M-002

Investigating Workloads of Broker Account in BrokerChain-based Sharding IoT-Blockchain

蘇悦† 向陽† グエン キエン†,‡ 関屋大雄†
Yue Su Yang Xiang Kien Nguyen Hiroo Sekiya

1. Introduction

The scalability remains a major challenge for IoT-Blockchain integration. Sharding has been proposed as a scalable solution, partitioning the blockchain network into multiple shards that process transactions in parallel, thereby reducing computational and storage overhead [1]. As the number of shards increases, the number of cross-shard transactions (CTX) (i.e., transactions where the sender and receiver reside in different shards) also grows. CTXs increase processing complexity, communication overhead, and synchronization delays, becoming a performance bottleneck in sharding blockchains. Thus, minimizing CTXs is critical to improving overall system efficiency. Currently, notable CTX mechanisms include Relay and Broker [2]. Unlike Relay, Broker converts CTX into intra-shard ones within shards, reducing their volume and improving efficiency.

In the process of reducing the CTX ratio by introducing Broker, understanding the behavior of Broker accounts is crucial for evaluating the scalability, efficiency, and reliability of CTX mechanisms in sharding blockchain systems. However, there is still a lack of related research (e.g., load characteristic) of Broker accounts. This gap limits our comprehensive understanding of the applicability and scalability of the Broker mechanism. Therefore, this study focuses on investigating the behavior of different Broker accounts and their workloads.

In more detail, we emulated a sharding IoT-Blockchain system based on the BlockEmulator [3] platform and introduced a dataset of historical Ethereum payment transactions for emulating classic paying load scenarios. We focused on emulating the behavior of IoT full nodes in BlockEmulator. We then analyzed the load of Broker accounts in processing CTXs and found that there is a significant load imbalance problem (i.e., hot broker). This finding is of great practical significance for the construction of a more efficient and scalable CTX mechanism.

2. Evaluation Methodologies

2.1 Sharding IoT-Blockchain System

In this study, we utilized the BlockEmulator platform which is developed on Goland software to emulate the sharding blockchain system. BlockEmulator contains a broker file (as shown in Fig. 1) that stores 100 predefined broker account addresses. Depending on the number of brokers configured, the system will pull the appropriate number of broker accounts from this file to execute. The IP addresses for blockchain nodes are

Fig. 1 The list of Broker accounts

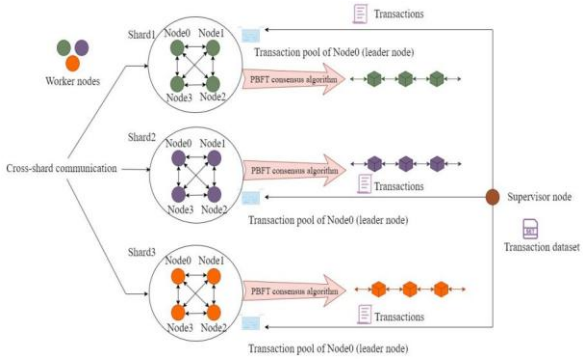

Fig. 2 A framework of Sharding IoT-Blockchain system

configured in the ipTable.json file. All nodes share the same IP address 127.0.0.1 while using different port numbers to ensure node uniqueness. The communication between nodes is realized using TCP connection. The transaction data used in this study comes from Ethereum On-chain Data provided by xblock [4], collected by running a full Ethereum node (blocks 0–17,499,999) and processed into multiple sub-datasets. We selected the BlockTransaction dataset, which splits transactions into 20 CSV files, and used two of them for injection. The data includes detailed block-level information on transfer transactions, such as sender, receiver, and amount.

An example framework, as illustrated in Fig. 2, demonstrates a sharding IoT-Blockchain system. The graph illustrates a blockchain network containing three shards, each consisting of four worker nodes. Within each shard, the worker nodes employ the PBFT consensus algorithm to collaboratively process transactions and generate blocks. Node0 in each shard serves as the leader node, responsible for coordinating consensus and transaction processing within the shard. The remaining nodes act as follower nodes, assisting in transaction verification. The datasets are injected by a supervisor node and deposited into the

† Graduate School of Science and Engineering, Chiba University, Yayoi-cho, 1-33, Inage-ku, Chiba-shi, Chiba, 263-8522 Japan
‡ Institute for Advanced Academic Research Chiba University, Yayoi-cho, 1-33, Inage-ku, Chiba-shi, Chiba, 263-8522 Japan

TABLE 1: Parameters configuration

Block_Interval	5 seconds
MaxBlockSize_global	2000 transactions
InjectSpeed	2000 transactions/sec
TotalDataSize	160,000 transactions
ShardNum	8
NodesInShard	4
BrokerNum	16
Committee_Method	2 (Value range:[0,4), representing: CLPA Broker, CLPA, Broker, Relay)

transaction pools of the leader nodes in each shard, which then process the transactions accordingly. Additionally, communication between shards is achieved through the CTX mechanism. We used two different datasets (i.e, 2000000 to 2999999_BlockTransction.csv and 5000000 to 5999999_BlockTransaction.csv). Other system configurations are detailed in TABLE 1.

2.2 Investigating Workloads of Broker Accounts

To investigate the operational characteristics and load distribution of the Broker mechanism, we designed a multi-step analysis. First, we extracted transaction data from each shard's database, structured it into readable CSV files, and parsed key fields including Sender, Recipient, Nonce, Value, and three Broker-related Boolean flags: WasBroker, SenderIsBroker, and RecipientIsBroker. We then identified each transaction's type and the role of selected Broker accounts by analyzing these Boolean combinations: false, false, false: intra-shard transaction; false, true, false: first phase of CTX (Broker1 TX, Broker as receiver); false, true, true: second phase of CTX (Broker2 TX, Broker as sender). Finally, we computed the number of CTXs each Broker handled in each shard, separately counting their roles as Broker1 and Broker2 to quantify load distribution.

3. Experimental Results

We only show results when using dataset 2000000 to 2999999_BlockTransction.csv due to insufficient space. Figure 3 shows a heat map of the number of CTXs processed by 16 Broker accounts in 8 Shards. Each cell in the graph represents the number of CTXs processed by a Broker in the corresponding shard. Its color is based on the color bar on the right side: the closer the color is to red, the more transactions are processed and the heavier the load, and vice versa, the lighter the load. The Broker load is clearly uneven: Broker d34d..... a375 has very high processing in every shard, especially in Shard#7 reaching 3843 CTXs, the darkest red area in the whole graph. In comparison, such as 1e99.... .0341, 9e0b.... .7606 etc. Broker hardly participates in CTX processing (all 0 or very low values). Most of the shards have CTX totals in the same range (15000-20000). However, the load on the Brokers within the shards is not even. The experiment clearly reveals that there is a serious load imbalance issue for the Broker accounts. A few Brokers are responsible for most of the CTX tasks for a long time, while most Broker nodes are idle. To improve system throughput and resource utilization, a dynamic Broker scheduling mechanism should be introduced in the future.

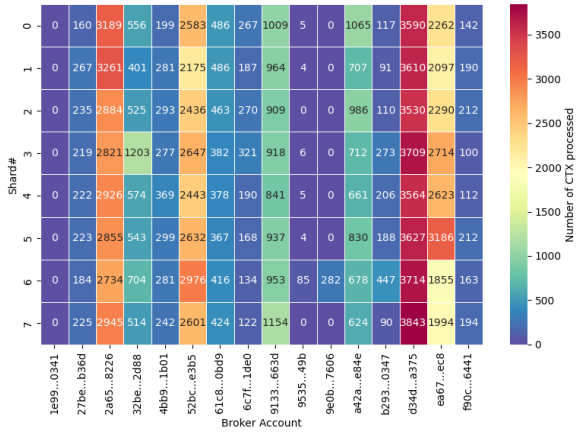

Fig. 3 Number of CTXs processed by each Broker account in all shards

4. Conclusion

Understanding the behavior and workload distribution of Broker accounts is essential for evaluating the scalability and efficiency of Broker mechanism. Despite its importance, this aspect remains under explored in existing research. In this study, we investigated the behavior of different Broker accounts using the BlockEmulator platform, where we emulated a sharding IoT-Blockchain system and injected historical Ethereum payment data to emulate realistic transaction loads. We created different scripts to read transactions information in databases and calculated the amount of handling CTXs for various Broker accounts in each shard. Our analysis revealed a significant load imbalance among Broker accounts, with certain accounts consistently handling a majority of CTXs. This "hot broker" phenomenon highlights a critical limitation in the current Broker mechanism and provides valuable insight into designing more balanced and scalable cross-shard communication strategies.

Acknowledgement

This work was supported in part by the Japan Society for the Promotion of Science (JSPS) under Grant 23H03377; and in part by the Japan Science and Technology Agency (JST), the establishment of university fellowships towards the creation of science technology innovation, Grant Number JPMJFS2107.

References

[1] Li, Mengya, and Yang Qin. "Scaling the blockchain-based access control framework for IoT via sharding." ICC 2021-IEEE International Conference on Communications. IEEE, 2021.
[2] Huang, Huawei, et al. "Brokerchain: A cross-shard blockchain protocol for account/balance-based state sharding." IEEE INFOCOM 2022-IEEE Conference on Computer Communications. IEEE, 2022.
[3] Huang, Huawei, et al. "Blockemulator: An emulator enabling to test blockchain sharding protocols." IEEE Transactions on Services Computing (2025).
[4] Zheng, Peilin, et al. "Xblock-eth: Extracting and exploring blockchain data from ethereum." IEEE Open Journal of the Computer Society 1 (2020): 95-106.