Programmable Consensus

The author
BUPT

***Abstract*—This paper conducts a holistic reevaluation of blockchain usability, examining it through the lens of a six-stage consensus lifecycle and introducing a three-dimensional framework for design principles. Building on these insights, we introduce the concept of Programmable Consensus. This approach moves away from the traditional order-execute framework and offers blockchain users programmable interfaces across the entire consensus lifecycle, fully unleashing the potential of blockchain.**

I. Introduction

Blockchain systems have gone beyond its cryptocurrency origins, transitioning into broad decentralized platforms that build trust for diverse applications. Protocol changes in both the blockchain data and consensus structure have been made to support the execution of Turing-complete codes and facilitate the deployment of complex logics based on smart contracts. In both proof-based and BFT-based blockchains, transaction execution follows the same manner, during which blockchain nodes reach agreement through consensus and realize operation through the execution of smart contract.

While the common requirement of a blockchain system is to ensure integrity, authenticity, and security without the third party, needs vary in different scenarios. Efforts have been made to enhance the versatility of the blockchain to meet diverse needs across various fields. Basically, these improvement can be examined in terms of the Validation Phase and the Execution Phase.

A. Validation Phase of Consensus

With respect to accessing the blockchain network, there are two main modes of operation: permissionless and permissioned blockchains.

Proof-based consensus are designed for permissionless blockchain to establish agreement in a probabilistic manner, mainly to prevent Sybil Attack and Double-Spending Attack. Although proof-based consensus has excellent node scalability, they can be resource demanding and suffers from long transaction confirmation latency and low throughput. BFT-based consensus work well in permissioned blockchain, where the identity of nodes is known and manageable. It enjoys resources efficiency and faster transaction processing that align with the nature of permissioned networks. However, BFT-based consensus faces scalability concerns since the intensive communication requirements surge exponentially with the number of nodes.

Identify applicable funding agency here.

To address the above drawbacks, some works involves sharding and function segmentation to reduce the communication complexity. Meanwhile, to address the bottleneck caused by sequential process of block validation and blockchain attachment, several efforts have focused parallel and asynchronous execution to enhance overall throughput. This necessitates resolving conflicts in data sharing across concurrent transactions and smart contracts.

B. Execution Phase of Consensus

As programs running on blockchain systems, smart contracts provide tremendous opportunities for use in many fields that rely on data-driven transactions. The majority of work have been done to improve the performance of smart contract execution. The incorporation of data structures with higher read and write speeds to address the issues of resource consumption and latency during processing. That is, replacing the Merkle Tree with the list structure to reduce the overhead of data reading.

Also, some blockchain systems utilize containers as sandboxed environments for smart contract execution, where the execution output does not update the state immediately. It separates smart contract execution from state management, decoupling the traditional consensus procedure. This approach essentially proposes the two-step order-execute model that allows for the separates discussions of verification and execution of transactions. Some systems, like Hyperledger Fabric, further evolve this model into an execute-order-validate framework, which, however, only consider the non-consensus nodes and offers minute improvement.

C. Motivation and Contribution

We see a progressive decoupling of the validation phase and execution phase, which are so-called consensus (procedure that nodes reach consensus on a certain amount of data through redundant validation) and smart contract (procedure to perform the designated functions). However, a holistic evaluation of how improvement of each aspect contributes to overall usability remain inadequately explored.

In order to better evaluate the usability of a blockchain, we treat the validation phase and execution phase as a whole, both are stages of a Consensus Lifecycle. A Consensus Lifecycle describes the complete sequence of stages that a single action undergoes.

We have also observed that adaptability, a crucial factor, is often overlooked in many blockchain-related studies and developments. Typically, when evaluating the adaptability of a blockchain system, the focus is primarily on the functionalities

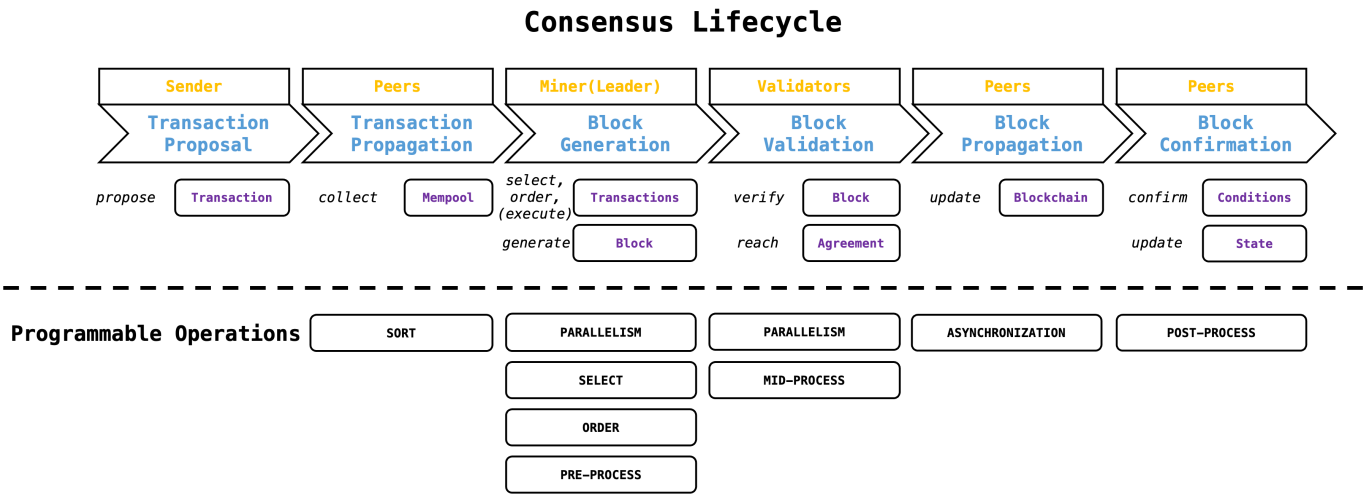

Fig. 1. Rethinking & Reaccessing

enabled by smart contracts. However, upon examining the entire Consensus Lifecycle, it becomes evident that various stages offer opportunities for customization to meet specific requirements. Present modifications are predominantly limited to parameter adjustments.

To solve the insufficient transplantation, we provide the programmable consensus, which enable blockchain to incorporate and function more seamlessly in alignment with specific scenarios, ensuring a more effective and tailored employment.

Thus, our contributions can be concluded as follows:

- We Rethink the relationship between Consensus and Smart Contract, gaining some insights. To better design the blockchain, we view the consensus and smart contract as a unified whole-the validation phase and execution phase of the entire Consensus Lifecycle. Therefore, we propose a six-step journey of transaction to elucidate the possible design principles.
- We Reassess the usability of a blockchain through three-dimension design metrics: Flexibility, Extension, Efficiency. These metrics help to assess the effectiveness of proposed design principles.
- We Redesign the Consensus Lifecycle holistically, naming it Programmable Consensus. Programmable Consensus provides users with interfaces among the Consensus Lifecycle, making many key components programmable. This enhancement greatly increases the adaptability of the blockchain in specific scenarios.

II. Holistic Review

A. Rethinking: Evaluation of Blockchain

To elucidate the Consensus Lifecycle, we rethink from the entire journey of a transaction: 6 crucial steps and Y key factors.

STAGE 1: **Transaction Proposal**: Node create a transaction.

STAGE 2: **Transaction Propagation**: Transactions are propagated to and collected by the validating peers.

STAGE 3: **Block Generation**: Leader node of this round of consensus select the collected transactions and generate a block according to the preset roles. Some system, like all the permissionless blockchain, would execute the smart contract and generate a receipt at the procedure.

STAGE 4: **Block Validation**: Undergo different stages of consensus mechanisms, all the consensus nodes would reach an agreement on the block. Afterwards, the block is regarded valid and executable. Block validation is a crucial stage that distinguishes blockchain from other distributed systems; it involves reaching unanimity on specific data through redundant computation.

STAGE 5: **Block Propagation**: The valid block is propagated to update the replica of other nodes.

STAGE 6: **Block Confirmation**: When the valid block is updated as the latest version and used as the alignment of the whole network. With this, the consensus lifecycle comes to an end.

Subsequently, considering the enhancement of usability, possible modifications to the crucial steps are conducted.

- In Step 2, transactions could be collected by designated transaction pools, filtered into different subsequent stages, which is called SORT operation.
- In Step 3, which is a crucial step, transactions are assembled into a block, where exists SELECT, ORDER, and PRE-PROCESS operations.
- In Step 4, which is a crucial step, block is validated through redundant computation. Here exist PARALLEL operation that use sharded and concurrent concepts. Also, to validation the transactions in the block, MID-PROCESS operation is appliable.
- In Step 5, block is attached to the blockchain, forming the latest version of the blockchain and diffused among the network to let peer nodes update their replica of

blockchain. Some works decouple this chaining procedure with previous block validation one, using another round of validation to eventually generation the blockchain, which could be summarized as ASYNCHRONIZATION operation of chaining.

- In Step 6, nodes confirm the update of the blockchain and update their ledger replica, eventually arrive to the end of the lifecycle. In certain system which decouple the smart contract execution from the validation steps, here exist POST-PROCESS operation, which basically is the execution phase mentioned in Character one.

B. Reaccessing: Programmable Consensus

III. Overview

A. Design Philosophy

B. Generalized Structure

IV. Main functions and procedures

A. Proof-based programmable consensus

B. BFT-based programmable consensus

V. Implementation & Experiment

VI. Conclusion