[Extended Abstract] Cross-Blockchain Transactions

Dongfang Zhao
dzhao@unr.edu
University of Nevada, Reno

1 MOTIVATION

Blockchains [1] offer an immutable, decentralized, and encrypted mechanism for transaction processing. However, realized by its first widely-used application Bitcoin [2], blockchain was not originally designed for OLTP workloads; instead, it aimed to offer an autonomous and highly-secure data management service among untrusted parties, partly by sacrificing the transaction throughput performance. Recent research [3, 4] advocates to leverage blockchains for OLTP workloads by proposing various techniques (e.g., sharding, sidechains [5]) to boost up the transaction throughput of blockchains, such that blockchains would deliver similarly high performance as relational databases and become a competitive alternative to the latter as a general-purpose data management system.

There is yet another critical issue that must be addressed before blockchains can be widely adopted as a general data management system: the interoperability across blockchains. While SQL is available between different database vendors, no such standardization or interface exists for blockchains. Recent attempts (e.g., Cosmos [6]) on such cross-blockchain transactions are all *ad hoc*: making strong assumptions on the blockchains such as their consensus protocols and programming interface. In addition, existing cross-blockchain systems exhibit design limitations such as poor scalability and huge overhead.

To make it more specific, what follows lists four outstanding limitations exhibited by state-of-the-art cross-blockchain systems.

(1) Centralized Broker. The transactions between heterogeneous blockchains are managed by a third-party, usually implemented as another blockchain (it is called a *hub* in Cosmos). This is against the decentralization principle of blockchains: the broker becomes a performance bottleneck and single-point-of-failure.

(2) Two-Party Transactions. The protocols used by existing cross-blockchain systems are derived from the *sidechain protocol* [5], which was originally designed for transferring assets between Bitcoin [2] and another cryptocurrency. Sidechain protocol speaks of nothing about three- or multi-party transactions; in fact, Cosmos only supports transferring assets between Bitcoin and Ethereum [7].

(3) Performance. The sidechain protocol [5] takes 1-2 days to commit the cross-blockchain transaction. The main reason for this is due to the possible branches from the participating blockchains. In any participating blockchain, only one (i.e., the longest one) branch will remain valid and any transactions from the shorter branches will rollback. This is not a problem if all of the transaction parties are from the same blockchain; But for cross-blockchain transactions, they must wait for the (longest) branch to stand out.

(4) Interface. The centralized broker requires the users to pack their cross-blockchain transactions with the provided interface. It would create portability issues when the users concurrently work with multiple cross-blockchain platforms. What we need is a common interface with which different blockchains (and their users) can communicate. SQL is an excellent example for relational databases.

2 PROPOSED APPROACH

We started with designing a multi-party transaction protocol across heterogeneous blockchains without a centralized broker. The most straightforward means to design multi-party protocols seems to be extending the sidechain protocol; however, it turns to be a challenging problem when the number of participants increases from two to an arbitrary n: we need to reapply the sidechain protocol for up to $\binom{n}{2} = n \cdot (n-1)$ times each of which takes 1-2 days—not a practical solution. To this end, we propose two techniques for efficient multi-party transactions: (i) resolving conflicts by complementary intra-blockchain transactions; (ii) parallelization of subtransactions.

Complementary Transaction. We come up a passive principle for the multi-party transactions: instead of waiting for the branches, we simply mark concurrent branches valid and will trigger additional intra-blockchain transactions between branches to resolve the inconsistency right before short branches are invalidated.

Parallelization of Subtransactions. We parse the given multi-party transaction into a directed acyclic graph (DAG) where each vertex represents a series of operations over the same set of parties. We then leverage multiple cores to parallelize the independent paths from the DAG.

We theoretically prove the safety (correctness) and liveness (non-blocking) of the proposed multi-party protocol. We implement the proposed protocol in a blockchain emulator BlockLite [8]. We evaluate the effectiveness and efficiency of the proposed multi-party transaction protocol on three popular blockchains (Hyperledger [9], Ethereum [7], Parity [10]) and an in-memory blockchain [11].

We have not started working on the standard interface to specify cross-blockchain transactions. We note that there are efforts on SQL wrappers over blockchains [12]. At this point, it is unclear whether a new standardization is needed for the fast-growing blockchain user base. We leave this as an open question to the community.

REFERENCES

[1] Kaiwen Zhang and Hans-Arno Jacobsen. Towards dependable, scalable, and pervasive distributed ledgers with blockchains. In *38th IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2018.

[2] Bitcoin. https://bitcoin.org/bitcoin.pdf, Accessed 2018.

[3] Hung Dang, Tien Tuan Anh Dinh, Dumitrel Loghin, Ee-Chien Chang, Qian Lin, and Beng Chin Ooi. Towards scaling blockchain systems via sharding. In *International Conference on Management of Data (SIGMOD)*, pages 123–140, 2019.

[4] Jiaping Wang and Hao Wang. Monoxide: Scale out blockchains with asynchronous consensus zones. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 95–112, 2019.

[5] Sidechains. https://blockstream.com/sidechains.pdf, Accessed 2019.

[6] Cosmos Network. https://cosmos.network, Accessed 2019.

[7] Ethereum. https://www.ethereum.org/, Accessed 2018.

[8] Xinying Wang, Abdullah Al-Mamun, Feng Yan, and Dongfang Zhao. Toward accurate and efficient emulation of public blockchains in the cloud. In *12nd International Conference on Cloud Computing (CLOUD)*, 2019.

[9] Hyperledger. https://www.hyperledger.org/, Accessed 2018.

[10] Parity. https://ethcore.io/parity.html/, Accessed 2018.

[11] Abdullah Al-Mamun, Tonglin Li, Mohammad Sadoghi, and Dongfang Zhao. In-memory blockchain: Toward efficient and trustworthy data provenance for hpc systems. In *IEEE International Conference on Big Data (BigData)*, 2018.

[12] Johannes Gehrke et al. Veritas: Shared verifiable databases and tables in the cloud. In *Biennial Conference on Innovative Data Systems Research (CIDR)*, 2019.