Blockchain Minus Theory

0xKJ 0xFBY

September 27, 2024

Abstract

Blockchain technology has evolved rapidly, from Bitcoin's decentralized transactions to Ethereum's smart contracts. However, this feature addition has introduced significant challenges in terms of scalability and computational complexity. Current blockchains struggle with limited transaction throughput (TPS), high gas fees, and state explosion due to the growing computational demands of on-chain smart contracts.

In response, we propose the Blockchain Minus Theory, a novel approach that simplifies blockchain architecture by removing computation from the consensus process. Rather than attempting to add layers of complexity to scale, the Minus Theory advocates for off-chain computation, allowing blockchains to focus solely on transaction ordering and security. This decoupling of computation from consensus significantly improves scalability, reduces costs, and increases flexibility for developers.

By transforming the blockchain into a streamlined consensus machine, the Minus Theory offers a path toward more scalable, efficient, and cost-effective decentralized systems, paving the way for mass adoption of blockchain technology.

1 Introduction

The evolution of blockchain technology, from Bitcoin to Ethereum, has seen rapid innovation, but also increasing complexity. Bitcoin's simple, decentralized transaction model laid the foundation, but its limitations in throughput and speed hindered scalability. Ethereum introduced smart contracts, expanding blockchain's utility but also introducing performance challenges like high gas fees and global state management issues.

As decentralized applications (DApps) proliferated, demand for blockchain scalability surged. However, the need for every node to compute and verify transactions created bottlenecks, limiting transaction throughput (TPS) and driving up costs. Solutions like Layer 2 networks, sharding, and rollups have been developed to address these issues, but they often add complexity and will soon reach the limit as the time between blocks is finite.

1.1 The Scalability Bottleneck

Blockchain's core limitation stems from the tight coupling of computation and consensus. Every node must execute all transactions and update the global state, limiting scalability. This results in:

- **Limited TPS**: Blockchains struggle to process more than a few dozen transactions per second, far from the needs of global-scale applications.
- **Rising Gas Costs**: With increased demand, the cost of processing transactions becomes prohibitively expensive.
- **State Explosion**: The continuous growth of the global state increases the storage and computational requirements for each node.

1.2 Blockchain Minus Theory

Blockchain Minus Theory proposes a radical simplification: separate computation from consensus. Instead of adding complexity to improve performance, Minus Theory advocates for offloading computation off-chain, allowing the blockchain to focus purely on ordering transactions and maintaining security.

This approach draws inspiration from earlier projects like Counterparty and recent innovations like ETHS and Facet Protocol, which successfully demonstrated off-chain asset management and computation. By removing the computational burden from the blockchain, Minus Theory enables leaner, more scalable systems.

1.3 Key Benefits of Minus Theory

- **Higher TPS**: Off-chain computation allows blockchains to process more transactions without being slowed down by complex smart contracts.
- **Lower Costs**: Transaction costs are reduced as gas fees for on-chain computation are minimized.
- **Scalability**: Off-chain systems handle intensive tasks, freeing the blockchain to scale more efficiently.
- **Security**: The blockchain remains a secure, decentralized ledger, while off-chain environments handle complex operations.

By simplifying blockchain's role to that of a consensus machine, Minus Theory offers a clear path to greater scalability and efficiency, enabling blockchain technology to meet the demands of global applications without sacrificing decentralization or security.

2 Problems

Blockchain faces many challenging problems, such as transaction throughput limitations and state explosion for smart contract chains. Previously, we solved transaction sharding, which brought down the blockchain node hardware requirements. There are similar solutions, including DAG. However, due to the high throughput, the on-chain smart contract execution became the bottleneck. The Minus Theory shows the way to unlock on-chain computational resources.

2.1 TPS Limited by Smart Contract and Global State

The blockchain throughput is measured in Transactions per Second (TPS), showing the blockchain's transaction processing capacity. Previously, our sharding solution increased the capacity so that each node does not need to store the full blockchain data. It reduces the hardware requirement, especially in terms of storage resources.

With the increase in throughput and TPS, the blockchain node needs to process more transactions. In a smart contract environment, each transaction evaluation requires certain resources and execution time. The node must process all the transactions within the block during the block interval gap. It is challenging to increase TPS when transactions invoke complex smart contract logic.

Bitcoin, the first blockchain, came with built-in simple rules to support coin transactions: a user sends some coins with a small amount as a transaction fee. The sender must have enough coins to pay for both the transaction amount and the fee.

When other blockchains with smart contracts emerged, they allowed customization of rules by deploying new contracts. The transaction in Bitcoin is simply about sending coins, but now it has evolved to a call of smart contract code. This leads to more complex smart contract evaluations. Each transaction, which used to perform a simple balance check, now needs to be run in a virtual machine (VM).

Moreover, the code run in the VM requires more resources. Smart contract-based blockchains allow a smart contract to call another smart contract. Additionally, each smart contract may read and write to a huge key/value database, named the global state. In the Ethereum blockchain, users can allocate new on-chain storage space in the global state if gas is paid. Thus, we see that the size of the global state in Ethereum keeps growing at a dynamic rate. It is always expanding, and so far, there is no way to reduce the global state size.

We can say that smart contract blockchains are limited in performance, and the requirements for VM execution/storage are growing. People keep adding new features to both enrich the blockchain and increase performance. Some milestones have been achieved, but we are still far from practical implementation.

2.2 Dapp Features Limited by On-chain Computation Resource

From Bitcoin to Ethereum, the biggest change is the emergence of smart contracts. Based on the EVM, many decentralized applications have emerged, including DeFi. Before Layer 2, most dapps were expensive to use. This is because on Ethereum Layer 1, the computation resource is scarce, requiring end-users to pay ETH as gas to run the dapp. With Layer 2, although the cost for end-users has been reduced, dapps still face the issue of limited computation and storage resources.

The reason for the current situation with smart contracts is due to their design. Ethereum decided to extend blockchain features to include a virtual machine by adding the programming language execution module into a blockchain node. The chain accepts anyone to deploy code to Ethereum, making it impossible to detect if there is a dead loop during code execution. In this design, the new block generation involves executing all the dapp call requests and saving the changes made by the requests to the block header (global state Merkle tree root). The cost of executing the VM is much higher than Bitcoin and must be done within the block interval, which has limited execution time. A gas limit is set for a block to prevent any dapp from taking too many resources like execution time or CPU cycles.

The Minus Theory proposes a novel approach to addressing the dapp execution limit problem by removing the smart contract execution from the blockchain node. Instead of executing smart contracts on-chain, the Minus Theory suggests offloading the execution to off-chain environments. By removing the smart contract execution from the blockchain node, the Minus Theory significantly reduces the computational burden on the blockchain. This approach allows the blockchain to focus on its core function, which is to reach consensus on the input message, without being bogged down by the resource-intensive process of executing smart contracts. Additionally, the Minus Theory allows for delaying the execution of dapp requests. This delay brings more elastic time to execute the smart contract, which gives more resources to execute and makes it possible to build more complex applications.

3 On-chain Application with Off-chain Computation

The idea of Off-chain Computation has a long history. It began to sprout as early as 2014 with the attempt of the Counterparty protocol. The Counterparty protocol emerged as an innovative solution to create new assets based on the Bitcoin network. Counterparty emerged as an alternative solution to Colored Coins, both aiming to enable asset issuance on the Bitcoin blockchain. While Colored Coins relied on Bitcoin's native scripting, Counterparty innovated by using an indexer approach to interpret and manage asset data. This pioneering method paved the way for future developments in off-chain computation and asset issuance protocols, inspiring more implementations such as BRC-20. These subsequent innovations built upon the foundational concept introduced by Counterparty, further exploring the possibilities of extending blockchain functionality through off-chain mechanisms.

3.1 Counterparty

Counterparty is the asset issuance protocol [1] on Bitcoin. Counterparty represents a seminal contribution to the exploration of Off-chain Computation within the Bitcoin ecosystem. As an asset issuance protocol built on top of Bitcoin, Counterparty introduced novel mechanisms for conducting off-chain computations while leveraging the security and immutability of the underlying blockchain. By enabling the creation and transfer of digital assets through off-chain processes, Counterparty paved the way for a more scalable and versatile blockchain ecosystem.

3.2 Ordinals and BRC-20

Building upon the foundations laid by Counterparty, subsequent developments in Off-chain Computation led to the emergence of protocols such as Ordinals and BRC-20. These protocols introduced standardized approaches to tokenization and asset management, further demonstrating the potential for Off-chain Computation to enhance the functionality and interoperability of blockchain networks. Ordinals and BRC-20 protocols expanded the scope of Off-chain Computation, offering practical solutions for a wide range of use cases within decentralized ecosystems.

3.3 ETHS and Facet

ETHS (ETHScription) is the inscription asset issuance protocol on Ethereum, operating without reliance on smart contracts. It demonstrates that off-chain computation is effective on both Bitcoin and Ethereum. The Facet Protocol is a rebrand of ETHS, now supporting off-chain smart contracts. It showcases the advantage that Ethereum users can enjoy lower gas costs while utilizing the same features as on-chain smart contracts.

In the context of Ethereum, Off-chain Computation has been realized through protocols such as ETHS (ETHScription) and Facet. ETHS, as an inscription asset issuance protocol, has proven the viability of Off-chain Computation on Ethereum, highlighting its compatibility with diverse blockchain platforms. By facilitating asset issuance without solely relying on smart contracts, ETHS has demonstrated the flexibility and scalability benefits of Off-chain Computation in decentralized environments.

Facet Protocol, a refinement of ETHS, has further emphasized the strengths of Off-chain Computation within the Ethereum ecosystem. By incorporating off-chain smart contract capabilities, Facet Protocol has offered Ethereum users enhanced efficiency and reduced gas costs while maintaining compatibility with on-chain smart contracts. This integration has exemplified the synergistic relationship between Off-chain and On-chain Computation, providing users with a more seamless and cost-effective experience.

The Facet Protocol, building upon the foundations of ETHS, introduces a significant advancement in the realm of Off-chain Computation for Ethereum. By focusing on providing a more cost-effective way to interact with the blockchain, Facet Protocol offers users the benefits of smart contract functionality without the typically associated high gas fees.

The protocol achieves this by executing complex computations off-chain and only submitting the results to the Ethereum blockchain for verification and consensus. This approach significantly reduces the computational burden on the network, resulting in lower gas fees for users. Essentially, Facet Protocol offers a bridge between the robust security of on-chain transactions and the cost-efficiency of off-chain computation.

By providing this cheaper alternative to traditional on-chain smart contract execution, Facet Protocol opens up new possibilities for decentralized applications (dApps) and use cases that were previously constrained by high transaction costs. This innovation could potentially lead to increased adoption of blockchain technology by making it more accessible and economically viable for a broader range of users and applications.

4 State Machine

State machine is a mathematical abstraction used to model the blockchain system. The state machine concept in blockchain can be represented by a classic algorithm that processes transactions and updates the global state. Here's a simplified representation:

Algorithm 1 Blockchain State Machine

```
globalState ← initialState
for each block in blockchain do
    for each transaction in block do
        input ← transaction.data
        globalState ← stateTransferFunction(globalState, input)
    end for
end for
return globalState
```

This algorithm illustrates how the blockchain processes transactions and updates the global state, embodying the state machine concept. The $stateTransferFunction$ represents the rules (smart contracts) that determine how each transaction modifies the state.

A state machine reads an input and changes to a different state based on the input. We model the blockchain system as a state machine.

$$State \xrightarrow[Function]{Input} State'$$

- *State* is the current state. In blockchain, the state refers to the global state or the users' balance.
- *Input* is the data applied to the current *State*. The all transactions of a chain block is the state machine input.
- *Function* is the rule to applying *Input* to current *State*, also called as State Transfer Function.
- *State'* is the new state after accepting *Input* with the rule *Function*.

Let's use a simple example to explain how the blockchain represents the state machine:

We have a blockchain with a smart contract that only has the mint feature for one user, 0x1. The user 0x1 has a coin balance of 5, and after minting, the balance will be increased by 1. Assuming there is no gas fee required to call the smart contract.

$$5 + 1 = 6$$

The addition (+) is the rule of the function that applies the input 1 to the current state 5. After the state machine evaluation, we get the new state 6 as the balance of user 0x1.

In the real-world blockchain, the smart contract global state is the state, and the transactions within the block are the input. All the deployed smart contracts serve as the rules, functioning as one large function. The more smart contracts deployed, the longer the list of rules would be. Just imagine that a smart contract may call another smart contract, and each smart contract invocation may require reading or writing any data entity within the global state.

Thus, we can say that the blockchain performance is limited by the on-chain smart contract execution.

4.1 State Transfer Function and Pure Function

The State Transfer Function plays a crucial role in the blockchain state machine model, yet its importance is often overlooked. To better understand its significance, let's delve deeper into its characteristics.

The State Transfer Function in a blockchain context is responsible for determining how the global state changes in response to inputs (transactions). It encapsulates the rules defined by smart contracts and the protocol itself. Key points about the State Transfer Function include:

- **Deterministic**: Given the same input and current state, it always produces the same output state.
- **Stateful**: It operates on and modifies the global state of the blockchain.
- **Consensus-Critical**: All nodes must implement this function identically to reach consensus.

In a blockchain system, the State Transfer Function would behave more like a pure function, with **Predictable Outcomes**.

This approach aligns with the principles of Minus Theory, which we will explore in the next section, by simplifying the consensus process and reducing the burden on blockchain nodes.

5 Minus Theory

Minus Theory explains why off-chain computation and related ideas are more useful than just distributing assets. Roughly speaking, the Minus Theory proposes to remove verification from the consensus.

5.1 Blockchain Consensus Process

We examine how the blockchain works during the consensus process:

1. Verification
2. Sequence
3. Election

When Bitcoin users send coins, they create a transaction with information about who to send the coins to, how much to send, and the transaction fee to pay. Then, the transaction is signed with the private key and broadcasted to the P2P network.

Next, the blockchain miners receive the transactions from the nodes, ordered by the gas fee. Before the miners decide to include the transactions in the block, it is essential to check two things: 1) if the transaction sender has enough coins to pay the miner fee, and 2) if the sender has sufficient coins to pay the recipient. These two types of verification are different, which we will explain later. If any of the conditions are not satisfied, the miners will refuse to include the transaction in the next block to mine. Even in Bitcoin, a simple balance check is required. This is the step of Verification.

Then, the miners create block candidates and start the Proof of Work computation over the block header (which contains the previous block hash, the Merkle tree root of all the transactions, and other information). This step is an election among a massive number of solo miners. Only one of the block candidates will be chosen as the next block and win the reward. In this step, the transactions within the block are in a fixed order (Sequence) through the consensus algorithm (Election).

5.2 Remove Verification from Consensus

Minus Theory eliminates the need for verification in the blockchain consensus process. Previously, we discussed two types of verification during the consensus process: 1) transaction fee 2) balance. The removal of verification pertains to the second type, which does not involve checking if the user has sufficient balance to send the specified amount after paying the transaction fee.

While the removal of simple verification may only result in a minor performance improvement for a blockchain system like Bitcoin, its impact becomes significant when applied to a smart contract based blockchain. In such a system, each transaction execution or verification requires VM level computation, making the process highly complex and expensive. By removing the verification from the consensus process, as proposed by Minus Theory, we can significantly reduce these complexities. This approach allows for a more streamlined blockchain system that focuses on its core functions of ordering and recording transactions, while moving the complex computations off-chain. This approach transforms blockchain nodes from complex systems into simple consensus machines. Miners and nodes no longer need to verify transaction details; they only focus on collecting and ordering transactions while maintaining the historical record. By reducing the responsibilities of the blockchain, this enables a lightweight system with lower hardware requirements. Essentially, it simplifies the blockchain's core function to that of a distributed, immutable ledger, offloading complex computations and verifications to off-chain processes.

It's important to note that while Minus Theory proposes removing complex verifications from the consensus process, blockchain systems still need to perform a basic check: verifying if users have sufficient coins to pay the transaction fee. This check is crucial for preventing Sybil attacks and spam in permissionless systems. By requiring a fee, the blockchain ensures that only legitimate users can interact with the network, effectively filtering out potential malicious actors or spam transactions. However, this simple fee verification is far less computationally intensive compared to the complex smart contract verifications that Minus Theory aims to move off-chain.

5.3 Bring Smart Contract off the Blockchain

Removing verification from the consensus process means that blockchain nodes no longer run a VM to check transaction validation. Instead, after blockchain consensus, an indexer role outside the blockchain node is responsible for replaying the ordered transactions and calculating the global state for any given block height. This separation allows the blockchain to focus on ordering and recording transactions, while the complex computations are handled off-chain by the indexer.

This leads to the concept of Asynchronous State Computation. In this model, the consensus process and the execution of global state updates are decoupled. The blockchain first reaches consensus on the order of transactions, and only afterwards, the global state is computed off-chain. This separation allows for a more efficient consensus process, as it no longer needs to wait for complex computations to complete before finalizing blocks.

The set of smart contracts are essentially the state transfer function that define how the global state should be updated. By deploying the smart contract code to the global state, we ensure that the state transfer function is maintained within the global state itself. As long as we keep the VM implementation aligned across different indexers, they will consistently generate the same global state for a given set of block transactions in a fixed order. This approach creates a decentralized system based on off-chain indexers that can reliably calculate the latest global state using the ordered blockchain transactions as input. The principles of state machine theory ensure that different indexers will arrive at the same state when computing from the consensus-ordered transactions. State transfer functions provide a deterministic set of rules for state updates.

The off-chain indexer system reuses the blockchain consensus mechanism. Thus the indexer does not need to the consensus algorithm. The original blockchain system was split into two components: a) the node without smart contract but only focus on the consensus. b) the indexer listening to the node for the consensus result and computing the state.

5.4 User State Verification

While the state machine principles ensure that honest indexers will calculate the correct state, there is still a possibility for malicious indexers to provide false information to end users. This raises the question of how users can verify the correctness of the state provided by an indexer. To address this issue, we propose a User State Verification mechanism.

The User State Verification mechanism allows users to independently verify the correctness of their account state without relying solely on a single indexer. This approach involves several key components:

1. **Merkle Proofs**: Indexers generate Merkle proofs for each account state. These proofs allow users to verify that their account state is included in the overall state root.

2. **State Root Publication**: The consensus nodes periodically publish the state root (e.g., the root of a Merkle Patricia Trie) on the blockchain. This serves as a trusted reference point for verification.

3. **Multi-Indexer Queries**: Users can query multiple independent indexers and compare the results. Any discrepancies would indicate potential malicious behavior.

4. **Fraud Proofs**: A system of fraud proofs can be implemented, allowing honest participants to challenge and disprove false state claims made by malicious indexers.

By implementing these mechanisms, users can maintain a high level of trust in the off-chain state computation system, even in the presence of potentially malicious indexers. While indexers may publish verification information, it is important to note that this information does not necessarily need to be placed on-chain. Instead, indexers can provide this data through off-chain channels, allowing for more flexible and efficient verification processes. This approach preserves the benefits of off-chain computation while ensuring the integrity and verifiability of the resulting state, without burdening the blockchain itself with additional data storage requirements.

6 Blockchain Complexity

The emergence of blockchain began with a simple task: sending transactions of coins. As demand grew, the complexity of rules relative to this task also increased, as shown in Fig 1. The advent of smart contract blockchains further increased the complexity of blockchain, as the computation and verification of blockchain transactions became more expensive.

The Minus Theory and related ideas first suggested reducing the complexity of blockchain to a simpler direction, rather than following the path of Bitcoin. From Bitcoin to Ethereum, and towards

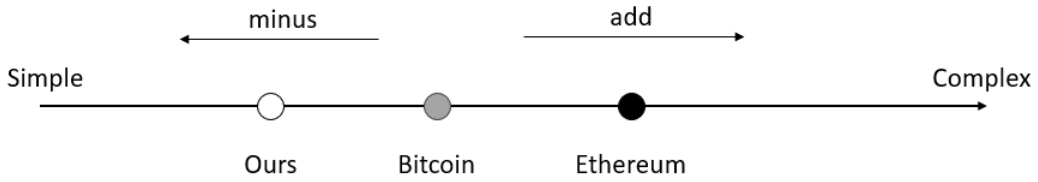

Figure 1: The complexity of blockchain computation and verification.

more blockchains with higher TPS, people have been trying to add more features to blockchain. While this enriches the chain's features, it also makes it harder to break the performance limitations.

Minus Theory advocates for a "less is more" approach to optimize blockchain performance. It proposes that by removing features and maintaining simplicity, blockchains can achieve higher performance levels. This counterintuitive strategy suggests that streamlining blockchain functionality, rather than adding complexity, is key to reaching enhanced efficiency and scalability.

6.1 Asynchronous State Computation

One of the key advantages of separating the indexer from the blockchain is the ability to perform asynchronous state computation. In some systems, this approach is called "delayed execution." In previous blockchain designs, state computation was tightly coupled with block production and validation, which could lead to performance bottlenecks. By moving state computation off-chain, we introduce a more flexible and scalable approach that allows for the delayed processing of state changes.

The asynchronous nature of off-chain state computation not only alleviates the performance pressure on the blockchain itself, but also makes building complex applications on blockchain possible. By moving resource-intensive state manipulations off-chain, developers can create more advanced and sophisticated applications without being constrained by the limitations of on-chain execution. This approach enables the development of complex systems that were previously impractical or impossible to implement directly on the blockchain, while still maintaining the security and decentralization benefits of the underlying blockchain infrastructure. Furthermore, this flexibility is crucial for scaling blockchain systems to meet growing demands without compromising the core consensus mechanism's efficiency.

This asynchronous approach to state computation introduces elasticity in computational time resources, addressing a significant limitation of on-chain virtual machines (VMs). In traditional blockchain systems, the VM's computational capacity is constrained by block time and gas limits, creating a rigid bottleneck. By moving state computation off-chain, we transform this fixed resource into an elastic one. Indexers can allocate more time and computational space as needed, allowing for complex operations that would be impractical on-chain. This elasticity enables the processing of more sophisticated transactions and larger data sets without being bound by the blockchain's inherent time constraints.

Moreover, this approach aligns with the Blockchain Minus Theory by effectively reducing the blockchain's responsibilities. The core blockchain can focus on its essential functions of maintaining consensus and ensuring security, while the elastic off-chain computation handles the heavy lifting of state updates. This separation not only enhances the efficiency of the blockchain's core infrastructure, but also provides scalability and versatility. As computational demands grow, the off-chain systems can adapt and expand more readily than on-chain solutions, paving the way for more robust and flexible blockchain ecosystems.

7 Conclusion

We summarize and propose the Blockchain Minus Theory as an alternative approach to optimize blockchain performance. Instead of adding more features or parallelizing the system, we suggest removing features to reduce blockchain duty. This theory, rooted in the long-standing idea of off-chain computation, demonstrates how simplifying the blockchain's responsibilities can lead to significant performance improvements. By removing certain verification processes from the consensus mechanism, we show how this approach can effectively address the blockchain performance bottleneck.

Inscription technology was initially regarded as a hack for blockchains without smart contracts to build asset distribution platforms last year. However, the idea behind such technology might lead the blockchain to achieve better performance towards massive adoption. The concept guides us to create thin blockchains focused on consensus with low hardware requirements, and move the smart contract features off-chain. This approach may free the blockchain from unnecessary burdens and unlock valuable VM computation resources.

References

[1] https://docs.counterparty.io/docs/basics/what-is-counterparty/a-bitcoin-protocol/. How does Counterparty work?