Towards A Novel Architecture for Enabling Interoperability Amongst Multiple Blockchains

Hai Jin, Xiaohai Dai, and Jiang Xiao
Services Computing Technology and System Lab, Cluster and Grid Computing Lab
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China
Email: jiangxiao@hust.edu.cn

***Abstract*—Next-generation blockchain ecosystem will be fuelled by a large variety of blockchain systems. These systems increasingly demand proper cross-chain cooperation to provide richer functionalities and enhanced capabilities in the future landscape. How to enable such 'interoperability' – effective communication and efficient data transfer across multiple blockchain systems is thus critical and facing many unprecedented theoretical and practical challenges. In this paper, we first clarify the definition of interoperability based on the cross disciplinary nature. Second, a roadmap of challenges needed to be addressed for interoperability has been laid out. Third, we articulate novel architectural approaches to fill in the gap by enforcing the interoperability from different blockchain layers.**

***Keywords*-Blockchain; interaperobility; consensus protocol; smart contract.**

I. Introduction

Recently we have witnessed the popularity of blockchain, which is envisioned to be the key enabler of trust establishment amongst untrusted network nodes in the digital world. Blockchain is understood as the distributed ledger technology by intelligently fusing the techniques including P2P, cryptography, distributed consensus protocol, and smart contract [1], [2]. With the promise of immutability, transparency, and consensus inherent, it has gained significant research and industrial attentions [3], [4]. In particular, it holds immense potential of advancing trustworthiness of data without any central authority across a wide range of applications dimensions, such as financial industry, healthcare assurance, and governmental services.

To satisfy diverse demands from different applications, there are ever-growing blockchain systems being designed and implemented, each of which processes and stores data independently. For example, the banks can leverage blockchain to accelerate transaction settlement among different parties with reduced headcount and operational expense, the consumers can track the producers and transportation processes of commodities in a more resilient and economic way, and the asset owners can provide related proofs of ownership more convincingly.

Proliferation of various blockchain systems creates a necessity for different blockchains to cooperate together, so as to provide richer services and greater value in the future landscape. This emerges as a new paradigm of establishing the connections between isolated blockchain islands. This will be highly beneficial in various areas of the economy and society. In financial industry, one typical scenario is digital payment among different blockchain-based cryptocurrencies. This allows Alice who owns Bitcoin [5] to reach a deal with Bob whose money is stored in the Ethereum [6]. Cross-border settlement also serves as a crucial sector in the area of finance, where it should widely benefit global capital markets. As reported by the People's Bank of China [7], the total amount of cross-border RMB trade settlement deal is over 4,360 billion in 2017. It is usually inefficient and error-prone to perform cross-border settlements with traditionally extensive manual efforts. Recent studies have been made to accelerate the intra-bank transaction processing by deploying a single blockchain system. However, existing research is lack of facilitating the inter-bank transactions transferred among different countries.

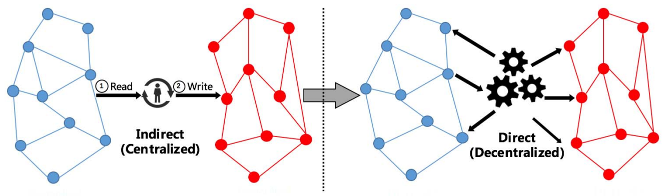


Figure 1: A sketch map of interoperability

Another typical scenario is to unleash the power of interaction when a supply chain meets a financial chain. As an example, the American retail giant Walmart is trying to innovate the financial system by designing a new scheme to facilitate the inter-chain communication. This can enable the financial activities safer and more efficient in the future. In particular, once the goods is received by Walmart's supply chain, a receipt message should be input into its financial system, so as to trigger an automatic remittance from Walmart's financial system to the producer. Apart from these, the realistic scenario can also be found in blockchain-based digital reputation system. It requires the mutual knowledge of an identity blockchain and a payment blockchain. This in return ties the two blockchains to link and merge the identity information and payment credentials for a unique user.

Consequently, our intention is to enable novel cross-chain interaction capabilities among multiple blockchains, each of which wishes to share and integrate the key information. Prior arts mainly rely on an intermediate organization (i.e., 'notary') to first read information from the source chain and then write it to the destination chain. Unfortunately, this indirect manner can suffer from a few security risks due to the fact that the notary may tamper the data consciously or unconsciously [8]. Furthermore, the efficiency of interaction can be degraded. In this context, it is required to discover a direct interaction paradigm, as the difference from the direct one is illustrated in Figure 1. In this way, data can derive from and flow into several different nodes without the participation of any designated notary, so as to maintain the security and performance of blockchain ecosystem. As a primitive, we cautiously introduce a rigid definition of cross-chain interaction as 'interoperability': an intelligent characteristic of effective communication and direct information exchange from one blockchain to another, while retains the essence of each individual blockchain, including irreversibility and traceability.

The vision of next-generation blockchain ecosystem is couched on wide scale interoperability behaving in two interoperation modes: passive interoperation and active interoperation. The passive mode is easy to implement while it incurs high costs. Note that the state-of-the-art researches all belong to this. On the contrary, the active mode has a requirement on the design of blockchain systems and is hard to implement, which limits its use scope, but it is more cost-effective. No matter which mode is adopted, there are some common challenges to be addressed in the design and implementation. To make the interaction reliable, the atomicity should be guaranteed without the sacrifice of performance. No or few modifications to the existing interaction protocol should be needed when a new blockchain joins in, so as to maximize the universality of the interaction protocol and minimize the redesign costs. To attract more developers to join in the development of blockchain interaction application, the interaction technology should be as friendly as possible to the high-level developers.

In this paper, we propose a novel architecture to guide the design and implementation of interoperability. The architecture demonstrates the benefits of unified transaction format in the data layer, the challenges faced by implementing the network layer, the new interpretation of consistency in the consensus layer, the possibility of realizing cross-chain smart contracts, and the difficulty in developing upper applications.

In summary, our contributions are the following:

- To the best of our knowledge, we are the first one to give a formal definition of blockchain interoperability.
- We divide the interoperation schemes into two categories: passive interoperation and active interoperation. We analyze the advantages and disadvantages of each category, respectively.
- We point out the possible challenges in realizing interoperability from five aspects: atomicity, efficiency, security, universality, and friendliness.
- We propose a novel blockchain architecture for enabling interoperability. Based on the architecture, we further talk about the key modules and potential design in each layer.
- To reduce the overhead introduced by passive cross-chain communication, we propose a solution inspired by I/O multiplexing, whose improvement effects are verified by experiments.

The rest of the paper is structured as follows. Section II introduces the two modes of interoperability among multiple blockchains. Section III discusses the challenges that we will face with the design. We propose a novel architectural approach to make it feasible in Section IV. Section V shows the Monitor Multiplexing Reading method can improve the performance of passive communication by evaluation results. Finally, we discuss related work in Section VI, and conclude the work in Section VII.

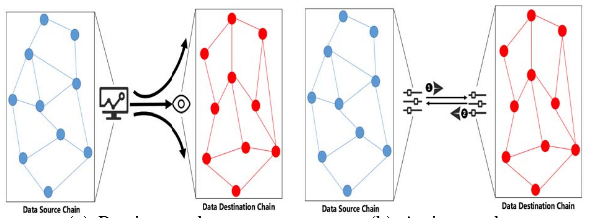
II. Two Modes of Interoperability

In this section, we elaborate on the two modes of interoperability, which motivates the design of our layered architectural proposal.

For ease of expression, we refer to every two dissimilar blockchains involved in interoperability as source chain and destination chain respectively, as shown in Figure 2. Suppose that these pair-wise blockchains try to interact with each other. Then, the information will flow from source chain to destination chain. During this interaction process, source chain will elect nodes of interest as source data, also called source nodes. In turn, destination nodes will be chosen depending on the correlation with source nodes on the pairing chain. Recall that interoperability must reserve the merits of irreversibility and traceability in each participated blockchain. As a consequence, it first requires that data cannot be altered once arrived at destination chain. One possible way is to apply signature mechanism for reaching such irreversibility. Traceability can also be achieved under two conditions: (a) the source data is considered as reliable only if it has been committed on source chain, and (b) the interoperability process is identified as successful only when the data is committed on the destination chain.

According to the role of an initiator playing in cross-chain interaction, we classify the interoperability into active mode and passive mode. In the active mode, the interoperation initiator acts as the sender of information. On the contrary, the initiator performs as information receiver in the passive mode.

In terms of the passive mode, destination chain will keep on monitoring the transactions or events on source chain, which is shown in Figure 2(a). Once the specific transaction is issued or an given event takes place on source chain, the destination chain will take a related action. Throughout



(a) Passive mode (b) Active mode

Figure 2: A comparison of two modes of interoperability

the whole interaction process, source chain has no need to feel the existence of destination chain. The passive mode is especially suitable for the situation where the source chain has been deployed for a long term, which is impossible or hard to make changes to its system architecture. A good example of this mode is BTCRelay [9]. As it is hard to modify the existing implementation of Bitcoin, which may result in a fork, BTCRelay just reads from Bitcoin chain. However, since the source chain will not inform the destination chain of its information actively, the flow of information is triggered by the polling-based read from destination chain, which either is not timely enough or brings high costs to destination chain.

Figure 2(b) depicts that source chain in active mode will firstly send information to destination chain positively, and then waits for the feedback from destination chain. Each blockchain has to be aware of each other for communication. The difficulty in implementing the active mode of interoperability is higher than that of passive mode, which further needs the functional supports of two blockchains. As an example of this mode, Interledger adopts the hash-locking trust model to implement cross-chain digital asset exchange. In the case of hash-locking asset exchange, source chain will send hash of a random secret S_{Src} to destination chain firstly, and then waits for a secret S_{Dest} from destination chain. Once source chain receives the S_{Dest} from destination, it will send the correct secret S_{Src} to destination. It is required that both blockchains have implemented the support to hash-locking technology in their system designs, which limits the usage scenario of active mode. Compared to the passive mode, active mode can leave out the need of polling-based read, which is more cost-effective.

III. Challenges in Realizing Interoperability

To achieve the goal of realizing interoperability is not straightforward, there exists challenges scaling along many dimensions: guarantee of atomicity, the improvement of efficiency, maintenance of security, the tolerance for diversification, as well as friendliness to application developers. To make things worse, there is no universal framework can solve these issues once and for all.

A. Guarantee of Atomicity

How to guarantee the atomicity of actions performed for interoperability is a challenging task. This means that all the actions involved in each chain have to be completely successful or failed at the mean time. Take assets transfer between two financial chains belonging to different companies as an example. If Alice wants to transfer assets on the financial chain A to Bob's account on the financial chain B, both the decrease in Alice's assets on chain A and the increase in Bob's assets on chain B should either succeed or fail.

One of the biggest threats to the atomicity lies in the fundamental lack of determinism in some blockchain systems. This can be even worse in the permissionless (or public) blockchains without any access control mechanism [10]. For instance, a Bitcoin block will be added to the chain only after it is confirmed by more than 6 successors. In reality, however, the blocks in Bitcoin network can never become deterministic based on the PoW consensus algorithm, and even be modified in the future when the assumption of most nodes are honest is invalid. As a result, current successful action on Bitcoin may fail later. It is non-trivial for another chain to acquire the updated results in the first place, thus sacrifices the atomicity of interoperability.

B. Improvement of Efficiency

The next challenge comes from the efficiency of multi-chain interaction. It is well-known that efficiency is one of most important bottlenecks to limit the wider adoption of blockchain technology [2]. While in terms of interoperability, the efficiency issue can suffer at higher scales and therefore deserves much attention since multiple parties will be affected. The efficiency of blockchain interaction can be accessed by the interaction times per second.

Existing implementations of active and passive modes are of questionable performance, which need to be further improved. As aforementioned, in the active mode, source chain has to wait for the commitment of data being received at destination chain, which depends on the commitment speed of destination chain. The passive mode based on polling-driven read from destination chain is not efficient enough.

C. Maintenance of Security

As we have mentioned in Section I, the realization of interoperability should not subvert the security of blockchain ecosystem. For the scope of an individual blockchain, security targets at protecting the overall system from any kinds of attacks [11]. In terms of multi-chain ecosystem, we focus more attention on the possible security risks introduced by the cross-chain operations, which leaves out the original security problems in an individual blockchain.

A cross-chain data transmission can be divided into 3 phases: data leaving the source chain, data be in transit, and data reaching the destination chain. As a result, different measures must be taken for each phase. To be specific, the maintenance of security across multiple blockchains should

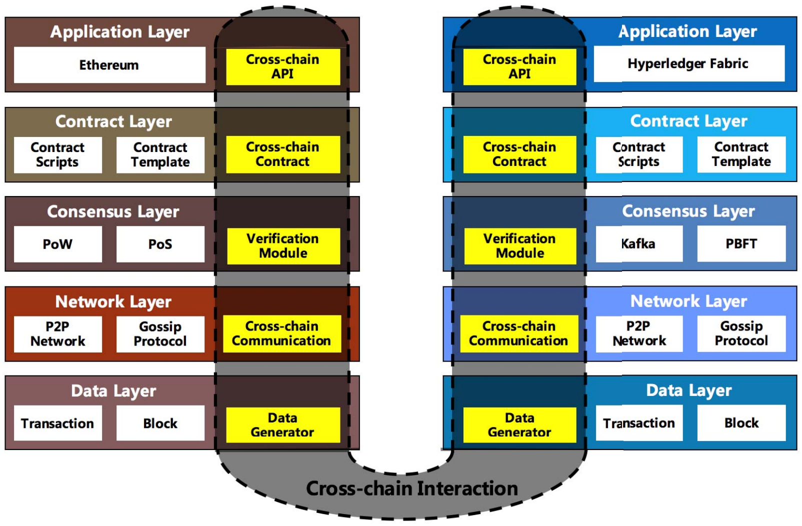


Figure 3: Overall architecture

satisfy: (a) the information to be transmitted is reliable, which should be committed on source chain and come from a random set of nodes; (b) the data in transmit cannot be tampered, which can rely on the signature check, and (c) there should be a final commitment on the destination chain, after the data reaches the destination chain.

D. *Tolerance for Diversification*

During the past decade, blockchain ecosystem exhibits significant diversification in both consensus protocol design and usage scenarios. On one hand, more and more companies and institutions have developed or are developing the blockchain systems owing to their diverse purposes, leading to the enormous discrepancies in consensus protocols. The Bitcoin and Ethereum running on differential consensus protocols (i.e., PoW and PoS) can be a most representative example. On the other hand, a lot of open-source blockchain systems are released recently. Leveraging the open source code, a company or institution can build up the blockchain for an application- specific domain easily. Further, it can be customized based on the requirements with some programming efforts. As a result, the blockchain systems show a great diversity in their usage scenarios. Imagine that the Ethereum can be used to implement both a supply chain and a voting system.

A reliable cross-chain consensus protocol should be available to support the diversification of blockchains. In other words, no or few modifications to the existing protocol of each system will be needed when a new blockchain joins in the ecosystem.

E. *Friendliness to Application Developers*

As an underlying technology, blockchain ecosystem requires substantial efforts with the participation of massive application developers. At the high level, the interoperability should provide friendliness in order to attract more developers to join the ecosystem. The shortcomings of application-transparent by design can lead to significant loss of universality.

A feasible scheme is to provide a set of API to transmit some data from one blockchain to another one with ease. For example, an API can be invoked to transfer assets from Bitcoin to Ethereum directly, without the application developers to consider the complicated and error-prone underlying implementation details. The invoker only needs to provide the source address of Bitcoin, the destination address of Ethereum and the number of transferred assets in BTCs.

A great difference between API of cross-chain interaction and general API is the way to return the execution result. The general API can return a result to the invoker quickly. However, as the cross-chain interaction depends on the block commitment on the blockchains, whose completion time may be long and cannot be predicted, the API will not return a result to the invoker directly. Instead, the invoker needs make attempts to query the execution result from time to time, until a result is got back.

IV. A Novel Blockchain Architecture for Interoperability

In this section, we propose a cross-chain framework by innovating on all five layers as shown in Figure 3. We then detail the design of the modules in each layer and how they adhere to address the above challenges.

A. *Data Generator in Data Layer*

Data management is an essential task in blockchain systems [12]. It gives birth to the data layer that contains the transaction format, block structure, storage model and so on. Different blockchain systems may inevitably differ in terms of the data layer.

The diversity of transaction formats is one of the dominated bottlenecks hindering the information exchange among different chains. For example, Ethereum and Bitcoin are different in the transaction format, which results in an impediment to the direct transmission of a transaction between Ethereum and Bitcoin.

A simple but feasible idea is to unify the transaction format. Hereby, data generator module is presented to support this universality. With data generator, we aim to build a middleware facilitating the direct interaction among blockchains without relying on the relays. The newly created blockchain system can easily adopt the unified transaction format and connect to the middleware. Nevertheless, changing the transaction format of the existing blockchain system would add complexity. Alternatively, a transaction translator can assist in translating the transaction from specific format to a general one.

B. Cross-chain Communication in Network Layer

Initially, the role of network layer is designed to support the communication among different nodes in a blockchain system. To realize the decentralization of communication, blockchain systems usually adopt the P2P network model. Moreover, gossip protocol is widely used to implement the distributed fault-tolerance and enhance the communication efficiency. Aggregating with both P2P network and gossip protocol, however, the current design of network layer only takes the intra-blockchain communication into consideration. We argue that a natural extension to inter-blockchain communication is needed. The extension in the network layer will be implemented in an augmented module, termed as cross-chain communication.

The implementation of cross-chain communication module will inevitably encounter several problems. First, how to discover the IP addresses of nodes in the counterpart blockchain when two blockchain systems communicate with each others. More specifically, there is no centralized server or authority could be trusted to obtain the list of IP address. Second, how to achieve good performance of cross-chain communication. The state-of-the-art literatures of both active and passive interoperation have limited efficiency, which need to be further improved. As we have discussed before, the passive mode based on polling-based read from destination chain is not efficient enough. In the active mode, source chain has to wait for the commitment of data on destination chain, which highly relies on the commitment speed of destination chain.

C. Verification Module in Consensus Layer

The consensus layer is of great importance to ensure the consistency of states for a blockchain system. Currently, there are mainly two categories of consensus algorithms: Nakamoto consensus and classical consensus. The former one comprises of PoW, PoS, DPoS and so on, assuming a totally asynchronized network without a guarantee of liveness. Making use of the Intel SGX [13], *Proof of Elasted Time* (PoET) algorithm is regarded to be more secure and efficient, which is adopted in Hyperledger Sawtooth project [14]. On the other side, classical consensus algorithms such as PBFT [15], either because their higher communication costs make it hard for the number of nodes to scale out, or because they cannot tolerate the Byzantine failures (e.g., Paxos [16] and Raft [17]). However, as an exception for that, Algorand [18] makes use of a novel mechanism based on *Verifiable Random Functions* (VRFs) [19] to tolerate Byzantine failures with a low communication cost, which can be adopted in the permissionless blockchains.

Since the states in different blockchain systems are totally different, it is meaningless to talk about the cross-chain consensus protocol here. However, from another perspective, as the functionality of consensus protocol is to maintain the correctness of a blockchain system's state, we reinterpret the consensus protocol in the context of cross-chain interaction as verification protocol. The verification protocol will verify that if the data has been committed in the source chain, if the transmitted data is tampered.

D. Cross-chain Contract in Contract Layer

Driven by the popularity of blockchain technology, smart contracts regain popularity after its first proposal by Nick Szabo in 1997 [20]. In general, smart contract is a computer protocol, which will self-execute and self-verify once it is established and deployed. It has the advantages of real-time update, accurate execution and little human intervention. Not only Ethereum and Hyperledger Fabric [21], but almost all the systems in Blockchain 2.0 are now starting out to embrace a smart contract fashion.

To extend the contract layer to cross-chain scenarios, a smart contract should be triggered by two conditions from two chains separately. One of the biggest challenges in implementing a cross-chain smart contract is how to unify the contract runtime environments and programming language, since various blockchain systems may adopt different runtimes and languages.

E. Cross-chain API in Application Layer

Taking advantage of the blockchain's decentralization, immutability, and traceability, many ingenious applications are developed and deployed. As we have observed, the most popular application is cryptocurrency.

To unlock the possibility of cross-chain applications, many challenges need to be further addressed. A simple challenge could be described as how to provide a friendly development interface so as to attract more developers to take part in.

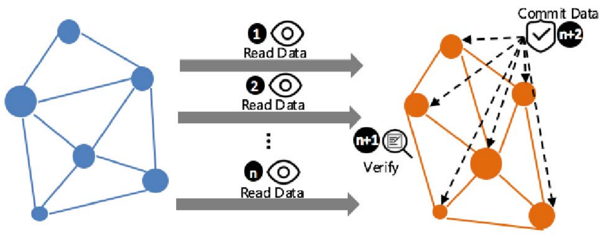


Figure 4: Schematic of PBR

Algorithm 1 Proxy Procedure

```
Input: $blockNum$, $ipBtcNode$, $initialHeight$, $period$, $targetTx$,
    $preDesignedTx$, $ipEthNode$
Output: $statusCode$
 1: $i = 0$
 2: $p = 0$
 3: while $i < blockNum$ do
 4:     $h$ =GetBlockCount($ipBtcNode$)
 5:     if $h < initialHeight$ then
 6:         continue
 7:     else if $h > p$ then
 8:         for $p < j < h + 1$ do
 9:             $hash$ = GetBlockHash($j$)
10:             $block$ = GetBlock($hash$)
11:             if $targetTx \in block$ then
12:                 IssueTxToEthereum($preDesignedTx$, $ipEthNode$)
13:                 return 0
14:             end if
15:         end for
16:         $p = h$
17:     end if
18:     Sleep($period$)
19: end while
20: return −1
```

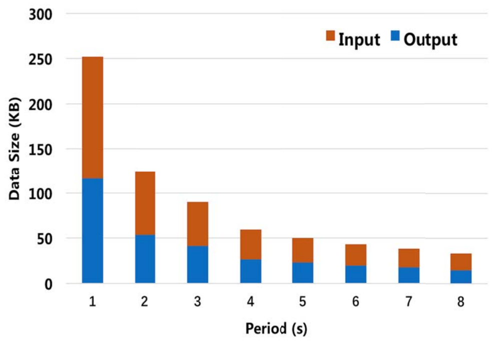
V. MMR Method for Improving Passive Cross-chain Communication

As previously mentioned in Section II, the traditional approach relying on polling-based reading is neither timely enough nor introduces substantial I/O overhead. In this section, we will firstly evaluate the I/O overhead of polling-based reading and then exploit the potential to reduce the overhead and ensure timeliness meanwhile.

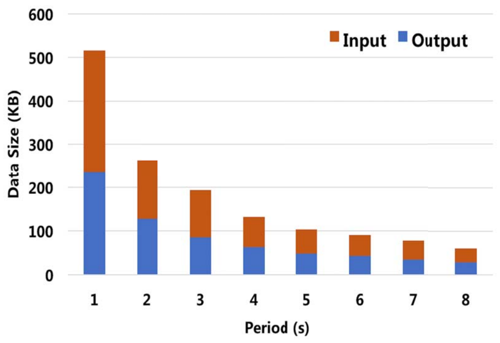
A. Evaluation of Traditional Method

For convenience, we will refer to the traditional method as PBR (*Polling-Based Reading*) in the rest of paper. PBR method requires the destination chain to continually make attempts to read information from source chain again and again, which is shown in Figure 4. Once the read operation is successful, the destination chain will verify the correctness of information. Once the verification is passed, a transaction containing the information will be proposed to the chain, which can be committed later.

To evaluate the I/O overhead, we simulate the working procedure of PBR method. The simulation system is implemented on our virtual machines. Each virtual machine consists of 1 core of Intel Core i5-4590HQ 2.2Ghz CPU with 2GB of DRAM and 60GB SSD, running on Ubuntu 14.04. We choose Bitcoin in regression test mode as source chain and Ethereum in testnet mode as destination chain, separately.



(a) Average I/O data size



(b) Maximum I/O data size

Figure 5: I/O overhead of PBR

On the Bitcoin side, we keep issuing transactions which transfers BTC from one address to another. The transactions are the same except that the numbers of BTC are different, which are generated randomly. On the Ethereum side, as Ethereum cannot read information from the outside directly, we carry out a simple proxy program to represent the Ethereum to read Bitcoin chain and propose transactions to Ethereum chain when needed, which can be regarded as a simple implementation of blockchain oracle [22]. Ethereum can monitor the Bitcoin chain through the proxy program, whose pseudocode is shown in Algorithm 1.

The proxy is only interested in the blocks in Bitcoin whose block number is not less than $initialHeight$ and at most read $blockNum$ blocks. Since the proxy and Bitcoin core may run in different nodes, the IP address of node running Bitcoin core ($ipBtcNode$) must be provided. The same goes for IP address of Ethereum node ($ipEthNode$). The proxy will read the current block count of Bitcoin every $period$ seconds, by calling the function $GetBlockCount$. Once newly committed blocks are found, the proxy will read the blocks down with the functions $GetBlockHash$ and $GetBlock$. It will further verify if the target transaction (i.e.,

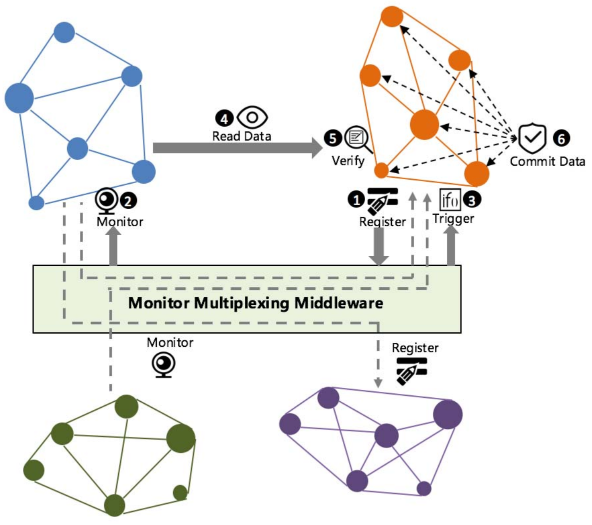
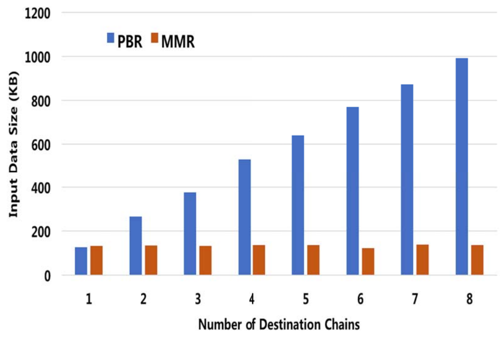
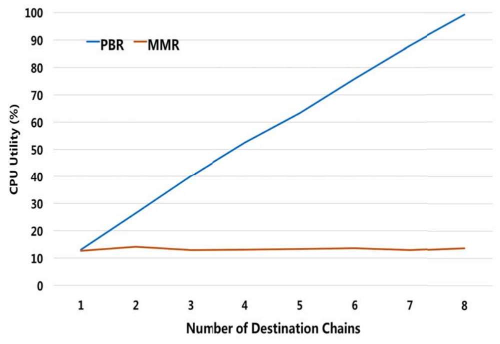


Figure 6: Schematic of MMR



(a) Impact of input data size



(b) Impact of CPU utility

Figure 7: PBR vs. MMR

Ethereum is interested in) is contained in the new blocks. If true, a pre-designed transaction will be issued to Ethereum; otherwise, we repeat the steps above.

We conducted 8 groups of experiments whose polling-based read period varies from 1 second to 8 seconds. In each group of experiments, we repeated 100 times to reduce the random errors. Each time, the I/O data sizes in Bitcoin node were recorded until the target transaction was found. Note that the size of block data is excluded in the input data size, since the block data is always needed to be transmitted regardless of whichever method used. The average and maximum data sizes are shown in Figure 5.

From the Figure 5(a), we can find that the average input and output data sizes are larger than 135KB and 116 KB separately when the polling-based read period is 1 second, which are both much larger than the size of a target transaction (Ethereum is interested in). What's more, the maximum input and output data sizes in Figure 5(b) are over 236KB and 278KB. On the other side, with the increase of period, both the average and maximum I/O data sizes are reduced. However, the reduce of I/O overhead is at the cost of timeliness of blockchain interaction. In detail, the larger the period is, the poorer the timeliness of blockchain interaction is. To reduce the I/O overhead without sacrificing the timeliness, we propose a novel method as follows.

B. Implementation and Evaluation of MMR

We propose a novel solution to improve the passive cross-chain communication, namely MMR (*Monitor Multiplexing Reading*). The underlying idea of MMR is mainly inspired by I/O multiplexing in Unix/Linux programming. Figure 6 illustrates the detailed flow of our solution, which is based on an event monitor. At first, the destination chain registers a desire to read source chain in the cross-chain event monitor. The desire specifies an event or transaction on the source chain, the destination chain is interested in. Then, if the source chain has not been monitored before, the event monitor will get started on monitoring the source chain. Otherwise, the event monitor will do nothing. Once the specified event takes place or transaction is committed on source chain, a signal will be sent to destination chain to inform it that it is time to read information from source chain. The next steps are the same as that in polling-based method. This method can help one destination chain to monitor several different source chains and several different destination chains to monitor one source chain simultaneously, which reduces the costs largely. It should be noted that the security of interaction will not be broken by a centralized event monitor, due to that the event monitor just inform the destination chain to read source chain while the verification process totally depends on the destination chain itself.

To evaluate the improvement effects of MMR, we implement a monitor multiplexing middleware, which takes on lots of work from Ethereum (proxy program). Obviously, with only one destination chain, the I/O data sizes in Bitcoin node introduced by the MMR method are close to that by the PBR method, which cannot reflect the improvements of MMR. Thus, we look at the advantages brought by MMR method from another perspective in this section. The hardware and software environments are the same as the Section V-A. We conducted 8 groups of experiments

Table I: A comparison of existing interoperability projects

Projects	**Atomicity**	**Efficiency**	**Security**	**Universality**	**Friendliness**
BTCRelay	Well	Poor	Well	Poor	Poor
RootStock	Well	Medium	Well	Poor	Poor
Interledger (Notary)	Well	Well	Poor	Poor	Poor
Interledger (Hash-lock)	Well	Poor	Well	Poor	Poor

whose number of destination chains increases from 1 to 8 for each method. Similarly, we repeated 10 times to reduce the random errors for each group experiment. In each experiment, we tracked the I/O data sizes and CPU utility in Bitcoin node. The average input data sizes introduced by PBR and MMR are shown in Figure 7(a). We can find that the input data size increases largely with the increase in the number of destination chains when the traditional PBR method is adopted. However, the input data size stays stable with the MMR method. The similar phenomenon occurs in terms of CPU utility, as shown in Figure 7(b). We can conclude from the experiment results that the MMR method has a better scalability, which can scale to a lot of destination chains with little increase in overheads.

VI. Related Work

A few prior proposals have aimed to implement the interoperability among blockchains, including BTCRelay, RootStock [23], Interledger [24] and so on. However, all of these projects reveal several deficiencies and shortages.

BTCRelay project is considered as the first sidechain of Bitcoin. By implementing the Bitcoin SPV (*Simplified Payment Verification*) wallet on the Ethereum smart contract, Ethereum can verify the transactions committed on Bitcoin network with ease. For instance, Alice can exchange her Bitcoin assets with Bob's Ethereum assets directly with BTCRelay. Its simple principle decides that BTCRelay is easy to implement. However, as the smart contract has to store the block headers of Bitcoin, the size of smart contract may grow too big to be cost effective. Besides, BTCRelay can only deal with the interaction between Bitcoin and Ethereum, which is lack of universality.

As another sidechain of Bitcoin, the principle of RootStock project is similar to that of BTCRelay, which combines the Bitcoin SPV wallet and smart contract too. Different from BTCRelay's developing a smart contract on Ethereum, RootStock implements the smart contract as a new blockchain. RootStock can exchange BTC to SBTC (i.e., a kind of cryptocurrency on RootStock) and vice versa. Due to its optimization of sidechain architecture design, RootStock has a better performance than BTCRelay. However, it still lacks the universality to enable more blockchain systems to join in.

Interledger project has undergone a scheme transformation from notary to hash-locking. In its early stages, notary scheme was adopted by Interledger, which is regarded as the simplest way to enable the interaction between different blockchains. In the notary scheme, a blockchain has to trust a set of entities to make decisions, which introduces the security issue. Without trusting a set of entities, the hash-locking scheme can facilitate the cross-chain atomic operations, so as to provide better security. Due to this reason, the Interledger project has replaced the notary scheme with hash-locking scheme later on. Nevertheless, the usage scenario of hash-locking is limited to assets exchange. Worse still, the user who wants to exchange assets has to find the partner on the other chain, which further constrains its flexibility.

A comparison among different projects is made in Table I. From the table, we can find that all of the projects are lack of universality and omit to provide interfaces to upper developers. Besides, all these projects do not transfer the transactions across chains actually, which limits the interoperability.

VII. Conclusion

In this paper, we show for the first time that the interoperability of multi-chain ecosystem can be grouped into active mode and passive mode. We discuss the proposal of establishing a novel architecture for supporting interoperability along with a couple of challenges. In particular, we propose the MMR method to reduce the reading overhead while maintaining the timeliness of interoperation. Experimental results show that it reduces the I/O overhead and CPU utility to 13.7% and 13.8% respectively compared with the PBR method, when the number of destination chains is increased to 8. The MMR method presents a better scalability so as to serve for a larger number of destination chains.

We take a courageous step towards making the blockchain ecosystem more practical and applicable. It also opens up exciting research directions that can build upon this cross-chain framework, and tackle the challenging issues in a divide-and-conquer manner.

Acknowledgment

This work was supported by National Basic Research Program of China (973 Program) under Grant No. 2014CB340600. Jiang Xiao is the corresponding author of the paper.

References

[1] M. Walport, "Distributed ledger technology: Beyond blockchain," *UK Government Office for Science*, 2016.

[2] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "Blockbench: A framework for analyzing private blockchains," in *Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD)*, 2017, pp. 1085–1100.

[3] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the internet of things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[4] M. Iansiti and K. R. Lakhani, "The truth about blockchain," *Harvard Business Review*, vol. 95, no. 1, pp. 118–127, 2017.

[5] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," https://bitcoin.org/bitcoin.pdf/, 2008.

[6] "Ethereum," https://www.ethereum.org/.

[7] "Financial statistical report by pbc," http://www.pbc.gov.cn/goutongjiaoliu/113456/113469/3461233/index.html.

[8] V. Buterin, "Chain interoperability," *R3 Research Paper*, 2016.

[9] "Btcrelay," https:btcrelay.org/.

[10] J. Garzik, "Public versus private blockchains part 1: Permissioned blockchains," *BitFury*, 2015.

[11] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *Proceedings of the USENIX Security Symposium*. USENIX Association, 2015, pp. 129–144.

[12] D. T. T. Anh, M. Zhang, B. C. Ooi, and G. Chen, "Untangling blockchain: A data processing view of blockchain systems," *Accepted to appear at IEEE Transactions on Knowledge and Data Engineering*, 2018.

[13] "Intel sgx," https://software.intel.com/en-us/sgx.

[14] "Hyperledger sawtooth," https://hyperledger.org/projects/sawtooth.

[15] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[16] L. Lamport, "Paxos made simple," *ACM Sigact News*, vol. 32, no. 4, pp. 18–25, 2001.

[17] D. Ongaro and J. K. Ousterhout, "In search of an understandable consensus algorithm," in *Proceedings of the 2014 USENIX Annual Technical Conference (ATC)*. USENIX Association, 2014, pp. 305–319.

[18] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM, 2017, pp. 51–68.

[19] S. Micali, M. Rabin, and S. Vadhan, "Verifiable random functions," in *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, 1999, pp. 120–130.

[20] N. Szabo, "The idea of smart contracts," *Nick Szabo ' s Papers and Concise Tutorials*, vol. 6, 1997.

[21] "Hyperledger fabric," https://www.hyperledger.org/projects/fabric/.

[22] "Blockchain oracle," https://blockchainhub.net/blockchain-oracles/.

[23] "Rootstock," https://www.rsk.co/.

[24] "Interledger," https://interledger.org/.