MCV: Mutual Chain Voting

Asynchronous BFT Protocol

as a distributed registry for ULTRANET infrastructure

(continuously updated)

www.ultranet.org

**Abstract.** There are several existing ABFT protocols that offer the highest throughput and minimal transaction fees, but they all have their own drawbacks and vulnerabilities. The most effective of them in terms of performance are those based on the Directional Acyclic Graph (DAG). However, classic DAG protocols require master nodes to prevent forks, double-spends and other attacks having a negative impact on network decentralization and reliability. Other approaches, such as Hashgraph, do not rely on master nodes but their confirmation time grows dramatically if the number of nodes is relatively high while network activity is relatively low. MBV is designed to achieve a throughput and cost close to those of DAG protocols but without sacrificing decentralization and scalability. It is based on a special data structure called Roundchain. Roundchain is similar to Blockchain but instead of chaining single blocks it chains sets of blocks called rounds. This allows all, or almost all, “members” to get their blocks added at each chain iteration, in contrast to a traditional blockchain where only one “winner” can place their block onto a chain at a time. MBV protocol, in turn, provides a simple voting mechanism that allows for reaching consensus for every round. Together, MBV and Roundchain make the technology inherently multithreaded, providing performance limited only by network throughput and without sacrificing decentralization and reliability.

***Disclaimer:*** *This document is for information purposes only. Ultranet Organization does not guarantee the accuracy of, or the conclusions reached in, this document, and this document is provided “as is”. Ultranet Organization does not make, and expressly disclaims, all representations and warranties, whether express, implied, statutory or otherwise, whatsoever, including, but not limited to (i) warranties of merchantability, fitness for a particular purpose, suitability, usage, title, or non-infringement; (ii) that the contents of this document are free from error; and (iii) that such contents will not infringe third-party rights. Ultranet Organization and its affiliates shall have no liability for damages of any kind arising out of the use of, reference to, or reliance on this document or any of the content contained herein, even if advised of the possibility of such damages. In no event will Ultranet Organization or its affiliates be liable to any person or entity for any damages, losses, liabilities, costs, or expenses of any kind, whether direct or indirect, consequential, compensatory, incidental, actual, exemplary, punitive, or special for the use of, reference to, or reliance on this document or any of the content contained herein, including, without limitation, any loss of business, revenues, profits, data, use, goodwill, or other intangible losses.*

# Contents

[Contents 3](#_Toc70068030)

[Glossary 4](#_Toc70068031)

[Consensus Algorithm 6](#_Toc70068032)

[Ultranet Specifics: Emission 10](#_Toc70068033)

[Ultranet Specifics: Fees and Costs 11](#_Toc70068034)

[Conclusion 12](#_Toc70068035)

# Glossary

**Account** – a cryptographic RSA public/private key pair

**Member** – an account who declared itself as a potential block issuer by placing a special transaction and notifying others that it is online. Members are eligible to generate blocks and participate in a consensus algorithm

**Block** – an element of a round that contains a set of transactions

**Transaction** – a cryptographically signed data structure that describes some write operation on a database

**Round** – a set of blocks that are sorted by their hashes. Each round has its own sequence index. Each member can place only one block per round.

**Parent round** – a round Ri is a parent of round Rj when i = j- Pitch, where Pitch is a constant. The rounds in the range [0… Pitch] are called *genesis rounds* and have a parent round reference set to zero.

**Roundchain** – a sequence of rounds where each block in each round has a reference to a parent round.

**Pitch** – the number of rounds between a round and its parent round

**Node** – a network computer with special software that can receive and validate blocks from other nodes, send its own, and relay received blocks to other nodes and participate in the MBV consensus algorithm

Figure 1. The chain of block rounds

A

B

Members

C

D

E

F

Ri

Ri+1

Ri+2

Ri+3

Ri+4

Ri+5

Ri+6

Ri+7

Ri+8

Ri+9

Rounds

- Normal Block

- Vote

Time

# Consensus Algorithm

The idea behind MBV is to create a distributed ledger algorithm with the following requirements:

* No Proof-of-Work, as it has lowest speed and highest hardware requirements among other algorithms
* No Proof-of-Stake, as it makes the richest miners take maximal profits so the rich get richer. It is also relatively slow without sharding
* No traditional blockchain, due to the principle of “one winner – one block”, which makes it slow and expensive
* No DAG, as despite its high speed and cheapness, it requires masternodes to prevent forks, which have a negative effect on the decentralization
* No Hashgraph and similar, as it can operate fast only with a very limited number of members, above which the confirmation time grows dramatically
* Minimal effort to create blocks
* Minimal possible transaction fees
* More than one block can be accepted at a time.

The core of MBV is a data structure called Roundchain, which represents a 2D grid of blocks where the column is a timeline of a particular member (like the blockchain of a single member) and the row (round) is an opportunity but not an obligation for each member to place its block there.

Each block refers to a whole parent round with some delta called Pitch. This guaranties cryptographic integrity for all data stored in all rounds similar to the ordinary blockchains.

In order to become eligible to add blocks, a particular account must satisfy some age and bail requirements. This measure limits the number of members, which is important for the voting mechanism. Any particular member can add only one block per round. Both these restrictions prevent the grid from growing infinitely in a horizontal direction. Vertical growth for each member, in turn, is limited by accepting only those blocks whose round lies in a specific range. This range is needed because the order in which a node receives blocks is not predictable for recent ones due to the nature of peer-to-peer networks. The advancement of this range is driven by a voting mechanism.

The following steps are required for an account to enable block production:

1. Have a non-zero balance.
2. Run a node and synchronize with the network.
3. Send and await a confirmation of a special transaction called “Declaration”, which locks some amount of tokens and tells the others that the account is going to act as a member
4. Send a special empty block that tells others that the member is online and ready to send blocks and vote for rounds.

A user can choose any member for sending transactions with its blocks. Unlike known single‑chain technologies, there is no competition between members as block-creation efforts are negligible and there is no difference for senders between all available members. This will force members to compete with each other and so to keep fees as low as possible, and also means that transaction pool overflow is not possible.

Each member can generate only one block per round – which prevents a chain from forking. If a member creates two different blocks with the same round index, then the network treats them as a cheater, seizes its bail, and distributes it between round members.

To prevent members from adding the same transaction to more than one block in the same round each transaction hash is generated from data that includes the chosen member account (its public key). Another, constrain “Maximum Round” prevents a member from deferring a transaction placement indefinitely long.

To be eligible to generate blocks, an account must “declare candidacy” by adding a special transaction to the ledger that locks some amount of tokens used to determine the member’s rank and to punish it in the event of malicious activity. If there are more than Mmax members present in the network then they are sorted by its age-bail rank (account age divided by accession bail) and only top Mmax members are eligible to create blocks and vote for rounds.

For each round, the algorithm expects all or top Mmax members to either create a normal block or send a vote block – either of them acting as a vote for a parent round. A vote block has no transactions and is not stored in the Roundchain. In this way, members can vote for a particular candidate (a subset of blocks) of a parent round if more than one exists. The parent round with the most votes is considered to be the winner. There must be M\*2/3 of blocks collected by any round Ri to elect round Ri-Pitch. As soon as Ri+Pitch round voted, which means Ri is elected, and all [R0 … Ri-‍1] rounds are confirmed, then round Ri is also considered as confirmed. In other words, for any round elected by a corresponding child round, if all previous rounds are confirmed then this round is also flagged as confirmed.

Figure 2. Voting and Confirmation

Ri

Ri+1

Ri+2

Ri+3

Ri+4

Ri+5

Ri+6

Ri+7

Ri+8

Ri+9

Ri+10

Ri+11

Ri+12

Ri+13

Ri+14

Collecting votes

Ri+15



Pitch = 8

First not yet voted, collecting votes

Elected by Ri+12 but unconfirmed because the previous Ri+3 is not confirmed

Not elected, cause Ri+11 not yet voted

Voted

Elected

Confirmed

Voted

Elected

Confirmed

Voted

Elected

Confirmed

Voted

Voted

Elected

Voted

Voted

Collecting votes

Collecting votes

Voted

Voted

Voted

Voted

Voted

Round voting has two further purposes. The second one determines how the chain grows. There are no time intervals used in this algorithm, so collecting a required number of votes means that a round is over and the chain can advance to the next one. The speed of MBV is thus limited only by the network speed. Since blocks can be received in a non-historical order, some “window” is required in a round sequence where blocks can be still accepted. For a first non-voted round Ri, this window lies in the range [Rlast\_confirmed + 1 … Ri+Pitch\*2].

The third purpose is to determine whether a member is offline. This is done at the confirmation stage for the round. Having a list of current members and their blocks received, the algorithm can determine whether any member has not sent a block or vote and remove this member from the list, and thus from participation in any further processing until the member signals its readiness to continue by sending a special “online” block again.

Figure 2. The Example of Transaction Fork caused by its Issuer

Block 3

A

B

Members

R7

R8

R9

Rounds

Time

Block 17

Tx Id=7

Rejected

A fork detected:

Transaction sender is punished

C

Block 32

Tx Id=7

Accepted

Tx Id=6

Accepted

Figure 2. The Example of Block Fork caused by its Issuer

w

Block 3

Accepted

A

Members

R7

R8

Rounds

Time

A fork detected:

Block issuer is punished

Block 4

Rejected

Block 4

Rejected

# Ultranet Specifics: Emission

In the Ultranet, the emission is done by transferring ETH coins from the Ethereum network. UNT is the native token of the Ultranet network. The emission mechanism works together with a special Ethereum smart contract. The following steps describe the whole process of ETH-to-UNT transfer:

1. A user tells the node application the amount of ETH and provides the Ethereum wallet of the account to debit ETH from, and initiates the transfer.
2. The application generates a unique signed key based on the account’s next emission transaction ID and the destination UNT account.
3. The application calls the special contract function that receives the key and the amount.
4. Smart contract stores the key and amount in the Ethereum blockchain.
5. All received ETH coins are burned by sending them to an 0x0 address.
6. The application sends a special transaction to the Ultranet with the key provided above.
7. After receiving the transaction, all nodes check the key against the record in the Ethereum blockchain, and if everything is correct, add it to the Roundchain.
8. If something goes wrong with the network and the emission transaction isn’t added to the Roundchain, then a user can manually repeat the sixth step until the transaction is placed and tokes  
   are credited.

The final amount a user account is credited depends on the current value of the Factor, which in turn depends on the total emission so far.

Initially, Emission = 0, Factor = 0, and a user receives 1000 tokens per 1 ETH. After each 10,000 ETH has been transferred through all accounts, the Factor is increased by 0.1 and a credited amount is calculated as follows:

Amount of UNT = ETH sent • (1000 – Factor)

This lasts until the Factor reaches the value of 1000 and at this moment the emission is over. Having this formula, nearly 10 million ETH will be required to burn and nearly 5 billion UNT will be created. Along with each transfer transaction, the Fundable Accounts are additionally credited by 10% of the resulting amount in equal shares, so the final total emission would be close to 5.5 billion tokens.

Below is the example of emission simulation:

Table 1. UNT Emission Simulation

|  |  |  |
| --- | --- | --- |
| Factor | ETH Spent | UNT Emission |
| 0  10  20  30  40  50  60  70  80  90  100  110  120  130  140  150  160  170  180  190  200  210  220  230  240  250  260  270  280  290  300  310  320  330  340  350  360  370  380  390  400  410  420  430  440  450  460  470  480  490  500  510  520  530  540  550  560  570  580  590  600  610  620  630  640  650  660  670  680  690  700  710  720  730  740  750  760  770  780  790  800  810  820  830  840  850  860  870  880  890  900  910  920  930  940  950  960  970  980  990 | 9848  109966  209999  309634  409493  509269  609769  709995  809904  909382  1009922  1109519  1209737  1309578  1409351  1509361  1609894  1709966  1809754  1909595  2009945  2109884  2209727  2309710  2409864  2509807  2609949  2709994  2809548  2909994  3009949  3109904  3209492  3309600  3409959  3509729  3609747  3709460  3809934  3909949  4009493  4109490  4209507  4309861  4409443  4509432  4609896  4709754  4809934  4909941  5009787  5109940  5209340  5309889  5409745  5509970  5609559  5709860  5809376  5909954  6009600  6109527  6209960  6309964  6409634  6509940  6609954  6709896  6809492  6909552  7009388  7109517  7209971  7309897  7409384  7509755  7609938  7709852  7809433  7909479  8009993  8109751  8209675  8309967  8409992  8509748  8609700  8709764  8809801  8909217  9009866  9109830  9209609  9309917  9409676  9509448  9609561  9709470  9809618  9909918 | 9848920  109416444  207899335  304995292  401313533  496555883  591483207  685145471  777512071  868488285  959430238  1048522357  1137168699  1224483040  1310742544  1396207500  1481111609  1564622185  1646898852  1728222230  1808956082  1888358407  1966687585  2044126769  2120697162  2196105805  2270662882  2344146165  2416274856  2488045934  2558464364  2627883823  2696055081  2763582387  2830273367  2895574452  2960038345  3023310376  3086059381  3147519206  3207696340  3267149305  3325614189  3383270996  3439488230  3494937885  3549644228  3603019857  3655565924  3707020261  3757393610  3806920923  3855083592  3902798032  3949183047  3994736919  4039006037  4082590046  4124838310  4166531330  4206840083  4246265759  4284884996  4322336802  4358668427  4394229309  4428684433  4462115756  4494437486  4525911199  4556316645  4585809989  4614391907  4641822190  4668140089  4693688911  4718185299  4741616187  4763975450  4785440640  4805998750  4825402859  4843841655  4861344546  4877798851  4893212279  4907658032  4921119387  4933576155  4944963881  4955486659  4964934729  4973368779  4980844193  4987280597  4992722404  4997182448  5000634129  5003092360  5004549189 |

Total ETH Spent: 10,000,777  
Total UNT Emission: 5,505,500,000 (with +10%)

# Ultranet Specifics: Fees and Costs

Transaction fee is set to 0.000001 UNT per byte. 10% of any transaction fee is distributed between Fundable Accounts and 90% goes to the members themselves.

# Conclusion

MBV is the first voting-based protocol that is designed to be as high-throughput as DAG-based asynchronous BFT algorithms but without a need of centralized masternodes and so to be able to deploy true homogenous decentralized networks for distributed registries.