RECYCLING REGENERATION CHAIN

Whitepaper

RRCHAIN Team

Version: v1.81

Update: 2018.06.01

Whitepaper open to the public and can be changed at any time

Introduction

Starting from Bitcoin, the advent of ASIC mining machines has severely undermined the fairness of the community and deprived most community users' equal rights to mine. It has become a paradox from the concept of decentralization. With the fast development of the cryptocurrency market, mining machines have become an independent industry. Influential cryptocurrencies can hardly escape the influence of the miner manufacturers.

Cryptocurrencies have tried various methods to prevent the miner from destroying the fairness of its communities. Even many cryptocurrencies have to choose multiple hard fork upgrade algorithms to deal with the miner, or just give up the original consensus algorithm.

RRChain is a public chain developed by RRC project team. RRChain adopts unique algorithm design (sandwich algorithm) and multiple algorithm sets. It can still ensure the fairness of mining under the premise of using PoW mechanism, relieving advantages of professional mining machines in order to ensure the fairness of ordinary users mining.

Through a P2P network, computers, mobile phones, and other computing terminals are used to allow application-side users (demanders) to rent computing power (suppliers) to other users on the network. The computing power can accomplish certain tasks, such as: big data analysis, artificial intelligence, image processing, game operations and many other aspects.

As an important part of the RRC ecosystem, RRChain has played the following two key roles.

1. Ensure the fairness of mining.

2. Efficiently and efficiently count the calculation workload of each node.

In addition to distributed computing scenarios, any scenario that has special requirements for fairness and authenticity can be built on the basis of RRChain to build application layer products.

RRCHAIN Technical model

Technical model diagram:

The model of the RRNC technology consists of a bottom-up data layer, a network layer, a consensus layer, an incentive layer, a contract layer, and an application layer.

1.Data layer

It encapsulates the chain structure of the underlying data blocks and related asymmetric public and private key data encryption technologies and time stamps. This is the lowest-level data structure in the entire technology model. These technologies are the basis for building RRChain.

1.1 Data block

1.1.1 Block header

```
struct block_header
  {
    uint8_t major_version;
    uint8_t minor_version;
    uint64_t timestamp;
    crypto::hash   prev_id;
    uint32_t nonce;

    // Serialization
    BEGIN_SERIALIZE()
      VARINT_FIELD(major_version)
      VARINT_FIELD(minor_version)
      VARINT_FIELD(timestamp)
      FIELD(prev_id)
      FIELD(nonce)
    END_SERIALIZE()
  };
```

There are five main fields; version number, minor version number (now used to do voting mechanism), timestamp, hash of last block， and nonce;

1.1.2 Block

```
struct block: public block_header
  {
  private:
    mutable std::atomic<bool> hash_valid;
    transaction miner_tx;
    std::vector<crypto::hash> tx_hashes;
    mutable crypto::hash hash;

    BEGIN_SERIALIZE_OBJECT()
      if (!typename Archive<W>::is_saving())
        set_hash_valid(false);

      FIELDS(*static_cast<block_header *>(this))
      FIELD(miner_tx)
      FIELD(tx_hashes)
    END_SERIALIZE()
  };
```

The fields are: the status of the block, identifying whether the block has been validated as valid; the coinbase transaction; a hash of all transactions in the storage block; a hash of the block;

1.2 Chain structure

The blocks are connected in sequence to form a chain structure.

1.3 Timestamps

The existence of the blockchain timestamp allows time-stamping for each piece of data, by arranging each transaction, each block of production, in chronological order. It ensures the uniqueness of the blockchain and cannot be tampered with.

1.4 Hash function

RRNC uses a hash-based proof of work (PoW) algorithm.

```
static void cryptonight_behe(const uint8_t *input, size_t size, uint8_t *output, struct cryptonight_ctx *ctx, int variant) {
        crypto::cn_slow_hash(input, size, (char*)output);
    }
```

1.5 Merkle Tree

The benefit of the Merkle tree is that a node can verify whether a transaction is included without downloading the entire block.

```
struct tx_extra_merge_mining_tag
{
  size_t depth;
  crypto::hash merkle_root;
 };
```

2. Network layer

Including P2P networking mechanism, data transmission mechanism and data verification mechanism.

2.1 p2p Networking mechanism

In addition to storing the entire blockchain, the full node in the RRNC also has roles for miners, wallets, routing nodes, etc. Their responsibilities are as follows:

Miner：

Just like the mining worker, doing manual work, constantly trying to fill the numbers in the random fields of the structured block header, to find a hash value that meets certain criteria, if found, connect this block to the top of the blockchain, and send the legitimate blockchain to the adjacency node.

Wallet：

The blockchain records coinage transactions and transfer transactions. This is different from the usual meaning of the account system. The usual meaning of the account system is to record account balances. In the blockchain, only transactions are recorded, there is no balance, and the wallet is used for calculate the balance of an address in a blockchain. If you have learned a relational database, you can think of the blockchain as an index of a database or as a directory of books.

2.2 Data dissemination mechanism

The routing node is responsible for passing transactions and blocks in a decentralized network environment. A node creates a transaction and sends it to an adjacent node. After verifying the neighboring node, it sends it to the adjacent node. Spread throughout the network. If a node goes through mining, it finds a block that meets the criteria. This node will also be passed to neighboring nodes in the same way, and then the neighboring nodes will continue to spread and all nodes of the network will agree.

2.3 Data validation mechanism

RRCHAIN confirms the correctness of the data by verifying the hash value and signature of the data.

3. Consensus layer

RRCHAIN uses the PoW mechanism. The miners get their rewards through mining, and ensure the stable operation of the system.

The consensus layer encapsulates various consensus mechanism algorithms for network nodes. The consensus mechanism algorithm is the core technology of the blockchain, because it determines who is going to account, and the decision-making method of accounting will affect the security and reliability of the entire system. At present, there have been more than ten kinds of consensus mechanism algorithms, of which the most well-known ones are the proof of work (PoW), the proof of Stake (PoS), and the DPOS (Delegated Proof of Stake).

The proof-of-work mechanism is actually a voting system. Users vote to choose the correct transaction sequence to implement the new features in the agreement and the honest distribution of money supply. Therefore, all participants have the same voting rights during the voting process. RRCHAIN makes equal average job pricing perfectly applicable to ordinary PCs. It uses built-in CPU instructions that are very difficult and expensive to implement on dedicated devices or on fast memory on-chip devices with low latency.

3.1 Current situation:

To realize the equalization of mining, we hope that the energy efficiency and cost of general-purpose devices (such as CPUs and GPUs) will not significantly lag those of ASIC mining machines. Historically, various PoW digital currencies have designed several anti-ASIC algorithms to achieve this goal, but these algorithms are not satisfactory. These algorithms can be roughly divided into two categories.

The first category is to try to increase the local storage required to complete PoW. For example, LiteCoin requires 128 KB of storage, while

Monero needs 2 MB of storage. However, in the 28nm and more advanced processes, it is no longer difficult to integrate 2MB of SRAM on integrated circuit chips. LiteCoin's ASIC mining machine has already been born. The Monero mining machine was also recently produced.

The second category is to try to increase the DRAM bandwidth and DRAM space required to complete the PoW, typically represented by Ethereum and ZCash. The current mining machine of these two digital currencies is dominated by video card mining machines. But it is worth noting that Bitland has already mass produced the Ethereum mining machine E3. It can be foreseen that the ASIC mining machine will gradually replace the graphics card mining machine will be an inevitable trend. The reason is that the bandwidth-intensive mining algorithm only uses the DRAM fetch bandwidth of the graphics card, and it is not used for the computing resources on the graphics card. Therefore, a chip similar to a graphics card but with a drastic reduction in computing resources can be designed, and its power consumption and cost are extremely low. However, bandwidth-intensive PoW algorithms can still be implemented.

3.2 Design ideas:

RRCHAIN hopes that each node involved in mining will use the CPU to mine, at the same time, it can seamlessly switch between mining tasks and big data computing tasks. Therefore, when designing the PoW algorithm, we follow the following principles to achieve the goal of the Anti-ASIC.

First, select a CPU with high energy efficiency as the adaptation target.

As far as a PoW algorithm is concerned, the most critical is energy efficiency, which is the amount of electricity required to complete a certain amount of work.

If we choose the CPU with very high power consumption to adapt, designing PoW algorithm makes them have good performance on this type of CPU, then leave room for ASIC mining machine. A negative example is Monero, which chooses desktop/server CPUs with higher power consumption and sets the frequently accessed local storage capacity to 2MB (equivalent to the average L3 cache capacity of a CPU core). On the CPU of mobile devices, the computing power is greatly reduced.

We should choose the CPU that originally had very low power consumption as the adaptation target. In the current market, the low-power cores in mobile phones and the low-power cores in embedded devices (such as the Raspberry Pi) meet this requirement. The CPU cores on mobile phones have almost all adopted the (big.LITTLE) architecture, that is, a number of large cores with low energy efficiency and high performance, which are combined with several energy-efficient and low-performance small cores. The high-power consumption of large cores cannot even be used for a long time. Otherwise, it will cause the phone to overheat. Generally, it can only be operated for a moment when it needs high performance instantaneously (for example, when running points). Obviously, the big core is not adapted as PoW algorithm.

When designing the PoW algorithm, we will adapt the parameters of various components of the mainstream small-core CPU in the mobile phone market, including but not limited to: throughput, emission width, and various values of the value decoding component. The number of execution components, the size of the Cache at each level, the Buffer depth for instruction and data prefetching, and the average access bandwidth for each core. Ensure that the small-core CPU's performance advantages are fully realized.

Second, make full and reasonable use of various components in the CPU. The components in the CPU can be roughly divided into two types. One is an operation unit, and the other is a hierarchical access system.

The arithmetic unit is responsible for the operation of integer and floating-point operations, as well as some logic operations and bit operations. Most PoW algorithms only use addition, logic, and bit operations because the hash function uses only these calculations. However, the addition, logic, and bit operations only occupy a very small area of silicon. Operating units that occupy most of the CPU's internal area, such as multiplication and division, floating-point units, etc., cannot play a role. ASIC miners omit this part of the unit, which can save a lot of costs and save some power consumption.

Monero's PoW algorithm uses 64-bit multipliers and AES instructions, slightly more advanced than other PoW algorithms, but far from fully utilizing various arithmetic units.

The PoW algorithm uses floating-point numbers to face problems of positive infinity, negative infinity, and NaN (Not-a-number). Some random floating point numbers, after several rounds of random operations, some data will overflow to positive infinity, while others will overflow to negative infinity. Positive infinity and negative infinity are contagious. They add, subtract, multiply, and divide other numbers. Both get positive infinity and negative infinity. Positive infinity and negative infinity add NaN. NaN participates in any calculation. The result is still NaN. Because of the nature of floating-point numbers, it is difficult to construct a hash algorithm directly using floating-point numbers, because after several rounds of random calculations, the majority of the results obtained are positive or negative infinity or NaN.

However, this problem is not irresolvable. As long as we limit the initial value of a group of floating-point numbers within a certain range and limit the round of operations, we can ensure that any member of this group of floating-point numbers can perform arbitrary addition, subtraction, multiplication, and division in any order. There will be no positive and negative infinity and NaN.

Hierarchical access systems include Cache (generally divided into L1, L2, L3) and DRAM memory access interfaces. The CPU's fetching system assumes that the program running on it has considerable temporal and spatial locality. That is, after a certain piece of data is accessed, data adjacent to it is likely to be accessed within a short time. The L1, L2, and L3 Caches are responsible for the locality of different sizes. The L1 cache is generally between 32KB and 64KB, the L2 cache is between 128KB and 256KB, and the L3 cache is between 256KB and 2MB.

The PoW algorithm that adapts to the CPU should cooperate with the L1, L2, and L3 caches, presenting a reasonable locality of access, spanning a small number of visits, and having a large span of visits. Monero is a typical example. Its locality of access can only be supported by L3 Cache. It is equivalent to L1 and L2 which have not played a role. The adaptation effect is not good, and CPU performance is not fully utilized. In addition, the hierarchical fetching system is optimized for both read operations and write operations, so both read and write must be available. Ethereum's PoW algorithm is a negative example. It only has read operations on data sets in DRAM and has no write operations. This makes it easy to start multiple threads to share data sets and reduce costs.

The DRAM memory access interface is necessary for all CPUs and occupies a large area of silicon, even exceeding the CPU itself. If the PoW algorithm can work without DRAM, then chips that do not include a DRAM fetch interface can be manufactured to reduce costs. Monero's PoW algorithm was designed to use the L3 Cache to fully house the entire data set, so it was only exploited by the miner. For this reason, the memory space occupied by the PoW algorithm must be large enough, at least 16 MB, to make the use of on-chip SRAM in the mine machine an expensive option.

Third, we must introduce as many algorithms as possible to construct a set of algorithms. Any single algorithm can always make use of its unique features. By customizing and tailoring the CPU, it can be greatly optimized. Only by introducing various CPU-appropriate algorithms developed in the history of computer development into the PoW algorithm set as much as possible, thus, allowing us to use the original CPU to mine.

Currently, there are few secure hash algorithms that are recognized in the industry. Most of the algorithms running on the CPU do not satisfy the randomness, fairness, and unpredictability required by the PoW algorithm.

Fourth, the collection of algorithms must maintain a certain degree of update and evolution.

The ASIC mining machine has a long cycle from project approval to cost recovery, including research algorithms, chip design, chip manufacturing, complete machine manufacturing, capacity ramping, sales, mining, etc., which will take at least one year. If the mining algorithm implements rotations on a block-by-block basis (but all from the set of algorithms), and the algorithm set keeps changing over time, for example adding new algorithms every six months, as long as the interval between changes is less than the life of the miner Cycle, then the optimization of ASIC mining machine will be unprofitable, can only be configurable, refresh algorithm mining machine, that is, can only take the CPU to do the mining machine.

3.3 Algorithm implementation:

To solve this problem, we use a sandwich structure that can transform almost any algorithm into a hash algorithm: First, using a standard hash function H to extend the nonce and random number seed seed into many random numbers. Input I=H (nonce, seed); use the algorithm A that the CPU excels at to process the random number input I, get some output O=A(I); then calculate the hash value h(O) of these output with another hash function h, with leading the number of zero determines whether it meets the difficulty requirement. This sandwich structure has been proven effective by ZCash. We only need to modify the middle part of the general birthday paradox problem to other algorithms that the CPU is good at. Because there is a standard hash algorithm for closing, h(A(H(nonce,seed))) can satisfy the avalanche and irreversible characteristics of the hash function, and satisfy the randomness, fairness, and unpredictability required by the PoW algorithm.

Since the birth of the computer, humans have developed many CPU-optimized algorithms. We plan to gradually select some algorithms and add them to the PoW algorithm set in the order of ease and difficulty. We plan to add the following algorithms in the first phase as follows: category:

1、 Establishment and access of common data structures: various trees (red and black trees, B-trees, B+ trees, etc.), priority queues, Fibonacci heap, HashTable, Cockoo-HashTable, Bloom-Filter, etc.
2、 Typical sorting algorithm
3、 Typical dynamic programming algorithm
4、 String matching
5、 Graph algorithm (shortest path, All-pairs shortest path, minimum support tree, maximum flow)
6、 Matrix algorithm (multiplication, transfer rank, inverse, find eigenvalues, etc.)
7、 Least Squares
8、 Newton iteration method
9、 Fourier Transform
10、 Viterbi decoding
11、 Linear programming
12、 Error Correction Algorithm (LDPC, ECC, etc.)
13、 Compression algorithm (gzip, LZ4, etc.)
14、 Regular expression
15、 Reverse Polish Expression Evaluation

Different algorithms have different memory access behaviors, and many algorithms make full use of L1, L2, L3 cache, and DRAM interfaces. Therefore, when necessary, we will add additional "filling logic" that is not related to the algorithm itself. The filling logic takes some random results when the algorithm itself is running as input. After a small amount of calculations, it generates numerous memory access operations and makes the storage system busy.

4. Incentive Layer

Integrate economic factors into the RRChain technology system, including the issuance and distribution mechanisms of economic incentives. In RRChain, motivating the nodes that follow the rules to participate in the accounting, and punishing the nodes that do not follow the rules, can make the entire system move in a virtuous circle.

4.1 Issuing mechanism:

Total amount: 1.8 billion
Token release amount: (Total amount - Total released amount) >> Token release speed

4.2 Distribution mechanism:

RRCHAIN will release gradually, based on a total amount, each reward is based on the total amount minus an algorithm that has already been issued.
The difficulty adjustment algorithm uses BCH's DAA algorithm
The DAA algorithm is designed to avoid sudden difficulty reductions and spikes. For example, when the RRC computing power changes exponentially, the network will quickly adjust the difficulty and also avoid feedback oscillations.
The difficulty will be adjusted for each new block based on the amount of work done by the previous 144 blocks and the elapsed time.

5. Contract Layer

Encapsulating various types of scripts, algorithms, and smart contracts is the foundation of the blockchain's programmable features. Bitcoin itself has a simple script writing function, and RRChain has greatly enhanced the programming language protocol and can theoretically write applications that implement any function. If you think of bitcoin as a global book, you can think of RRChain as a "global computer." Anybody can upload and execute arbitrary applications, and the effective execution of programs can be guaranteed.

5.1 Script code:

RRCHAIN uses Solidity as a programming language and uses solc as a compiler.

5.2 Algorithm mechanism

After the smart contract is compiled and sent to the web, you can call it via RRCHAIN web3.js javascript api, and you can create a web app that interacts with these contracts.

5.3 Smart contract

In RRCHAIN, smart contracts are written in a dedicated Solidity language, then uploaded to the block and persist. Because blockchain data is secure and immutable, one can be confident that the smart contracts on the RRCHAIN are executed as expected, and that the enforceability of the code makes it impossible for levies and defaults to occur.

Standard transaction and contract transaction

In the RRCHAIN blockchain system, the VM part exists independently of the original scripting language, and transactions are divided into two types, Standard Transaction and VM Transaction.

StandardTransaction: Standard transaction type, we mark it as V1 type;
VMTransaction: Contract transaction type, we have marked V2 type.
Normal bitcoin-like transaction types we have labeled Version1, Version1's transaction maintains similarities to Bitcoin's network standard transactions, and we create a new transaction type for the part that uses the virtual machine, which we label as Version2.
In the Version2 transaction type, we will design the following areas:
Inputs and Outputs There is a "vm" block, with 1 representing the standard transaction type and 2 representing the type of transaction that needs to be performed through the VM.
This new VM contract can be built by creating a new output, and RRCoin Token can then be sent to a contract address through a standard transaction. Tokens can be assigned via new operators, for example, via an assign-to opcode.

RRCHAIN system storage contract code

By extending the bitcoin network's scripting language and adding new operators, such as OP_VM, the contract's bytecode will be encoded into the transaction output. The number of contract bytes depends on the MAX_SCRIPT_SIZE of the system design.

Creation of contracts in the RRCHAIN system

When the contract is created, the contract is established by the creator.

```
contract Escow { address owner   Escow() {
owner = msg.sender   }
}
```

The OP_VM operator is executed through the interpreter of the existing scripting language, and the control is transferred to the VM to execute the corresponding contract.

Contract costs RRCoin

When the contract is output as a transaction, the output can be used as input to the 'send' opcode, and the Send opcode can send V1 and V2 outputs and a certain amount of RCRoin to another output or address.

Let's look at the process of sending an RRCOin to an RRCoin contract.

Assuming that in the 100th block, we have a contract with a TXID of 1234 and an output of 0

```
contract MoneySender{
function sendmoney(outputScript){
send(outputScript, this.balance / 2)
}
}
```

There are two RRCoin standard transactions sent to this contract address

Value of 100 coins at block 150, tx id 1

assign-to 100.0

Value of 50 coins at block 200, tx id 2

assign-to 100.0

We call this contract later in the 300th block

call(contract100.0.sendmoney, myBitcoinAddressScript) //

Finally, we will call Sendmoney and Send opcode. And a valid transaction for this block will contain all the send opcodes after all the contracts have been executed. This will have tx1 and tx2 2 in a transaction in this block. Input, and contains two outputs of myBitcoinAddressscript and change address.

The preservation of contract status in RRChain system Each contract script has its own state. The state of the contract is stored in Statedb. Statedb can be reconstructed through contracts in the blockchain. We call this the reconstruction process. The state can always be rebuilt/created from a contract in an existing block.

Which StateDB should be able to return to the specific transaction to deal with some conflicting transactions or short-term forks in the blockchain. The Berkley DB used by the Bitcoin network is not the best choice.

The status of Statedb will only be affected by the confirmed block. Unacknowledged transactions and the contract itself will not affect the state of Statedb.

Contractual account book and its readability

In the RRCHAIN system, in addition to traceability, we will also build Contract Ledger, a contractual content, to facilitate auditing and reading smart contracts.

In the Ethereum system, the writer of a smart contract may choose not to publish the contract's express content and contract intent.

In the RRCHAIN system, we will build a Contract Ledger to store all the contents of the RRCHAIN's plain-readable text. Users can download their own contract code and contract explanations that are of interest to them through P2P. RRCHAIN client.

The construction of Contract Ledger can bring more transparency, readability, and auditability to the contracts in the RRCHAIN system.

6.Application layer

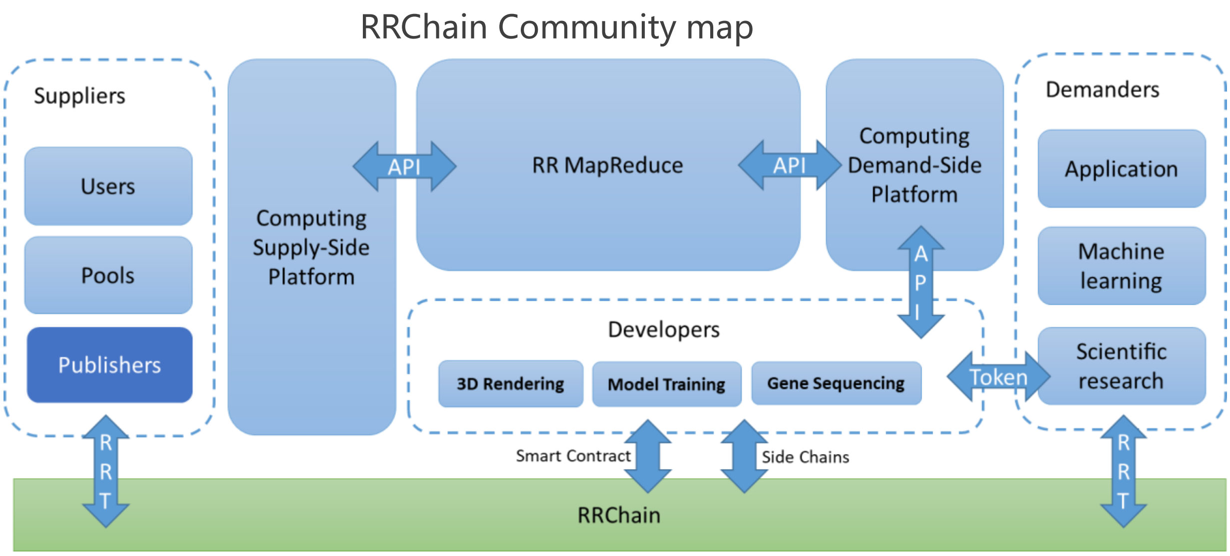
6.1Computing power recycling and regeneration

Based on RRChain's algorithmic features, the operational threshold for user participation in hash calculations can be minimized. The user does not have to download a client program required for general mining. Instead, people can start hashing (mining) by simply opening a browser that supports JavaScript. During calculation, the user can adjust the speed autonomously, and can truly perform various calculations including hash calculation through idle CPUs without affecting the user's experience, achieving a wide collection of “residual computing power”.

At the same time, the advantages of RRChain's natural competitive governance relieve the advantage of mining machine, which makes many individual users willing to join RRChain's community.

This feature of RRChain can provide website owners and software developers with a new profit model in addition to advertising and membership fees. For example, a video website owner can exchange the user's computer's remaining computing power by providing users with free videos, and then sell the computing power the to make profits for other demand parties.

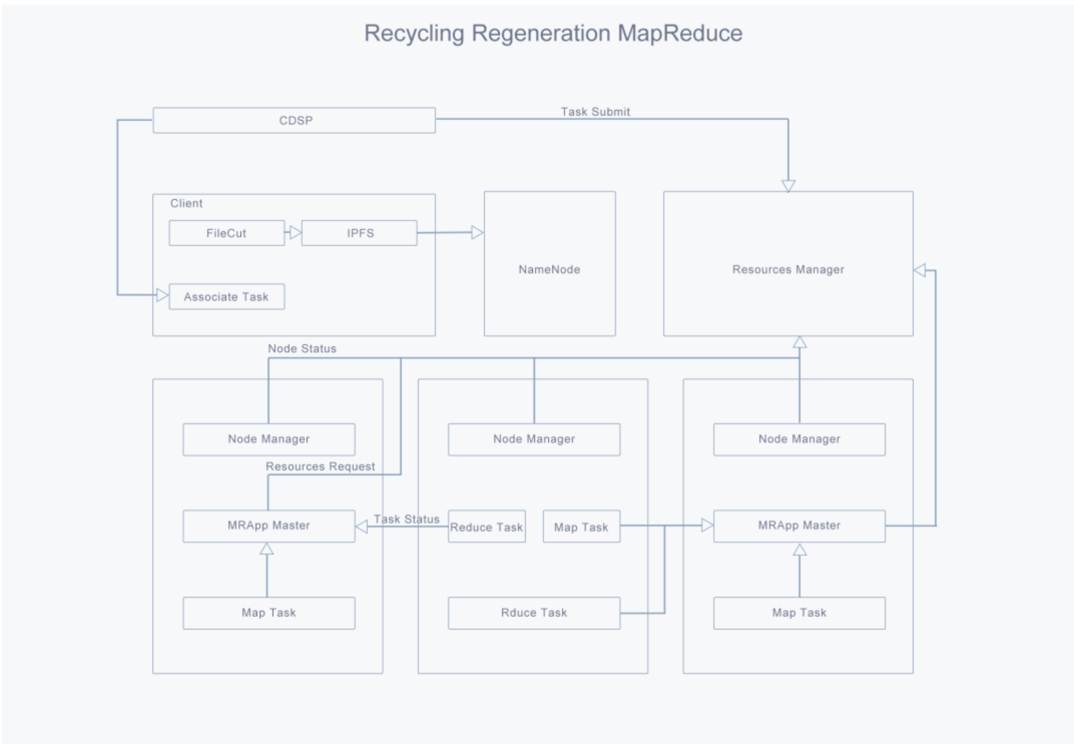
Therefore, based on RRChain, we will construct a brand-new residual computing power collection trading ecosystem.
At the same time, a wide range of third-party developers can develop various types of Dapps based on RRChain and share the computing power in the RRChain community.



After RRChain is released and the user volume reaches a certain order of magnitude, RRMapReduce, a big data distributed computing platform based on the MapReduce model will be released.

The platform mainly solves the problems of subdividing and disassembling of distributed computing tasks and fine management of task distribution of massive micro-computing nodes, adjusting the redundancy of task allocation according to the dimensionality characteristics such as area and time to ensure the balance between resource input and timeliness of tasks.

The product structure design is as follows:



After the release of RRMapReduce, it will provide the market with a cheap computing service solution. Compared to traditional self-built big data clusters or cloud computing services, the computing products generated by RRChain and RRMapReduce will have more price advantages, and they are highly competitive for off-line computing services that require low real-time performance.

The test case is as follows: Log extraction task, extract the valid data in the log.

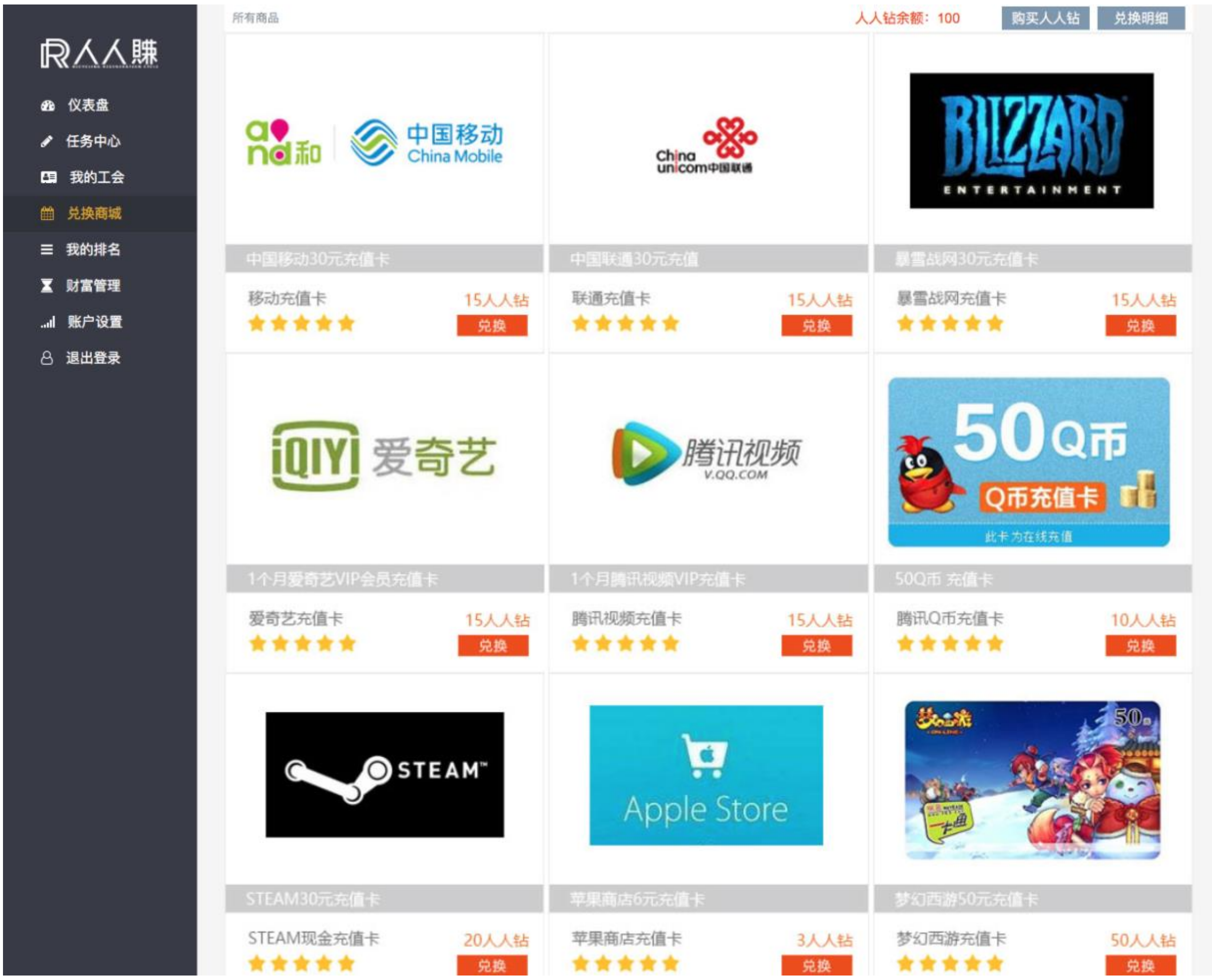
Extraction of 4T original log file, 600G data sent from Mapper to the Reducer, in the 40-node 600-core cluster, by using the Hadoop solution, was divided into 10000 subtasks, the calculation takes 60 minutes to complete. Using Alibaba Cloud EMR costs about 550 yuan.

The same task uses RRMR calculation. It is assumed that each terminal only provides 1 core resource for calculation, and the task is distributed to 6000 nodes. The calculation task time is also 60 minutes. If we reward each node with 5 cents RMB, then completing the same task using RRMapReduce will only cost about 1/2 of Alibaba Cloud. At the same time, if the user continues to contribute one-core computing power, then the user's monthly income is 36 yuan. The income of 36 yuan is already greater than the average monthly fee for most video, game, software and other service providers. The value generated by the remaining computing power when the user uses the computer normally can support the user to use the software for free and enjoy the network service. This will constitute a healthy community ecosystem.

6.2 RRCoin Circulation

Based on the wide applicability of RRChain and high competitive governance capabilities, it is possible to rapidly aggregate many ordinary users into miners and integrate it with their existing network product usage habits. Therefore, it can be imagined that with the development of RRChain, there will be a large base of RRCoin holding groups. How to make RRCOin fast and efficient circulation has become an important factor in maintaining the vitality of RRChain community.

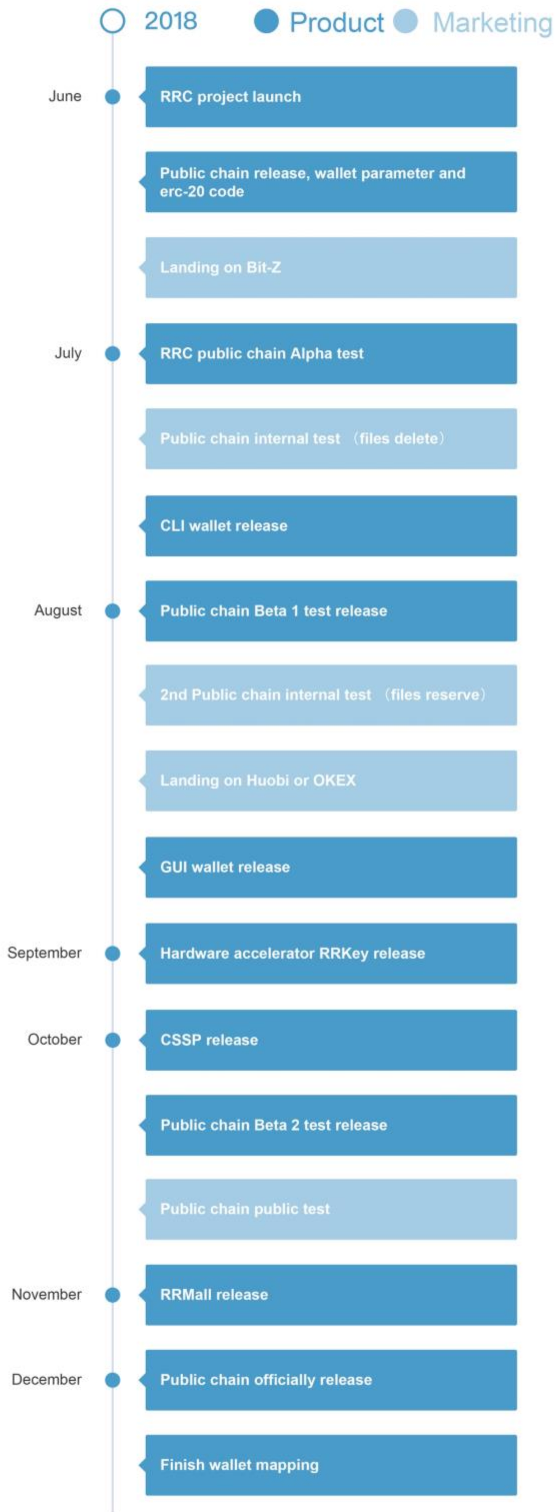
We have designed a shopping system, RRMall that can use RRCoin to quickly and easily consume to meet the needs of users.



RRMall will solve the problem of RRCoin's rapid tradable. We will provide users with a web version of the light wallet, using a semi-centralized approach to deal with the speed of small transactions. When users purchase goods through RRMall, they do not need to wait for a long period of synchronization to complete the transaction quickly.

At the same time, RRMall will also open APIs and SDKs.Thus, many third-party platforms (e-commerce, software, games, etc.) can quickly access and support the payment mode of RRCoin.

Project RoadMap



Team and consultants

Team

Liu Shuang, Founder&CEO

Has 14 years of experience in the digital marketing industry in China. Has been responsible for displaying advertising products for Alibaba Group since 2008, and has established well-known advertising product systems in various industries such as diamond exhibition and TANX, bringing more than 10 billion yuan in annual revenue to Alibaba Group. After leaving Alibaba Group, he joined WPP Group, established the first company based on e-commerce and advertising in the 4A system, called KUVERA, and became the CEO.

Wang Qiang, Chief engineer of the chain

Served as a senior engineer for device-driven of Founder's Electronic Graphics Division , and was responsible for multi-platform driver for Konica Minolta-related devices.. he chief architect of a blockchain project is responsible for the creation, development, and maintenance of the chain. Years of work experience in encryption algorithms and security systems.

Wang Xin, Chief of Product and Marketing

Over 10 years of product design and marketing experience. Used to be the director of product and marketing of Intercom and Youyi Interactive, and now he is the general manager of Behe Adtech. Leading the design of China's first programmatic marketing system.

Consultant:

Shen Bo, Expert consultant

The founder of FBS capital. One of the founders of Invictus

Innovations Incorporated, which is a founding team of BitShares projects. He is also a veteran of traditional finance and has over 12 years of experience in securities, hedge funds and investment banking.

Yuan Ye, Expert consultant

MBA from Tsinghua University. He is currently an executive director of the Shuimu Tsinghua Alumni Foundation, a member of the Expert Committee of the "Internet+" college student Innovation and entrepreneurship competition of the Ministry of Education, and a youth tutor of the China Entrepreneur Training Camp of Tsinghua University. He has many years of asset management experience in the secondary market, early stage investment experience in the primary market, and entrepreneurial experience in the financial data field. He has invested in dozens of start-up companies in the fields of financial technology and intelligence.

Li Zongcheng, Expert consultant

Co-founder of Timestamp Capital, CFO of 8 BTC. Established two major product lines, including 8 BTC and the Bytom chain. 8 BTC provides the underlying basic information and data services for blockchain entrepreneurs and investors. Bytom chain provide solutions for asset registration and circulation on the blockchain. Li successfully combined vertical portals and public chain in an ecosystem.

Feng Chi, Expert consultant

A representative of the first batch of post-90 entrepreneurs, started business at university and quickly obtained multiple rounds of financing. After joining 36Kr, he established a high-quality entrepreneurial board of private entrepreneurs of the Internet "Monsters Academy", and later became the head of the marketing department of the "Whale" marketing division of the primary market data and system provider. He has unique insights into the primary market. He is currently CEO of Genesis Capital and is known as "the investor who knows the most about entrepreneurs.

Liu Junfeng, Expert consultant

14 years of continuous entrepreneurial experience, the founder of the Behe Adtech, used to be the former media director of McCann

Advertising Co., Ltd.; in 2005, founded Kuree (the predecessor of StormPlayer); founded Yoyi in 2007.

Dai Jun, Expert consultant

Worked as senior engineer of Sina Weibo platform; leader of Youyi interactive data project, led the team to design and develop the first DSP system in China; leader of Sohu accurate advertising project, responsible for the design and research and development of Sohu accurate advertising system and AdExchange; 360 commercial products leaders in technology, upgrading the navigation display advertising system; Behe technology CTO, forming a technical team, developed a complete digital marketing system from scratch.

Main Investment