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

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.

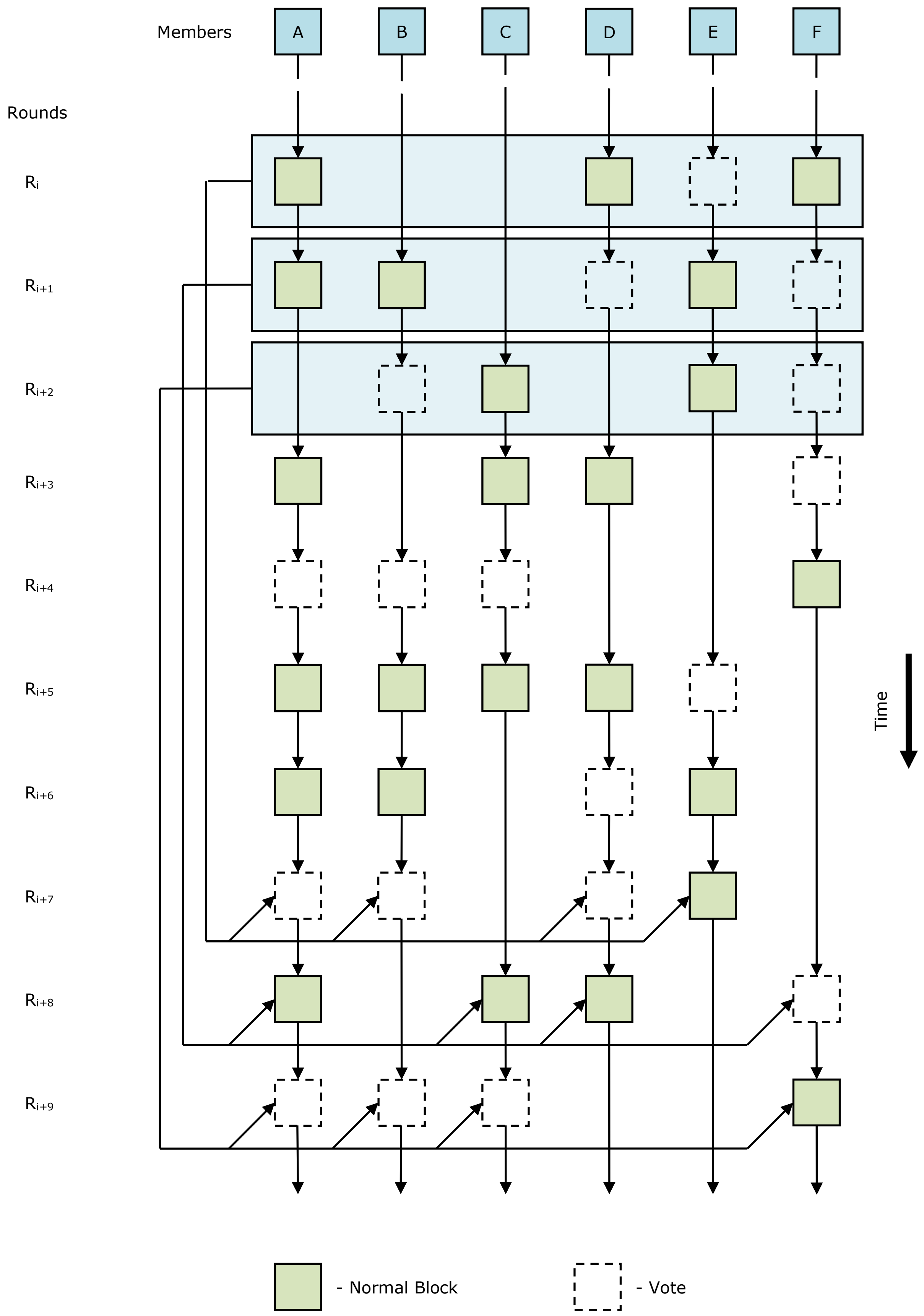
Parent round – a round R_i is a parent of round R_j when $i = j$ - Pitch, where Pitch is a constant. The rounds in the range [0… Pitch] are called *genesis rounds* and have a parent round reference set to zero.

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



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.

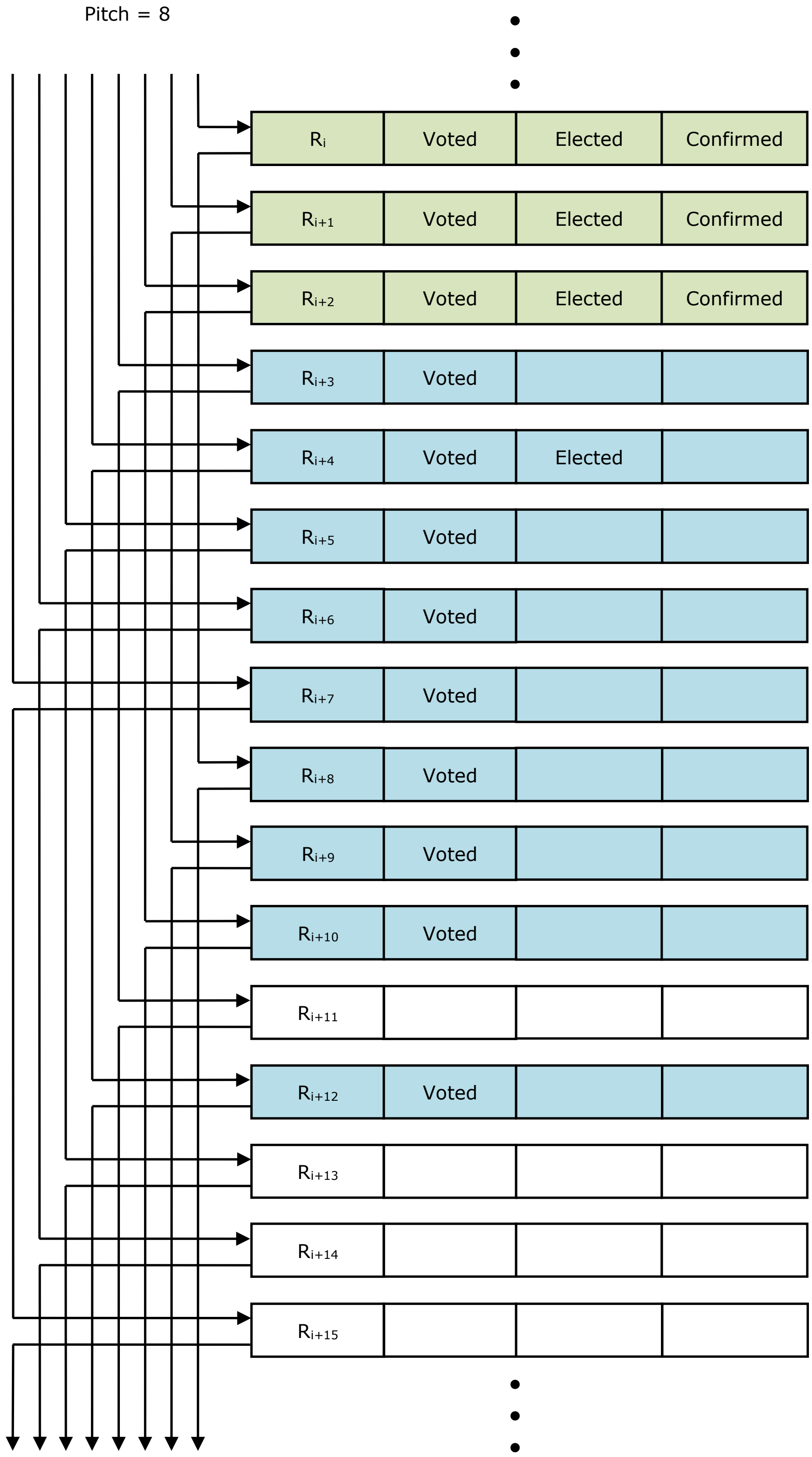
Each member can generate only one block per round – which prevents a chain from forking. If a member creates two different blocks with the same round index, then the network treats them as a cheater, seizes its bail, and distributes it between round members.

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 M_{max} members present in the network then they are sorted by its age-bail rank (account age divided by accession bail) and only top M_{max} members are eligible to create blocks and vote for rounds.

For each round, the algorithm expects all or top M_{max} 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 R_i to elect round $R_{i\text{-}Pitch}$. As soon as $R_{i+Pitch}$ round voted, which means R_i is elected, and all [R_0 … R_{i-1}] rounds are confirmed, then round R_i 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



Not elected, cause R_{i+11} not yet voted

Elected by R_{i+12} but unconfirmed because the previous R_{i+3} is not confirmed

First not yet voted, collecting votes

Collecting votes

Collecting votes

Collecting votes

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 R_i, this window lies in the range $[R_{last\_confirmed + 1} \ldots R_{i+Pitch*2}]$.

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

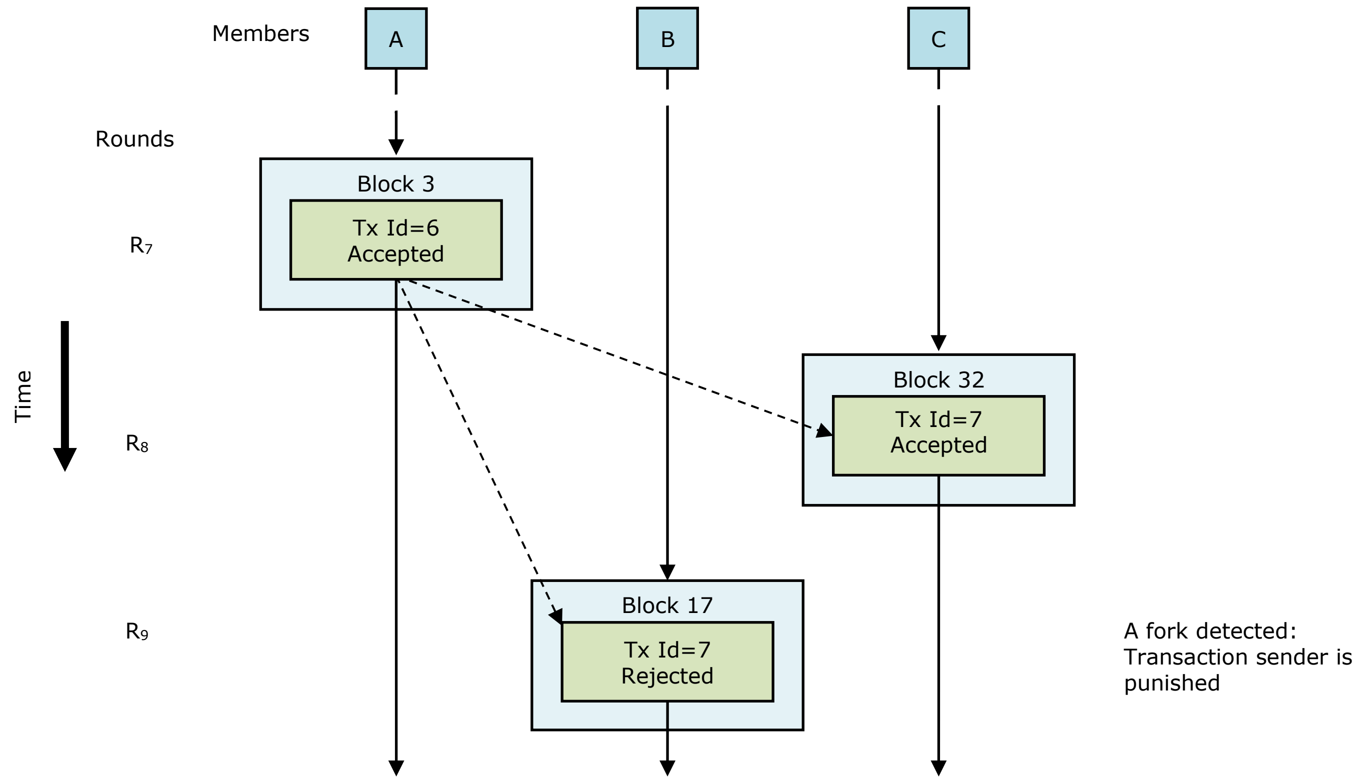
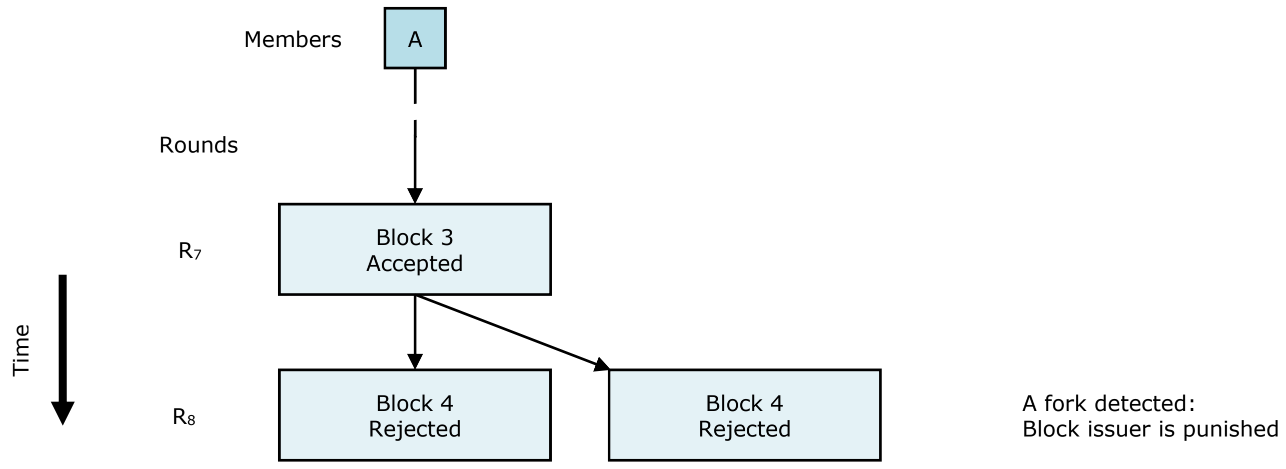


Figure 2. The Example of Block Fork caused by its Issuer



w

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:

$$\text{Amount of UNT} = \text{ETH sent} \bullet (1000 - \text{Factor})$$

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	9848	9848920
10	109966	109416444
20	209999	207899335
30	309634	304995292
40	409493	401313533
50	509269	496555883
60	609769	591483207
70	709995	685145471
80	809904	777512071
90	909382	868488285
100	1009922	959430238
110	1109519	1048522357
120	1209737	1137168699
130	1309578	1224483040
140	1409351	1310742544
150	1509361	1396207500
160	1609894	1481111609
170	1709966	1564622185
180	1809754	1646898852
190	1909595	1728222230
200	2009945	1808956082
210	2109884	1888358407
220	2209727	1966687585
230	2309710	2044126769
240	2409864	2120697162
250	2509807	2196105805
260	2609949	2270662882
270	2709994	2344146165
280	2809548	2416274856
290	2909994	2488045934
300	3009949	2558464364
310	3109904	2627883823
320	3209492	2696055081
330	3309600	2763582387
340	3409959	2830273367
350	3509729	2895574452
360	3609747	2960038345
370	3709460	3023310376
380	3809934	3086059381
390	3909949	3147519206
400	4009493	3207696340
410	4109490	3267149305
420	4209507	3325614189
430	4309861	3383270996
440	4409443	3439488230
450	4509432	3494937885
460	4609896	3549644228
470	4709754	3603019857
480	4809934	3655565924
490	4909941	3707020261
500	5009787	3757393610
510	5109940	3806920923
520	5209340	3855083592
530	5309889	3902798032
540	5409745	3949183047
550	5509970	3994736919
560	5609559	4039006037
570	5709860	4082590046
580	5809376	4124838310
590	5909954	4166531330

600	6009600	4206840083
610	6109527	4246265759
620	6209960	4284884996
630	6309964	4322336802
640	6409634	4358668427
650	6509940	4394229309
660	6609954	4428684433
670	6709896	4462115756
680	6809492	4494437486
690	6909552	4525911199
700	7009388	4556316645
710	7109517	4585809989
720	7209971	4614391907
730	7309897	4641822190
740	7409384	4668140089
750	7509755	4693688911
760	7609938	4718185299
770	7709852	4741616187
780	7809433	4763975450
790	7909479	4785440640
800	8009993	4805998750
810	8109751	4825402859
820	8209675	4843841655
830	8309967	4861344546
840	8409992	4877798851
850	8509748	4893212279
860	8609700	4907658032
870	8709764	4921119387
880	8809801	4933576155
890	8909217	4944963881
900	9009866	4955486659
910	9109830	4964934729
920	9209609	4973368779
930	9309917	4980844193
940	9409676	4987280597
950	9509448	4992722404
960	9609561	4997182448
970	9709470	5000634129
980	9809618	5003092360
990	9909918	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.