BRIDGE

Proof of Concept

Project description

Long Hoang Le

Scaling blockchain: optimized partitioning as a service

1. Summary

Since Bitcoin's initial launch in 2009, blockchain technologies have attracted extensive worldwide attention. Several blockchain implementations have been proposed, each focusing on solving specific shortcomings of the original blockchain proposal. Some implementations focus on special features, such as smart contract support [1, 2], and privacy [3]. Some others allow enterprises to create private blockchains, where only a defined set of entities can participate [4]. The growing adoption of blockchain-based systems have raised many new challenges. Two primary challenges in this context are scalability and interoperability.

Scalability is the ability to achieve a target throughput and latency in the presence of increasing workload without compromising the decentralized nature of blockchains. Scalability could be achieved by state partitioning (e.g., [5, 6, 7]). In order to be effective, the partitioning must ensure that most requests or transactions access one partition only and are equally distributed among partitions. That way, partitions can work in parallel to improve performance. Besides, when partitioning a blockchain, other factors need to be considered: computation, storage, and bandwidth.

Interoperability allows communication and data exchange between multiple different blockchains (e.g., sharing data between public and private blockchains, exchanging cryptocurrencies such as bitcoins for ether). Interoperability is gaining more attention in the blockchain community. Some blockchain systems are introducing inter-blockchain communication (IBC) protocols that enable communication and data exchange between multiple blockchains. This technology allows users to choose where to place smart contracts [8, 9, 10, 11]. Moreover, some solutions start to appear that allow smart contracts to move between existing blockchains [12, 13, 14].

Although scalability and interoperability are different aspects of blockchain, they share a similar concern: They both need a mechanism that tracks and provides some information about blockchains. A partitioned blockchain needs an overview of the partitioning of the system to compute an optimized location to place contracts and objects, thereby maintaining the balance of the network and increasing the performance. In inter-blockchains transactions, where data could be shared and exchanged between different blockchains, it is necessary to track the movement of the data, as well as to have some information about the parties, including the cost of transactions and the performance of the network.

This project proposes an oracle service that provides such information. The oracle service consists of two main components, one on-chain and one off-chain. The on-chain component is in the form of a smart contract that provides interaction interface for user smart contracts. The off-chain component is a service that monitors the blockchains by analyzing transaction history. To process user requests (e.g., for object location, for optimizing transaction costs) the on-chain and off-chain components communicate to extract the necessary information for the request. User transactions pay per-query fees to get the hints from the oracle service. The cost of each query depends on the time and resources that the system has spent for that query. With the information provided by the oracle, the blockchain system can effectively choose a partition to place objects to maximize the balance of the network, thus increasing the transaction speed, as well as optimizing the execution cost of transactions.

2. Project description

2.1. Research background

In this section, we briefly summarize the state-of-the-art in the proposed research areas and outline the current state of the principal investigator own research.

2.1.1. Current state of research in the field

Since the introduction of Bitcoin, the demand for blockchain-based solutions has increased. A multitude of blockchain technologies have been proposed to address different aspects of digital life. Being the first generation of blockchain, Bitcoin essentially supports the transferring of assets. Later generations of blockchain systems (e.g., Ethereum, NEO, Stellar, Cosmos) have extended blockchains with support for *smart contracts*. Smart contracts provide a general-purpose programmable infrastructure, where the blockchain does not only store financial data but also has the capability of deploying and executing custom code. More technically, smart contracts implement the abstraction of state machine replication. Ethereum is after Bitcoin the second-largest cryptocurrency platform by market capitalization. Unlike Bitcoin, Ethereum supports programmable smart contracts with an entire programming language along with it, running in Ethereum Virtual Machine (EVM) [15]. The flexibility within a smart contract allows users to implement their own autonomous execution logic on blockchains, which enables more options for optimization. However, several challenges in blockchain technology remain to be addressed, two of which are scalability and interoperability.

Scalability. The performance of mainstream blockchains is still a barrier to the adoption of blockchain technology. As a global payment system, Visa is capable of processing around 24,000 transactions per second on average [16], while Bitcoin and Ethereum can handle 7 and 15 transactions per second [17, 18]. Scaling the performance of blockchain systems has become a hot topic for both industry and academia, with many proposals. Those proposals vary from optimizing the performance of the consensus protocols [19], partitioning the blockchain [20], to shifting the computation from the blockchain (on-chain) to the outside (off-chain) [21, 22]. Among all the proposed methods for scaling the performance, sharding the blockchain state is an effective and practical solution, as it can overcome both performance and scalability problems. Partitioning (or *sharding*) is a technique that divides the state of a service or a system in multiple partitions so that most requests or transactions access one partition only and are equally distributed among partitions (e.g., [5, 6, 7]). As a result, partitions can work in parallel to maximize performance and improve throughput. In the context of blockchain, state partitioning is challenging. With its decentralized nature, there is no single entity in a blockchain system that contains information of all objects in the network, which makes it difficult for blockchain clients to locate objects in the presence of partitioning. RapidChain [23] builds a routing overlay network for users and committee leaders to locate the committee where to send their transactions. Moreover, with the lacking of overall information, choosing a partition to place the objects in order to maximize the balance between partitions is not effective. Even if enough information is available, finding a good partitioning is a complex optimization problem [24, 25]. To the best of our knowledge, no work has focused on achieving an optimized partitioning in the context of blockchain.

Interoperability. With the increasing number of blockchain systems and distributed applications (DApps), there is a rising demand for interoperability. Interoperability in the context of blockchain means connecting multiple blockchains to share, exchange, or change the state of the own or the other blockchain [26]. In the recent evolvement of blockchain technologies, some blockchain systems have supported interoperability in their core, in the form of an inter-blockchain communication (IBC) protocol [8, 9, 10, 11]. Moreover, some solutions start to appear that allow smart contracts to move between existing blockchains [12, 13, 14]. In [12], Fynn *et al.* introduced a move primitive [12] that allows programmable blockchains to interoperate by moving contracts and data from one to another. The possibilities of interoperability are endless. An application in a private blockchain can access data from a public blockchain [27], A contract can shift an expensive computational off-chain [21, 22], or a DApps can access data from multiple blockchains. However, it also leaves new open questions. An important question is how to locate objects in a multi-blockchain environment where objects can move from one blockchain to another. Another crucial question is how to optimize the economics of object placement. This project will address both questions.

2.1.2. Current state of principal investigator's research

Investigator Long Hoang Le conducts research in distributed systems. Much of his work has focused on scalable and fault-tolerant systems, particularly in the context of state machine replication (SMR) [28, 29, 30]. His work on scalable SMR [28] was nominated for the best paper award at the 46th IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), the flagship conference on fault-tolerant systems.

State machine replication is a common approach to building fault-tolerant systems [31, 32]. Participants in a replicated state machine system form agreement on a sequence of transactions and apply them in sequence order deterministically to reach the same state. With its simple yet effective execution model, state machine replication is at the core of blockchain systems [33, 34]. This approach provides a configurable degree of availability but limits the achievable throughput of the replicated service to what a single replica can process sequentially. In the work DynaStar [29], Le *et al.* proposed an approach to scaling the performance of SMR by allowing the SMR system to partition the state and reconfigure its data placement on-the-fly. DynaStar maintains a location oracle with a global view of the application state. The oracle helps users locate objects in the system and optimizes data placement for performance. A detailed experimental evaluation with standard benchmarks has revealed performance improvements of two orders of magnitude.

2.2. Innovation potential and market review

This project's main goal is to develop an oracle service that analyzes blockchain transactions to provide object information about placement and economics. The service will answer several questions such as where objects and smart contracts are located, where to place objects to optimize the partitioning of the network to improve performance, or to minimize transaction costs. Therefore, it is aimed to be the catalyst that supports the widespread adoption of blockchain technologies.

Blockchain analysis is becoming increasingly important in the cryptocurrency market [35]. Some blockchain monitoring and intelligence platforms are now offering analytics tools and software for

investigating entities, individuals, and transactions interacting with a blockchain [36, 37, 38]. Private companies use blockchain analysis solutions to track their customers' transactions to learn more information about them. Security and government firms use those solutions to effectively identify money laundering operations, illicit activity and other financial crimes. The information provided by our oracle service serves different purposes. First, it provides a location mapping for smart contracts and objects within and across partitions. This enables blockchain clients to quickly locate to which partitions or blockchain they should send their transactions. Second, the location mapping opens up possibilities for further improvements, e.g., computing the optimized location for the upcoming transactions and objects. Third, with the information of the execution cost and time provided by our oracle service, the blockchain client can decide to off-load expensive computation to a side-chain to optimize the cost. Our main customers are blockchain developers and distributed application designers whose main concerns are about the scalability and cost efficiency of the blockchain or its applications.

Figure 2.1 shows the mean value in USD of transaction fees in the first six months of the year 2020 [39]. This data suggests that the average fee for executing an ordinary transaction on the Stellar network is 1,000 times cheaper than on Ethereum Classic network, and is 100,000 times cheaper than running on Ethereum. If this information is available at the time of execution, the blockchain can make an effective decision of where to execute the transaction. However, it does make two important assumptions: (i) the transaction is executable on all parties (i.e., there is a smart contract of this transaction type available on both blockchains). (ii) there exists a move operation that could send the necessary data from one contract to another (such primitive was proposed in [12]).

Our proposed service could be quickly and easily integrated with existing blockchains, DApps, or non-blockchain-based applications. Our solution doesn't require any changes in the blockchain code, and therefore makes the integration simple without hassle. The system is accessible in different ways. On-chain smart contracts or DApps can request necessary information by issuing a call to our on-chain components. Non-blockchain based applications can also access our data with REST API. In addition, the design of our system is flexible enough to add support for new blockchain systems.

Our main source of revenue is from blockchain users. Users will be charged based on the time it takes to execute their queries. This time is calculated from the time our computational unit begins executing the query until it returns. The price depends on the maximum amount of memory that users are willing to pay (e.g., a query for the location of objects results in a lower cost than a query for an optimized location that takes more time and resources to calculate). With our pay-per-call pricing model, the users only pay for the queries that they run. This payment model has several benefits. There is no entry barrier for users, no commitment required, and it offers the users more flexibility around the services they want, and when they want them. For instance, the application could decide to cache some queries results once in a while for further usages to optimize costs.

2.3. Implementation strategy

To fulfill the goals of the project, the proposed system needs to address the following requirements:

- The system should be able to work with a number of different blockchains. While most of the existing blockchains share the core principles, their implementations and data structures

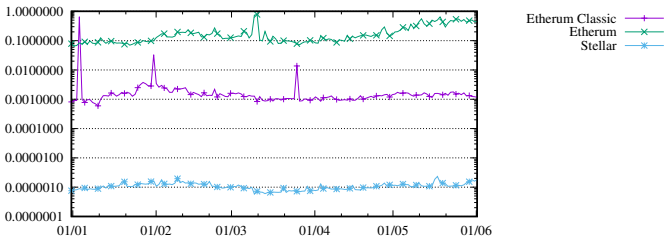

Figure 2.1: Transaction fee of Ethereum, Ethereum Classic and Stellar in USD value in first half of 2020

are different. The system should have an abstraction to standardize the communication with different blockchains, simplifying the process of adding support for new blockchains.

- The integration with existing blockchains and DApps should be simple. Ideally, the deployment of the system on existing blockchains should be ”plug-and-play”, without introducing any changes in the protocol of the blockchain.

Figure 2.2 presents the overall architecture of our proposed oracle service. The core components of the system are divided in two parts: the on-chain smart contract component and the off-chain service component. The on-chain smart contract (OSC) is a set of smart contracts that are developed separately for each supported blockchain. The OSC plays the roles of the interface of the service, receives requests from other contracts in the form of contract calls, and dispatches those requests to the off-chain component. The OSC is also responsible for computing the fee of each request. Figure 2.3 proposes an example of the OSC in the form of Ethereum's smart contract. The off-chain component consists of a computational unit (CU) and a set of blockchain analyzers. The analyzers monitor the blockchains by actively fetching and analyzing the transaction history to aggregate multiple metrics of a standard blockchain, for example, the location of contracts and objects in the case of movement, the average execution cost, or the pending transactions count. The CU is a set of computational nodes that serves requests by historical data fetched by the analyzers. Upon receiving requests from user smart contracts, the OSC validates the requests and forwards those requests to the CU. The CU queries the data from the backlog and replies to the OSC. Once receiving the response from CU, the OSC calculates the fee for the request, and send back the data to the user smart contracts.

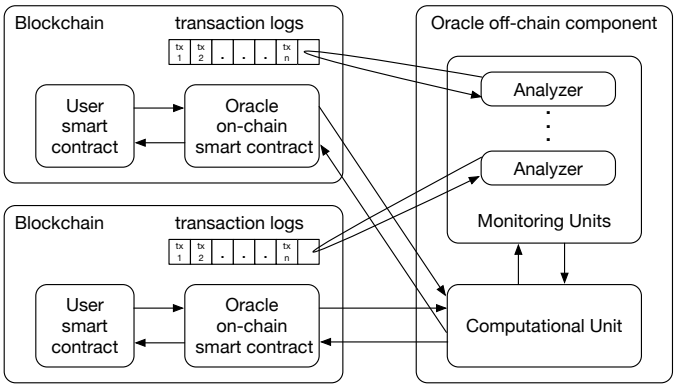

Figure 2.2: The architecture of the proposed oracle service

```
contract Oracle {
      function getTransactionFee(address sidechain) public view returns (uint);
      function getContractLocation(address contract) public view returns (uint);
      function getOptimizedLocation(address contract) public view returns (uint);
}
```

Figure 2.3: The service interface on Ethereum's smart contract

2.4. Project plan

The project plan will divide the implementation strategy described in Section 2.3 into three phases. Figure 2.4 identifies the major tasks and milestones for each phase.

Phase 1: Developing off-chain component. In this phase, we will to develop an abstraction that helps standardize the monitoring process for the blockchain analyzers. With this abstraction, we aim to implement analyzers for at least three existing blockchains. We will also implement the computational unit in this phase. The outcome of this phase is the fully functioning off-chain component.

Phase 2: Developing on-chain smart contracts. Similar to Phase 1, we will develop an abstraction for the interface for the service in the form of smart contracts. Then we will implement this abstraction on at least three different blockchains.

Phase 3: Deploying and evaluating. We will deploy the system on the supported blockchains and start the evaluation process. To benchmark the system, we will develop a sample DApps that could leverage the information of the oracle to make an optimized decision. The evaluation will consider the scalability of the blockchain, as well as the financial benefits based on the hints of the oracle service.

Risks. The proposed project is adventurous, as with any research proposal, carries with it a certain amount of uncertainty. In this project, some use cases of cost-efficient are based on the assumption of the existence of a move protocol that enables the transferring of smart contracts and objects from one blockchain to another. Although such primitive was presented in a recent work [12] that allows the moving of a smart contract between Ethereum and Burrow, developing and integrating such protocol to our system may require a considerable amount of time and effort. To mitigate this risk, we will collaborate with Fynn Enrique, the main author of that work, to help us with the integration.

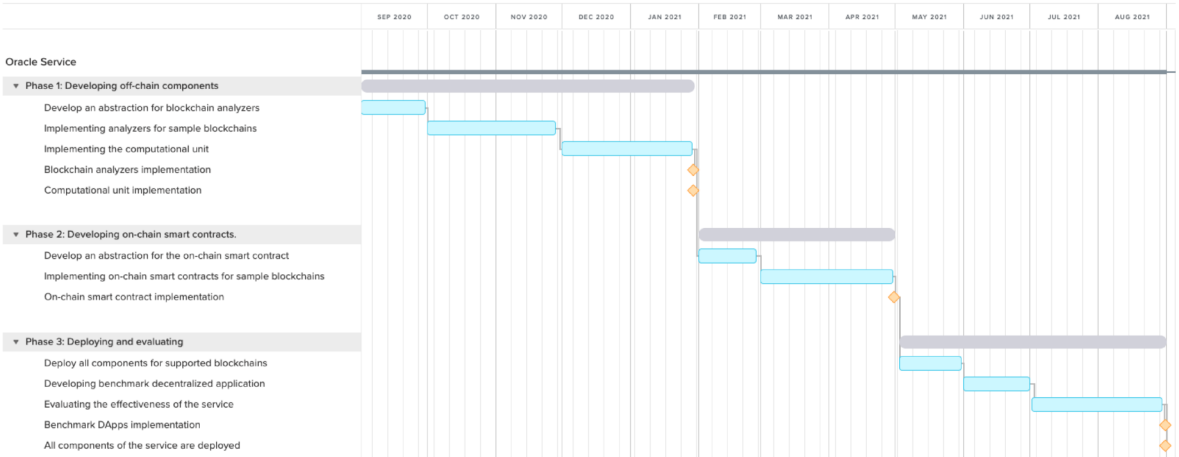

Figure 2.4: A tentative project timetable.

References

[1] V. Buterin *et al.*, "Ethereum white paper," *GitHub repository*, vol. 1, pp. 22–23, 2013.

[2] E. Elrom, "Neo blockchain and smart contracts," in *The Blockchain Developer*. Springer, 2019, pp. 257–298.

[3] K. M. Alonso, "Monero-privacy in the blockchain," 2018.

[4] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, 2018, pp. 1–15.

[5] N. Bronson, Z. Amsden, G. Cabrera, P. Chakka, P. Dimov, H. Ding, J. Ferris, A. Giardullo, S. Kulkarni, H. Li *et al.*, "Tao: Facebook's distributed data store for the social graph," in *USENIX ATC*, 2013.

[6] D. Sciascia, F. Pedone, and F. Junqueira, "Scalable deferred update replication," in *Proceedings of the 2012 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, ser. DSN '12. IEEE Computer Society, 2012, pp. 1–12.

[7] M. K. Aguilera, A. Merchant, M. Shah, A. Veitch, and C. Karamanolis, "Sinfonia: A new paradigm for building scalable distributed systems," in *SOSP*, 2007.

[8] J. Kwon and E. Buchman, "Cosmos: A network of distributed ledgers," *URL https://cosmos.network/whitepaper*, 2016.

[9] S. Thomas and E. Schwartz, "A protocol for interledger payments," *URL https://interledger.org/interledger.pdf*, 2015.

[10] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, "Omniledger: A secure, scale-out, decentralized ledger via sharding," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 583–598.

[11] M. Al-Bassam, A. Sonnino, S. Bano, D. Hrycyszyn, and G. Danezis, "Chainspace: A sharded smart contracts platform," *arXiv preprint arXiv:1708.03778*, 2017.

[12] E. Fynn, A. Bessani, and F. Pedone, "Smart contracts on the move," in *Proceedings of the 2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, ser. DSN '20. IEEE Computer Society, 2020.

[13] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timón, and P. Wuille, "Enabling blockchain innovations with pegged sidechains," *URL: http://www.opensciencereview.com/papers/123/enablingblockchain-innovations-with-pegged-sidechains*, vol. 72, 2014.

[14] M. Herlihy, "Atomic cross-chain swaps," in *Proceedings of the 2018 ACM symposium on principles of distributed computing*, 2018, pp. 245–254.

[15] E. foundation, "Ethereum virtual machine wiki," https://eth.wiki/en/concepts/evm/ethereum-virtual-machine-(evm)-awesome-list, 2017, [Online; Accessed: 2020-06-10].

[16] Visa, "Visa acceptance for retailers," https://usa.visa.com/run-your-business/small-business-tools/retail.html, 2010, [Online; Accessed: 2020-06-10].

[17] E. foundation, "On sharding blockchains faqs," https://eth.wiki/sharding/Sharding-FAQs, 2018, [Online; Accessed: 2020-06-10].

[18] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Manubot, Tech. Rep., 2019.

[19] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, "Towards scaling blockchain systems via sharding," in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 123–140.

[20] G. Wang, Z. J. Shi, M. Nixon, and S. Han, "Sok: Sharding on blockchain," in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, 2019, pp. 41–61.

[21] J. Teutsch and C. Reitwießner, "A scalable verification solution for blockchains," *arXiv preprint arXiv:1908.04756*, 2019.

[22] R. Network-Fast, "cheap, scalable token transfers for ethereum," 2018.

[23] M. Zamani, M. Movahedi, and M. Raykova, "Rapidchain: Scaling blockchain via full sharding," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 931–948.

[24] C. Curino, E. Jones, Y. Zhang, and S. Madden, "Schism: A workload-driven approach to database replication and partitioning," *Proc. VLDB Endow.*, 2010.

[25] R. Taft, E. Mansour, M. Serafini, J. Duggan, A. J. Elmore, A. Aboulnaga, A. Pavlo, and M. Stonebraker, "E-Store: Fine-grained elastic partitioning for distributed transaction processing systems," *Proc. VLDB Endow*, 2014.

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

[27] N. Prusty, *Blockchain for Enterprise: Build scalable blockchain applications with privacy, interoperability, and permissioned features*. Packt Publishing Ltd, 2018.

[28] L. Hoang Le, C. E. Bezerra, and F. Pedone, "Dynamic scalable state machine replication," in *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, June 2016, pp. 13–24.

[29] L. Hoang Le, E. Fynn, M. Eslahi-Kelorazi, R. Soulé, and F. Pedone, "Dynastar: Optimized dynamic partitioning for scalable state machine replication," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, July 2019, pp. 1453–1465.

[30] C. E. Bezerra, L. H. Le, and F. Pedone, "Strong consistency at scale." *IEEE Data Eng. Bull*, vol. 39, no. 1, pp. 93–103, 2016.

[31] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Communications of the ACM*, vol. 21, no. 7, pp. 558–565, 1978.

[32] F. B. Schneider, "Implementing fault-tolerant services using the state machine approach: A tutorial," *ACM Computing Surveys*, vol. 22, no. 4, pp. 299–319, 1990.

[33] M. Baudet, A. Ching, A. Chursin, G. Danezis, F. Garillot, Z. Li, D. Malkhi, O. Naor, D. Perelman, and A. Sonnino, “State machine replication in the libra blockchain,” *The Libra Assn., Tech. Rep*, 2019.

[34] C. Cachin *et al.*, “Architecture of the hyperledger blockchain fabric,” in *Workshop on distributed cryptocurrencies and consensus ledgers*, vol. 310, 2016, p. 4.

[35] G2, “Blockchain analysis software,” https://www.g2.com/categories/blockchain-analysis, 2019, [Online; Accessed: 2020-06-10].

[36] Chainalysis, “Chainalysis kyt,” https://www.chainalysis.com/chainalysis-kyt/, 2020, [Online; Accessed: 2020-06-10].

[37] “Bloxy,” https://bloxy.info/, 2020, [Online; Accessed: 2020-06-10].

[38] Blocwatch, “Blocmonitor,” https://www.blocwatch.com/blockchain-monitoring, 2020, [Online; Accessed: 2020-06-10].

[39] C. Metrics, “Coin metrics network data charts,” https://coinmetrics.io/charts/#assets=eth,etc,xlm_left=FeeMeanUSD, 2020, [Online; Accessed: 2020-06-10].