White Paper

Matcoin: The First Blockmatrix-based Decentralized Application Platform

Zhichao, Duan

April 2018

Abstract

The paper introduces *blockmatrix*, a novel data structure different from blockchain. The consensus protocol of blockmatrix is also provided. With the newly designed data structure, blockmatrix resolves the inherent low-capacity problem in blockchain. Instead, a controllable and measurable solution is proposed to expand capacity with the guarantee of security, practicality and complete decentralization. While directed acylic graph (DAG) has been a pioneer in addressing the above-mentioned issues in blockchain, it is not a complete replacement due to its limit in applications. However, blockmatrix is a data struture that can potentially replace blockchain to the entirety. MatCoin, proposed by the creators of blockmatrix, is the new type of cryptocurrency based on the decentralized platform of blockmatrix. Compared to Ethereum (ETH), MatCoin is readily to support the operation of multiple industries. MatCoin continues to adopt proof-of-work (POW) consensus protocol with an estimated block generation rate of every 10 minutes and each block capacity of 1 million, while over 7000 transactions are processed per second. Combing techniques such as the lightning network and separate handling of small payments, the transaction rate can reach over 100,000 each second.

Contents

Chapter 1

Background

In this chapter, we first introduce the background for the birth of blockmatrix, which are blockchain and its decentralized applications. Recent advances based on blockchain as well as their challenges are also discussed. In particular, the efficiency concern of the public blockchain is analyzed as an example to show the limitation of the data struture for blockchain.

1.1 Blockchain and Existing Developments

Blockchain has recently generated a significant impact on business and cryptology. The decentralized applications based on blockchain are redefining business models in a variety of industries such as finance, supply chain, insurance, internet of things (IoT) and philanthropy. Currently, there are two avenues of the decentralized applications: the blockchain created by individual developers or the *private chain*, and the blockchain based on public and decentralized application platforms or the *public chain*. Both chains have its strengths and limits. For example, private chain tends to incur a high expenditure in creations and lacks public creditability. On the other hand, existing decentralized application platforms such as ETH, are often associated with low efficiency, which undermines the potential for large scale business deployment.

1.2 Analysis of Low Efficiency in Public Chain

Essentially, the key to the low efficiency in public chain is the limit of storage capacity at each unit time in blockchain, which is formed by a linked data structure. Next, a brief explanation of the data flow is provided by using transactions per second (TPS) to indicate the efficiency of decentralized

application platforms.

In blockchain, every transaction needs to take place in a block of the chain and the capacity of each block is definite. Therefore, it is also fixed of the maximum number of transactions that can be stored at each block. In a blockchain system, it is expected that every two new blocks can appear with a certain time difference. In other words, the expected time interval of block generation is also fixed. Then the maximum number of transactions stored in one block is the largest quantity of transactions which can be processed in the time interval of block generation. The number of transactions per second is the largest number of stored transactions divided by the time interval of block generation.

Obviously, how to improve the number of transactions per second is equivalent to how to increase the unit storage capacity of the blockchain. Without changing the data structure of blockchain, there are only two options:

- Increase the capacity of each block
- Reduce the time interval of block generation

Currently, there are two types of consensus protocols: proof of work (POW) and proof of stake (POS). POS-based blockchain systems typically increase capacity by reducing the time interval of block generation. Next, we discuss the practicality and safety of a blockchain system based on empirical knowledge, followed by an elaboration on the issues of POS. At last, we conclude the discussion, in hopes to provide readers a clearer picture of why it is such a dilemma to increase capacity within the data structure of blockchain.

Since this manuscript is created for a broader range of audience including both developers and blockchainn euthusiasts, precise definitions of concepts are not given here. Further study and related work will be published on eprint where more academic definition and explanation can be found.

1.2.1 Evaluation Criteria for Practicality and Safety of A Blockchain System

We often see the following situation in reality: In the Bitcoin system, '3 comfirmations' is often used to describe that one transaction is trustworthy and unchangeable. In general, '3 confirmations' means it takes 20 minutes for a transaction from being recorded to being finally confirmed in a blockchain, or two expected time intervals of block generation.

The example above also sheds light on an important and widely-accepted fact: The shorter it takes from recording a transaction to confirming it, the more practical the blockchain system is and vice versa.

Regardless what is transacted, the block gains credibility provided the transaction inside is considered trustworthy.

In other words, for each blockchain, if we remove the k topmost blocks, the rest blocks should be accepted by all the honest nodes and the rejection probability decreases exponentially with respect to k, in which $k \in Z^+$. In fact, this is a critical attribute of blockchain, also referred as *common-prefix property* by cryptologists.

In the above-mentioned '3 confirmations' scenario, $k = 2$. Based on our observation, provided the expected time interval is fixed for generating each block, the smaller k is, the more practical a blockchain system is; the blockchain system might even not be usable with a very large k.

Hence, a criterion for evaluating the practicality of a blockchain system can be described in the following:

- The practicality criterion: for a blockchain system, the expected block generation rate is r, and k is the minimal integer to guarantee the *common-prefix property*. The system is more practical with a smaller kt and vice versa. Therefore, the criterion is defined as kt.

On the flip side, with possible 'bad guys' and forks in a blockchain system, there is still a slim chance that the transaction is not recognized even though it has '3 confirmations'. This also be described as a safety criterion:

- The safety criterion: Suppose the network environment is the same for different blockchain systems (same number of the honest nodes and 'bad guys' as well as same capacity of machines and network transmission), if the k topmost blocks are removed in a blockchain system, the probability is p for the rest blocks to be accepted by all the honest nodes, in which $p \in Z^+$. Apparently, the system is safer with a larger p and vice versa. p is defined as the safety criterion.

As it can be seen, safety consideration is de facto another expression of practicality. Note that the discussion on safety in this paper upholds the premise that a blockchain system is viable. Safety is a direct reflection of the reliability of the system. A poor safety criterion makes a transaction more difficult to be confirmed, which has been proven a key problem in applications. The safety criterion can also be interpreted as an indicator to predict the safety of a transaction within a prefixed duration.

Strictly speaking, as long as a blockchain system is applicable by meeting certain requirements, there is always an eventual moment at which one certain block can be known as accepted, or equivalently the system is safe if the wait time is not a concern. Note that this eventual safe mode does not interfere the significance of the safety criterion discussed earlier.

1.2.2 Scalability in POW

In 'The Bitcoin Backbone Protocol: Analysis and Applications', Juan A. Gray modeled blockchain as a mathematic problem and analyzed the necessary conditions to guarantee the *common-prefix property* and *chain-quality property*, which are two prerequisites for an effective blcokchain system. We provide a few conclusions from the paper above that are relevant to our discussion. Readers are referred to the original article for specific definitions and proof.

Here, we focus the discussion on the two possible venues of scalability in POW:

- Capacity expansion at each block
  The acceptance of block is prolonged once the unit capacity is increased. Of course, it is trivial when the expansion is not significant. However, the added time for block acceptance can not be ignored once the unit capacity surges to a different magnitude such as from 1M to 50M. In the same network environment, this gives rise to an increase of k, the minimal integer to guarantee *common-prefix property*. As a consequence, the practicality and safety of the system drop.
  Meanwhile, a sufficient large unit capacity can cause the dysfunction of the blockchain system since *common-prefix property* and *chain-quality property* are not guaranteed. Therefore, there is an uplimit to which the unit capacity can rise.
  Conclusion: In POW protocol, capacity expansion at each block might lead to a decrease of practicality and safety, and the scalability is bounded by the uplimit of capacity.

- Time reduction for block generation
  The only way to reduce the duration for generating new blocks is to lower the expected difficulty of POW, which incurs k to increase. An enlarged k, together with a reduced t (the expected time interval for block generation) does not necessarily undermine the system's practicality. Nonetheless, the increase of k greatly outruns the decrease of t. As long as t keeps abating, the blockchain system will gradually become less useful. The safety is also expectedly weakened.

  Compared to capacity expansion at each block, the limit for block generation time reduction is even lower. Take bitcoin (BTC) system for example, once the time interval for new block generation reaches 10 seconds (only 60x accelaration), the system is almost unable to function in the current network. In this case, the system is very unsafe because one transaction might not be confirmed in a long time.

Conclusion: In POW protocol, time reduction for block generation might cause the system less safe. The practicality also deteriorates over a long period. The scalability is bounded by the downlimit of time reduction.

1.2.3 Scalability in POS

In essence, POS-based systems scale up capacity by significantly reducing the time of block generation. The chain can be attacked by assembling past accounts with assets in them, or 'history attacks present.' When designing POS, it is hard to complete voting if every vote requires too many stocks. Conversely, the system is vulnerable to attacks if few stocks are required for voting. Diffusion of initial stocks allievates the danger, but does not technically solve the problem. For the majority of current POS projects, initial stocks, by and large belong to the few founders. Undoubtedly, this clustering of stocks increases the risk of potential attacks.

Arguably, the authenticity of each honest node can be protected by the node's self-awareness to deny the longest chain. But new nodes are inevitably susceptible to the lure from attackers. Hence, we are forced to trust certain center nodes which, on the contrary, conflicts the notion of decentralization.

Conclusion: In POW protocol, systems have to trust certain center nodes at the expense of decentralization.

1.2.4 Summary and Discussion

Summary: Capacity expansion in blockchain system is gained by sacrificing key factors including safety, practicality and decentralization.

In particular, it is absolutely vital for a public chain to be practical, safe and completely decentralized. Therefore, it is theoretically impossible to develop a decentralized application platform with high capacity based on the data structure of blockchain.

POW-based system starts to centralize on mining pools in the wake of their advent. However, there are always new mining pools built to compete with old ones as long as economic benefits suffice. Hence, POW can still be considered as decentralized since both theoretically and practically it allows anyone to become a mining pool.

In POS-based systems, those that create foundations are the most correct. If a longer chain by 'bad guys' appears, participants need to trust the server node where the foundation was founded so as to operate properly. In reality, a new block is generated every few seconds in POS-based systems. If nodes of a POS-based system are from the everywhere in the world, it is

also economically significant to own a server with the power to both vote and generate new blocks. Similar to POW, corporate involvement is needed for the generation of new blocks, making it close to the nature of mining pools.

Chapter 2

Objectives of Matcoin

Chapter 1 discussed the limits of data structure in blockchain. In the chapter, a new type of data structure, blockmatrix, is introduced. We first briefly demonstrate its advantages and elaborate its potential as a full replacement of blockchain. Then we will show that Matcoin is a decentralized application platform based on blockmatrix.

2.1 Blockmatrix: Full replacement of blockchain

As discussed previously, it is theoretically impossible to build an application platform based on blockchain while guarenteeing safety, practicality and complete decentralization. The objective of MatCoin is to establish a safe, practical and completely decentralized application platform based on blockmatrix. Based on POW protocol, the new platform can process maximally over 7000 transactions provided the time interval of block generation is 10 minutes and unit block capacity 1M. Combing techniques like lightning and processing small accounts individually, the number of transactions go up to at least 10,000 per second in the current bitcoin network capacity worldwide. In fact, the scalability of this system based on blockmatrix is theoretically indefinite. The only possible limit is the data-processing capability at node servers, which can be addressed by constructing large-scale computing networks while the computing power in hardware also keeps growing rapidly.

Blockmatrix, which MatCoin is based on, is a graphic data structure, notwithstanding that it is far different from directed acyclic graph (DAG). Capacity expansion in DAG faces three major challenges: i) One piece of data is stored in multiple blocks, causing waste of storage space; ii) It is not sure how many new blocks are generated for a time period, or the horizontal development is not limited. This might generate high uncertainty in real

applications; iii) The acceptance rate of a block depends entirely on how many new blocks use this block, making it impossible to measure and predict the wait time for a block to be confirmed. Consequentially, DAG applications are very restricted.

Blockmatrix provides solutions for the challenges above. In a way, blockchain can be viewed as a simplified version of blockmatrix. Projects based on blockchain, once converted to blockmatrix, can scale up while still maintaining safety, practicality and complete decentralization. The capacity expansion has no theoretical uplimit and is bounded by network capacity and server space.

Our MatCoin cofounding team will not only build the decentralized application platform, but we will also help current blockchain-based teams and projects with a smooth transition to the blockmatrix-based platform. For example, we will assist projects on ETH to continue operation on MatCoin, enabling a faster performance and a larger capacity for user accommodation.

2.1.1 The decentralized application platform of MatCoin

MatCoin hopes to build an application platform similar to an operating system for users and developers. In comparison with current platforms, MatCoin improves the user and developer experience in the following three ways: 1) account and permission; 2) smart contract and resource allocation; 3) script and virtue machine.

Chapter 3

Implementation of Matcoin

In this chapter, section 3.1 introduces the new data structure: blockmatrix. 3.2 discusses how blockmatrix generates, validates and accepts blocks. How blockmatrix encodes transaction information is shown in 3.3, followed by the discussion of forks in blockmatrix in 3.4. The 'length' protocol in blockmatrix is given in 3.5. 3.6 illustrates the overall operation mechanism in blockmatrix-based systems, or the consensus protocol. 3.7 provides a brief theoretical analysis of blockmatrix. Last but not least, 3.8 and 3.9 describes how to build a decentralized application platform based on blockmatrix.

3.1 The Data Structure of Blockmatrix

Blockmatrix begins with n independent and equivalent genesis blocks. n is a constant defined before the advent of the system.

To help understand, we start the discussion by creating n blockchains based on the n genesis blocks. These blockchains are parallel as shown in Fig.1.

Each block on the n parallel chains is designated with a unique set of coordinates (i, j). Here, i represents the ith genesis block and $1 \leq i \leq n$, and j is the height of the block on the chain and $j \in Z^+$. An example of blocks with their coordinates can be seen in Fig.2 when $n = 3$. Unlike blockchain, blockmatrix requires coordinates be encoded by the block header at the current block, denoting the location in the matrix.

Each block in blockchain only needs to reference the hash value of the previous block. But in blockmatrix, every block needs to reference the hash values of all newly generated blocks in the rest of $n - 1$ chains, in addition to the hash value of the previous block in the same chain. Suppose a node produces a block at (i, j) $(j \in Z^+)$, $S_{i,j-1}$, the set of blocks at $(i, 0)$,...,$(i, j -$

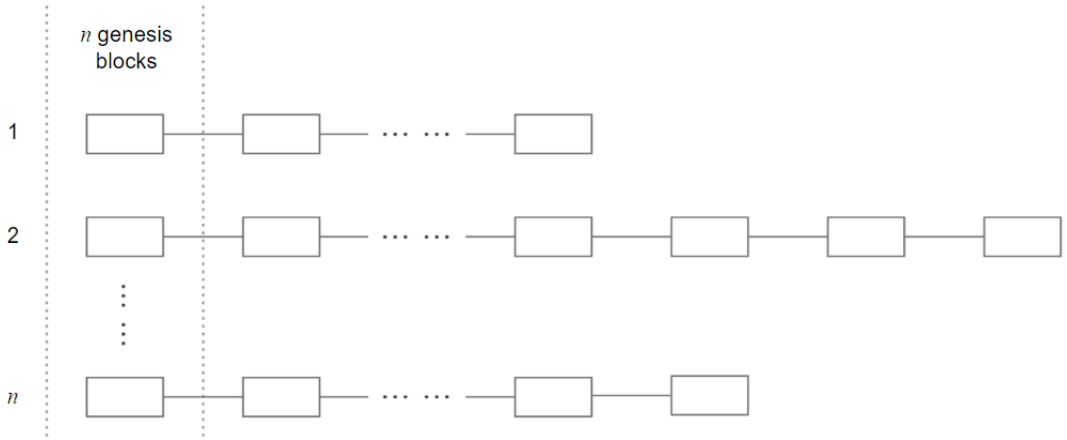


Figure 3.1: n parallel blockchain

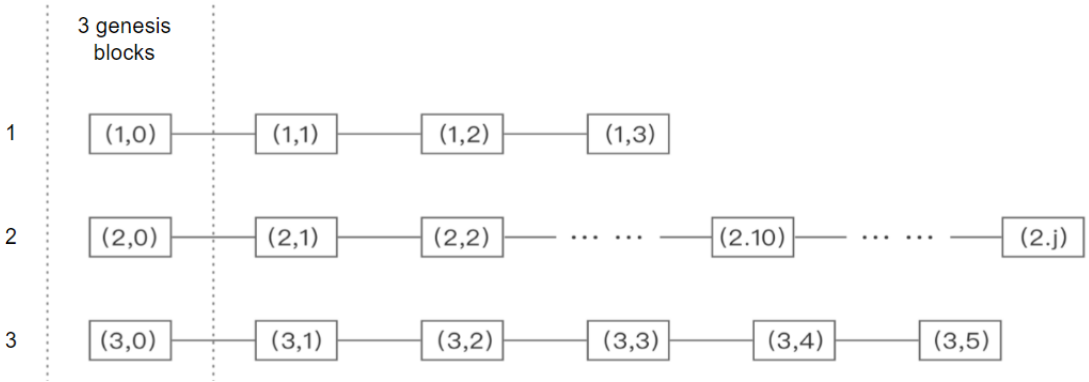


Figure 3.2: Block coordinates at $n = 3$

1) is encoded, then the set of all accepted blocks at current node $S_{i,j}$ is also encoded. Obviously the difference of sets $S_{i,j-1}$ and $S_{i,j}$, or $S_{i,j} \setminus S_{i,j-1}$ contains at least one element, the block at $(i, j-1)$.

To reference $S_{i,j-1} \setminus S_{i,j-1}$, all blocks in the set are sorted by coordinates to construct a Merkle tree. The hash value at the root node in the tree is the referenced hash value. Again, hash values and coordinates of all referenced blocks are included in the block header.

So far we have provided the basics of the data structure in blockmatrix. As we can see, the cross-reference among blocks in blockmatrix forms a 'graphic' data structure with the width of n. Fig.3 gives an example of the graphic data structure when $n = 3$ in blockmatrix.

3.2 The Growth of Blockmatrix

The growth of blockmatrix can be categorized to three steps: i) block generation; ii) block validation; iii) block acceptance. In this section, we discuss these steps regardless of the encoded content, since it also endures the process of generation and confirmation.

3.2.1 Block Generation

When a node generates block (i, j), a consensus protocol P, either POW or POS, is required. As discussed in 3.1, block (i, j) can be directly generated after block $(i, j-1)$ without considering blocks in other locations are generated correctly, as long as the new block accepts and is referenced by the the current node. This behavior can be viewed as the n blocchains generate blocks independently.

For the n chains which have grown from the original n genesis blocks, blocks generated from any two of those chains are independent. In other words, blocks in one chain is not effected by the delay or advance of block generation in another chain. In Fig.4 where $n = 3$, a node receives block (2,2) only after it starts generating block (1,3). Then block (2,2) is not referenced.

If POW is adopted, the difficulty for the current block is determined only by blocks at $(i, 0)$,...,$(i, j-1)$.

For POS-based systems, POS decides the number of votes at the current block and number of stocks required in every voting. Because the n chains grow separately in blockmatrix, POS reduces the total stocks required to generate each block.

As a result, the computing power in POW is distributed to each chain and so are stocks required in each POS voting. A relatively even distribution can be achieved by designing reward mechanisms, but not necessary since blockmatrix has no such requirement.

3.2.2 Block Validation

When a node receives a block at (i, j) $(j \in Z^+)$, three criteria needs to be validated regardless of the encoded content:

- the block meets the requirements in the consensus protocol P.
- one coordinate (i, j) can not accept two identical blocks.
- all blocks referenced by the newcoming block at (i, j) should be accepted by the node. Alternatively, the set of referenced blocks at (i, j)

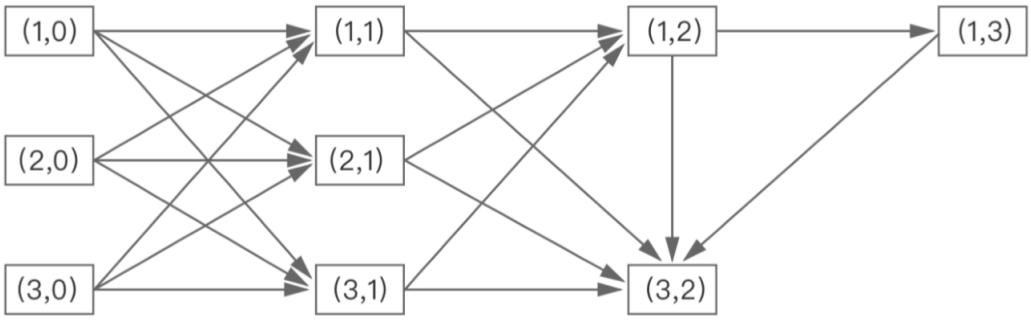


Figure 3.3: Note the referencing relationship between blocks, assuming blocks (1,1), (2,1), (3,1), (1,2), (1,3) and (3,2) are generated in chronological order.

for the newcoming block should be a subset of all blocks in the blockmatrix.

Two identical blocks should have the same coordinates and hash values. Note that when the content encoded in the two blocks is considered, two blocks may have different hash values if their content is different.

Without considering the content, two blocks at (i, j) and (i', j') is acceptable if the three criteria above are satisfied.

3.2.3 Block Acceptance

It is the same to accept a block in blockmatrix and blockchain, as long as it is referenced when a new block is generated.

3.3 Blockmatrix Encoding

There is one core principle of encoding in blockmatrix: if one type of content is encoded in a block, the content encoded in the rest of blocks should be the same type. For example, if digital currency is produced in a block, the rest blocks also produce and circulate the same type of digital currency. The status change of the digital currency can be encodeded on any block in the blockmatrix as long as it meets the principle.

The rationale behind the principle is that blockmatrix is one entity. It is pointless to consider the environment of each block individually. In consortium blockchains, blocks in two chains are prohibited to reference mutually. Hence, digital currency does not circulate directly in consortium blockchains

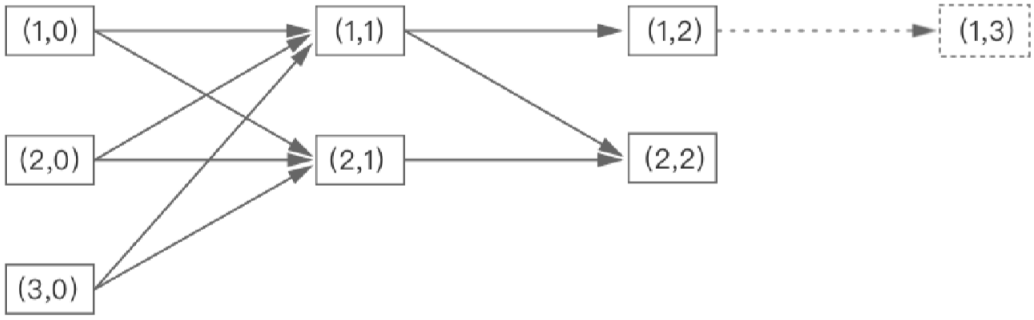


Figure 3.4: Having received (1,2), the node starts to generate (1,3). It also receives (2,2) before (1,3) is produced. Then block (2,2) is not referenced due to the independence of block generation.

and needs to a medium to transit. This is different from the dynamics in blockmatrix.

In a blockmatrix system, every newly generated block will receive the broadcast of all information. From 3.2 we know block generation can be considered as an independent process in n chains from the n genesis blocks. By mapping the encoded information to the n chains, we guarantee each block will not encode the same content.

One way to build mapping is to change the content to a string in json format, then take the hash value of the string, convert the hash value to a decimal number and evetually take the modulo of the decimal number.

Since every honest node receives the broadcast of all information, if two contents conflicts each other, the blockmatrix system will demend the two contents be discarded and not encoded. For the blocks generated by honest nodes, the contents of two blocks at (i, j) and i', j' does not conflict.

3.4 Blockmatrix Fork

A fork in blockmatrix incurs when there is a conflict between blocks from different nodes. The nature of this conflict is the conflict between two blocks, since a conflict among multiple blocks can be seen as multiple conflicts between two blocks. Conflict will not exist in blockmatrix if there is no conflict between any pair of blocks.

During the growth of blockmatrix, forks are caused by two possible reasons:

- Two blocks have the same coordinates

- There is a conflict of content encoded in two blocks at i, j and i', j', $i \neq i'$.

Fig.5-1 and Fig.5-2 demonstrate a fork at two different nodes when $n = 3$.

From 3.2 and 3.3, we know that the content encoded in two blocks at i, j and i', j' respectively should not conflict if blocks are generated by honest nodes. The second reason above for the existence of a fork is fundamentally due to the Byzantine node.

In a peer-to-peer (P2P) network, a node does not receive the information from other nodes in real time. Assuming in a P2P network, the duration for a node from generation to acceptance is Δ. A conflicting block is generated after another block with a duration of *Delta*. Then this does not cause a fork because the honest code has already accepted the block produced earlier. Hence, if there is a fork in blockmatrix, a pair of conflicting blocks are generated within a duration shorter than Δ.

Same as in blockchain, forks eventually converge in blockmatrix. Therefore, *common-prefix property* remains in blockmatrix. In 3.7, we will show that *common-prefix property* holds in blockmatrix if it also holds in blockchain, provided the network environment stays the same.

3.5 The 'Length' Consensus in Blockmatrix

In a network environment, forks are inevitable in blockmatrix as the complexity keeps building on the n genesis blocks. Similar to blockchain, we choose the 'longest' blockmatrix as the corrent blockmatrix. Specifically, the 'length' of a blockmatrix is determined by the following two factors:

- L, the total number blocks in the blockmatrix
- W, the total weights of all blocks in the blockmatrix

Consider two random blockmatrices A_1 and A_2 with 'length' of (L_1, W_1) and (L_2, W_2) respectively, the comparison is defined as the following:

- If $L_1 > L_2$, then $(L_1, W_1) > (L_2, W_2)$
- If $L_1 = L_2$, $W_1 > W_2$, then $(L_1, W_1) > (L_2, W_2)$
- If $L_1 = L_2$, $W_1 = W_2$, then $(L_1, W_1) = (L_2, W_2)$

The motivation to consider weights is to reduce the probability of forks and enable a faster selection of the correct fork.

If there are Byzantine nodes in the network, we define the following two conditions as the minimum requirement:

- For two conflicting blocks, weights are not known before generation
- For two conflicting blocks, allow them to grow till the final 'length' of (L_1, W_1) and (L_2, W_2), respectively. If $L_1 = L_2$, the probability of $W_1 > W_2$ is much greater than $W_2 > W_1$

In other words, weight can not predicted. When a fork appears, it is wise to choose the blockmatrix with a large 'length.'

Block weight can be defined as follows. Consider a set W of m sorted arrays, each element in W is $(W_1, W_2, ..., W_m)$ of which $W_i \in Z^*$ and $1 \leq i \leq m$. Next, we define the operations of addition and comparison. For any two elements in W, $(W_1, W_2, ..., W_m)$ and $(W'_1, W'_2, ..., W'_m)$:

- the addition operator +: $(W_1, W_2, ..., W_m) + (W'_1, W'_2, ..., W'_m)$ $= (W_1 + W'_1, W_2 + W'_2, ..., W_m + W'_m)$
- the equivalence operator =: $(W_1, W_2, ..., W_m) = (W'_1, W'_2, ..., W'_m) \iff W_i = W'_i$, $1 \leq i \leq m$
- the less than operator ¡: For the smallest i that makes $W_i \neq W'_i$, $(W_1, W_2, ..., W_m) < (W'_1, W'_2, ..., W'_m)$ if $W_i < W'_i$

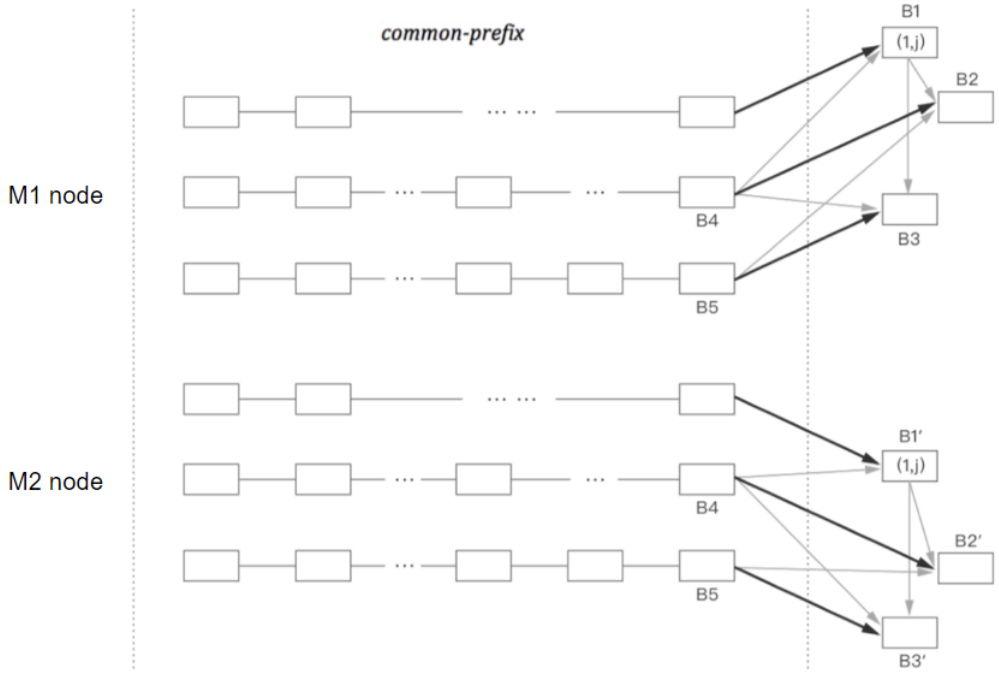
The operations above still holds when m is positive indefinite. For a blockmatrix with n genesis blocks, we use $H(B)$ to denote the hash value in decimal of block B at (i, j) and s as the safety parameter (preferably large), the weight of B is $W(i, j) = (W_1, W_2, ..., W_{(j+1)n^3})$. In addition, $\forall i \neq H(B) mod sn^2, W_i = 0$ and $\forall i = H(B) mod sn^2, W_i = 1$. The rationale of introducing n^2 and s will be discussed in 3.7.

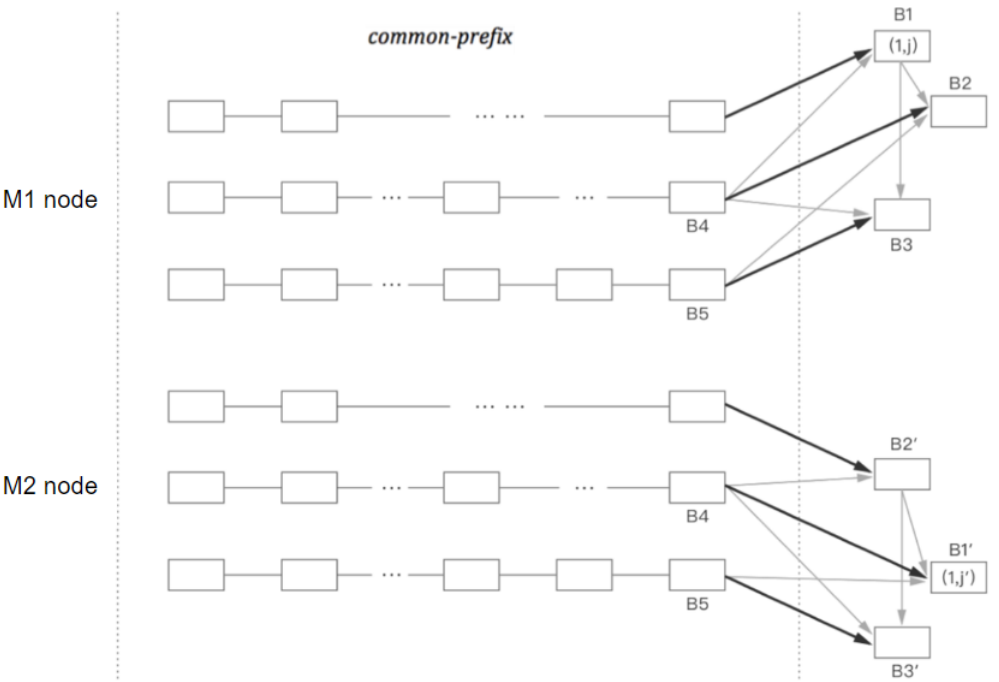
3.6 The Consensus Protocol in Blockmatrix

From 3.1 to 3.5, we have introduced the basic concepts of blockmatrix. Next we focus on the necessary agreements in a blockmatrix system, or the consensus protocol of blockmatrix. Every node acts according to the protocol and eventually all honest nodes form a blockmatrix with consensus. Here, we provide the general idea of the consensus protocol. More details of the protocol can be seen in articles following the whitepaper.

Specifically, the consensus protocol can be described in following steps:

- When a new transaction takes place, it will be broadcast to all the nodes in the P2P network


common-prefix
M1 node
M2 node
B1
(1,j)
B2
B3
B4
B5
B1′
(1,j)
B2′
B3′
B4
B5


common-prefix
M1 node
M2 node
B1
(1,j)
B2
B3
B4
B5
B2′
B1′
(1,j′)
B3′
B4
B5

- Each node first sorts the transactions received through the broadcast. While sorting through each transaction, the node also checks whether the transaction meets the requirements. Transactions falling short of the requirements are discarded. The rest are packaged according to different transaction channels and then put into the block of the corresponding blockchain
- Each node generates new blocks as described in 3.2.1
- When a new block is generated at a node, it is broadcast to the all the nodes in the P2P network
- When a node receives the new block broadcast, it needs to verify:
  - whether the content encoded in the block meets the requirements
  - whether the block meets the three criteria in 3.2.2
  - if the received block conflicts with previous ones, the blockmatrix is considered to have a new fork and a backup space is initiated. According 3.5, the longest fork is chosen for the current blockmatrix. If two forks have the same length, the one with an earlier timestamp is chosen.
- The blockmatrix is updated when the node accepts the block and creates a new block according to 3.2.1

3.7 A Brief Analysis of *Common-prefix Property* and *Matrix-quality Property* in Blockmatrix

To replace blockchain entirely, blockmatrix also needs to have three attributes: *common-prefix property*, *matrix-quality property* and *matrix-growth property*. First, we give a description of the three and then elaborate more on the first two.

- *common-prefix property*: The blockmatrix maintained by honest nodes (players) possess a large common prefix. More specifically, for a blockmatrix system with n genesis block, if any honest node prunes (i.e. cut off) k blocks from the end of each chain generated by the n genesis blocks, the probability that the remaining pruned chains will not be mutual prefixes of another honest node drops exponentially in the

safety parameter. The blockmatrix is of no value without consensus manifested by the common-prefix property.

- *matrix-quality property*: In the blockmatrix maintained by honest nodes, the number of blocks contributed by honest nodes should be greater than that by 'malicous' parties. In fact, if this property fails to hold, the blockmatrix system is controlled by 'malicous' parties and therefore loses its meaning.

- *matrix-growth property*: The blockmatrix maintained by honest nodes keeps growing with more blocks generated, verified and accepted as time goes. A majority of new blocks are generated by honest nodes so the content spread by honest nodes can be encoded in the blockmatrix at a certain time.

Any functional, decentralized system must have these three properties. The essence of a decentralized system is to find consensus in a fault-tolerant and distributed system. Hence, the applicability of a system can only be achieved under certain constraints. The practicality and safety of a system in Chapter 1 are indicators of applicability. Strictly speaking, these two indicators should be quantified by parameters that are able to impact the outcome, instead of obtained via qualitative analysis. In blockchain, a system is applicable if it complies with the three properties. From the applicability, we can also derive different insights such as a shorter duration of block generation will require the ratio of the blocks owned by 'malicous' parties to all the blocks to decrease. Then the indicators we defined in this paper will not be able to reflect the similar insights. Hence, indicators like safety and practicality are defined to help developers and users to better understand the concepts. More strict and accurate constraints will be given in future articles.

3.7.1 *Common-prefix Property*

Note that we will not discuss how Byzantine nodes prolongs blockmatrix by creating more blocks or the double-spend attack since conclusions for these are fundamentally similar to blockchain. Instead, we focus on new issues resulting from the data structure of blockmatrix. The purpose of this section is to demonstrate that the probability of forks is lower in blockmatrix than in blockchain by judging the 'length' of a blockmatrix. This can allow capacity expansion with the same practicality and safety.

One new issue is that forks in blockmatrix undermines the *common-prefix property* of a blockmatrix system.

Reason: Suppose the probability that a node in a blockchain system generates a new block in a round is f. If we distribute the computing power at this node to the n chains (from n genesis blocks) in blockmatrix, and the probability of generating a new block at each chain stays the same (the distributed computing power does not affect block generation in blockmatrix), then the probability that a node in a blockmatrix system generates a new block in a round is $f_1 = 1 - (1 - f)^n$. Apparently, forks are more likely to appear as the probability of block generation increases. Next, we analyze what is probability of a fork in blockmatrix.

Consider k nodes are generating new blocks in the blockchain system, the probability of newly generated blocks in a round is $f_l = 1 - (1 - f)^k$. The probability of newly generated blocks from only one node is $f_l = k(1 - f)^n$. Hence, the probability that more than one node generate blocks in a round is $f_l - f_{sl}$. Note that when $f \in (0, 1), k \geq 2$, we can obtain $f_{sl} = kf(1 - f)^{k-1} \geq kf(1 - f)^k \geq f_l - f_l^2$ through Bernoulli's inequality. Therefore, in the blockmatrix system, the probability of newly generated blocks in a round is $f_m = 1 - (1 - f_1)^k$, and the probability that more than one node generate blocks in a round is $k^2 f_1^2 \leq k^2 f^2 n^2$. We can see that the probability that more than one node generate blocks in a round is less than 1, whether it is in blockchain or blockmatrix.

In fact, if a blockmatrix start to fork, it will also end soon if one fork has a bigger 'length' in the next round of block generation. If forks remain, it is because new blocks are generated 'simultaneously' and forks have the same 'length'. Here, 'Simultaneously' means the duration for a block from generation to acceptance is smaller than Δ.

In the same network environment, if we can ensure the probability of forks in blockmatrix is lower than that in blockchain as well as the *common-prefix property* holds in blockchain, then the *common-prefix property* also holds for blockmatrix with the same practicality and safety.

In 3.5, by selecting reasonable weights of blockmatrix, we enable the probability of forks to be lower than that of blockchain. sn^2 in 3.5 is the generalization of $k^2 f^2 n^2$ in the analysis above. The bigger s, the probability of forks in blockmatrix is more likely to be lower than that of blockchain.

A lower probability of forks in blockmatrix implies the following: For any block in the n chains (from the n genesis blocks), if the block is confirmed in one chain, the blockmatrix is also confirmed provided the consensus protocol and network environment remains the same. Without any doubt, this means blockmatrix improves the capacity from blockchain while still maintaining the same level of practicality and safety.

3.7.2 *Matrix-quality Property*

As we know from the analysis in 3.7.1, if k nodes are generating new blocks in the blockchain system, the probability of newly generated blocks in a round is $f_l = 1 - (1 - f)^k$. Suppose malicious nodes have the same computing power as honest nodes and all of them attach one of the chain. In other words, malicious nodes have a computing power kn times of a honest node in one chain, then the probability that malicious nodes generates new blocks in this chain is $1 - (1 - f)^{nk}$. Hence, the probability of obtaining new blocks is the same whether the computing power is focused or distributed.

In blockmatrix, the outcome is the same from attacks with either distributed computing power or distributed one. As long as honest nodes are at advantage, the constraints to satisfy *matrix-quality property* in blockmatrix are more or less the same as in blockchain with little impact on practicality and safety.

3.8 Implementaion of the MatCoin system

The implementation of the MatCoin system is based on blockmatrix from 3.1 to 3.6. In this section, we introduce the implementation plan of MatCoin public matrix from five segments: 1) the selection of n in blockmatrix; 2) the selection of consensus protocol, unit block capacity and expected duration of block generation; 3) rewarding mechanism for block generation; 4) solution for encoding transactions in blocks; 5) solutions for distributing computing power to the n chains from the n genesis blocks. The functions of MatCoin as a decentralized platform such as smart contract will be introduced in the next section.

- In MatCoin blockmatrix, $n = 1024$. 1024 genesis blocks will be part of the MatCoin consensus protocol.

- MatCoin will use POW-based consensus protocol with the maximal size of a block as 1M and the expected duration of block generation 10 minutes. What is different from BitCoin is that the block difficulty will be adjusted for every 24 blocks, or every 4 hours, ensuring every of the 1024 chains can generate a block within 10 minutes. The ideal scenario is there are 1024 blocks generated every 10 minutes with different $i-$coordinate.
  In the future, MatCoin will change to POS+POW. POS+POW will be a novel and efficient Byzantine consensus protocol which will 'borrow' POW from the world after choosing the candidate of POS. The

POS protocol will be inspired by Algorand. We decide not to use the combined protocol at the beginning because fakes are highly likely if there are not enough POW-based computing power available to be borrowed. A less-than-ideal POS will also lead to the issue of trusting certain nodes, which is the opposite of decentralization.

- Like BitCoin, every mined block in MatCoin will contain mining rewards and transaction fee. In MatCoin, every block will have the digital currency with the same name MatCoin. The transfer and generation of this currency is reflected in the growth of blockmatrix. The allocation of MatCoins will be explained in Chapter 5.

- MatCoin will use the same transaction encoding as ETH: the account status rather than UTXO. The reason is that MatCoin aims to maintain a public ledger of a massive size, so we hope to access the information we want from the massive database while also minimizing the use of resources on computers.
  In MatCoin, all transaction records will be sorted and then encoded in each block without repetitive encoding as shown in 3.3. This will avoid the waste of storage space as DAG.

- In 3.2.1, we introduced how to generate blocks in blockmatrix. The computing power will be naturally distributed to 1024 chain corresponding the 1024 genesis blocks. In MatCoin, we will provide user interface and enable users to decide the allocation of computing power.

Based on the 5 segments above and the key concepts from 3.1 to 3.6, we can build the first publc matrix based on blockmatrix: MatCoin.

3.9 MatCoin Platform Functions

The objective of MatCoin is not just a public matrix based on blockmatrix. Instead, it aims to become a real decentralized application platform. Hence, we hope to make it simple and convenient to build decentralized applications. By learning the experiences from current ETH projects, we will optimize the the following functions as a start for developers and users:

- Account and permission
  As a decentralized application platform, it is necessary to provide information of accounts and their permissions. We use demo@domain as an example to introduce naming rules. '@domain' is exclusive for a

completely decentralized application platform. Each user is entitled to an account when s/he starts to create and particpate smart contract. It will cost a large amount of MatCoins to create an account with '@domain', whereas a child account of '@domain' is cheaper. Role-based permissions will also be provided for creations such as EOS.

- Smart contract and resource allocation
  All smart contracts are based on decentralized applications. In MatCoin, an application is required before the corresponding smart contract can be used. To some extent, it seems to hamper the development of smart contract. Nonetheless, this will spawn more decentralized applications and in turn, provide more justification for existing contracts. Application owner will have at least one account with @domain. Account transfer between owners is also allowed.

  Compared to the free transaction in current decentralized applications, we consider public ledger is a public resource. If a transaction is free, then public resource is free. However, we believe it is necessary to pay to use public resource. Otherwise, a public ledger can be filled with trivial data. In an environment of free resource, one often prioritizes to throw the data into a public ledger rather than develop a third-party decentralized application. This benefits neither the development of decentralized applications nor the interest of miners.

- Script and virtual machine
  We adopt the Ethereum Virtual Machine (EVM) as a start for script and virtual machine. A new script and virtual machine will be added in the future in hopes to save developers from learning a new development language.

Chapter 4

Initial Team

Zhichao Duan: The creator of blockmatrix; Founder of Huangzhou Jiudeng Network Technology Company; Mathematics major at Zhejiang University

Xue Bai: Former CEO of Guojun Jewelry; Many years of experiences of business operations and strategy in finance industry; Graduated from Chinese University of Hong Kong

Jianhua Li: Dean of School of Cyber Security, Shanghai Jiao Tong University; Director of Engineering Research Center for Network Information Security Management and Service of Chinese Ministry of Education; Director of Shanghai Key Laboratory of Integrated Administration Technologies for Information Security

Chen Cui: Post-doctoral research fellow for machine learning and artificial intelligence at Harvard University

Chapter 5

Solutions of Matcoin Allocation

The MatCoin system will issue 67,276,800 MatCoins in total, of which 13455360 will belong to the MatCoin foundation to support its development in the future. The rest 53,821,440 will be mined evenly in the 3650 days following the start of the project. Each mined block will be rewarded with 0.1 MatCoin. We expect a new block to be generated every 10 minutes and the total numnber of genesis blocks is 1,024. Hence, overall the number of minable MatCoins is 53,821,440.