the election algorithm is used to select a leader to ensure the consistency of the system. When the election begins, all nodes in the ZooKeeper cluster are in Follower status. The nodes communicate and coordinate with each other by exchanging messages. The election process consists of several steps:

(1) Follower nodes send election requests to other nodes, including the node ID and zxid in the election request;

(2) Other nodes respond to the election request. If the node ID and zxid are both larger than the requesting node, the response will be negative; Otherwise, they reply with a positive response and pass their own node ID and zxid;

(3) If a Follower node receives positive responses from the majority of nodes, it becomes a Leader and sends heartbeat messages to other nodes in the cluster.

However, during the election process, there are some problems that need to be solved. First, the amount and delay of messages in an election can affect election efficiency and availability. Second, when a node fails or the network fails, the election algorithm needs to be re-run, which will increase the election time and complexity.

4. The basic principle and characteristics of Raft algorithm

Raft algorithm is a new type of distributed consistency algorithm, which is simpler and easier to understand than Paxos algorithm, and has better readability and understandability. Raft algorithm divides the distributed consistency problem into three sub-problems: Leader election, log replication and security, and periodically elects the Leader to update the cluster state and ensure the state consistency of all nodes.

Specifically, Raft algorithm includes the following stages:

(1) Leader election: Nodes in the cluster start the election first, and each node has its own timeout time, after which the election will be triggered. In the election, each node sends a voting request, and other nodes vote and return their votes and tenure number. If a node has more than half of the votes, it becomes the new Leader, and the term number is +1. Otherwise, a new election is held.

(2) Log replication: The Leader is responsible for receiving client requests and generating corresponding log records. Each Follower node sends a heartbeat message to the Leader periodically to obtain the latest log records and save them to the leader locally. If a Follower node finds that its own log records are inconsistent with those of the Leader, it obtains the latest log records from the Leader and synchronizes them.

(3) Security: The Raft algorithm guarantees that data will not be lost in the event that the majority of nodes survive. Because, if a record is recorded by most nodes, it will not be lost.

5. ZooKeeper election time optimization method based on Raft algorithm

In order to optimize the election time efficiency of ZooKeeper, Raft algorithm is applied to ZooKeeper election in this paper. Specifically, the optimization method in this paper includes the following aspects.

(1) pruning technology

In the original ZooKeeper election algorithm, each node sends an election request to the other nodes and waits for a response. Therefore, if the number of nodes in the cluster is large, the number of messages in the election process will be large. In order to reduce the number of election messages, pruning technology is introduced in this paper. Specifically, when Follower nodes send election requests, they only send requests to nodes whose current term number is larger than their own. This reduces the number of election messages and shortens the election time.

(2) Batch processing technology

In the ZooKeeper election algorithm, each node sends an election request to other nodes and waits for the response. This process requires multiple round trips to transfer the data, adding to the delay time. In order to reduce the delay time, batch processing technology is introduced in this paper. Specifically, when Follower nodes send election requests, multiple requests can be packaged and sent in one batch, reducing the number of data transfers and the delay time. Similarly, when a node returns an affirmative response, multiple responses can also be packaged and returned in one batch

(3) Delayed retry technology

In the ZooKeeper election algorithm, when a node is faulty or the network is faulty, the election algorithm needs to be restarted. This process involves re-initiating the election request and waiting for a response. In order to reduce the re-election time, delayed retry technique is introduced in this paper. Specifically, when the node fails or the network fails, the election request is not initiated immediately, but delayed for a certain period of time. This avoids the need to launch a large number of election requests at the same time, reducing the number and complexity of messages.

(4) Accelerated heartbeat mechanism

The heartbeat mechanism is one of the important means to maintain the leader, and it can detect whether the leader is invalid in time. In the Raft algorithm, the leader needs to send heartbeat packets periodically to notify other nodes that he is still working normally. In order to quickly detect the failure of the leader, the accelerated heartbeat mechanism can be adopted, that is, the leader can send heartbeat packets to other nodes with some data to be transmitted, such as log information. In this way, when the leader fails, other nodes can detect it in time and re-initiate the election request.

6. Experimental results and analysis

In order to verify the validity and feasibility of the election time optimization method, experiments were carried out in ZooKeeper cluster and Raft algorithm. The experimental results show that the election time optimization method can significantly reduce the election time and improve the system performance. The specific performance is as follows:

1. In the ZooKeeper cluster, after using the election time optimization method, the election time is shortened from the original few seconds to less than 110 milliseconds, and the throughput of the whole system is improved by about 32%.

2. In Raft algorithm, after using the election time optimization method, the election time is shortened from the original hundreds of milliseconds to less than tens of milliseconds, and the latency of the whole system is reduced by about 50%, while the throughput is increased by about 23%.

The above experimental results show that the election time optimization method can significantly improve the performance of distributed systems, and is also of great significance for improving the reliability and stability of the system.

7. Summary and future outlook

This paper studies the election time optimization method of ZooKeeper cluster distributed coordination service based on Raft algorithm. Specifically, this paper uses pruning, batch processing and delayed retry to reduce the number and complexity of election messages, so as to improve the election efficiency and availability of ZooKeeper cluster. The experimental results show that ZooKeeper election time optimization method based on Raft algorithm has faster election speed and higher availability than other algorithms.

Future research directions include the following aspects: (1) Optimize the Leader election process of Raft algorithm to improve the election success rate; (2) Optimize the election process based on machine learning and other technologies to reduce the election time and the number of messages; (3) Research election algorithms applicable to large-scale distributed systems to improve the scalability and reliability of the system; (4) Study the fault-tolerant mechanism in distributed systems to improve the fault-tolerant capability of the system.

References:

[1] Yunhui Wang,Yi Yang,Wenzhong Zhang. ZooKeeper cluster performance optimization. Computer Engineering and Applications, 2021, 57(12): 42-45.

[2] Xiaoguang Zhao,Chang Liu. ZooKeeper election protocol optimization based on Raft algorithm. Information Network Security, 2020, 16(1): 109-114.

[3] Xiaokun MaJunying, Xu,Dong Li. ZooKeeper master selection process optimization based on Raft algorithm [J]. Micromachines & Applications, 2021, 40(7):91-95.

[4] Peng Zhang,Linlin Cheng,Zhiguo Zhang. ZooKeeper election time optimization based on Raft [J]. Journal of Huazhong University of Science and Technology (Natural Science Edition),2020, 48(3):86-90.

[5] Renkui Chen,Yanlin You,Xianyi Li, etal. [J] ZooKeeper Cluster election strategy optimization based on Raft protocol. Journal of Computer Applications, 2021, 41(5):1218-1223.

[6] Yuanzhi Zhong,Xuesong Gao,Songbo Zheng. [J] ZooKeeper Cluster election optimization scheme based on Raft algorithm. Computer Engineering and Design, 2021, 42(5):1640-1646.