Advances in Computer Science Research, volume 87

3rd International Conference on Mechatronics Engineering and Information Technology (ICMEIT 2019)

A Transaction Processing Method for Distributed Database

Zhian Lin [a], Chi Zhang [b]

School of Computer and Cyberspace Security, Communication University of China, Beijing, China

[a]282666409@qq.com, [b]949659861@qq.com

Abstract. This paper introduces the distributed transaction processing model and two-phase commit protocol, and analyses the shortcomings of the two-phase commit protocol. And then we proposed a new distributed transaction processing method which adds heartbeat mechanism into the two-phase commit protocol. Using the method can improve reliability and reduce blocking in distributed transaction processing.

Keywords: distributed transaction, two-phase commit protocol, heartbeat mechanism.

1. Introduction

Most database services of application systems will be distributed on several servers, especially in some large-scale systems. Distributed transaction processing will be involved in the execution of business logic. At present, two-phase commit protocol is one of the methods to distributed transaction processing in distributed database systems. The two-phase commit protocol includes coordinator (transaction manager) and several participants (databases). In the process of communication between the coordinator and the participants, if the participants without reply for fail, the coordinator can only wait all the time, which can easily cause system blocking. In this paper, heartbeat mechanism is introduced to monitor participants, which avoid the risk of blocking of two-phase commit protocol, and improve the reliability and efficiency of distributed database system.

2. Distributed Transactions

2.1 Distributed Transaction Processing Model

In a distributed system, each node is physically independent and they communicates and coordinates each other through the network. Because of the transaction mechanism, data operations on each independent node can be guaranteed to satisfy ACID. However, the transaction performance of other nodes cannot be accurately known between independent nodes. So in theory, the two machines cannot achieve the same state. If you want make data consistency in distributed deployed machines, you need to ensure that all node successfully write operations, or all node fail to write operations. However, when a machine executing a local transaction, it cannot know the execution result of the local transaction in other machines, so it does not know whether the transaction should be commit or rollback. Therefore, the better solution is to introduce a "coordinator" to uniformly schedule the execution of all distributed nodes.

The distributed transaction processing model, shown in Figure 1, includes applications, transaction manager (TM) and resource manager (RM). Applications encapsulate business operations. Transaction managers as a "coordinators", including transaction processing monitors which be used to run user applications, and it assign identities to transactions. Transaction managers monitor the processes of user applications, and command transaction commit or rollback. When user applications request commit or rollback, transaction managers coordinate among all participating resource managers through XA interfaces; the common resource managers are databases.

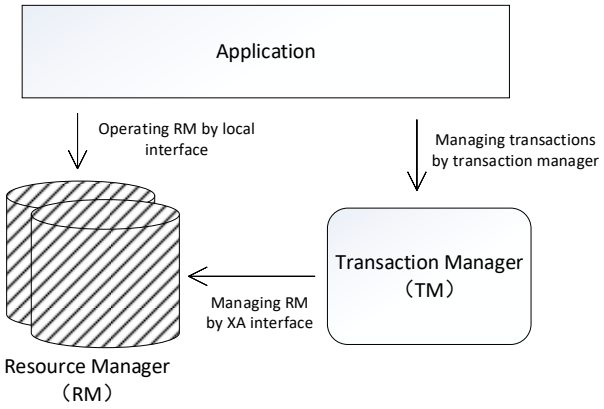

Fig.1 Distributed Transaction Processing Model

2.2 Mediator Pattern

The mediator pattern is one of the commonly used design patterns. It encapsulates a series of object interactions by a mediator object. The mediator makes the objects do not need to refer to each other explicitly, so that the coupling of objects is loose and the interaction between them can be changed independently. This model encapsulates the relationship between objects through intermediary objects, so that each object communicates with others through intermediary objects without knowing the specific information of other objects. At the same time, it reduces the relationship between system objects by referring to intermediary objects, and improves the reusability of objects and the extensibility of the system.

Figure 2 shows the interaction between objects without using the mediator pattern; Figure 3 shows the interaction between objects through the mediator in the mediator pattern.

By the mediator pattern, the interaction between objects becomes more concise.

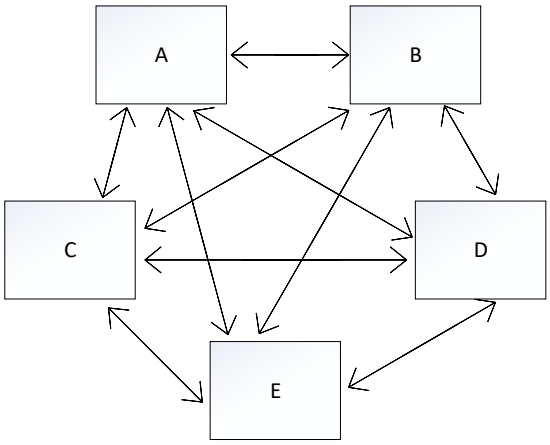

Fig.2 Unused Mediator Pattern

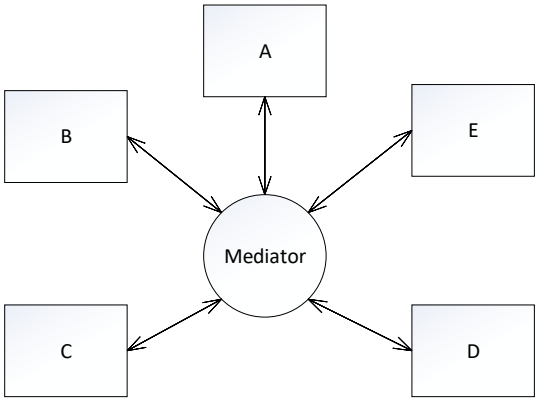

Fig.3 Using The Mediator Pattern

2.3 Two-phase Commit Protocol

When using multiple resources to solve a problem, it is necessary to make these resources cooperate in one transaction. To achieve this goal, transaction commit is divided into two phase.

The algorithm of two-phase commit protocol can be summarized as follows: the participants will inform the coordinator that success or failure of their operation, and then the coordinator will decide whether to commit or discontinue the operation according to the feedback information of all participants. Two phase refer to: the first stage, request and voting phase; the second phase, commit and implementation phase.

Request and voting phase. The transaction coordinator notifies each participant that they are ready to commit or cancel the transaction, and then proceeds to the voting process. The participant executes the transaction locally, writes the local redo and undo logs, but does not commit them, reaching a state of "everything is ready, only owes the East wind". During the request phase, participants will inform the coordinator of their own decisions: agree, which mean the transaction participant execute local job successfully, or cancel, which mean failing to execute local job.

Commit and implementation phase. At this phase, the coordinator will make decisions based on the vote of the first phase: commit or rollback. If and only if all participants agree to commit the transaction, the coordinator notifies all participants to commit the transaction, otherwise the coordinator will notify all participants to cancel the transaction. The participant will perform the corresponding operation after receiving the message from the coordinator.

When the two-phase commit protocol handles distributed transactions, the ID of branch transaction (XID) is generated. The XID must be unique, and any transactional execution begins and ends with XID.

The two- phase commit protocol is actually based on the mediator model. The coordinator is a mediator, which centrally schedules the interaction of all participants, and each participant do not need knowing the specific information of others. Figure 4 shows a model of two-phase commit protocol.

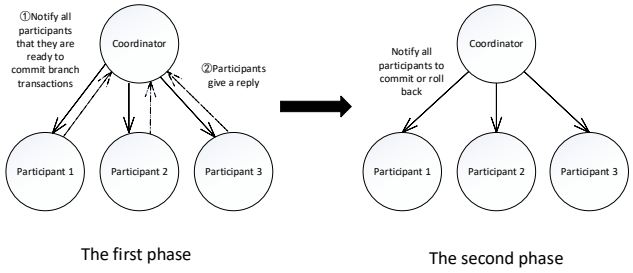

Fig.4 Model of Two-phase Commit Protocol

3. Distributed Transaction Processing of Two-phase Commit Protocol

3.1 Processing

This paper takes the database of news system as an example. The news system includes many subsystems, two of which are news adding system and news statistics system. When adding a news to the news table of news adding system, the value of total of news field in the news statistics table of news statistics system will increase by 1. Because news adding system and news statistics system are two different subsystems of the news system, their databases are distributed. These two databases are distributed on two servers, which is Server1 and Server2. Server1 deploys news adding system, its database is rm1, Server2 deploys news statistics system, and its database is rm2. In news system, transaction manager acts as coordinator. Figure 5 shows the process of adding a news.

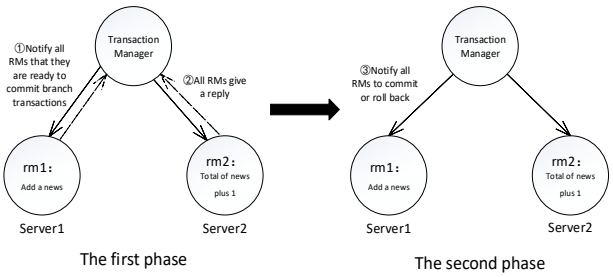

Fig 5. The Process of Adding A News

When a global transaction is started, the transaction manager asks whether these two RMs can commit their transaction. If both RMs reply to transaction manager that they can commit their local branch transaction, the transaction manager will inform the two RMs that they start commit their local branch transaction, and the global transaction executes successfully. If at least one RM reply cannot commit its local branch transaction, the transaction manager will inform these two RMs to roll back their local branch transaction and the global transaction failed to execute.

3.2 Deficiencies

Mentioning the process of distributed transaction execution, the above-mentioned processing method can correctly implement the distributed transaction processing without server failure, and maintain the consistency of data in database server.

However, if the server fails, such as power failure and hard disk damage, during the execution of distributed transactions, the execution of distributed transactions will be problematic.

Figures 6 and 7 show possible problems in the process of adding a news.

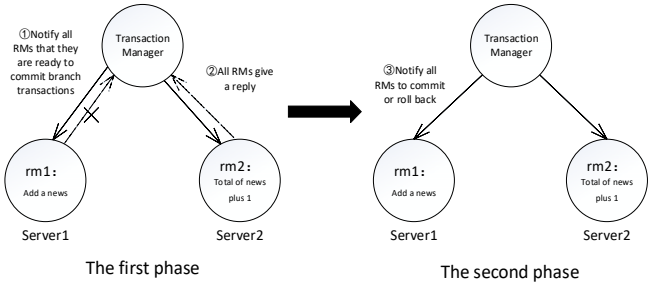

Fig.6 RM Break Down In The First Phase

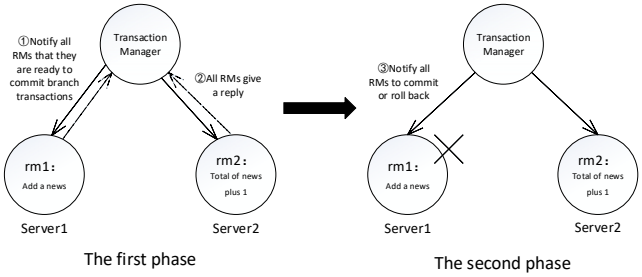

Fig.7 RM Break Down In The Second Phase

As shown in Figure 6, in the first phase, when the transaction manager notifies all RMs that they are ready to commit branch transactions, if RM1 has failed, RM1 will not reply whether it can commit local branch transactions to the transaction manager, and the transaction manager will wait for the response of rm1 all the time, and the system will be blocked.

As shown in Figure 7, in the second phase, if RM1 fails after every branch transaction vote, the transaction manager will still notify RM1 and RM2 to officially commit their local branch

transactions, since both RM1 and RM2 have previously informed the transaction manager that they can commit local branch transactions. Then RM1 cannot commit due to the failure, while RM2 commit successfully. The result is that the news adding system did not add a news, but the news statistics system increased the total of news by 1.

The above situation can be solved by determine which branch transaction failed to commit, and then to re-execute, or by determine which RM has failed and roll back all transactions by means of checking the log file saved on the transaction manager side. But doing so will make the whole process very complicated and these is not the best solution. Therefore, the two-phase commit protocol based on heartbeat mechanism is introduced to handle distributed transactions.

4. Two-phase Commit Protocol with Heartbeat

4.1 Heartbeat Mechanism

In software design architecture, heartbeat detection is very important. For example, in Web API invoke, the consumer side needs to know whether the provider side is alive or not, and if not, switch to call another provider.

If two systems are connected for a long time, there may be no data exchange for a long time. In theory, the connection is always maintained, but in practice, it is difficult to know what fault will occur in the intermediate node. If some nodes open the firewall, they will automatically disconnect the connection without data interaction within a certain period of time. At this point, heartbeat detection is needed to maintain long connections. To judge whether certain node (equipment, process or other network elements) is moving normally, a simple communication package is sent at regular intervals. If one node does not receive a response of a node other within a specified period of time, the other node is considered broken.

In distributed systems, nodes distributed on different hosts need to detect the status of other nodes, such as server nodes need to detect whether slave nodes fail. In order to detect the validity of other nodes, a fixed message is sent to others in a fixed time, and other nodes replies to a fixed message. If the response of one node is not received for a long time, the connection with these node is disconnected.

The sender can be either a server or a client, which depending on the specific implementation. Because it is sent in a fixed time, and similar to heartbeat, so the fixed message is called heartbeat packet. Heartbeat packet are usually smaller, which can be structured according to specific circumstances. Heartbeat packet are mainly used to maintain long connections and short links.

Generally, the client should actively send heartbeat packets to the server, because the server sending heartbeat packets to the client will affect the performance of the server.

Figure 8 illustrates the process of the heartbeat mechanism.

Server maintains a Client status table, which records whether all Clients are alive or not, so that Server can know the survival status of each Client by querying the Client status table. The status of Client is judged by the heartbeat mechanism.

Client uses a timer to send heartbeat packets continuously. Server receives heartbeat packets and replies to a reply packet. Server starts a timeout timer for each Client and considers Client invalid if it does not receive the heartbeat packets within a specified time.

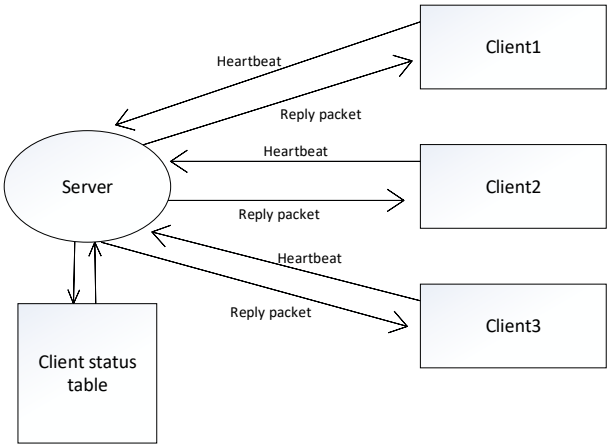

Fig.8 Heartbeat Mechanism

4.2 Coordinator based on Heartbeat Mechanism

The two-phase commit protocol has two roles: coordinator and participant. On this basis, the heartbeat mechanism is introduced. Through the heartbeat mechanism, the coordinator can detect the failure of each participant and maintain a "participant status table". Before the coordinator notifies all participants that they are ready to commit their branch transaction, the coordinator queries the participant status table to find out whether the participants are malfunctioning. If certain participant fails, the coordinator will not send any notification. After every participant reply that they can commit a branch transaction, the coordinator will query the participant status table again. If a participant failure is found, the coordinator will notify all participants that roll back their transaction. In this way, the reliability of the whole distributed database system can be improved and the system blocking can be reduced.

4.3 Distributed Transaction Processing with Heartbeat Mechanism

The example described in Section III.A is implemented by the two-phase commit protocol with heartbeat mechanism. Add a news and the total of news plus 1. If any link has problems, the whole process will be rolled back. If both processes are successful, then the news is added and the total of news is increased by 1. As a coordinator, the transaction manager maintains an RM state table by the heartbeat mechanism.

Figure 9 illustrates the process of adding a news with the heartbeat mechanism.

In the first phase, the transaction manager needs to query the RM status table firstly. If one RM fails, the transaction manager does not send any notifications to all RM.

After the transaction manager queries the RM status table, if all RMs is normal, the transaction manager will asks all RMs if they can commit local branch transactions. After all RMs vote, the process enter the second phase.

In the second phase, if an RM reply to that it fails to commit its local branch transaction, the transaction manager will notify all RMs to roll back their transaction. If all RMs reply to that they can commit their local branch transaction, the transaction manager will query the RM status table again. If at least an RM fails, the transaction manager will notify all RMs rollback transaction which has no failure. If all RMs is normal after the RM status table is queried again, the transaction manager will notify all RMs to commit their transaction.

In this process, the problems caused by the failure of RMs in the first phase and the second phase are effectively handled.

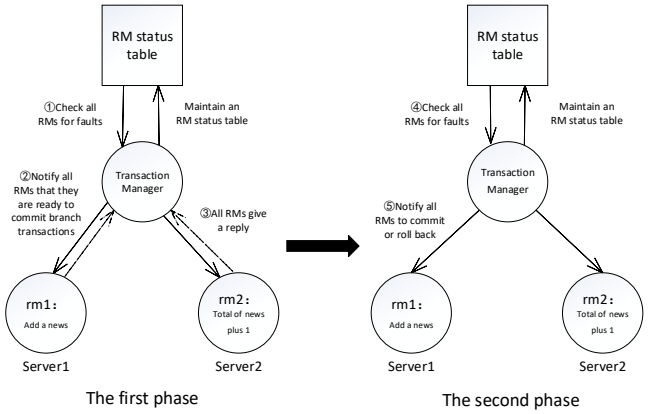

Fig.9 Add An News With Heartbeat Mechanism

4.4 Advantages of Introducing Heartbeat

In practical applications, once problems arise in distributed transaction processing, their losses will be incalculable and irreversible. Although the general two-phase commit protocol processing method can achieve distributed transaction processing, the application must consider the whole transaction processing process. It need to know the specific situation of each branch transaction at each time, so that it can judge whether they are failures, and the application needs to deal with the problems raised in section III.B. If a branch transaction fails, the application must know in time to deal with such failures.

The advantage of distributed transaction processing based on two-phase commit protocol with heartbeat mechanism is that it can know the running status of each database server in time. If there are abnormal situations, it can roll back the global transaction in time, avoid system blocking and improve system reliability. If the general two-phase commit protocol method is used to deal with global transactions with multiple branch transactions, the rollback situation will become extremely complex when an exception occurs.

5. Summary

Database system has been widely used in various industries, among which distributed database is an important application, and distributed transaction processing is a key point of distributed database. This paper introduces the model of distributed transaction processing and the process of two-phase commit protocol in distributed transactions. We taking the distributed database of news system as an example, it analyses the shortcomings of general two-phase commit protocol in distributed transactions processing. In view of the possible problems in the distributed transaction processing of general distributed database, we introduces heartbeat mechanism, which maintain a participant status table. According to the participant status table, the status of participant can be monitored in real time. The failure of participant in the first phase and the second phase of the two-phase commit protocol can be effectively solved. Finally, the distributed transaction processing of the two-phase commit protocol after introducing the heartbeat mechanism and its advantages is expounded. The method presented in this paper can effectively avoid the blocking problem that may occur in distributed transaction processing, and improve the reliability of distributed database system.

Acknowledgements

We wish to thank the timely help given by our classmates in analyzing the process of the two-phase commit protocol.

References

[1]. liancheng. Discussion on distributed transaction implementation technology and application scenarios [J]. Information technology and informatization,2018(10):187-189.

[2]. Zhi - yong Liu. Distributed Database Integrated Transaction Processing Technology Research [A]. Journal of Web Systems and Applications (2017 Vol. 1 Num. 1) [C]. : information Technology and engineering Technology association international (Inernational Informatization And Engineering Associations), 2017-5.

[3]. Luo da, zhang yuanzhou, zhou xiaocong. Aspect-oriented implementation of intermediary model [J]. Computer applications and software,2008(11):46-47+111.

[4]. Hu yu. A database real-time synchronization method based on intermediary model [D]. Jilin university,2008.

[5]. Huang meizhi, Chen junhua. An improved 1-2pc protocol based on multi-site backup [J]. Computer and digital engineering,2011,39(10):125-129.

[6]. Zhang genrong. Improvement of two-phase commit protocol for distributed database [J]. Heilongjiang science and technology information,2008(30):79+78.

[7]. Li yongsheng, cui jiadong, qin huibin. An adaptive heartbeat interval preserving TCP connection method [J]. Computer applications and software, 2008,35(01):149-153.