Whitepaper

Building the Internet of Value for the future.

Seele is a blockchain 4.0 platform that aims to spearhead development in the Internet of Value era. We aim to solve the scalability, security, and efficiency problems found in current blockchain networks by developing a heterogeneous, multi-chain ecosystem powered by a novel Neural Consensus Algorithm.

seele.pro

Table of Contents

0. Abstract

In recent years, blockchain systems and their applications have expanded the boundaries of the crypto space and resulted in deeper decentralized application development. Bitcoin, Ethereum, Hyperledger Fabric, Corda, and other public and consortium chains have continued to emerge, creating a highly competitive environment.

With the development of blockchain technology and its applications, various problems have gradually been exposed. Issues such as the inability to scale for large-scale performance, the inability to support diverse business use cases, and the inability to exchange information and share assets across different blockchain networks have become more prominent.

In response to these problems, we are going back to the core values of blockchain technology. We want to solve blockchain's current problems by improving some of the base-level issues – consensus algorithms, ecological topologies, cross-chain agreements, underlying network communication protocols, collaborative convergence computing, application ecosystems, and more, to promote the wider application of blockchain and Internet of Value technology.

We propose the following major technological solutions:

- A novel Neural Consensus Algorithm to improve fault tolerance from 33% to 40% without any loss of performance compared to Byzantine Agreement (BA) algorithms.
- A Heterogeneous Forest Network (HFN) architecture with high scalability for use in a wide variety of application scenarios. This network will be secure, include resource isolation, and can be customizable for any demand.
- Value Transport Protocol (VTP) and Value HTTP (VHTTP) to aid in the naming, discovery, and addressing of Value Internet assets and entities. These seamlessly integrate with internet resources to build underlying protocols and infrastructure services for blockchain ecosystems.
- A TCP/UDP-based low-latency Quick Value Internet Connection (QVIC) protocol to better adapt to and meet the requirements of blockchain network interactions compared to the traditional internet TCP/UDP protocols used in current networks. Advantages include the ability to handle more connections, higher security, and lower latency, especially in the transmission of packets of a specific block size (1M, 2M). Compared with UDP, transmission efficiency can be improved nearly 1 order of magnitude.

1. Our Name

Seele is "soul" in German, which refers not only to the human soul itself but also to the core of an idea or action. The name Seele implies our own goal – to innovate and develop the infrastructure behind the upcoming Internet of Value era.

2. Mission and Goals

- To promote the development of protocols, networks, and communities in the Internet of Value era.
- To integrate the idea of the layered internet protocol suite model into blockchain and to provide interoperability of resources and collaborative computing in heterogeneous blockchain networks.

- To provide a powerful, foundational platform for the free and safe circulation of resources within and across blockchains.
- To create a collaborative, win-win developer community around Seele. To provide developers with integrated, flexible technologies that allow them to deliver value within this community.
- To use the token economy to encourage community contributions to create and share within Seele's decentralized ecosystem.
- To promote the overall development of blockchain technology and spearhead technological advancement in the Internet of Value era.

3. Design Principles

- Open System Architecture: meet the requirements of portability, customizability, and interoperability in our fundamental protocols, functional frameworks and upper-level applications.
- Efficiency First: provide an efficient and stable business platform with fast and accurate value transfer.
- Dynamic Expansion: provide a blockchain platform and network structure that can dynamically expand and adjust to fulfill the computing and storage needs for various business requirements.
- Resource Isolation: adopting a partitioned, hierarchical, multi-chain model to blockchain to avoid multi-service interference.
- Experience First: provide full-scale development and service support for users through standardized communication protocols, module interface specifications, SDKs and IDEs, community support, developer conferences, industry application associations, and more.

As pioneers and practitioners of the next generation of blockchain, we will work to create a value ecosystem with our partners by combining heterogeneous forests, neural network consensus algorithms, computational power sharing between inside and outside chains, and scalable transactions per second (TPS).

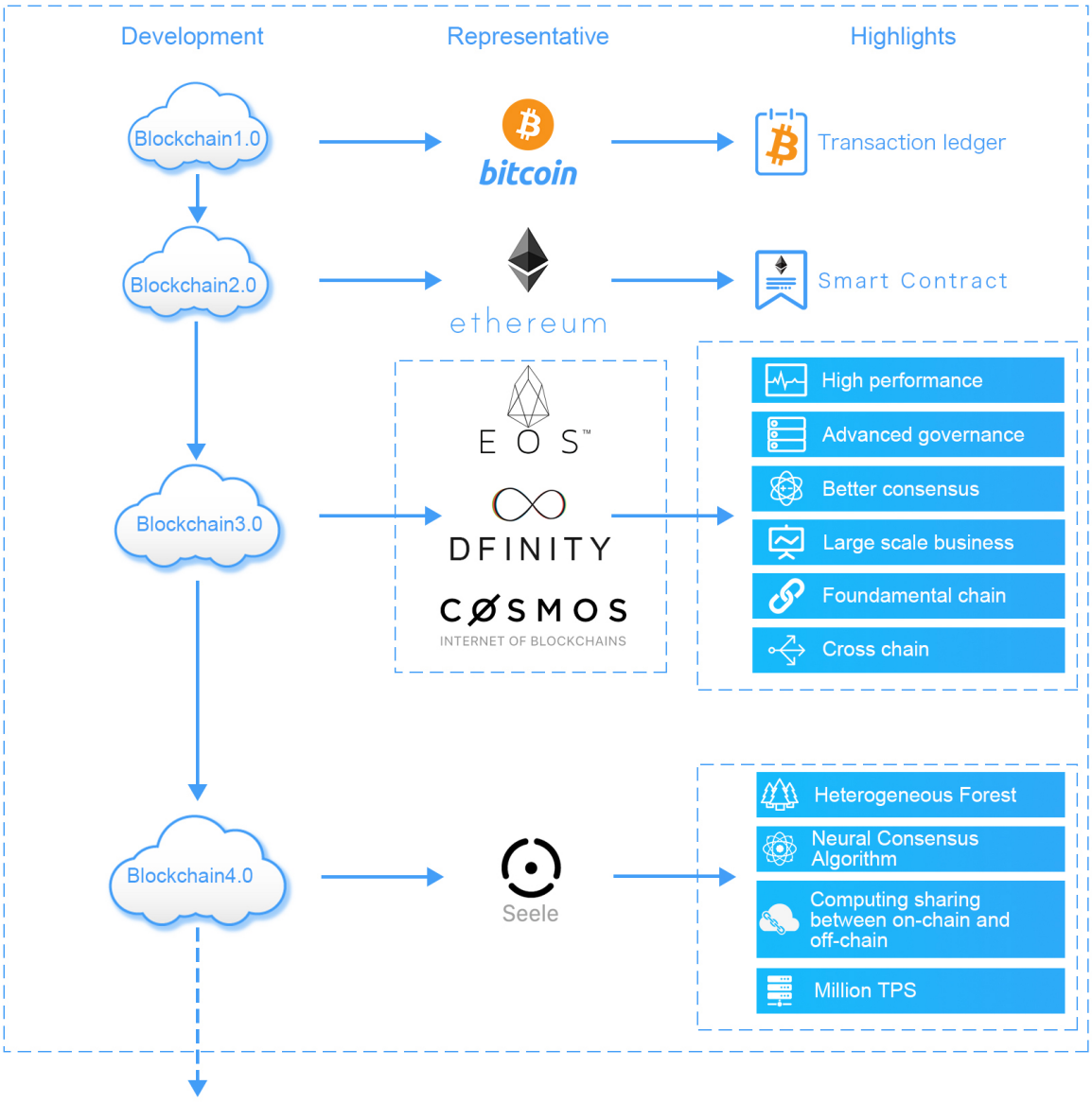


Figure 3.1: Seele, Blockchain 4.0

4. Neural Consensus Algorithm

4.1 Background

Current blockchain consensus algorithms are not scalable, secure, and efficient, forming the SSE problem. For example, proof of work (PoW) lacks efficiency due to its inherently high computational overhead. Proof of stake (PoS) lacks security, for example, by being at risk of the possibility of "nothing-at-stake" attacks. Delegated proof of stake also lacks security, by utilizing only a small number of decision-making delegates. Practical byzantine fault tolerance (PBFT) lacks scalability, as network overhead increases rapidly with each additional node. Hashgraph is scalable and efficient but has relatively large confirmation delays due to its structural and procedural constraints, such as the

connectivity and partition of network nodes. Lastly, Algorand is scalable, secure, and efficient, but has strict requirements on the overall connectivity of the network, so the time it takes to reach consensus becomes unpredictable in the case of a network partition.

4.2 Introduction

Seele synthesizes the advantages and disadvantages of current mainstream consensus algorithms and proposes a Neural Consensus Algorithm based on ε-differential agreements (EDAs), with the goal of solving the SSE problem.

Neural Consensus uses EDA to converge a network upon a singular, correct value. This value could be a specific transaction order, block order, or any other metric that a decentralized network would need to agree upon. This large-scale convergence process is fully asynchronous, and each node executes its own part separately. When all nodes' values are within the micro-real number ε-differential range, the network is considered to be in consensus.

The time it takes for the network to reach consensus decreases linearly with the number of nodes, making Neural Consensus a remarkably scalable algorithm. In a 100,000-node network environment, transactions per second (TPS) reached 100,000 and the transaction confirmation time decreased to just a few seconds.

Furthermore, EDA is very robust in terms of the overall connectivity of the network, it can function even if the network connection is less than 50%. It is also able to handle up to 40+% malicious or failed nodes.

4.3 Overview

To reach consensus, nodes conduct multiple rounds of sampling. In each round of voting, every node randomly samples values from a certain percentage of the other nodes in the network. Each node computes a statistic (median, mean, etc.) from their sample. This statistic is each node's value, or vote. After multiple rounds of sampling and voting, the values of all the nodes in the network will converge towards consensus. This procedure is similar to how the network of neurons in the human brain reaches consensus. Notably, the discrete voting $[T|F]$ of traditional consensus agreements is replaced by continuous values [0%, 100%] for all nodes.

The network is considered to be in consensus when all nodes' voting values are within the range of the preset threshold ε. At this point, the network is considered statistically consistent with a singular correct value. As the number of nodes in the network increases, the number of rounds necessary to reach this consensus decreases.

It is important to note that a node can vote on and sample multiple different items in parallel, so that the network can concurrently reach consensus on multiple items. This provides an effective and practical method to greatly improve the efficiency of the system.

This algorithm is secure, scalable, and efficient, resolving many of the issues found in traditional consensus algorithms. This makes it applicable in a wide variety of use cases, such as finance, healthcare, and trade.

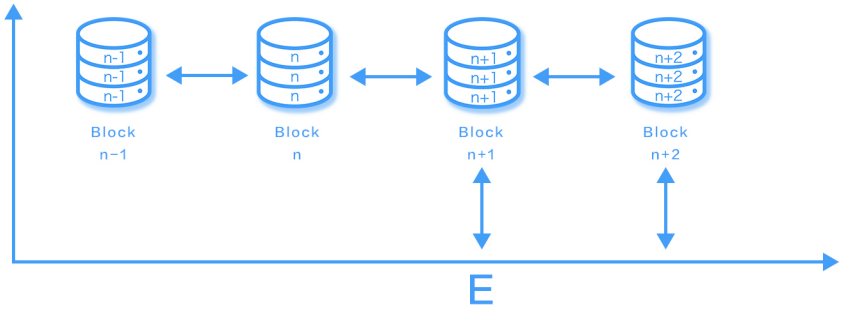


Figure 4.1: Consensus Model

4.4 Highlights

- Consensus process changed from discrete voting to continuous voting

In traditional consensus algorithms, nodes vote for or against blocks using 1 or 0 respectively. In neural consensus, each node can vote continuously along some fixed spectrum.

- Energy savings

Unlike PBFT, no primary node is selected. Furthermore, there is no PoW or PoS required. This greatly reduces energy consumption during operation. Note: Although Neural Consensus requires floating-point arithmetic, the rest of the operations are based on integer arithmetic which requires extremely low performance overhead. This further reduces energy costs.

- Low transmission overhead

Unlike PBFT, every node does not need to connect with most other nodes during the consensus process. This reduces data transmission overhead and reduces nodes' dependence on the network structure as much as possible.

- Adjustable parameters for different use cases

The efficiency of this algorithm is based on several different adjustable parameters, including the convergence interval ε and the sampling rate s. Optimal system efficiency can be obtained by adjusting the relevant parameters in real time.

- Compatible with a variety of network structures

This consensus algorithm can be adapted to many different network structures, including the traditional chain structure and the directed acyclic graph (DAG) structure.

4.5 Security Analysis

To prevent Sybil attacks, users will be assigned weights depending on the amount of currency they hold. A node's likelihood of being sampled is based on its weight. This avoids Sybil attacks as long as the total combined token value of all Sybil nodes is less than half of the total network token amount. Furthermore, the procedure of the neural consensus algorithm avoids forks and hence double spend attacks.

For the random sampling of nodes in EDA, a random computable function is used. Users calculate from their private key whether or not they have been selected and broadcast this result to other users. This random selection process is non-interactive, attackers cannot know in advance which nodes have been

selected. The random selection process leads to different nodes being selected during each round, also increasing the cost of an attack.

4.6 Experimental Results

Throughout these experiments, we compare EDA to the PBFT algorithm. 2,000 Amazon EMC cloud nodes were used in the testing and ε was set to <0.001%.

4.6.1 Influence of Node Failure on the Consensus Process

Three different tests were performed:
A: 10% (200) failed nodes
B: 33% (660) failed nodes
C: 40% (800) failed nodes

PBFT fails to operate when the total number of failed nodes (including malicious nodes) is more than 33%, meaning it does not work for test C. For every 10% increase in failed nodes, we increase the sampling percentage s by 20.

For test A, it takes up to 6 votes to complete consensus and converge upon an order.

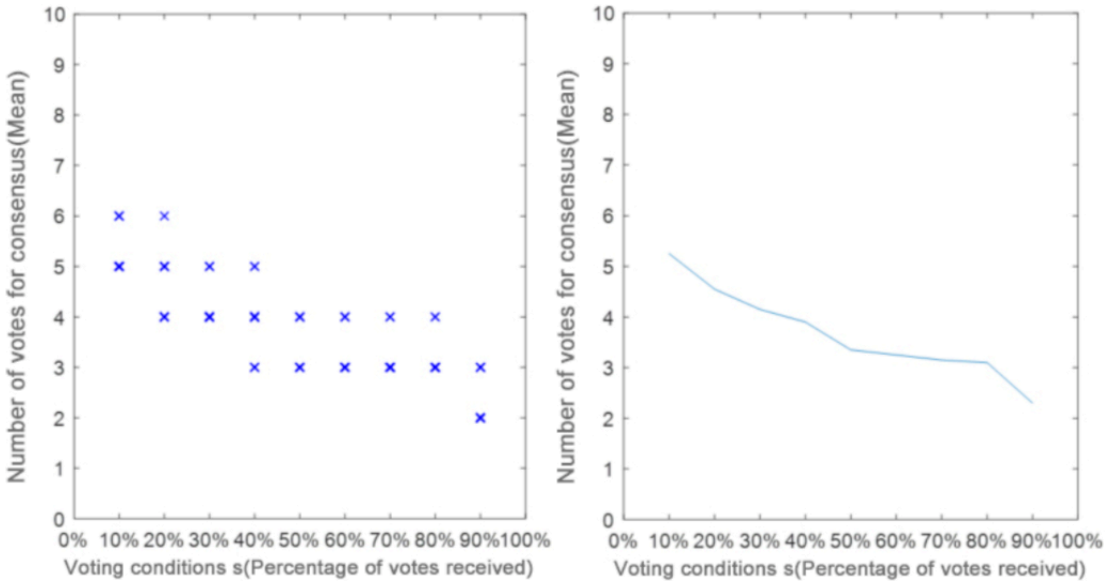


Figure 4.2: Effect of sampling rate s on the votes required to reach consensus (accounting for 10% faulty nodes).

For test B, it takes up to 7 votes to complete consensus and converge upon an order.

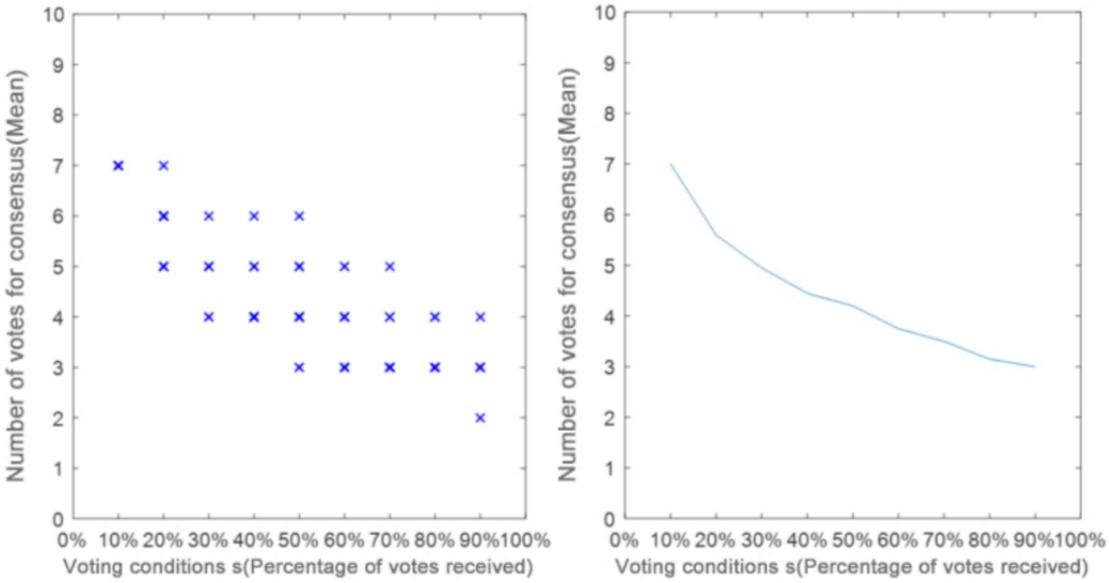


Figure 4.2: Effect of sampling rate s on the votes required to reach consensus (accounting for 33% faulty nodes).

For test C, when s is above 20%, it takes up to 8 votes to complete consensus and determine the order. PBFT could not complete consensus in this environment because the number of failed nodes is higher than 33%.

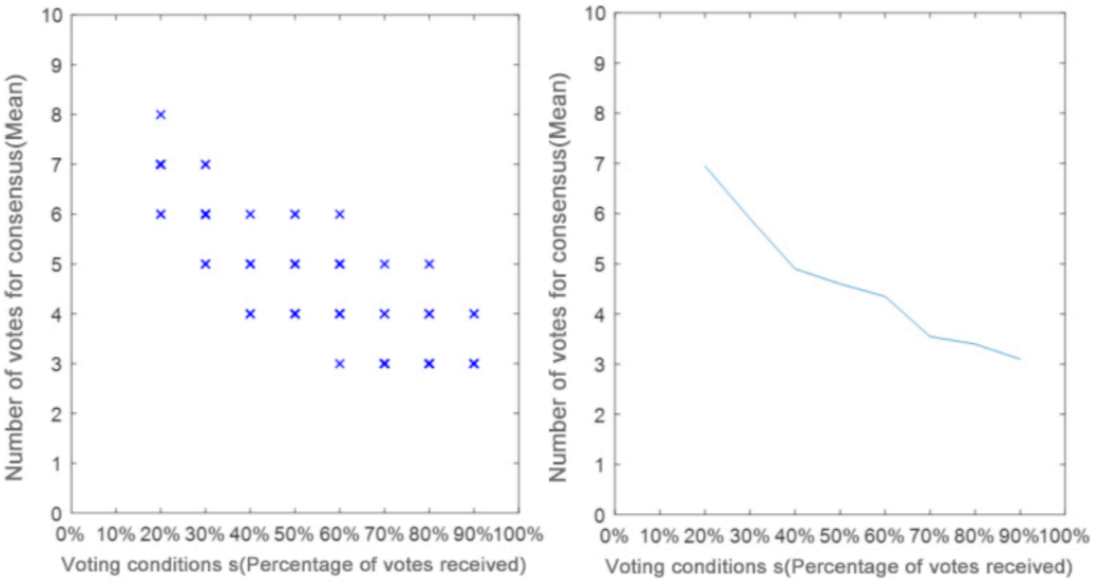


Figure 4.4: Effect of sampling rate s on the votes require to reach consensus (accounting for 40% faulty nodes).

4.6.2 Overall Coverage of Multiple Sampling

This test consists of a large-scale node environment (N = 10,000) and a sample rate of 3% (n = 300). At round r, each node has access to every other node's results from round r - 2. In other words, it takes 2

rounds for every node in the network to directly or indirectly become aware of every other node's value at a specific round. This phenomenon was also noticed in a medium-scale node environment (N = 1,000) with a sampling rate was 3% (n = 30).

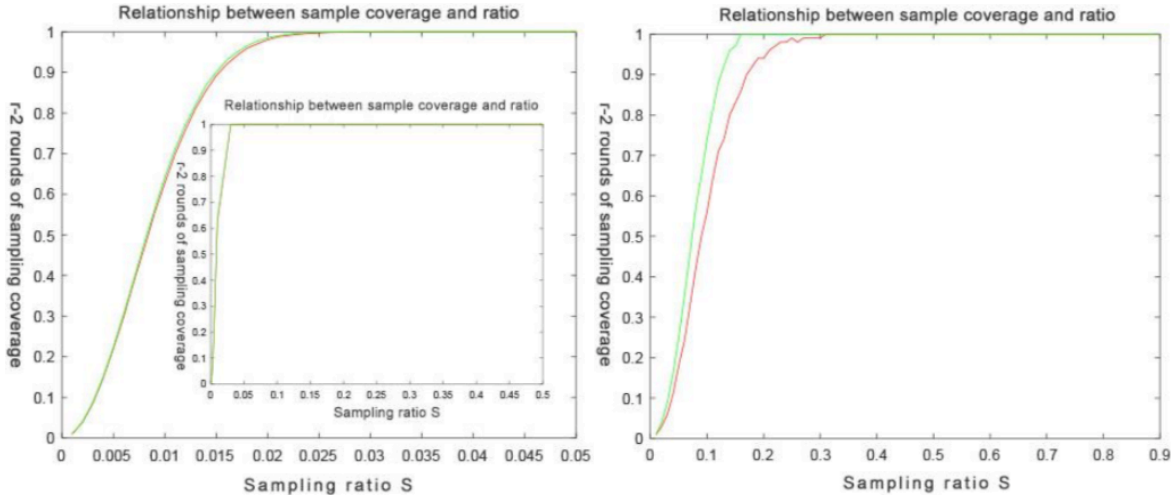


Figure 4.5: Overall Coverage of Multiple Sampling

4.6.3 Number of Communications for Sampling

Under the large-scale condition, EDA transmission has an obvious effect on bandwidth usage – there are significant savings in the number of transmissions required for voting. In the small-scale network condition, EDA is not much different from PBFT. EDA is superior to PBFT when r*s < 2. In small-scale networks, bandwidth consumption is negligible.

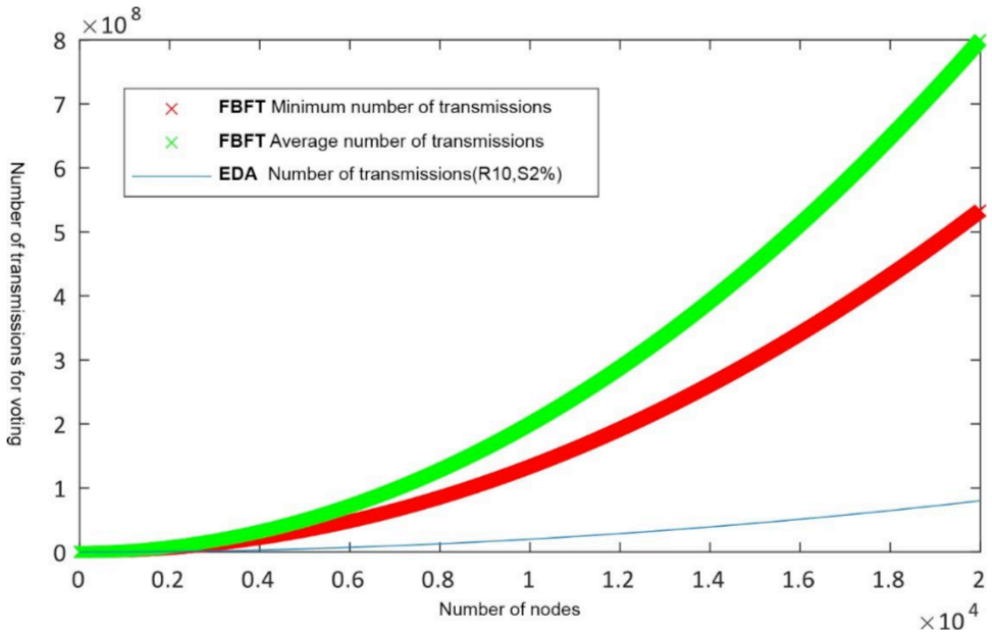


Figure 4.6: Comparison of the number of communications.

4.6.4 Scalability

With 100,000 nodes and s = 0.9%, two rounds of EDA cover 100% of voting components. This is the same as PBFT.

With node size equal to 100K, s = 0.9%, and r <= 10, the number of transmissions is far less than PBFT. In PBFT, parallel voting produces meaningless results. In EDA, parallel voting results can be used directly to achieve consensus.

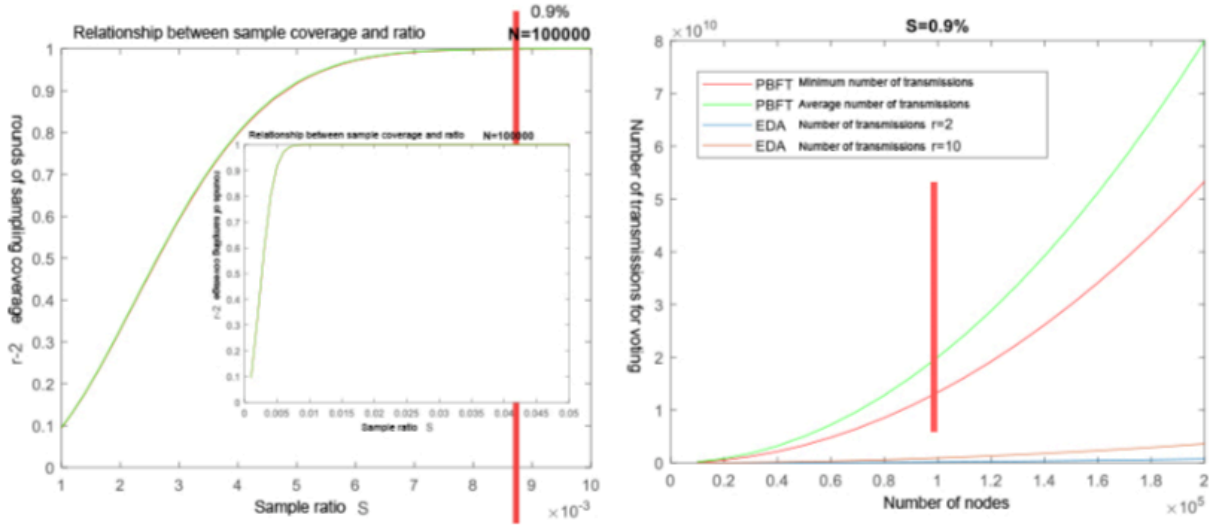


Figure 4.7: Comparison of scalability.

4.6.5 Fault Tolerance

In PBFT, blocking occurs when the failure ratio of nodes exceeds 33%. EDA allows the nodes to continue voting until the failed node(s) return to normal.

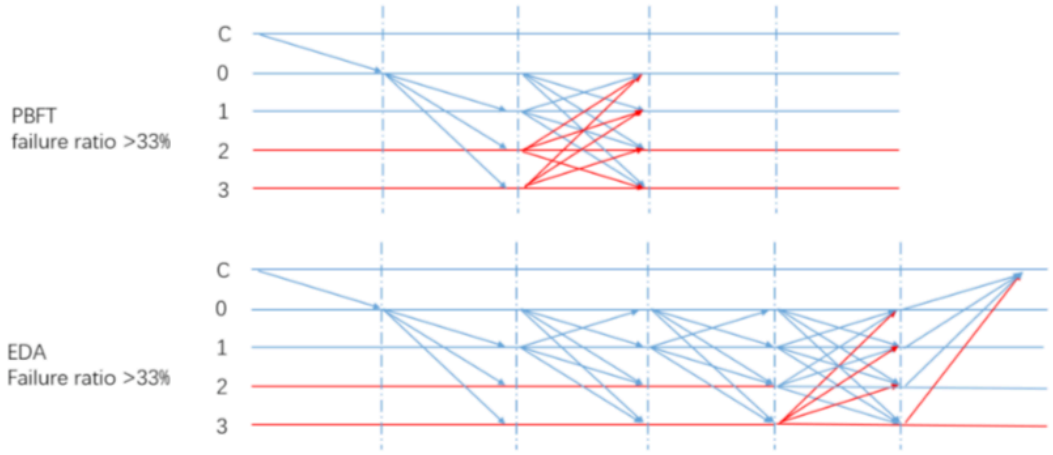


Figure 4.8: Comparison of the fault tolerance between PBFT and EDA.

EDA in a small-scale environment:

If we set s = 100% and ε = 100%, we lose the ability to parallel sort and degrade to typical PBFT.
If we set s = 100% and ε < 100%, we keep the ability to parallel sort.
If we set s = 100% and ε = 0%, we reduce fault tolerance and allow parallel sorting.

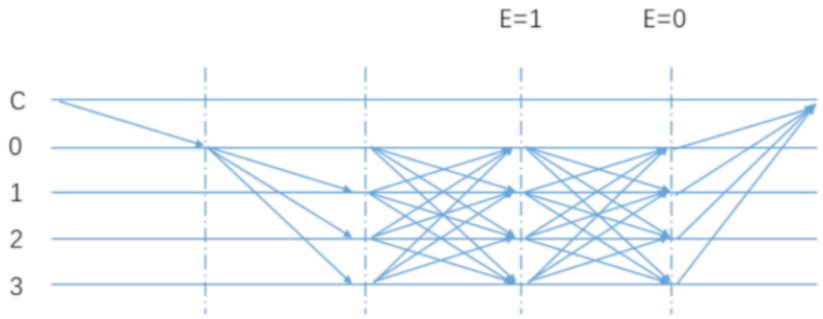


Figure 4.9: Comparison of the failure rate between PBFT and EDA.

4.6.6 Conclusion

From the experimental results above, it can be concluded that with an s above 20% and a network with under 40% node failure, EDA is able to achieve overall system convergence. The higher the value of s, the fewer the number of rounds required.

5. Heterogeneous Forest Network

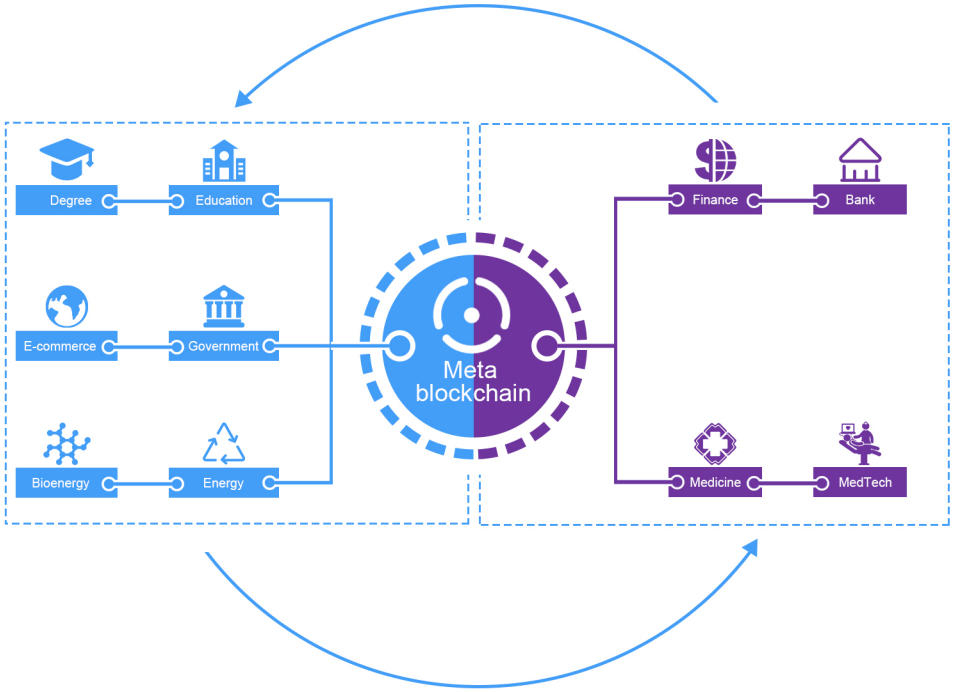


Figure 5.1: Heterogeneous Forest Network

5.1 Introduction

Human society has experienced massive development over time – from the primitive single-tribe model, to the multi-tribe model, to cultural groups, to institutional countries. The development of the internet has experienced similar massive growth – from the stand-alone era, to multi-machine interconnectivity, to multi-machine LAN, to heterogeneous LAN interconnectivity, and finally to PC internet, mobile internet, and the Internet of Things. The development of blockchain is no different – starting with Bitcoin, growing into smart contracts, and expanding into today's booming variety of blockchain products.

5.2 Single Blockchain Structure

Traditional blockchain networks, such as Bitcoin and Ethereum, are all single-chain structures. Their transactions take place on one chain.

The main advantage of a single-chain structure is that the transaction and consensus process is relatively simple. It provides an easy way to meet all the user requirements. However, with the development of more complex blockchain technology and the increasingly varied market demand for its services, the single-chain architecture begins to reveal its weak points:

- There are bottlenecks in throughput and performance. Bitcoin can only execute around 7 TPS and users need to wait several blocks (6 are recommended) in order to ensure a transaction remains on the chain. Ethereum experiences similar issues, taking 10-20 seconds to produce a new block. These severely hamper the growing demand for blockchain services.
- Applications built on the same chain interfere with each other. Single-chain systems can easily be overwhelmed by a single, busy application. For example, the popular game CryptoKitties crowded out the entire Ethereum network, delaying normal Ether transaction processing and confirmation.
- Single-chain networks have a closed structure – there is no interaction between different chains and they cannot meet the needs of businesses that require interaction between multiple platforms.

5.3 Multiple Blockchain Structure

In order to overcome the previously-stated limitations of single-chain structures, multi-chain structures have been proposed. Proposed multi-chain structures have taken many varieties, mainly in the form of multiple parallel chains or main/side chain structures. These partially meet the diversified needs of businesses. However, there are still some shortcomings in terms of flexibility and customization.

- With multiple parallel chains, each chain's function is generally preconfigured. This makes it difficult to meet the rapidly changing and diversifying business needs of blockchain. This kind of structure also does not have fully developed methods of sharing computing and data resources across chains.
- In the main/side chain architecture, different side chains can be created in accordance with the growth of the network and the introduction of new use cases. However, main chain consensus is often closely coupled with main chain consensus, leading the main chain to potentially become a bottleneck.

5.4 Forest Blockchain Structure

In the traditional internet, we can open a browser, input a website name, enter the site, and navigate through its various resources. The Domain Name System (DNS), one of the basic internet protocols, has made a tremendous contribution to our ability to easily search and navigate the web in this way.

The Internet of Value built by blockchain will consist of a huge network of clusters around the world. It will consist of multiple chains, each with its own individual use cases and functions. There will be a large number of cross-chain requests, and humans will need to interact with this technology in an easy-to-use way. Drawing on the success of DNS and the necessity of multiple-blockchain structures, we propose a heterogeneous forest network architecture that enables the definition, storage, transfer, and transformation of Internet of Value resources and assets to promote the integration of Internet of Value services with the traditional internet.

The heterogeneous forest network consists of different subnets. Each subnet can be seen as a hierarchical tree – the top of the tree is a global service chain, called the Meta Chain, which provides global configuration and scheduling services. From top to bottom, each chain has various business uses. The tree is divided according to these business uses, as well as isolation requirements and various performance costs. The upper layer of the network provides addressing and scheduling services to the lower layers. Each layer can have its own independent governing mechanisms, such as access rights, flow control, and security mechanisms. Each layer is its own small ecosystem; the combination of these small ecosystems constitutes the larger heterogeneous forest network.

Due to the wide range of blockchain use-cases in the real world, it is difficult for a single-chain structure to be comprehensive. In a heterogeneous forest network, each chain only serves a small subset of functionality, and each separate business use case runs on a separate chain. This not only provides effective security isolation, but also helps the network more efficiently use computing resources.

The heterogeneous forest network structure can meet many different types of complex business needs in the real world. Use cases with different characteristics can run in different sub-chains – for example, a computation-intensive project can run separately from an IO-intensive project. Use cases with varying security requirements can run on different, customizable chains to maximize efficiency and security. High-security use cases such as banking can be isolated in more secure layers of the network.

6. Value Transport Protocol

6.1 Internet Protocol

The widespread adoption and success of the traditional internet is partially due to its full set of normative protocols. The internet protocol suite connects the equipment in the network, unifies resource identification, and makes the exchange of internet assets convenient and straightforward. Parts of the internet protocol suite include:

- Internet Protocol (IP): Any device and software can seamlessly access the internet and share resources if it complies with and implements IP.
- Uniform Resource Identifier (URI): URI uses strings of characters to uniquely identify internet resources such as pictures, text, and video clips. This makes it significantly easier for human users to interact with these resources. URLs are a form of URI.

6.2 Current Status

Current blockchain networks cannot share data and communicate with each other, forming independent blockchain islands like Bitcoin and Ethereum. These separated ecosystems differ in terms of data models, interactive protocols, and code – leaving them incompatible.
Current Bitcoin and Ethereum addresses look like this:

33YV5wC11kF67AuGSwTpUSpDTHBPTS4qDh

Strings like this are easy for machines and programs to handle but are unfriendly for human cognition and memory. This greatly limits the user value of blockchain networks. Although the ENS service based on Ethereum provides a DNS-like naming service, it is still lacking in terms of efficiency and coverage.

6.3 VTP

Transmission between blockchain networks cannot be effectively carried out. In order to solve this, we propose the Value Transport Protocol (VTP) for use in the heterogeneous forest network architecture. VTP covers the uniform identification of assets on the chain and the routing strategy for asset discovery. It is a full set of transport protocols for the blockchain value network.

6.3.1 Naming Mechanism

In blockchain networks, assets are the pieces of data on the chain. The unique identification and naming of each asset on a chain is of great importance to that asset's registration, discovery, and transfer.

We define the Uniform Asset Identifier (UAI) to work with the VTP protocol. Hierarchical, structured naming of assets is helpful for people's cognition and memory while retaining uniqueness and scalability.

An example would be:

CHAIN://edu.pku.cs/account/data

In this example, 'CHAIN://' is the default protocol header, 'edu', 'pku', and 'cs' are the chain identifications at all levels, 'account' is the account on the chain (or contract), and 'data' is some piece information, like an account balance, note, or even a contract interface. In the heterogeneous forest network, different namespaces are used between chains, and the same namespace is used with parent chains. This facilitates the addressing and routing of content through parent-child relationships.

6.3.2 Content Addressing

Each chain provides sub-chain address lookup services, which are implemented by the system contract and initialized when building the chain. When a new sub-chain is added, the sub-chain sends a registration request to the parent chain, and the parent chain records the sub-chain address. The "Meta Chain" is a global configuration chain; it manages the entry addresses of the entire forest network.

When looking for specific assets using UAI, the first step is to find the entry in the Meta Chain. Then, go down to the desired sub-chain, account, and data field to find the assets.

The Meta Chain will not become a performance bottleneck, because routing information to each destination sub-chain can be cached.

6.3.3 Route Cache

In order to ensure efficient network utilization, access efficiency, data availability, and overall upper-level utilization, we will use a routing cache mechanism. On each chain, the built-in system contract manages route caching and is initialized when the chain is built. We have several main strategies for cache replacement:

- Last interviewed time interval
- Access frequency
- Last visit time and frequency of access
- Random replacement

If cache routing fails, the cache is cleaned up immediately. When a new sub-chain joins the heterogeneous forest network, its information must be registered with the meta chain and its route is added.

6.3.4 VHTTP

To facilitate easy cross-chain access for upper-level applications, Seele proposes a new Value HTTP (VHTTP) protocol for the Internet of Value. This protocol implements the exchange of values between and onto chains. VHTTP is compatible with HTTP and can recognize the format of HTTP request packets, so users outside the chain can access assets and data directly through HTTP protocol. For HTTP requests coming into the blockchain network, mappings between methods are automatically established. A VHTTP request consists of three parts – a request header, a message header, and a body. The request header begins with the name of the method and is followed by the requested address and version of the asset identified with UAI. This is the complete format:

Method UAI Version CRLF

Request method types are as follows:

GET: Request to obtain the UAI resource information
POST: Create assets (store assets on the chain)
TRANSFER: Transfer assets between two UAIs

7. Quick Value Internet Connection

7.1 Introduction

Blockchain nodes are located world-wide, leading to a complex network environment. Network jitter and latency can seriously affect the performance of consensus algorithms and the synchronization of blocks between nodes. We propose the Quick Value Internet Connection (QVIC) protocol to better adapt to the needs of a blockchain value network at the transport and application layers compared to traditional TCP and UDP. QVIC's low latency has obvious advantages, and for certain specifically-optimized block sizes (1M, 2M), QVIC improves transmission efficiency by nearly 1 order of magnitude.

7.2 Technical Advantages

- Using a non-transparent proxy mode, the client connects to a nearby server and the connection is switched to QVIC.
- Data remains safe from source to destination without caching.
- UDP packets transferred between servers are encoded to increase security.
- Load balancing can be used to improve robustness.
- High tolerance for packet loss.

7.3 Framework

QVIC has been optimized for wide area networks with network jitter, packet loss, and other unstable characteristics. It not only retains the fast and efficient features of UDP but also provides the integrity of TCP data transmission.

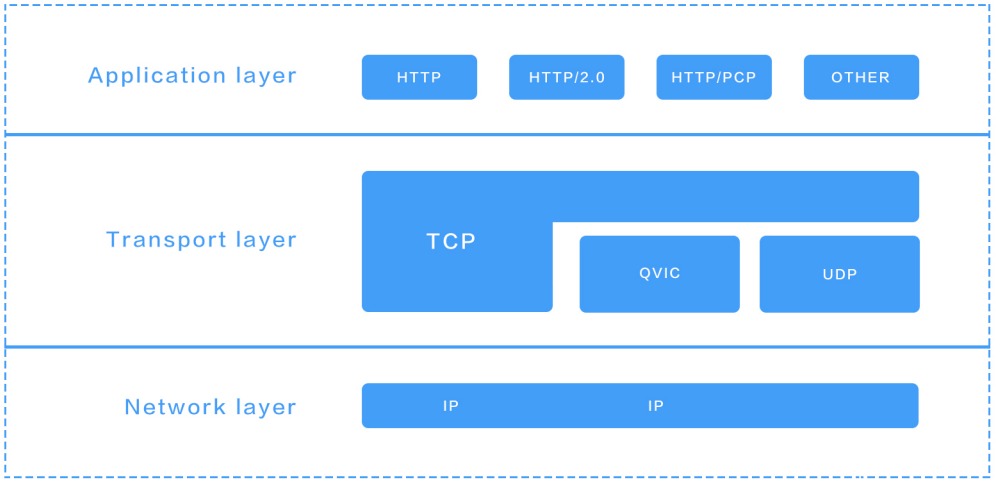


Figure 7.1: Framework of QVIC

QVIC utilizes handshake control done at the sending end and a pre-connection mode to improve the transfer rate of data packets and make the transmission process more efficient.

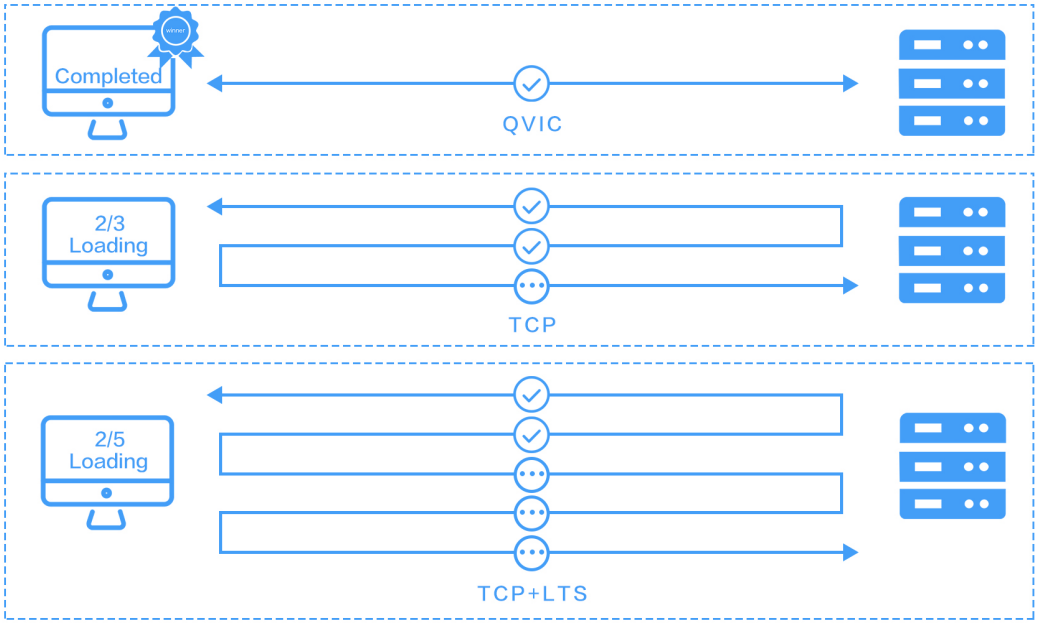


Figure 7.2: Comparison of handshake mechanism.

7.4 Experimental Comparison

50 machines in different data centers in Beijing, Shanghai, Guangzhou and London were used to test data transmission. Using QVIC protocol, the transmission rate of a 1G file was increased from 100Kbps to 1Mbps.

The four data centers above were also used to build a blockchain test network with 1K nodes. Using the QVIC protocol, the data transmission process during consensus and block synchronization was significantly more efficient, and the confirmation time for a single transaction was reduced by 70%.

7.4.1 Transport Bandwidth

As shown by the figure below, QVIC protocol has greatly improved the transmission rate of packages as compared to TCP. The transmission rate of a 1GB data transmission was improved by 500%.

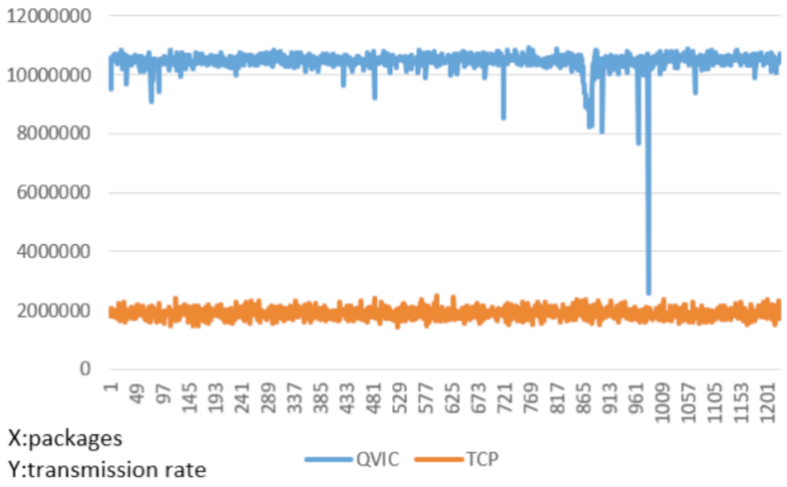


Figure 7.3: Comparison of Transmission Rate

7.4.2 Stability

QVIC has a customized transmission control algorithm and a forward error correction (FEC) dynamic compensation mechanism, improving stability and efficiency over UDP.

8. Computing Integration

8.1 Current Situation

Blockchains provide reliability, security, and tamper resistance. More and more data has been transferred from centralized sources to blockchains.

Unfortunately, the cost of storing files and data on the blockchain is large. In addition, many agencies, including government and corporate enterprises, have high-value data. It is difficult to store this private and secure data on an open blockchain. Traditional application execution does not guarantee data security

and correctness, but virtual machines lack the computing power to execute modern, more demanding computing needs.

Seele defines a new type of technology that integrates off-chain storage and computational resources using blockchain and smart contracts.

1. Blockchain and IPFS are combined, storing the encrypted hash value of the file directory on the blockchain. This saves storage cost. The encrypted file can be permanently stored in the distributed IPFS file sharing network.
2. Blockchain-based data desensitization technology will be used to ensure data privacy for data islands and off-chain data. This will allow for the reliable exchange and ownership of data within the system.

Internet computing presents a new distributed cloud computing infrastructure that enables Seele's blockchain computing to have lower operating costs. Seele provides two different methods of secure, contract computation. On-chain virtual machines are suitable for scenarios with applications that require low resources and are safe to transfer. Off-chain computing resources, including desktop systems and distributed clusters, can be controlled through contractual entities.

8.2 Resource Definition On-Chain

Naming Value Transfer Protocol (NVTP) defines a complete set of blockchain VTP, using a hierarchical structured naming method to name information in chains. This structured naming scheme defines a protocol header, chain identifier, account name, and resource name. This section mainly describes the storage definition of resources on the chain.

8.2.1 Metadata Directory Specification

Metadata Directory Specification (MDDS) is used to describe the original features of off-chain data, including a static description of the semantic features and basic attributes and a dynamic description of the data storage mechanism. MDDS has been well verified in actual use cases. We use MDDS to uniquely identify original data on the blockchain.

8.2.2 Metadata Directory Description Method

Seele extends MDDS. We propose Resource Description Framework (RDF), a description protocol for specific information contained by a UAI. In addition to including the static and dynamic description of data in MDDS, RDF adds a description of computing resources, and calculates the characteristics of these resources, including memory, CPU, hard drive size, and more. This is helpful for implementing tasks that have minimum requirements for memory or CPU cores.

As an important component of distributed storage and computation, automatic addressing is used to address content through the system contract. According to UAI identification, we first find the entry in the Meta Chain and then search downwards until we find the desired sub-chain. Here we can find the RDF corresponding to the data and initiate a request for the on-chain (or off-chain) resource pointed to by the RDF. After the request is confirmed by the signature of the data owner, the smart contract can link the resource requester and the resource supplier.

The automatic matching of the resource supplier and the requester needs to be done using different strategies depending on the needs of users. Each chain provides a sub-chain address lookup service,

which can be implemented by the system contract and initialized when building the chain. When a new sub-chain is added, the sub-chain sends a registration request to the parent chain, and the parent chain records the sub-chain address. The Meta Chain is the global configuration chain; it manages the entire forest system and all the entrance addresses. When it receives a query for information it first finds the entry in the Meta Chain, and then looks down until it finds the desired sub-chain, and then accesses the resources on that chain.

8.3 Storage and Computing

8.3.1 Internet Storage

Using MDDS, we provide two kinds of blockchain-based distributed storage methods. IPFS is at the core of data and metadata storage on chains. IPFS storage provides a decentralized network, where users store data using a directory structure that allows each user to define files. This directory structure includes links to other IPFS files and their descriptions. The user uploads the data to the blockchain, the producer determines that the data has been accepted by the broadcast, and the other blockchain nodes copy the file over the IPFS network. A hash address is calculated for all published content and a general hash address index is built – when a user accesses a file, IPFS broadcasts a hash request, finds the node that stores the file, and sends it to the user.

The public network environment is suitable for the exchange of metadata usage due to its trusted nature. Trusted data exchange in the actual product has been verified. Through the interconnection protocol provided by the public chain, the user can also synchronize the original data of files to the blockchain and establish a mapping relationship between the data directory on the blockchain and the user-owned data using MDDS.

Data needs to be secure and private while it is stored on chain and while it is going through the transmission process. Traditional data encryption technology must be used to ensure that this is the case. Sometimes, different users need different levels of access clearance – so we need fine-grained control of data encryption. The public chain uses attribute encryption and a security sandbox to achieve this.

8.3.2 Grid Computing

Blockchain can provide computing resource integration and distributed computing capabilities. Smart contract virtual machines can be run on nodes in a public network, creating a verifiable, distributed computing environment similar to Ethereum. During this process, the data is securely encapsulated within the smart contract virtual machine.

A traditional data exchange scenario works as follows. A user, C, obtains data from sources S1, S2, and S3. It wants to run an algorithm, M, on the data. This algorithm is provided by an algorithm provider, P. C must provide P with the data, and P returns the result back to C. Unfortunately, this reveals to P the data in S1, S2, and S3.

Computing resource integration based on secure multi-party computing provides a new way. The data provider can upload encrypted data into a smart contract virtual machine – the data is divided into N parts and distributed to N smart contract virtual machines. The intermediate results obtained from this stage are reorganized to obtain the final calculation result.

Figure 8.1: Framework of Grid Computing

Seele provides a new computing method based on collaborative computing. Through contract control and coordination, the distributed computing process can be extended to off-chain computing resource providers, and data is distributed to the desktop system through the sandbox or distributed cluster computing power platform. Blockchain verifies the correctness of the computing process.

8.3.3 Multi-Domain and Multi-Level Scheduling

Task and resource scheduling is a very important component of distributed systems. The design of the scheduler algorithm has a direct impact on the utilization rate of the entire cluster. In distributed systems, scheduling goes through single scheduling, two-level scheduling, shared state scheduling, fully distributed architecture evolution, and hybrid architectures.

We propose multi-domain and multi-level scheduling. We divide the blockchain into different domains based on security and trust factors. We divide indices such as cost, performance and security that are required by users into different levels; the same scheduling algorithm can choose different domains and different levels to allocate tasks and computing resources to, and the scheduler can be fine-tuned to predict mission performance and reduce neighbor interference to support the user's special computing needs. The following scenarios describe different schedules based on different user needs: Consumer A does not have high execution time or data security requirements for a task, and the computation time is relatively long. Consumer B needs fast execution time but the security of the data is not important; Consumer C's demand for data security is higher, but the task execution time can be longer, and the cost for the calculation can be higher.

8.4 Client Design

8.4.1 Metadata On-Chain

The client provides two interfaces for the on-chain metadata directory. The first is for metadata directory extraction and the second is for metadata directory updates.

Metadata directory extraction refers to using technical methods to assist the manual methods to extract the metadata directory information from the original data. The metadata directory is divided into the public metadata directory and the complete metadata directory (including private information).

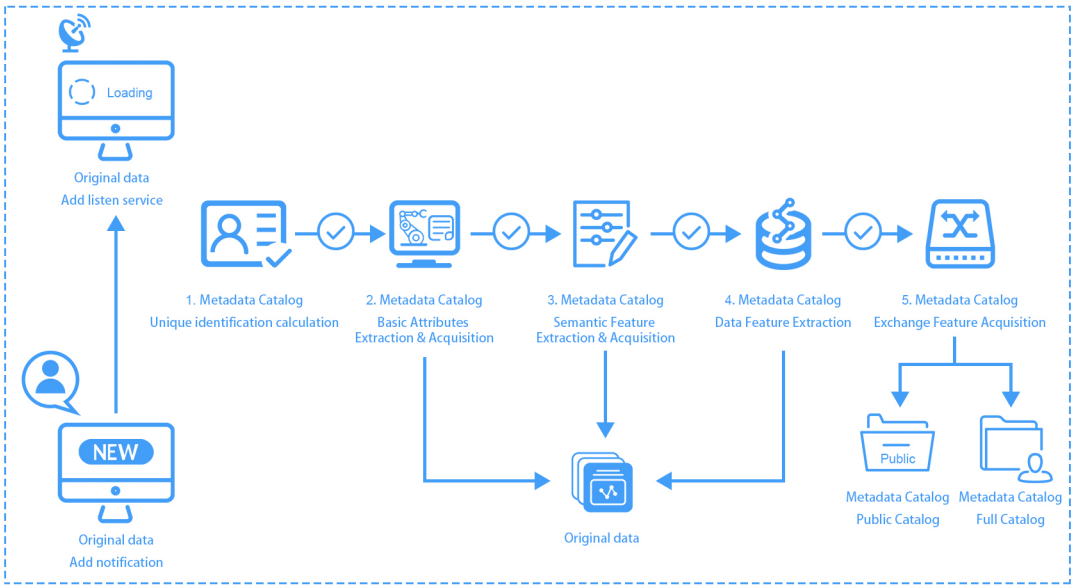


Figure 8.2: Procedure of Uploading Metadata to the Blockchain

Metadata directory updates mean that when the original data changes, the metadata directory needs to be changed as well. The main process is as follows:

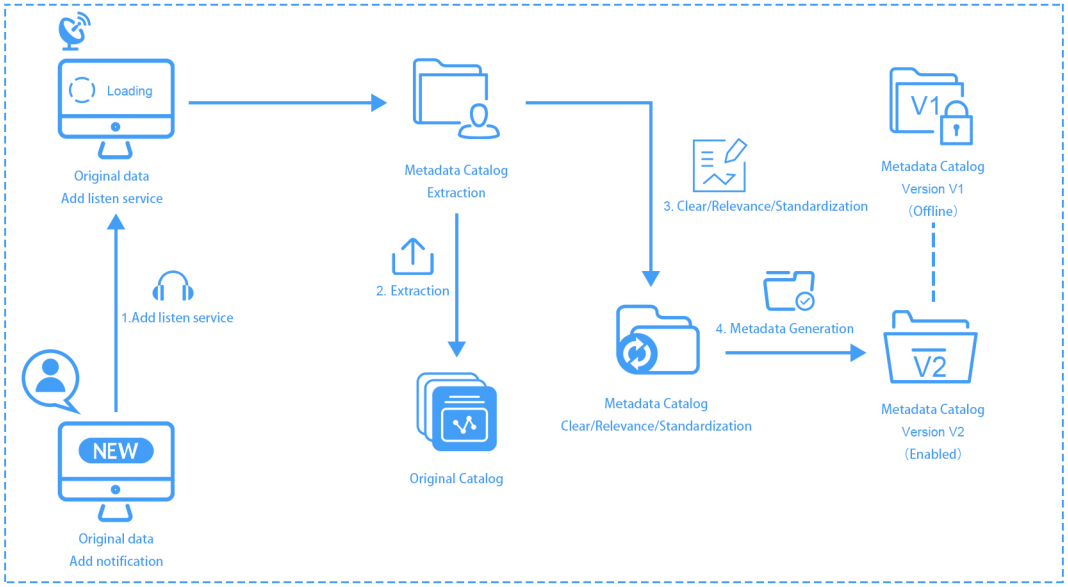


Figure 8.3: Procedure of Updating Metadata

1. Listen for a change notification from the original data.
2. Metadata directory information is extracted from the raw data according to the change notification and the metadata directory standard.
3. ETL (Extract, Transform, Load) operations according to the metadata directory standard.
4. The generated metadata directory data is recorded as a V2 version, storing the metadata directory information into the chain.
5. The upper application enables a new V2 version of the metadata directory, gradually replacing the V1 version of the metadata directory.

8.5 Data Privacy and Confidentiality

Through smart contracts, the data is dispatched to the execution environment in the encapsulated sandbox, and the execution environment is divided into two types of cases. It is executed on the contract virtual machine and on the computing resources outside the chain.

Through the secure multi-computing cryptography algorithm, the multi-party data combination computation is completed on the contract virtual machine of the smart contract cluster without compromising the original data of each party, and the code is processed under the premise of ensuring the privacy of user data. For scenarios executed on computing resources outside the chain, the smart contract controls the delivery and execution of data, encapsulates the data in the sandbox to the computing resources, and returns the results to the user through the security sandbox. The computing process and the environmental information are uploaded in real time for on-site inspection and post-audit.

8.5.1 Attribute-Based Encryption

At its simplest, attribute-based encryption is a type of public key encryption that ensures that non-authorized users are not allowed to view or access specific data. Its biggest feature is that the user's private key and the ciphertext are dependent upon certain attributes - like the user's identity. Only when the user's attribute set and the properties of the ciphertext match can the ciphertext be properly decrypted.

When implementing the contract computing, the security sandbox decrypts the received encrypted data using the private key of the user. During the process of generating the private key of the user, we can generate the key according to the predetermined user's permission and access rules to implement line-grained permissions, time limits, frequency limits, and content limits of security access control.

Attribute-based encrypted function definition:

1. Parameter initialization: Given the system size^{m}, initialize the parameters, and output the public parameters^{mpk}, and the main private key ^{msk}: $Setup(m) \rightarrow (mpk,msk)$
2. Key generation: The main private key^{msk}, and the attribute assignment A as input, calculate output user private key$^{sk}_{k}$: $KeyGen(msk,A) \rightarrow sk_{k}$
3. Encryption algorithm: Enter the public parameters^{mpk} and encryption policy Policy, calculate the session key ^{ek} and ciphertext ^{c} : $Encrypt(mpk,\text{Policy}) \rightarrow (ek,C)$
4. Decryption algorithm: Enter the public parameters^{mpk}, user's private key$^{sk}_{k}$, calculated to recover the session key ^{ek} : $Decrypt(mpk,sk_{k},C) \rightarrow ek$ iff Match(policy,A)=1

For instance:

Let's create a collection of the following values:

University Name: = {....., "Harvard University", "Stanford University", "Tsinghua University", ...},

Department: = {..., "College of Biology", "College of Chemistry", "School of Information", ...},

Year: = {...,"2013", "2014", "2015", "2016"...},

Role: = {..., "Professorial Committee," "Academic Committee," "Academic Degrees Committee," ...}.

The above security policy can be defined for any resource (including files, storage, network channel, process, etc.) as

Policy: = (university name ∈ {"Harvard University", "Stanford University"} AND year = 2015 AND role = "academic degrees committee" AND department {"Biology", "Chemistry"}).

Suppose a user has the following identity attributes:
A: = {University Name: = "Stanford University", Year: = 2015, Role: = "Academic Degrees Committee", Department: = "School of Information"}, this means that the user was at Stanford University's School of Information in 2015 and is a member of the Academic Degrees Committee. In this case, the user's identity passes the above security policy authentication, and therefore, will be allowed to access the encrypted resources.

8.5.2 Secure Multi-Party Computing

Theoretical models behind secure, multi-factor computing have been developed in the past. In 1982, Dr. Andrew Chi-Chih Yao proposed the Yao's millionaire problem. From 1983 to 1987, Israeli scholar Oded Goldreich proposed several research papers and improved the concept of secure multi-party computing.

In the case of collaborative computing based on blockchain and smart contracts, participants need to complete a certain computing task together, but all parties want to retain ownership and control of their source data. Participants only want to share limited amounts of their own data with each other.

9. Ecology/Governance/Incentive

9.1 Developer Ecology

In order for a public blockchain to succeed, it must be able to attract developers to invest in the platform's network, community, and application development. Public blockchains must provide these developers with excellent underlying technology and an easy-to-use environment for application development.

9.1.1 Problems

As an example, let's look at the development of Ethereum dApps:

1. There are still a lot of imperfections to overcome in the development process in spite of the rich development tools and framework. It can be very complicated to collect materials because of the different uses of varying languages and frameworks.

2. There is quite a lot of documentation, but it can be messy and out-of-date. Upgrades to the protocol have left many examples needing updates.

3. There is no perfect cross-platform access solution. Although now it's easier to develop some web applications, many mobile applications don't have multi-terminal access functionality as a result of imperfect SDK support.

4. Although there are various development environments like a private blockchain, testing blockchain and main blockchain, a perfect guide still needs to be produced for the development, testing and deployment of smart contracts.

9.1.2 Solutions

In response to these problems and using our team's experience with the Unity gaming engine, our team will develop a complete smart-contract development tool, SeeleEditor, with the following features:

1. A complete application development toolbox. It will support the development, testing, and deployment of smart contracts.

2. Cross-platform SDK support and continuous updates of technical documents to improve development efficiency and the diversity of applications across different platforms.

3. A plugin store, like Unity, to provide contract development components and sample programs to developers to improve development efficiency. This way, developers can generate revenue not only by developing specific applications but also by developing support components.

4. IDE with community modules. Any problems relating to development can be communicated and promptly resolved on it.

9.2 Industry Application Ecology

Blockchain technology at its present stage has already gone beyond its original ledger function. Seele, at its core, is a super-distributed cloud computer with its own complete storage system that can process millions of TPS. The goal is that we can foster the development of richer, more complex blockchain applications in addition to improving already-existing applications involving financial asset transactions, token issuance, and prediction markets. The following are some examples of spaces we find interesting:

1. Social networks. Build a social platform like Steemit that can encourage users to create better content by awarding them with tokens. A blockchain platform can be built for users to tip contributors directly. The transparency of payment can eliminate many problems on the platform like self-voting and opaque commissions.

2. Blockchain gaming. The emergence of the Ethereum-based CryptoKitties has inspired a new direction for the gaming industry. Blockchain-secured digital ownership provides a new dimension for trading-based games, allowing users to fully "own" their assets and hold onto them permanently, even after the game developers abandon the game. However, the network usage generated by CryptoKitties massively clogged the Ethereum blockchain and made transaction costs skyrocket - the industry needs better infrastructure to support high-traffic content like games. Seele will help solve this issue.

3. Internet of Things. Blockchain has always been a good solution for the Internet of Things due to its superiority when it comes to distributed calculation, data management, security, and transparency. However, due to the heavy power consumption and inefficiency of traditional, PoW blockchains, it has failed to fully penetrate the market. Seele, with its EDA consensus algorithm providing ultra-low power consumption and high concurrency, is suitable for this scenario. Our consensus algorithm can also be used with IOTA's tangle structure, which currently cannot completely avoid "double spend" and DDOS attacks.

4. Other enterprise applications. Using our forest structure, it's easy to create a chain for a specific enterprise application. A variety of customized services can be developed on Seele's technology, and through NVTP, these enterprise applications can make cross-blockchain communications and value delivery, bringing more flexibility and convenience.

9.3 Economic System

9.3.1 Token

One of the good features of our consensus algorithm is that transaction confirmation time decreases as the number of nodes increases. Given that, we will encourage nodes to join the network to improve performance and security. To do this, we use a token. This token will mainly be used in two ways - first, it will be used to reward participating nodes, and second, it is used to pay transaction fees. Transaction fees allows reimbursement for bandwidth and computing resources used by nodes and helps prevent against Sybil attacks.

9.3.2 Incentives

Currently, mainstream blockchains utilize transaction fees as well as giving fixed rewards to nodes as a way of issuing currency.

There are three main ways to charge transaction fees:

1. Change mechanism used by Bitcoin. The change of each transaction is used to reward miners.

2. Ethereum's gas mechanism. Calculate the consumption of gas and then make a price for gas value in strict accordance with the calculation and storage behind the transaction. The gas value multiplied by the price equals the final transaction fees.

3. EOS' exemption from transaction fees. The premise is that the user initiating the transaction will hold certain tokens to enjoy resources in the system. The number of tokens determines the amount of resources.

After seeing the transaction and their corresponding fees, miners will then choose to include the transactions with the highest fees to maximize profits. Therefore, the essence of an incentive system lies in the allocation of resources - how to improve the maximum utilization of resources under limited resources. For this problem, the traditional mainstream approach is essentially the following two simple solutions:

1. The party that has the most money will be the one in charge.

2. The party that seizes a seat first will be the one in charge.

In our opinion, the allocation of resources should not be decided in such an easy way. The blockchain technology selection and programs shall refer to the actual modes of governance in society. Here, our principles are efficiency and fairness.

In terms of incentives, we adopt a hierarchical incentive mechanism based on participation and value. It mainly includes the following two aspects:

1. Consensus participation incentive. Our system becomes more efficient as the number and stability of nodes increases. Therefore, we want to encourage more nodes to join the network and participate in consensus to improve performance and security. Tokens are rewarded to converging nodes, and the number of rewarded tokens will decrease over time. In addition, nodes can earn tokens from transaction fees. Unlike traditional POW mining, our consensus algorithm requires nodes to have good network connectivity instead of high computing power. In our system, the contribution of nodes depends on their bandwidth - because this is a distributed, national resource, it avoids the concentrating effects that hashrate-based mechanisms experience. Moreover, the lower-level of our network has strong penetration ability, which can allow a large number of intranet nodes participate in transaction consensus to greatly increase the size of nodes.

2. Packing block value incentives. Each time a node generates a block, the system will reward block-creators with certain tokens based on the value of the block's transactions. The system will provide value according to the different types of transactions.

This way, we can encourage more valuable transactions to be identified faster to improve the efficiency of the network.

As for transaction fees, we employ model similar to Ethereum's gas, namely by calculating gas costs of the transaction. But the cost of gas will be dynamically adjusted according to running status of the system to ensure fairness.

Through the mechanism above, we hope to give consideration to both efficiency and fairness to improve efficiency of the entire system.

9.3.3 Governance Structure

More subjective blockchain problems cannot be solved by algorithms and need to be solved by the community - otherwise, problems can arise. A few examples are as follows:

1. The serious disagreement between Bitcoin's core team and its miners has caused a division within the entire community. Various forked coins have emerged within this conflict.

2. A Bitcoin or Ethereum account is completely lost if the private key is lost or forgotten. In Bitcoin, many addresses have already been lost forever and these coins will not be able to contribute to the network.

3. Ethereum’s fork caused by the DAO hack.

4. Frozen Ether from the Parity multi-sig wallet bug.

These problems have led to serious consequences and have caused many people to lose confidence in blockchain technology. We need to take some measures to solve such problems using solutions in the real world.

In Seele, the governance power stems from the token holders who delegate power to the block builders. The block builders are given limited and supervised privileges to freeze accounts, update defective applications, and make changes to the underlying protocol.

Token holders are selected by our consensus algorithm, which will randomly select a group of people from the fastest responding nodes, determine a list by weighing their held tokens, and update this list regularly. These nodes will undertake the governance functions within this time frame. Nodes that have good network connectivity and a fast response time will be more likely to be chosen as governance nodes. This combines the advantages of PoS and PoW, and takes nodes’ participation into consideration, which reflects fairness and will help avoid collusion attacks.

The block builder election is a part of Seele, and the blockchain can only be changed upon the approval of block builders. Block builders can be voted down if they refuse to make expected changes. If block builders make some changes to blockchains without permission of the token holder, all other non-productive full-node verifiers (switches, etc.) will refuse to change. This allows for issues to be resolved by node voting.

10. Core Team

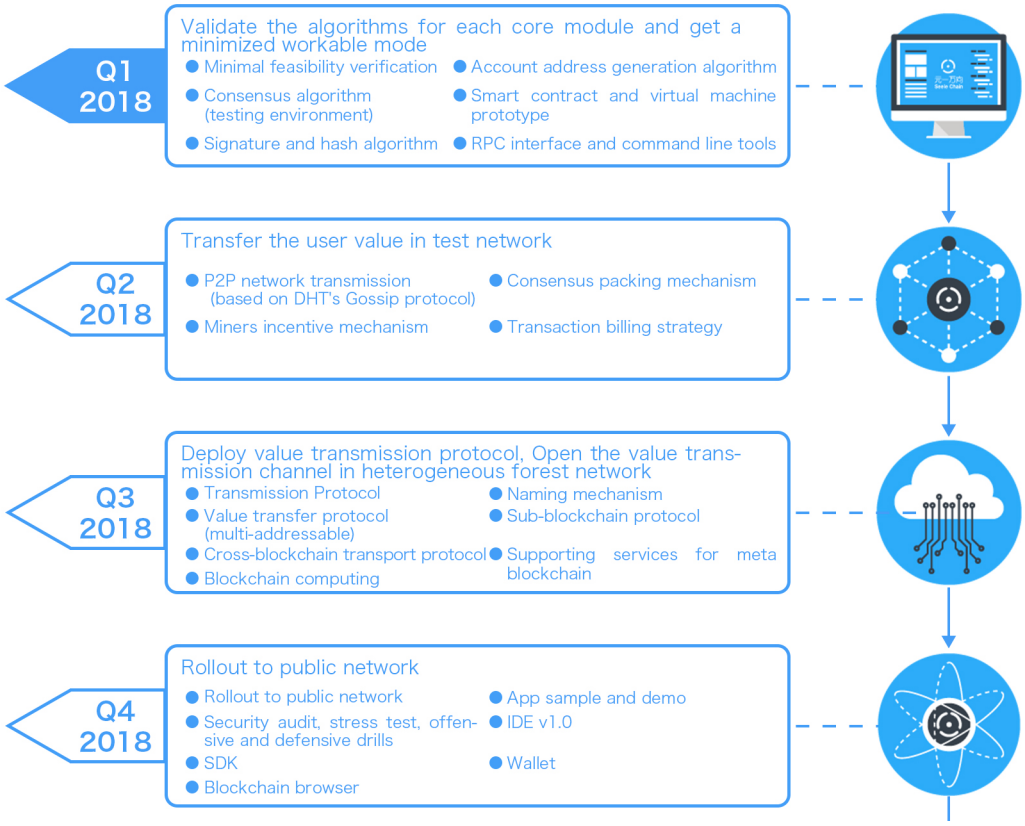
Dr. Maolin Zheng, Chief Executive Officer
Postdoctoral Research Fellow, NSERC, GERAD, University of Montreal
Decision Optimization Expert, FICO

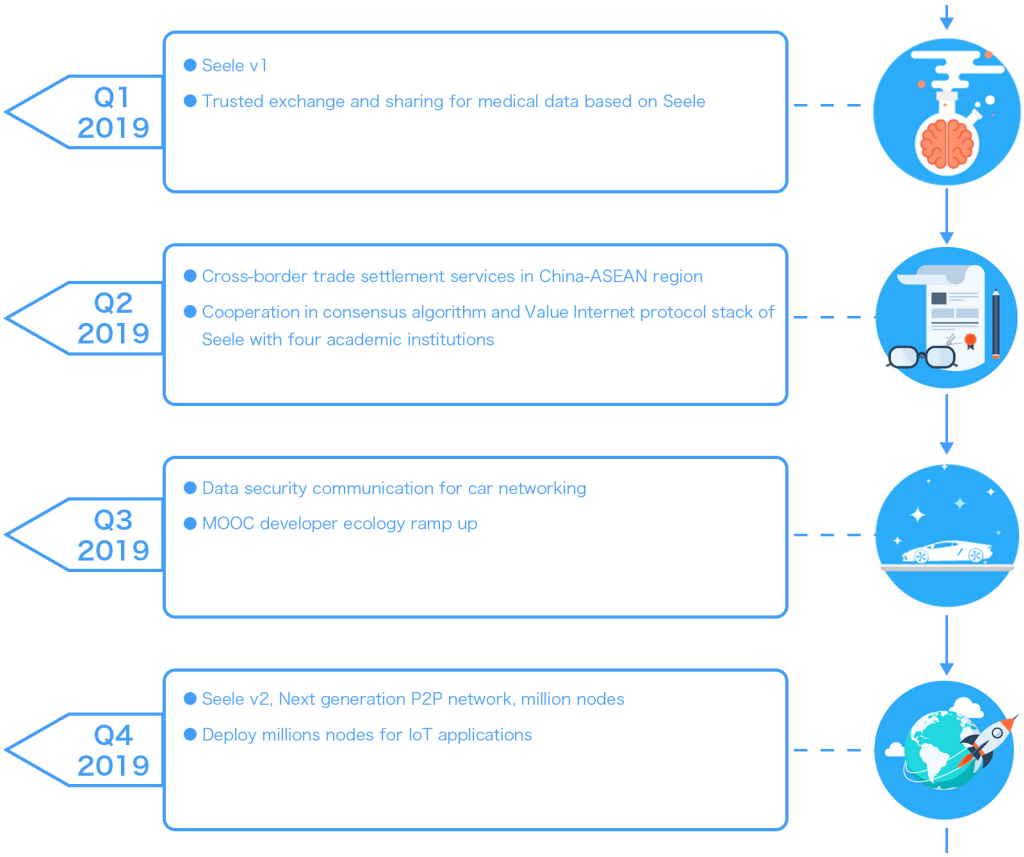
Former CTO and Chief Scientist, Beijing Guozhengtong Technologies
Former Chief Scientist in Credit Risk Management, Creditease Beijing
Experience in big data computing, mathematical modeling, predictive analytics, credit risk management, decision optimization, and discrete optimization.

Dr. Bi Wei, Chief Scientist
PhD in Visual Science, City, University of London
MSc in Computer Science, The University of Oxford
Deputy Secretary General, China Blockchain Technology Innovation and Application Alliance
First author of 8 blockchain-related technology patents (Pending, submitted in 2017)
Former fellow and graduate doctoral tutor, London University
Research interests: blockchain, cryptography, data analytics, image processing, and visual science
Dr. Bi has been invited to attend international academic conferences and has published work in journals such as the New England Journal of Medicine. His articles and opinions have been collected by BBC, London Chinese Radio, Complex UK, and other media sources.

Dr. Nick Smith, Chief Operating Officer
PhD in Visual Science, City, University of London
MRes, Design and Evaluation of Advanced Interactive Systems, Lancaster University
MSc in Advanced Computer Science, Lancaster University
Experience in data analysis, software development, and hardware development. Skilled in research, feature extraction, and project management.

11. Roadmap





12. Postscript Note

This white paper is a partial overview of the key technologies and ecosystems covered by Seele. The development and progress of this technology is endless, new forms of applications are also emerging. The white paper of Seele will be continuously updated as our technology advances and our applications expand. In line with the ambitious goal of innovating the new era of Internet of Value, Seele welcomes developers and service providers to join the community and help build a new ecosystem of innovation, development, and win-win cooperation.