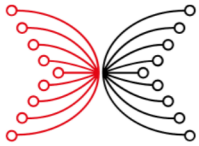

INPUT | OUTPUT

ETHEREUM CLASSIC TREASURY SYSTEM PROPOSAL

RESEARCH REPORT
March 7th 2017

The Veritas Team

Dmytro Kaidalov, Lyudmila Kovalchuk, Andrii Nastenko,
Mariia Rodinko, Oleksiy Shevtsov, Roman Oliynykov

IOHK.io

Abstract

We propose the Ethereum Classic Treasury System (ECTS) whose main purpose is to establish a decentralized funding mechanism for development and maintenance of the Ethereum Classic platform. Anyone can submit a project proposal and the stakeholders will decide on its necessity through a transparent voting procedure. The system is fully verifiable, having no significant influence on ETC performance. The report also takes into account ECIP 1017 that changes Ethereum Classic monetary policy and emission schedule. We expect that the ECTS implementation will increase stability and the outlook of the system, providing additional benefits both for stakeholders and miners from a more stable and predictable system.

Contents

1 Introduction **4**

1.1 Objective and desired properties of the treasury system . . . 4

1.2 Overview of the proposed treasury system . . . 5

2 Ethereum Classic Treasury System **6**

2.1 Ethereum Classic Blockchain . . . 7

2.2 Basic Treasury Model . . . 8

2.2.1 Funding epochs . . . 8

2.2.2 Epoch stages . . . 9

2.3 Proposal submission . . . 12

2.3.1 Proposal ballot formation . . . 12

2.3.2 Submission rules and cost . . . 13

2.3.3 Miscellaneous: revocation and money reducing . . . 13

2.4 Voters . . . 13

2.4.1 Deposit submission and redemption . . . 14

2.4.2 Incentives . . . 15

2.5 Treasury source . . . 16

2.5.1 Funding within the ECIP-1017 monetary policy . . . 16

2.5.2 Legacy monetary policy . . . 18

2.6 Election method . . . 18

2.6.1 Tie vote . . . 20

2.7 Delegation . . . 20

2.7.1 Full Delegation . . . 21

2.7.2 Disposable Delegation . . . 21

2.7.3 Rewards distribution . . . 22

3 Technical implementation approaches **22**

3.1 Core consensus implementation . . . 22

3.2 Smart contract implementation . . . 23

4 Design rationale **24**

4.1 Funding epoch and its size . . . 24

4.2 Blockchain based voting . . . 24

4.3 Treasury amount . . . 25

4.4 Voting process . . . 25
4.4.1 Fixed proposals and voters . . . 25
4.4.2 Secret ballots . . . 26
4.5 Incentives structure . . . 26
4.5.1 Expected number of voters . . . 26
4.5.2 55% attack . . . 28
4.5.3 Self-balanced set of voters . . . 28
4.5.4 Freezing period . . . 28
4.6 Election rule . . . 28
4.6.1 Types of voting systems . . . 29
4.6.2 A voting scheme for the treasury system . . . 31
4.7 Computational and storage burden . . . 32
4.8 Directions for further development . . . 33

5 Attacks overview **33**
5.1 Taking control over treasury coins . . . 34
5.1.1 Direct attacks on the voting procedure . . . 34
5.1.2 Influence on the voting process . . . 34
5.1.3 Taking control over a "treasury account" . . . 34
5.1.4 Bribing influential part of voters . . . 34
5.1.5 Attack of 55% . . . 34
5.2 Blocking the treasury operation . . . 35
5.2.1 Issuing large amount of proposals . . . 35
5.2.2 Creation of excessive number of voting accounts . . . 35
5.2.3 Generation of many commitments and openings from each voting account 35

6 Conclusions **35**

7 References **36**

A Appendix. Voting Model - Fuzzy Threshold Voting **39**
A.1 Introduction . . . 39
A.2 Definition of FTV model . . . 39
A.3 Some properties of FTV model . . . 45
A.4 The necessity of secret ballots . . . 51
A.5 Conclusions . . . 52
A.6 References . . . 52

1 Introduction

The cryptocurrency phenomenon emerged in 2008 with Bitcoin [1]. Then hundreds of different cryptocurrencies were invented, and now the market capitalization only of 10 most popular cryptocurrencies exceeds US$17 billion [2]. Traditional financial institutions and governments show interest in adopting blockchain technologies developed by this industry [3,4,5].

Absence of a centralized control over the operation process is a key feature expected from the blockchain systems. It can be ensured with decentralization and honest majority governance. When there is no central authority controlling coin emission and transactions it is considered a serious guarantee of operation stability and drives users' interest in cryptocurrencies.

In short-term perspective, a cryptocurrency operation requires having a decentralized honest majority of nodes that supports a distributed ledger maintenance.

In the middle- and long-term perspective, an additional factor is crucial. Any cryptocurrency exists as a complex technical system that requires continuous designing and implementation of multiple complex protocols and algorithms that, in turn, requires the appropriate funding level. Lack of funding would lead to cryptocurrency operation freeze, and to a deficient issue resolution routine. Occasional funding with weakly managed voluntary contributions may result in security vulnerabilities and system instability that would affect the exchange rate and lead to dramatic halt of its market extension. Thus, stable funding for the system support and development is a key to the cryptocurrency middle and long term market outlook.

We may consider different options to provide the needed funding. It can be money raised from venture funds which are interested in a potential of the blockchain technology or it can be community donations raised through the crowdfunding process. The third approach is a cryptocurrency self-sufficiency to ensure sustainable funding [6,7,8].

This report introduces a proposal for the ETC Treasury System (ECTS) that is intended to provide a sustainable funding system for the cryptocurrency development under the community control.

1.1 Objective and desired properties of the treasury system

The ETC Treasury System is a community controlled and decentralized process for funding of the Ethereum Classic development. The presented treasury system is not a comprehensive governance system with a hard fork adoption subsystem, but it is only a tool for the cryptocurrency development funding.

The ECTS is designed to have the following properties.

1. Supported and boosted stability and future prospects of the Ethereum Classic platform. Cryptocurrency miners as well as regular users should have no negative influence but benefits from a more stable and predictable system.

2. Acceptable computation and blockchain space load for the system, with no significant impact on performance.

3. Possibility for anyone to submit proposals for funding (with the system protected from DoS), and selection of the best ones.

4. Community members have incentives to select proposals for funding. Voters are interested in stability and further development of the cryptocurrency, so take part in the process.

5. An immediate economic incentive to become a voter and to be involved in the decision making. The ETCS should support a self-balanced set of voters to ensure a good participation level (if the number of participants drops, then the reward per participant increases that would attract new members).

6. Decisions made are transparent and traceable. Everyone accessing ETC blockchain can completely and independently verify correctness of the treasury operations.

7. Regular decision making on funding with ample time for analysis and limiting of the treasury losses from inappropriate proposals.

8. Voting procedure should minimize complexity of voting strategies. No voter should win or lose from voting earlier or sooner than others.

9. A delegation procedure allowing voters to follow trusted qualified participants as to decisions that need time- and skill-intensive analysis.

10. Adequate security properties to protect the system from malicious participants. An attacker must involve significant resources for small potential gains. Ability for honest participants to interfere and prevent further attacks.

We designed the relevant functionalities for the proposed ECTS, the formal analysis was postponed for the further stages of research.

1.2 Overview of the proposed treasury system

Here we provide an overview of the proposed treasury system. The detailed description is given in the following sections.

The development funding pipeline is designed as the process of coin transfers to accounts linked to community approved proposals. The proposals to be funded will solve tasks of the cryptocurrency enhancing and maintenance.

Coin transfers will be made in a constant number of blocks called a treasury epoch, with duration of approximately one month. At the end of each epoch, a payment block with extra coinbase transactions for proposals wallets is issued.

The epoch funding budget will have a constant upper limit. It can be presented as follows: each issued block within the epoch provides predetermined number of coins (obtained within the monetary policy) for the treasury. The project proposals are funded at the end of the epoch up to 80% of this treasury coin amount.

Anyone can submit a proposal before the start of an epoch when the proposal is to be funded. The proposal submission burns 5 coins and is included into the blockchain (thus, a list of proposals to be funded within the epoch is always fixed in advance). A proposal may request funding for a single epoch or for multiple epochs.

Proposals are approved for funding in the current epoch via fuzzy threshold voting. Each voter may issue a ballot that indicates "yes", "no" or "abstain" for a specific proposal (ballots for different proposals from one voter are usually grouped together to save the blockchain space).

The proposal score is the number of "yes" votes minus the number of "no" votes. Proposals that got at least 10% (of all votes) of the positive difference are acceptable candidates, and all the rest are discarded. Acceptable candidates are ranked according to their score, and the most popular proposal is funded according to the requested amount of money, then the next acceptable candidate, and so on, till the limit is reached (some portion of epoch treasury may remain unspent; coins for it are not issued at all).

Ballots are kept secret during the first stage of the epoch (with the duration of approx. 26 days), instead every voter is supposed to provide corresponding commitment for a voting ballot. The last stage (approx. 4 days) is opening, where each voter is incentivized to open her ballot (that must strictly correspond to the previously issued commitment). Open ballots are also included into the blockchain.

Voters are cryptocurrency stakeholders who made a special locked deposit, from 500 coins (a special locking transaction is issued). A deposit can be withdrawn by the owner by issuing of another transaction, and that stake becomes available after the following three full funding epochs. Thus, a list of voters in each epoch is also always fixed, similarly to the list of proposals. The voting power is proportional to the deposit size with respect to the sum of all voting deposits.

There is an incentive to make voting deposits and to take part in the voting process. Each voter gets a reward from the remaining 20% treasury coin amount, according to her share with respect to the sum of all voting deposits (similarly to the voting power). The voter's treasury reward is paid within her share proportionally to the number of voted proposals (with respect to total number of proposal in the current epoch). If a voter skipped some proposals, the corresponding coins are not issued at all (there is no redistribution to other voters).

A voter can make delegation for a specific proposal ballot or for a set of them to another voter. In this case her open ballot (and the corresponding commitment) contains the special value "follow the voter ID" instead of "yes"/ "no"/"abstain". On proposal ranking the delegated voting power for a proposal (or its set) is added to the delegate ballot. Full delegation to some particular delegate is also provided.

Note that all introduced system parameters (such as number of coins for the treasury fund or amount of voters reward) as well as architectural solutions are not set in stone and the subject for further analysis and discussion with the community members.

2 Ethereum Classic Treasury System

In this section we give an abstract description of a treasury system which can be implemented for establishment of a self-funding mechanism for a blockchain-based cryptocurrency system. We focus on the Ethereum Classic blockchain, but the model could be used for other cryptocurrencies, possibly with some modifications.

First, we briefly discuss the Ethereum Classic blockchain, its rewards scheme and then continue with a detailed description of the proposed treasury system.

2.1 Ethereum Classic Blockchain

In a nutshell, the Ethereum Classic is a proof-of-work based decentralized public ledger with a built-in Turing-complete programming language which allows to write smart contracts and decentralized applications [9,10].

The foundations of the Ethereum Classic blockchain are basically the same as in Bitcoin, which is the most well-known and successful cryptocurrency at the moment [1]. Consistency of a public ledger of blocks in a proof-of-work based scheme is secured through the decentralized mining process. The miners continuously try to solve a hard computational problem of finding of a hash value which is lower than some target. In the case of success, they get an opportunity to generate a block and claim a reward from this block (Fig. 2.1).

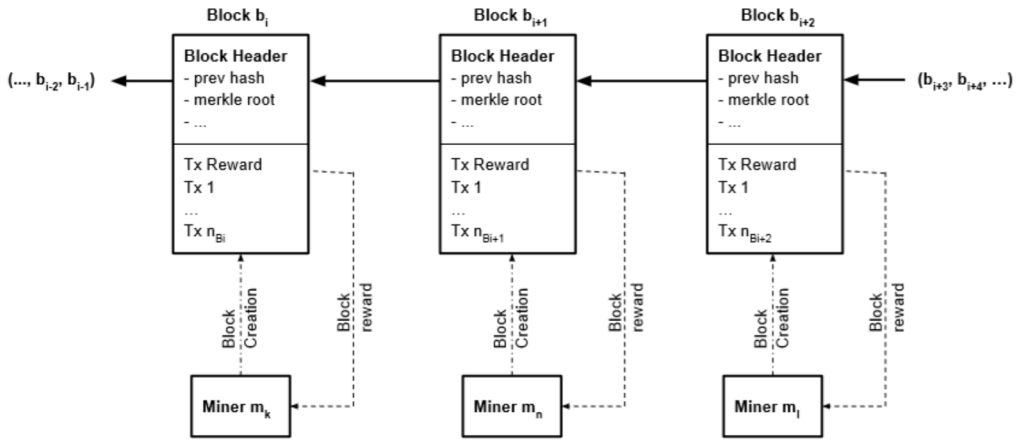

Figure 2.1: General scheme of a blockchain system

We do not go too deeply into technical details of the Ethereum platform, though there are many remarkable changes compared to Bitcoin, including accounts instead of UTXO, improved internal structures, Turing-complete scripting languages etc. An interested reader could find all necessary information in the original sources [8,9]. Instead, we highlight only a few things which are related in some way to the proposed treasury system

First of all, it should be mentioned that an average block time is about 14 seconds. It means that approximately $B_{month} = 1851428$ blocks are generated every 30 days.

The second important difference is a block reward. Each block includes a special reward payment to the miner who generated this block. Currently in the Ethereum Classic, this reward equals to 5 newly created coins (uncle blocks are not considered here). It is approximately $R_{month} = B_{month} \cdot 5 = 9257140$ coins per month. The full amount of rewards is paid to miners, and this is the only source of new coins in the system.

To summarize, we can say that though the Ethereum provides advanced features like Turing-complete programming language, improved Merkle trees, modified GHOST protocol, the basic principles are similar to those used in Bitcoin and other proof-of-work altcoins.

2.2 Basic Treasury Model

The basic idea is to introduce a mechanism for funding of different proposals from a treasury store. The funds for proposals could be allocated through the reward mechanism. At the moment the most well-known and successful treasury system was deployed in the Dash cryptocurrency [6,11,12]. It is called the Dash Governance System and supposed to use a simple scheme where 10% of coinbase block reward is collected for funding different proposals.

The proposed treasury model for the Ethereum Classic platform relies on the block rewards as a source of funding. The general idea is presented on the Fig. 2.2. There is an entity – "Treasury Store" – that serves as an accumulator of funds for future usage. Such store should not necessarily be implemented as a special account or a smart contract; the treasury fund can be easily calculated for a specific block height and created when it is really needed. In this case there is no need of special reward transactions to put money into the treasury store. The proposals could be funded directly from the coinbase transactions. The decisions regarding funding of the particular proposals could be made collectively by the stakeholders through a voting procedure.

A proposal itself is a funding request. It exists in the form of a proposal ballot which is submitted to the blockchain so every valid voter can review it and vote. Anyone can submit a proposal ballot. For instance, a core development team representative could request a payment for IT infrastructure services, developer's salary etc.

In this section we present high-level architecture of the proposed treasury system for Ethereum Classic.

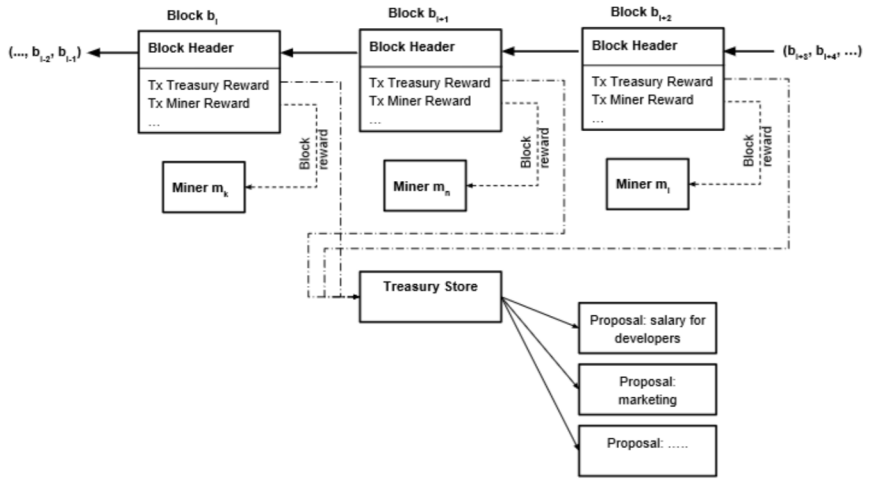

Figure 2.2: Treasury store

2.2.1 Funding epochs

The funding procedure includes several stages. An important point is that this process is iterative. We will call each iteration a *funding epoch*.

A *funding epoch* is a period of time (measured in blocks) during which a predefined set of voters decides on funds allocation among predefined set of proposals. The funds are accumulated

during the epoch.

More formally, let $B = (B^{(1)}, B^{(2)}, \dots)$ be a blockchain of a cryptocurrency with a treasury system, where $B^{(i)} = (b_1^{(i)}, b_2^{(i)}, \dots, b_{len}^{(i)})$ represents an i^{th} funding epoch consisting of len blocks. Funding limit for the epoch i is defined as

$$Tr(B^{(i)}) = \sum_{j=1}^{len} tr(b_j^{(i)}), \tag{2.1}$$

where $tr(b_j^{(i)})$ is a portion of the block reward which is assigned to the treasury store.

There is also a list of projects $P^{(i)} = (p_1^{(i)}, p_2^{(i)}, \dots, p_{\rho(i)}^{(i)})$ at the ith funding epoch which should be voted by the set voters $V^{(i)} = (v_1^{(i)}, v_2^{(i)}, \dots, v_{\vartheta(i)}^{(i)})$ to decide on funds $Tr(B^{(i)})$ allocation (voters also get some portion of these funds as a reward, it is discussed in section 2.4.2).

Generally speaking an entire blockchain is divided into the epochs of a constant duration (Fig. 2.3). The epoch duration is equivalent to 1 month (30 days). For the Ethereum Classic platform where an average block time is close to 14 seconds it means that an epoch duration would be $len = 185142$ blocks.

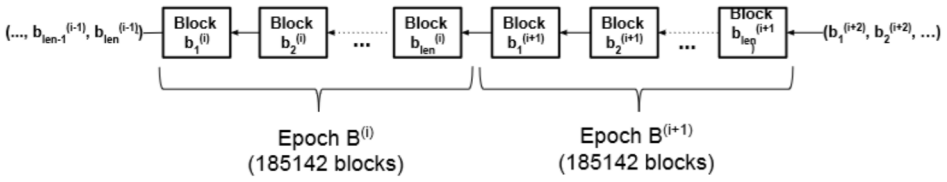

Figure 2.3: Funding epochs

2.2.2 Epoch stages

In this section we define different stages of the funding epoch and discuss purposes of each of them. In general, the further stages could be marked out as follows: submission, voting and finalization.

The necessity of separate voting and finalization stages stems from the fact that we use a simple commitment/reveal scheme [13] to keep voting ballots in secret during the voting. Disclosing of ballots happens only at the finalization stage when all voters have already submitted their commitments.

Submission stage The purpose of the submission stage is to collect proposals which should be reviewed by the voters. This stage lasts before the funding epoch begins because a set of proposals $P^{(i)} = (p_1^{(i)}, p_2^{(i)}, \dots, p_{\rho(i)}^{(i)})$ should be fixed at the moment when the voting starts.

The proposals should be submitted in one of the epoch which precedes the epoch where it should be voted and paid (Fig. 2.4). The purpose of such restriction is rather simple: there should be at least one full voting cycle, so the voters have enough time to review all proposals.

There is no restriction regarding the early submission, but voting for a particular proposal will take place only in the epoch where that proposal requests funds.

So if a proposal is supposed to be voted and paid in the epoch $B^{(t)}$, then it should be submitted in the epoch $B^{(i)}$, where $i < t$.

Detailed specification of a proposal ballot is given in further sections. Now it is only need to mention that all valid proposals are included into the blockchain, so it is trivial to find out when a specific proposal was submitted.

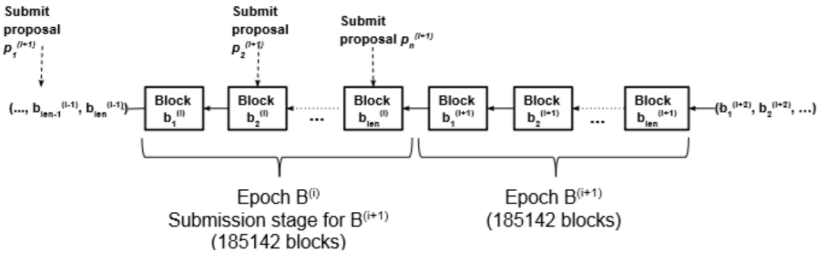

Figure 2.4: Proposals submission

Voting stage During the voting stage voters express their opinions regarding the submitted proposals. Selections of voters is discussed later; at the beginning of each epoch there is always a uniquely defined set of voters $V^{(i)} = (v_1^{(i)}, v_2^{(i)}, \ldots, v_{\vartheta(i)}^{(i)}))$.

At this stage the voters submit commitments for their voting ballots. The voting ballots themselves remain secret at this moment. They should be opened only at the finalization stage when all voters have already submitted their commitments. In such simple way secrecy is provided for the voting procedure that is a very desirable property of voting schemes (see Appendix A and [14]).

More formally, in the funding epoch $B^{(i)}$ a voter $v_1^{(i)}$ computes and publishes a commitment $c_{v_1}^{(i)}$ which uniquely represents her voting ballot $vb_{v_1}^{(i)}$:

$$c_{v_1}^{(i)} = H(vb_{v_1}^{(i)} \mid\mid r), \tag{2.2}$$

where $H(x)$ is a cryptographically strong hash function and r is some random value generated by the voter.

The voting stage lasts first k blocks of the epoch where $k = 160457$. The value of k is chosen so that it is approximately equal to 26 days (Fig. 2.5).

The commitments for the current epoch are taken into account only if they are included into the blockchain (in blocks $b_1^{(i)}, b_2^{(i)}, \ldots, b_{160457}^{(i)}$).

A particular voter can issue only one commitment per epoch. A commitment is considered valid only if it is signed by a valid voter.

So at the end of the voting stage there will be a set of commitments $C^{(i)} = (c_{v_1}^{(i)}, c_{v_2}^{(i)}, \ldots, c_{v_n}^{(i)})$. Note that it is not necessary that all voters submit their commitments. In this case, their votes will be automatically counted as "abstain".

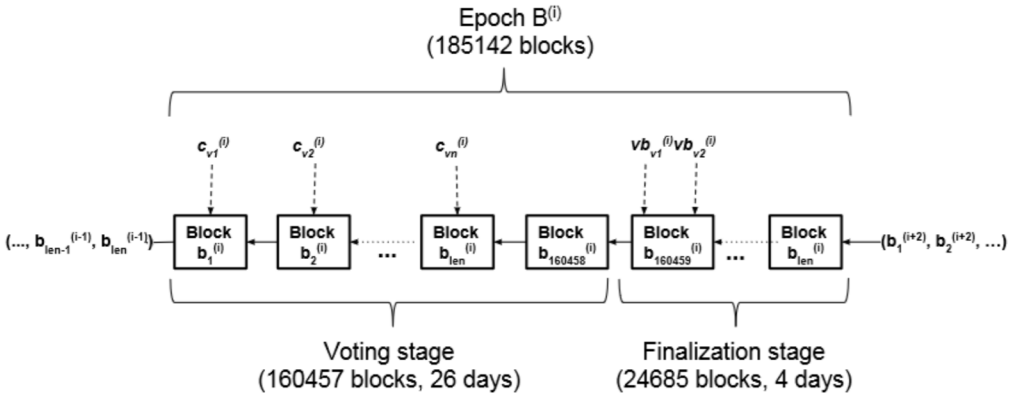

Figure 2.5: Epoch stages

The specific voting procedure and the structure of a voting ballot are discussed in further sections.

Finalization stage The finalization stage lasts 24685 blocks (approximately 4 days) at the end of the funding epoch (Fig.5). During this stage those voters who submitted commitments in the voting stage are supposed to submit their voting ballots $vb_{v_j}^{(i)}$ together with the randomness r (openings in terms of commitment/reveal scheme), so everyone can check that the voting ballot corresponds to some previously submitted commitment $c_{v_j}^{(i)}$.

A voting ballot contains all information regarding user preferences among proposals. The voting ballot is valid only if it is properly constructed:

- the user who signed the voting ballot is a valid voter;
- the commitment for this voting ballot was previously submitted by the same voter;
- the voting ballot contains votes only for proposals from the set $P^{(i)} = (p_1^{(i)}, p_2^{(i)}, \ldots, p_{\rho(i)}^{(i)})$;
- it is included in one of the blocks $b_{160459}^{(i)}, \ldots, b_{len-1}^{(i)}$.

It is not necessary that all voters who submit commitments will also submit voting ballots at the finalization stage, but we assume that they will (an incentive mechanism for this will be discussed further).

Finally at the end of the finalization stage (after block $b_{len-1}^{(i)}$) there will be determined set of voting ballots $VB^{(i)} = (vb_{v_1}^{(i)}, \ldots, vb_{v_m}^{(i)})$ which represents choices of voters. All voting ballots are included directly into the blockchain so everyone can see and verify the results.

The last block in the epoch $b_{len}^{(i)}$ is a payment block. It doesn't contain commitments or voting ballots but instead it contains transactions with payments for the winning proposals.

2.3 Proposal submission

The first step for obtaining funds from the treasury system is to submit a project proposal. Anyone can submit a proposal to the network. In this section we describe the process of submission together with the format of a voting ballot.

2.3.1 Proposal ballot formation

A proposal $p_j^{(i)}$ which is submitted to the network should follow a special format. Only valid proposals are included into the blockchain. There are six mandatory fields, they are represented in Table 2.1.

Table 2.1: Proposal data

Field name	Description
1. Proposal name	The symbols can be [*a-zA-Z0-9-_*]. It should be unique among any existing proposal names.
2. URL to the proposal description	A char string representing an URL to the description.
3. Hash of the proposal description	Hash value of the description. For example, if URL links to a pdf file with description, this hash value is a hash of the pdf file.
4. Start block	A block number when the first payment is expected. Note that a payment can occur only in the last block of the epoch $b_{len}^{(i)}$. So the Start Block number must be divisible by *len*.
5. End block	A block number when the last payment is expected to be made. It must be greater or equal to Start Block and as well be divisible by *len*.
6. Payment address	Should be a valid account address.
7. Payment amount	Should be a non-negative value, not more than the max budget of a funding epoch ($0 < paymentAmount \leq Pr(B^{(i)})$).
8. Collateral transaction ID	An ID of a collateral transaction. A collateral transaction is minimal payment for proposal submission.

Note that there are two values $startBlock(p_j^{(i)})$ and $endBlock(p_j^{(i)})$ which represent the funding period for the proposal $p_j^{(i)}$. If $p_j^{(i)}$ is a one-time payment, then both values are equal: $startBlock(p_j^{(i)}) = endBlock(p_j^{(i)})$. Otherwise there is multi-payment proposal:

$$k \cdot len = endBlock(p_j^{(i)}) - startBlock(p_j^{(i)}), \tag{2.3}$$

where k represents a number of payments and *len* is the size of a funding epoch.

2.3.2 Submission rules and cost

Prior to submission of a proposal, a special collateral transaction must be created (one such transaction may be referred by one proposal only). This transaction burns 5 coins, it is a minimal payment for proposal submission. The purpose of such payment is DoS attacks prevention so it would be harder and more costly to spam the network with malicious proposals.

After submission, the proposal is relayed through the p2p network to all nodes like a typical transaction. For the proposal to be valid it should be included into the blockchain at least len blocks prior to the $startBlock(p_j^{(i)})$ (see "Submission stage"paragraph).

2.3.3 Miscellaneous: revocation and money reducing

The proposed treasury system allows to revoke a previously submitted proposal. It could be quite useful in different situations, e.g. in the case of another approved proposal implementing more advanced functionality (developers cancel previously approved multi epoch funding in favor of their own newer already approved proposal).

In this case the owner of the proposal should have a possibility to submit a revocation transaction with no cost. To take effect, such transaction should be included into the blockchain before the payment block. In this case voting ballots for this project should not be counted, and the proposal itself should not be considered during the winner's selection procedure. The money which have already been allocated cannot be revoked.

It is also possible to reduce the amount of the requested coins. The same approach is applicable in this case: such transaction should be included into the blockchain before the payment block.

2.4 Voters

One of the most important parts of the developed treasury system is to define a set of voters $V^{(i)}$ who are empowered to participate in a voting procedure in the epoch $B^{(i)}$. An importance of this part of the system comes from the fact that selected voters should be properly incentivized to act not only for their benefit, but for the benefit of the entire system.

One easy solution to this problem is to give voting power to all stakeholders in the system (according to the amount of stakes they possess). In this case a list of stakeholders is fixed at some moment in time (for example, after the last block $b_{len}^{(i-1)}$ in the previous epoch) and those who have some positive balance can participate in voting in the epoch $B^{(i)}$. But such a scheme has significant disadvantages:

- it does not introduce direct personal incentives to participate. Why would someone spend time to review proposals and vote for proposals, if she does not have any benefits to do so compared to those who does not participate;
- the stake of participants is not locked so they can spend it right after the block $b_{len}^{(i-1)}$. It means that non-stakeholders will decide on proposals, which is undesirable property of the system because it could lead to non-optimal or even malicious decisions;
- uncontrolled participation will complicate estimations of the computational and network burden. In the worst case it would lead to a situation when the stakeholders are completely agnostic and do not want to participate.

All arguments mentioned above show the need of a more robust voters selection procedure with strong incentives to participate and behave honestly.

The proposed solution is based on locked deposits which should be submitted by the voters who want to participate in a voting procedure. An incentive to do so is a special reward which is paid at the end of each funding epoch.

The idea of deposits utilization is also used in other cryptocurrencies, for example, in Tezos [15,16] and Dfinity [8].

2.4.1 Deposit submission and redemption

Technical implementation of locked deposits is quite simple: the user v_j posts special transaction $Depo_{v_j}$ to the blockchain where she specifies a number of deposited coins $amount(Depo_{v_j})$. Note that for the $Depo_{v_j}$ transaction to be valid, the user should have the sufficient number of coins on her balance:

$$balance(v_j) \geq amount(Depo_{v_j}), \tag{2.4}$$

where the function $balance(v_j)$ returns an amount of the stake owned by the voter v_j.

The user will get voting rights in the epoch $B^{(i+1)}$ which follows the epoch $B^{(i)}$ where the deposit was made (Fig. 2.6).

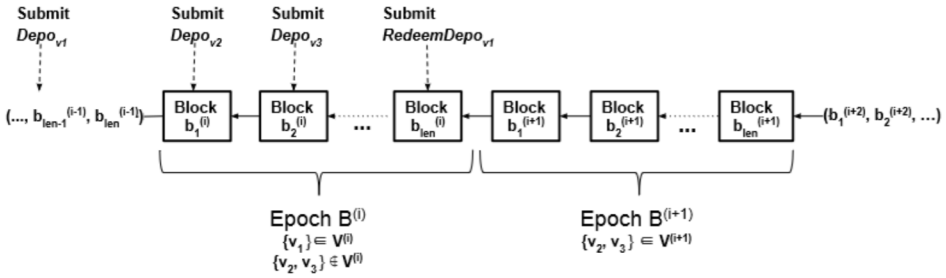

Figure 2.6: Deposit submission and redemption

The number of the deposited coins must not be smaller than the minimal threshold $amount(Depo_{v_j}) \geq minDepo$. For now we assume that $minDepo = 500$. Such threshold is needed for several reasons which will be substantiated later.

Since the $Depo_{v_j}$ transaction has been submitted, the corresponding stake cannot be spent by the user. To unlock the deposited funds, the user needs to submit another transaction $RedeemDepo_{v_j}$. Then after the freezing time t_{freeze} she will be able to access her funds. At the moment we suppose that

$$t_{freeze} = 3 * len(blocks),$$

where len is the size of a funding epoch. It means that the user should wait 3 full epochs. The freezing period t_{freeze} starts only from the epoch $B^{(i+1)}$ which immediately follows the epoch $B^{(i)}$, when the redemption was made. The user does not have a right to participate in voting during the freezing period t_{freeze}, but keeps this right till the end of the current epoch.

The proposed scheme has one significant advantage: it always keeps a particular funding epoch $B^{(i)}$ consistent, the sets of voters $V^{(i)}$ and proposals $P^{(i)}$ are determined and can not be changed during the epoch. This reduces some possible vectors of attacks and provides better

approaches to future formal analysis. The deposit size also constraints creation of excessive number of voting accounts that also protects system from flooding attacks.

2.4.2 Incentives

A special reward mechanism is introduced to incentivize stakeholders to deposit their stakes and participate in voting. As we discussed in section 2.2.1, the number of coins intended for the treasury is equal to $Tr(B^{(i)})$ per epoch. A portion of these funds is directed towards rewards for the voters.

Let $Vr(B^{(i)})$ be a reward for voters for the current epoch $B^{(i)}$. Following the suggestion that rewards for the voters are taken from the treasury fund, we have that

$$Vr(B^{(i)}) = k_{Vr} \cdot Tr(B^{(i)}), \tag{2.5}$$

where k_{Vr} is a portion of the treasury fund paid to voters, $0 \leq k_{Vr} < 1$.

From this it follows that the amount of the treasury fund assigned to the proposals themselves will be

$$Pr(B^{(i)}) = (1 - k_{Vr}) \cdot Tr(B^{(i)}) = Tr(B^{(i)}) - Vr(B^{(i)}). \tag{2.6}$$

We take $k_{Vr} = 0.2$, the rationale of this choice will be given in further sections.

The second important question is how the voters' reward $Vr(B^{(i)})$ will be allocated among the set of voters $V^{(i)} = (v_1^{(i)}, v_2^{(i)}, \ldots, v_{\vartheta(i)}^{(i)})$. Recall that the set $V^{(i)}$ and the reward $Vr(B^{(i)})$ are constants during the funding epoch $B^{(i)}$. To introduce the distribution function we first need to define the notion of a *voting power*.

Definition 2.1. A particular voter $v_j^{(i)}$ has a *voting power* $vp_{v_j^{(i)}}$ which reflects her ability to influence the final result. The voting power $vp_{v_j^{(i)}}$ is defined as follows:

$$vp_{v_j^{(i)}} = \frac{Depo_{v_j^{(i)}}}{\sum\limits_{v_k^{(i)} \in V^{(i)}} Depo_{v_k^{(i)}}}, \tag{2.7}$$

so the voting power roughly equals to the portion of the deposited stake of the voter $v_j^{(i)}$ in relation to the total amount of deposited stake by all voters in the epoch $B^{(i)}$(including $v_j^{(i)}$).

Now it is easy to define an upper bound of the voting reward for a particular voter. It is proportional to her voting power:

$$vr_{v_j^{(i)}} \leq Vr(B^{(i)}) \cdot vp_{v_j^{(i)}}. \tag{2.8}$$

The exact reward function will depend on one more parameter which is called *Participation Rate*.

Definition 2.2. Given a set of all proposals $P^{(i)} = (p_1^{(i)}, p_2^{(i)}, \ldots, p_{\rho(i)}^{(i)})$ in the epoch $B^{(i)}$ we define a *Participation Rate* of a voter $v_j^{(i)}$ as follows:

$$prate_{v_j^{(i)}} = \frac{\#\left\{p_k^{(i)},\ so\ that\ p_k^{(i)}\ \in\ vb_{v_j}^{(i)}\right\}}{\#P^{(i)}}. \tag{2.9}$$

Simply speaking, it is a portion of proposals which are reviewed and voted by the voter $v_j^{(i)}$.

Given all that the reward function is defined as follows:

$$vr_{v_j^{(i)}} = prate_{v_j^{(i)}} \cdot Vr(B^{(i)}) \cdot vp_{v_j^{(i)}}. \tag{2.10}$$

So it is easy to see that a reward is dependent on the voter's participation. Such scheme introduces a direct incentive to participate in a voting procedure. Together with the concept of a freezing period, it creates an extra incentive to behave honestly. Supporting malicious proposals will immediately reflect on the coins price and accordingly reduce the actual value of the deposited stake.

2.5 Treasury source

As mentioned in section 2.2.1 "Funding epochs", the only sources for the treasury are block rewards and transaction fees. The treasury function was introduced as follows:

$$Tr(B^{(i)}) = \sum_{j=1}^{len} tr(b_j^{(i)}),$$

$$tr(b_j^{(i)}) = br(b_j^{(i)}) + fees(b_j^{(i)}),$$

where $br(b_j^{(i)})$ is the portion of the block reward of the particular block $b_j^{(i)}$, and $fees(b_j^{(i)})$ is the portion of the transaction fees which is intended for the treasury.

2.5.1 Funding within the ECIP-1017 monetary policy

A new monetary policy and a coins emission schedule were proposed for open discussion in the Ethereum Classic community at the end of 2016 [37]. Nevertheless, the final decision on the policy adoption has not been made yet, and we are going to discuss how the presented treasury system fits the new monetary policy.

The ECIP-1017 "Monetary Policy and Final Modification to the Ethereum Classic Emission Schedule" proposal introduces several significant changes to the existing monetary policy. The major changes are as follows:

1. A block reward is not constant in time anymore. It is assumed that it will be decreased by 20% with each new era (recall that the era in Ethereum Classic is equal to 5,000,000 blocks that is approximately 2.3 years).

2. A reward for uncle blocks is significantly reduced. According to the new policy, a miner of an uncle block will get only 0.125 coins instead of 4.325 coins as it happens now.

The most important property the new rules bring to the system is that the money supply has a fixed cap. It is generally acknowledged by the Ethereum Classic community that a capped model will bring long-term stability for the entire system and attract more investors.

Joint Statement on Ethereum Classic's monetary policy [38] declares the total supply to be approximately 210 million ETC, not to exceed 230 million ETC. Thus, as the main option, we consider the treasury funding to be completely in line with the new monetary policy.

The treasury is filled from the block rewards and transaction fees. Block rewards will be reduced by 20% in each new era. To compensate for this reduction and to ensure stable treasury funding, a part of the transaction fees is also assigned to the treasury.

We propose to send to the treasury up to 20%[1] of the block reward, leaving to the miners 80% or more (approx. $\frac{4}{5}$ of the block reward remains to miners, and up to $\frac{1}{5}$ goes to the treasury).

A deployment period for 18 months is also considered, when the treasury funding sequentially would increase from 1/18 of its value to the full amount. For the first month, the treasury receives $\frac{1}{18} \cdot \frac{1}{5} = \frac{1}{90}$ of the block reward, or approx. 1.1% of an epoch reward, and the miners receive 99%. For the second month, the treasury is funded for $\frac{2}{18} \cdot \frac{1}{5} = \frac{1}{45}$ of the block reward, or 2.2% goes to the treasury, and 97.8% is to be received by miners, and so on ($3.3\% + 96.7\%, 4.4\% + 95.6\%$, etc.). For 18th month, at the end of the deployment period and later, the treasury is funded at the level of 20% ($\frac{1}{5}$) of the block reward, and the miners receive 80% of the block reward.

Considering that at the time of writing the transaction fees are negligible compared to the block rewards, initially the block rewards will play the main role in the funding, while fees will be important in the future. For the first era, after the full treasury deployment, it is possible to direct to the ETC treasury up to 185142 coins (185142 blocks × 5 coins × 20%) per one epoch. Fiat value of this fund is expected to increase with time as it happened to Dash after DGS deployment.

We assume that this option would be preferable as it completely agrees with ECIP-1017.

An alternative option of the treasury funding that we found to be less preferable, but consider being worth to discuss, is to increase a block reward specifically for the treasury, so there would be no need to reallocate miners‘ rewards. Such option would require changing the ECIP-1017 monetary policy because additional issuance of coins would be needed for the treasury. In the case of increasing of the block reward by 1 coin, it will consequently increase the final number of issued coins in the 2288 year from 210,725,892.25 (as it was suggested in ECIP-1017) to 235,725,892.27 (which is approximately by 11.5%). Furthermore, this change could be introduced at the beginning of the second era when the miner‘s reward will be reduced by 20% (from 5 to 4 coins). Simultaneously, an additional coin could be introduced for the treasury. So the total block reward would be unchanged (equal to 5 coins) for the second era. Note that in this case it is also supposed that the treasury reward would be reduced by 20% per era according to the ECIP-1017 rules, so the capped monetary supply model is not broken. To compensate for this reduction, we assume that a part of the transaction fees would be assigned to the treasury.

Considering these two options, it seems that the alternative one that requires further changes in monetary policy is more advantageous for the miners, as it provides more coins directly to the miners. But taking into account that a miner's profit depends on the coin price, the miners' preference could be changed to the treasury that exactly fits ECIP-1017. Increasing of the exchange rate, as one of the main expected consequences of the treasury adoption, will give more miners' profit than the alternative option with risks to cryptocurrency investors, etc. Exchange rate increase by 20% (required within the 18 months period) fully compensates to the miners for such modification. A further increase improves the miners' profit (as well as the profit for investors etc.) with stable and predictable monetary policy.

Thus, the treasury adoption, together with the strict compliance with ECIP-1017 gives the following benefits favorable to the miners.

1. Cryptocurrency investors will be assured that there are no risks from extra coin emission (keeping their investments free from inflation, at least, in terms of the coin amount) that would result in growing investments into the ETC with positive influence on the ETC exchange rate.

2. From less dependence to complete independence of the ETC from external support (that

[1] Choice of an exact value of a block reward assigned to the treasury is a matter of further discussion.

would provide extra stability and predictability).

3. Granted permanent professional expertise of the ETC community for the ETC development projects.

4. Marketing and other activities for the ETC promotion and further expanding, with increase of the ETC market share.

2.5.2 Legacy monetary policy

In the unlikely case if ECIP-1017 is rejected, the treasury can also be utilized within the current ETC monetary policy.

At the moment, the block reward in Ethereum equals to 5 coins, and all this amount is taken by the miner. We propose to reallocate a block reward, so that 1 coin (that is 20% of the total block reward) would be allocated to the treasury.

Amount of the treasury per funding epoch is:

$$Tr(B^{(i)}) = \sum_{j=1}^{len} tr(b_j^{(i)}) = len = 185142 \text{ (coins)}.$$

However, it should be pointed out that for the ETC further development we prefer the option with adoption of the ECIP-1017 and with strict compliance of the treasury with its principles.

2.6 Election method

In this section we will give a description of the election rule which is actually a core of any voting process.

In a nutshell the voting method is rather simple: given a set of voters and a set of proposals, each voter issues a voting ballot where she marks each proposal with one of the possible values: Accept, Abstain or Reject. After gathering all voting ballots it becomes possible to count the final score for each proposal, which is roughly equal to the difference between the number of coins which accept proposal and those which reject. If the score of a proposal is at least 10% of the total number of deposited coins, it is considered to be accepted. Then all accepted proposals are sorted in the descending order by their scores and funded one-by-one until run out of epoch treasury fund (or proposals). If a proposal does not fit into the remaining budget, then it is skipped and the process continues with the following proposals (also recall that the necessary condition for a proposal to be accepted is at least 10% support by the voters, according to their voting power).

Now we give a formal description for an election E and the election rule R, which defines a set of winners $W(B^{(i)})$.

An *election* $E = \{P^{(i)}, V^{(i)}, VB^{(i)}\}$ is given by a set $P^{(i)} = \{p_1^{(i)}, \ldots, p_{\rho(i)}^{(i)}\}$ of project proposals, a set $V^{(i)} = \{v_1^{(i)}, \ldots, v_{\vartheta(i)}^{(i)}\}$ of voters and a set of voting ballots $VB^{(i)} = \{vb_{v_1}^{(i)}, \ldots, vb_{v_m}^{(i)}\}$, where each $vb_{v_j}^{(i)}$ represents preferences of the voter $v_j^{(i)} \in V^{(i)}$. The voting ballot of the $j-th$ voter consists of 3 sets $vb_{v_j}^{(i)} = \{vb_{v_j}^{Y}, vb_{v_j}^{A}, vb_{v_j}^{N}\}$:

- $vb_{v_j}^{Y} \subseteq P^{(i)}$(a set of proposals $vb_{v_j}^{Y} = \{p_{\mu^j(1)}^{(i)}, p_{\mu^j(2)}^{(i)}, \ldots, p_{\mu^j(n_j)}^{(i)}\}$ which were voted positively by the voter $v_j^{(i)} \in V^{(i)}$);

- $vb^A_{v_j} \subseteq P^{(i)}$(a set of proposals $vb^A_{v_j} = \{p^{(i)}_{\varrho^j(1)},\ p^{(i)}_{\varrho^j(2)},\ \dots\ ,\ p^{(i)}_{\varrho^j(m_j)}\}$ which were voted neutrally by the voter $v^{(i)}_j \in V^{(i)}$);
- $vb^N_{v_j} \subseteq P^{(i)}$(a set of proposals $vb^N_{v_j} = \{p^{(i)}_{\eta^j(1)},\ p^{(i)}_{\eta^j(2)},\ \dots\ ,\ p^{(i)}_{\eta^j(k_j)}\}$ which were voted negatively by the voter $v^{(i)}_j \in V^{(i)}$).

A voter $v^{(i)}_j$ cannot issue different votes for the same proposal: $(vb^Y_{v_j} \cap vb^N_{v_j} = \oslash)\ \wedge\ (vb^Y_{v_j} \cap vb^A_{v_j} = \oslash)\ \wedge\ (vb^N_{v_j} \cap vb^A_{v_j} = \oslash)$. It follows that $vb^{(i)}_{v_j} \subseteq P^{(i)}$(a set of proposals which were voted by the voter $v_i \in V$ is always a subset of P).

Note that it is possible that some proposal $p^{(i)}_t \notin vb^{(i)}_{v_j}$. It means that the voter $v^{(i)}_j$ did not vote for this proposal. In this case it is assumed that the voter has abstained. Recall that if a proposal is not explicitly included into the voting ballot, the reward of the corresponding user would be reduced according to the rules presented in section 2.4.2 "Incentives".

An *election rule* R is a mapping that given an election $E = \{P^{(i)}, V^{(i)},\ VB^{(i)}\}$ outputs a set of winners $W(B^{(i)}) = \{p^{(i)}_{\varpi(1)},\ p^{(i)}_{\varpi(2)}, ...,\ p^{(i)}_{\varpi(k)}\}$ that are selected to be paid in the funding epoch $B^{(i)}$. A set of winners is eventually a subset of all proposals $W(B^{(i)}) \subseteq P^{(i)}$.

To define election rule R, we will recall two utility functions $amount(p^{(i)}_j)$ and $Pr(B^{(i)})$:

- $amount(p^{(i)}_j)$ given a proposal $p^{(i)}_j$ returns an amount of money requested by this proposal;
- $Pr(B^{(i)})$ given a funding epoch $B^{(i)}$ returns a reward limit for this epoch (that is maximal amount of money which can be paid to all proposals).

An election rule $R(E)$ returns a set of winners according to the following rules.

1. Let $score(p^{(i)}_k)$ be a function given a certain proposal $p^{(i)}_k \in P^{(i)}$ returns a score for this proposal $score(p^{(i)}_k) = \sum_{j=1}^{\rho(i)} (pref(p^{(i)}_k, vb^{(i)}_{v_j}) \cdot depo_{v_j})$, where $depo_{v_j}$ is the number of deposited coins[2] by the voter v_j and function $pref$ returns her preference regarding the proposal $p^{(i)}_k$:

$$pref(p^{(i)}_k, vb^{(i)}_{v_j}) = 1 \quad if\ p^{(i)}_k \in vb^Y_{v_j},$$

$$pref(p^{(i)}_k, vb^{(i)}_{v_j}) = -1 \quad if\ p^{(i)}_k \in vb^N_{v_j},$$

$$pref(p^{(i)}_k, vb^{(i)}_{v_j}) = 0 \quad if\ p^{(i)}_k \in vb^A_{v_j} \vee p^{(i)}_k \notin vb^{(i)}_{v_j}.$$

2. Let $P^{(i)}_{accepted} = \{p^{(i)}_{\chi(1)},\ p^{(i)}_{\chi(2)}, ...,\ p^{(i)}_{\chi(\iota(z))}\} \subseteq P^{(i)}$ be a sorted list of proposals so that $score(p^{(i)}_{\chi(1)}) \geq\ score(p^{(i)}_{\chi(2)}) \geq .... \geq score(p^{(i)}_{\chi(\iota(z))})$, and for each $p^{(i)}_k \in P^{(i)}_{accepted}$ it holds that $score(p^{(i)}_k) \geq 0.1 \cdot \sum_{v_j \in V^{(i)}} depo_{v_j}$ (proposals which have $score(p^{(i)}_k) < 0.1 \cdot |V^{(i)}|$ are considered unaccepted and are not included into the set $P^{(i)}_{accepted}$).

[2] Generally speaking multiplying by depo reflects a voting power of a particular voter. In this case score function shows amount of stake that supports proposal (instead of number of voters).

3. A set of winners $W(B^{(i)}) = \{p^{(i)}_{\varpi(1)}, p^{(i)}_{\varpi(2)}, ..., p^{(i)}_{\varpi(k)}\}$ consists of the proposals from $P^{(i)}_{accepted}$ where $score(p^{(i)}_{\varpi(1)}) \geq score(p^{(i)}_{\varpi(2)}) \geq .... \geq score(p^{(i)}_{\varpi(k)})$, so that $\sum_{j=1}^{k} amount(p_{\varpi(j)}) \leq Pr(B^{(i)})$.

Recall that the reward limit $Pr(B^{(i)})$ of the funding epoch $B^{(i)}$ is defined as follows

$$Pr(B^{(i)}) = (1 - k_{Vr}) \cdot Tr(B^{(i)}) = (1 - k_{Vr}) \cdot \sum_{j=1}^{len} tr(b^{(i)}_j),$$

where $tr(b^{(i)}_j)$ is a portion of the block reward intended for the treasury, and k_{Vr} is a portion of the treasury fund paid to voters ($k_{Vr} = 0.2$).

2.6.1 Tie vote

It is theoretically possible that there are two or more proposals which have equal scores $score(p^{(i)}_n) = score(p^{(i)}_m) = ... = score(p^{(i)}_k)$ but cannot be included into $W(B^{(i)})$ simultaneously because it would exceed funding limit $Pr(B^{(i)})$. In this case we need additional rules to rank such proposals to be able to define which of them are winners.

In our model, we assume the following simple rules for breaking ties which apply one by one until tie vote is solved:

1. The proposals are ranked according to the amount of funds gathered from the previous funding epochs. We denote the function which returns such amount as $score_{prev}(p^{(i)}_j)$. A particular proposal would have $score_{prev}(p^{(i)}_j) > 0$ only in the case if it is a multi-epoch proposal (which requests continuous funding during more than one epoch) and it has already received funds during previous epochs. Such rule will provide an advantage to the well-known and previously accepted projects:

$$score_{prev}(p^{(i)}_n) \geq score_{prev}(p^{(i)}_m) \geq .... \geq score_{prev}(p^{(i)}_k).$$

2. If there is a still tie vote among some proposals, then it is solved by the rule "first come first served" (FCFS). The proposals which have been submitted earlier (included to earlier block; have a lower index in the case of the same block) have an advantage.

Nevertheless, when we consider tie vote resolution rules, it is expected that such events could happen quite rarely.

2.7 Delegation

To get rewards, voters need to be online to track the list of proposals, review them and vote. To mitigate risks of low participation rate or poor proposal analysis, and to provide voters with more flexibility, we put forth a *delegation scheme*, whereby voters can automatically follow some other voter.

There are two types of delegation: full delegation and disposable delegation for a specific proposal. While the first one needs a specific transaction, the second one is implemented directly through a voting ballot.

2.7.1 Full Delegation

Implementation of the full delegation is rather simple. We only need two additional types of transactions: the one for the delegation itself and another one for revocation.

Fig. 2.7 represents the basic idea. To delegate her voting right, a user needs to submit a special transaction $Delegate_{v_j}$. This transaction will point to the particular voter (delegate) which should be followed (for example, it may contain her public key).

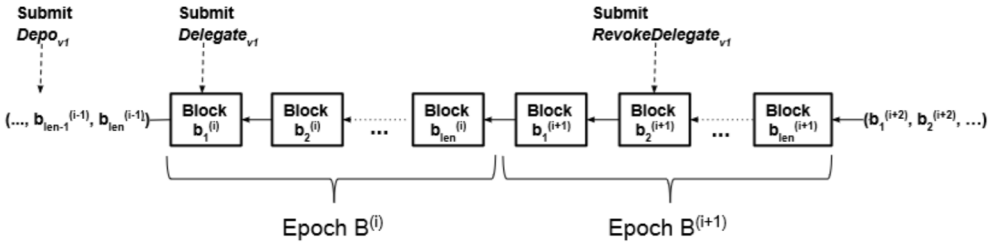

Figure 2.7: Delegation

The $Delegate_{v_j}$ transaction could be issued at any time by a valid voter (who made a deposit). Note that delegation and revocation will take effect in the current epoch only if they are submitted before the finalization stage.

Even if the delegation has been made, the voter still has a right to vote herself. For instance, she may vote for some proposals which she is interested in, and delegate decisions about others to someone else. The voting ballot of a user takes precedence over the voting ballot of a delegate. So the votes of the delegate would be taken into account only if the original user does not include them into her voting ballot (or she does not issue the voting ballot at all).

To revoke delegation rights a voter need to submit $RevokeDelegate_{v_j}$ transaction.

To change a delegate, a voter needs to revoke the previous one. Only one delegation and one revocation is allowed during an epoch.

Note that if a delegate has not voted for some proposals, then a voter who delegates her voting rights will not receive a reward for these proposals. If a delegate herself delegates her votes, then the delegation reward should be paid to those who actually voted.

2.7.2 Disposable Delegation

A delegation for a specific proposal allows the voter to include into her voting ballot a special value "follow the voter ID" instead of "yes"/"no"/"abstain". In this case the vote for this proposal will be the same as in the voting ballot of the corresponding delegate.

The same rules as for the full delegation are applied here. If the delegate has not voted for the proposal, it is considered as non-voted by the original voter (so she would not receive the reward).

The disposable delegation has a precedence over the full delegation, so if the voter includes "follow the voter $Delegate^t_{v_j}$" statement for some proposal, the vote of the $Delegate^t_{v_j}$ would be taken first. If $Delegate^t_{v_j}$ has not voted, then the vote of the full delegate would be taken into account.

2.7.3 Rewards distribution

In the case of delegation, some portion of the voting reward is redirected to the delegates. From one side, it incentivizes the delegate to vote more deliberately (so more voters want to follow her), from the other side it prevents thoughtless delegation.

There is a parameter $0 \leq k_{Del} \leq 1$ which defines the portion of the voting reward which goes to the delegate ($k_{Del} = 0.05$). Recall from the section 2.4.2 "Incentives" that the reward function for a voter is defined as

$$vr_{v_j^{(i)}} = prate_{v_j^{(i)}} \cdot Vr(B^{(i)}) \cdot vp_{v_j^{(i)}},$$

where

- $prate_{v_j^{(i)}}$ is a participation rate (how many proposals were voted in relation to the total number of proposals);
- $Vr(B^{(i)})$ is a total voting reward for the epoch $B^{(i)}$;
- $vp_{v_j^{(i)}}$ is the voting power of the voter $v_j^{(i)}$.

As far as one voter could have several delegates in a particular epoch (one full delegate and several disposable delegates), we will define a set of delegates for the particular voter v_j as $Delegates_{v_j} = \{Delegate^1_{v_j}, ..., Delegate^n_{v_j}\}$.

If a voter made a delegation (either full or disposable), her reward function will be defined as follows:

$$vr_{v_j^{(i)}} = (prate_{v_j^{(i)}} + \sum_{t=1}^{n} pdrate^t_{v_j^{(i)}} \cdot (1 - k_{Del})) \cdot Vr(B^{(i)}) \cdot vp_{v_j^{(i)}},$$

where

$$pdrate^t_{v_j^{(i)}} = \frac{\#\left\{p_k^{(i)},\ so\ that\ p_k^{(i)} \notin vb_{v_j}^{(i)} \wedge p_k^{(i)} \in vb^{(i)}_{Delegate^t_{v_j}}\right\}}{\#P^{(i)}}.$$

Literally speaking $pdrate^t_{v_j^{(i)}}$ is a portion of proposals, which where voted by the $Delegate^t_{v_j}$. Now it is easy to define the reward of the $Delegate^t_{v_j}$:

$$dr^t_{v_j^{(i)}} = pdrate^t_{v_j^{(i)}} \cdot k_{Del} \cdot Vr(B^{(i)}) \cdot vp_{v_j^{(i)}}.$$

3 Technical implementation approaches

In this section we briefly present a high level description of the implementation options.

3.1 Core consensus implementation

The first possible solution is to integrate the treasury into the core consensus algorithm directly. All data related to the voting need to be included into the block together with the usual transactions (for instance, they could be combined into the Merkle tree). Block validation rules get additional criteria.

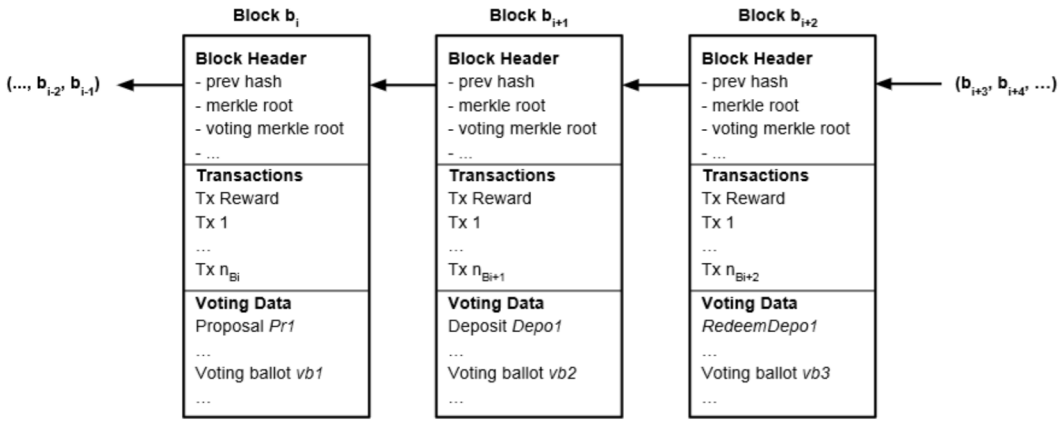

Figure 3.1: Extending block with the data related to voting

This implementation has many advantages and seems to be the best option for the treasury. It provides a highly scaled efficient, flexible and robust system that would easily support tens of thousands of voters, many proposals and has a great potential for further development (for advanced cryptographic voting protocol etc.). The disadvantage of this approach is that it requires additional improvement of the block format and of the validation rules.

Thus, the core consensus implementation looks as the optimal for the mature version of the treasury.

3.2 Smart contract implementation

Considering that Ethereum platform has a powerful internal smart-contract system with a Turing-complete scripting language, it becomes possible to implement a treasury system as a special smart contract. In this case, the voters will communicate only with smart contracts where all data related to the voting process will be stored.

Such approach also requires a hard-fork. The advantage of this approach is that the core consensus protocol needs less modification comparing to the previous option (implementation of the new block reward distribution is required, with the treasury contract involvement). But such approach has disadvantages, e.g. restrictions on EVM capabilities, limitations on implementation efficiency, potential problems with support of a necessary number of voters and limited further development scope (e.g., impossibility to implement advanced voting protocols in the future). Besides, smart contracts will also require fees (resulting in inefficient utilization of computation resources: a part of the block reward goes through a smart contract to be returned to the miners).

A smart contract could be used at the initial implementation stages, however, on our opinion, the core consensus implementation appears to be the optimal approach for the mature version of the treasury.

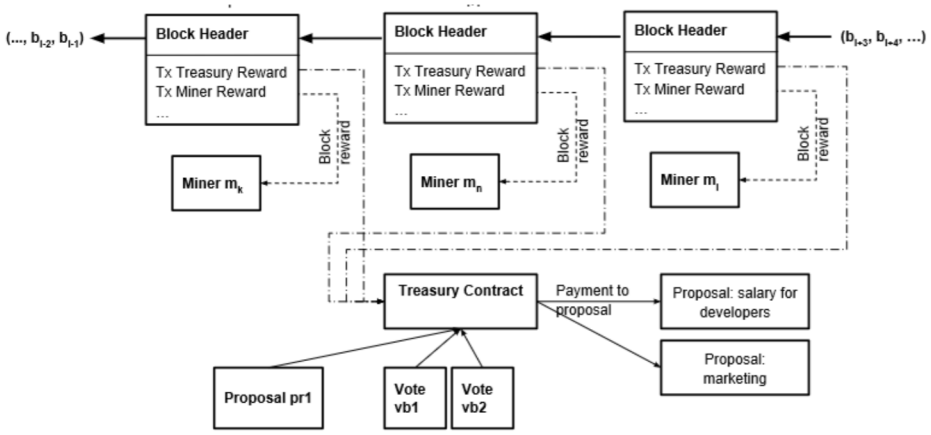

Figure 3.2: Treasury smart-contract

4 Design rationale

In this section we provide design rationale for the architectural solutions which were presented above.

4.1 Funding epoch and its size

An important aspect while developing a treasury system with regular funding is the duration of each funding period. The epoch duration should not be too large, so the delay before proposal submission and payment is suitable for real-world applications; on the other hand the duration of an epoch should not be small, so the voters will have enough time to review and vote for all proposals, etc.

A period of 1 month is a well-established value in practice. A lot of different activities in the financial world are bounded by the scope of 1 month. In the case of the treasury, it is also appropriate for the following reasons:

1. It allows to control a project execution on the monthly basis. For instance, if there is a long-term project, it would be reevaluated each funding epoch, and if the voters community is not satisfied with results, they could stop funding. And that cutoff could be done on the early stages when the financial losses remain small.
2. One month is a quite long period, so the voters have enough time to make well-considered decisions.
3. The reward for voting is also paid at the end of the funding epoch, so voters do not wait too long for their rewards.

4.2 Blockchain based voting

The only communicational channel in the proposed treasury system is a blockchain itself. It means that all messages (e.g. deposits, proposals, voting ballots, delegation, etc.) are stored

directly in a form of special transactions. So the whole voting procedure is non-interactive and does not require direct communication among voters.

Constructing the whole procedure on the blockchain gives great benefits over simple message exchange via p2p network and keeping a separate voting database. It provides much higher level of the general transparency and verifiability of the system, as well as allows full verification without need of trusted parties. Everyone can easily check that the voting results correspond to the genuine preference of the voters because all voting ballots could be recalculated. Validity of the voters' list could also be easily verified by checking of the relevant deposit transactions.

Blockchain voting satisfies the following desirable voting properties [14, 17]:

1. **Eligibility.** With consistent chain of blocks it is very easy to check that only legitimate voters made their votes and only once. It could be done by checking that a deposit transaction was made before the voting ballot is issued, and that here is only one voting ballot for a particular voter.

2. **Fairness.** No early results can be obtained which could influence the remaining voters. This is also true for the presented voting scheme because on the voting stage the only commitments are submitted to the blockchain, so no one can see the results of other voters during the voting stage.

3. **Individual verifiability**. A voter can verify that her vote was really counted. It is easy to see that this property also holds for blockchain based voting.

4. **Universal verifiability.** The published outcome really is the sum of all the votes.

4.3 Treasury amount

The only possible sources of the treasury fund are block rewards and transaction fees. Specifically, each block gives a part of the reward or transaction fees (or both) to the treasury (more details in section 2.5 "Treasury Source").

Now in Ethereum Classic each new block creates 5 new coins that are paid completely to the miner. With a treasury system we assume that each block will contribute one coin to the treasury (for the both options of ECIP-1017 adoption). A number of coins needed for the treasury system was chosen with respect to the price of the Ethereum Classic coin in a real market. At the time of writing, the price of 1 ETC token was approximately US$1.3[3]. It means that the total treasury monthly fund is estimated as US$241000, from which US$193000 are supposed to be paid for proposals themselves, and US$48000 to voters as their reward. These values also provide sufficiently large safety margin for possible risks of decreasing of the ETC exchange rate.

4.4 Voting process

4.4.1 Fixed proposals and voters

An important property of the developed voting scheme is that the sets of proposals and voters are fixed before the voting begins. Such architectural solution reduces possible malicious actions during the voting process (for example, it becomes impossible to inject a malicious proposal at the end of voting stage and quickly accept it, so the other voters don't have enough time to

[3] According to USDT/ETC pair on https://poloniex.com/exchange#usdt_etc

analyze and reject it). This also significantly reduces possible adversarial voting strategies and simplifies further theoretical analysis.

4.4.2 Secret ballots

Secret ballots are the key point to provide ***fairness*** property. It is widely accepted that this property is very important for a good voting scheme [14, 17, Appendix A.3]. It guarantees that nobody can gain any knowledge about the partial tally before the counting stage begins. The knowledge of the partial tally could affect the intentions of the voters who has not yet voted and gives them an advantage to modify their voting strategy according to a new knowledge.

By providing fairness property with secret ballots, we guarantee that all voters have equal rights and possibilities.

4.5 Incentives structure

In section 2.4.2 "Incentives" we presented the paid role of voters in a treasury system. The reward function for voters has also been presented. The main idea of the incentive scheme is to involve in a voting procedure a reasonable number of honest and rational stakeholders. On the other hand, a number of voters should be at least predictable, and it should not be too large, so the computational and storage load is suitable for the blockchain system.

4.5.1 Expected number of voters

Recall that the total voters reward is defined as follows:

$$Vr(B^{(i)}) = k_{Vr} \cdot Tr(B^{(i)}),$$

where k_{Vr} is a portion of the treasury fund paid to voters. This reward is distributed among all voters according to their voting power and participation rate. Previously we mentioned that the suitable value for parameter k_{Vr} is 0.2 which means that 20% of the monthly treasury fund is paid to the voters and the remainder of it (80%) is paid towards the proposals.

It is reasonable to assume that the stakeholders would deposit their stake only if an interest rate is at least some value $k_{MinReward}$. So the parameter $k_{MinReward}$ defines the annual income which is measured in a proportion to the amount of the deposited stake.

It is a highly nontrivial task that requires external complex economical analysis to estimate the exact value of $k_{MinReward}$ which would be suitable for stakeholders, because it depends on variety of different economic factors which are hardly predictable in the real world. Probably the most influential factor is a price volatility risks.

From the real world examples[4] we can assume that the minimal interest rate for the cryptocurrency investors is about 10% of annual return. Given that, we have also taken $k_{MinReward} = 0.1$ as a starting point in our calculations.

[4] In some sense, a similar incentive structure is used in the Dash cryptocurrency, where so called masternodes deposit their stake, provide some service (including voting for proposals) and get reward for this [4,5]. As far as Dash has been deployed and is quite common (top 10 by Volume according to http://poloniex.com) it is useful to see the real reward which is achieved. At the moment of writing, our calculations show that the annual return of investments for Dash masternodes is approximately equal to 10%. These calculations closely agree with calculations provided on http://dashmasternode.org.

As far as total reward of the voters per month is fixed by $Vr(B^{(i)})$, we can estimate the total amount of reward per year. We will assume that a funding year consists of 12 funding epochs (in reality it should be slightly less than 365 days because a funding epoch is equal to approximately 30 days). Given that we have that the annual reward Ar is defined as follows:

$$Ar(B^{(i)}) = Vr(B^{(i)}) \cdot 12 = k_{Vr} \cdot Tr(B^{(i)}) \cdot 12.$$

Recall that the $Tr(B^{(i)}) = \sum_{j=1}^{len} tr(b_j^{(i)}) = 185142$, so finally

$$Ar(B^{(i)}) = k_{Vr} \cdot 185142 \cdot 12 = k_{Vr} \cdot 2221704 \; coins.$$

Given that the stakeholders expect a return of investments at least at the level of 10% per year, we could easily count expected amount of stake EAS which would participate to keep up this rate:

$$EAS = \frac{k_{Vr} \cdot Tr(B^{(i)}) \cdot 12}{k_{MinReward}} = k_{Vr} \cdot 22217040.$$

As it is known the coins supply in Ethereum Classic is equal to $60102216 \cdot (1.198 + 0.26 \cdot n)$, where n is the number of years after the genesis block [8]. At the time of writing total coins supply is approximately equal to 88,390,704. From this we can evaluate percentage of the total stake which is expected to be deposited:

$$EAS_{\%} = \frac{EAS}{88390704} \cdot 100\% = \frac{k_{Vr} \cdot Tr(B^{(i)}) \cdot 12}{k_{MinReward} \cdot 88390704} \cdot 100\%.$$

So there are two main parameters which have significant influence on the $EAS_{\%}$: amount of the block reward intended for the treasury, and the portion of the treasury fund which is given to voters. Adjusting these two parameters, we can find a balance between the treasury supply and the expected amount of the participating stake.

As we have already stated, the initial values of the mentioned parameters are taken as follows:

1. $k_{Vr} = 0.2$, so that 20% of the treasury fund goes to voters;
2. $tr(b_j^{(i)}) = 1$, so that 1 new coin per block is issued to treasury needs.

Following that we have $EAS_{\%}8\%$ which means that given such parameters we will expect that 8% of the total stake will participate in a voting procedure[5]. At the first glance, it may seem that participation rate is not high enough to secure the voting procedure, but recall that these 8% of stake would decide on allocation of approximately 0.2% of the total stake (issued to the treasury) per month.

An absolute number of voters will also depend on the $minDepo$ which regulates the minimal amount of the stake to be deposited. Adjusting this parameter, it is possible to effectively regulate the total number of participants in a voting procedure. Taking $minDepo = 500$, we can estimate the maximal expected number of voters ($MENV$):

$$MENV = \frac{EAS}{minDepo} = \frac{4443408}{500} 8886 \; voters.$$

Note that it is highly unlikely that all deposits would be on the minimal level of 500 coins. So the actual number of voters should be lower.

[5] We need to mention that the expected participation rate will decrease from year to year, as well as the relative number of issued coins due to creation of new coins for block rewards.

4.5.2 55% attack

For the guaranteed control over the voting process, an attacker needs to have 55% of the voting power which is expected to be approximately 4.4% of the total stake in the system. In this case, she will be able to push any malicious proposal up. Given that a profit from this action is equal to 0.2% of the total stake per month, it follows that the attack is reasonable only if the negative impact on the tokens price is smaller than $\frac{0.2}{4.4} \cdot 100\% = 4.5\%$.

Taking into account that any successful malicious actions are expected to significantly impact the tokens price and as well as the deposit of an attacker (the expected price drop is much more than 4.5%) it makes such attacks unreasonable.

4.5.3 Self-balanced set of voters

A fixed amount of the total reward brings an important property to the developed system. It becomes self-balanced in terms of the amount of the participating stake (which consequently will be reflected on the actual number of voters). It means that after the initial bootstrap, an amount of the participating stake will reach some consistent value (we expect up to 8% of the total stake) and then this value mostly holds.

If a significant amount of deposited stake is revoked, then it would automatically boost up rewards for other voters. Overstated rewards, in turn, would attract new stakeholders to make a deposit until finally the rewards reach a consistent value. The same argumentation also works to the opposite side, and it will prevent from too much stake to be deposited.

As system grows, it becomes more stable and reliable that accordingly increases the$EAS_{\%}$ value, as there are fewer risks for stakeholders, so they are able to accept lower interest rate.

4.5.4 Freezing period

The rationale behind introduction of a freezing period t_{freeze} is to provide a mechanism for restricting voters from malicious actions and incentivize them to make optimal choices. If a voter acts maliciously, it would be immediately affect the coins price that respectively reduces an actual value of the deposited stake.

So the voters are incentivized to act in such a way that it will at least preserve an entire stability of the system and, accordingly, the tokens price. In reality, assuming that voters are willing to increase their profit, we expect rational decisions.

Together with direct rewards for active participation, the freezing period creates a well-established system for making optimal decisions regarding investments into different proposals.

4.6 Election rule

An election rule (voting system) is one of the main components of cryptocurrency treasury system which should be chosen very carefully. There is a big variety of voting systems: plurality voting, preferential voting, cardinal voting etc. [18, 19, 20, 21 etc.] They have different approaches and properties, but all of them try to meet the same goal - to satisfy as much as possible voters with election results. It is a difficult task because voters may act strategically (stepping down their direct preferences) in order to increase likelihood of their own satisfaction. The analysis of such behavior is within the scope of the game theory methods.

The voting systems comparison, usually, is based upon a set of specific criteria: majority rule, Pareto condition, independence of irrelevant alternatives (IIA), Arrow's impossibility theorem, Condorcet's method, Condorcet winner (loser) criterion, monotonicity criterion, participation criterion etc. [22] However, there is no voting system that satisfies all criteria. Furthermore, while choosing the election rule for a governance system, some other properties, like ballot and calculation simplicity, low voting machine costs etc. should be taken into account.

4.6.1 Types of voting systems

There are the following main types of voting systems [21]:

- plurality voting [18];
- majority voting;
- preferential (ranked) voting (Borda, STV, IRV etc.) [19,20];
- cardinal (rated) voting (Approval, Range, Cumulative) [21].

These systems are essentially oriented to single-winner elections, but most of them can be also used in multi-winner elections.

A **plurality voting system** is a voting system in which each voter is allowed to vote for only one candidate, and the candidate who polls more votes than any other candidate (a plurality) is elected [18]. Plurality rule is also known as a simple majority (not absolute majority) rule. If there are only two alternatives, then it is a very good procedure. However, for more than two alternatives the plurality voting has several problems [18, 21]:

- the information on voter preferences is gets ignored (only the most favorite candidate can be chosen);
- there is a dispersion of votes across ideologically similar candidates;
- It encourages voters not to vote for their true favourite, if that candidate is perceived to have little chance of winning;
- encourages voters to vote for one of the two candidates they predict are most likely to win, even if their true preference is neither, because a vote for any other candidate will likely have no impact on the final result.

Plural voting is distinguished from a majority voting system, in which, to win, a candidate must receive an absolute majority of votes (more votes than all other candidates combined) [18]. Majority voting has almost the same problems as plurality one.

Preferential or ranked voting describes certain voting systems in which voters rank outcomes in a hierarchy on the ordinal scale. When choosing among more than two options, preferential voting systems provide a number of advantages over plurality voting. One of the main advantages is that election results will more accurately reflect the level of support for all candidates.

There are many preferential voting systems: Instant-runoff voting (IRV) [19], Borda count [23], Single transferable vote [24], Schulze method [25], an optimal single-winner preferential voting system (the GT system) based on optimal mixed strategies computation [26] etc.

For example, IRV system has majority winners [19], but does not satisfy several criteria (IIA, Condorcet winner and monotonicity criteria) and has higher probability for ties in elections as compared to plurality voting [22].

Borda voting [23] system is simple, transparent and stable to small changes in rankings, but it also does not satisfy several criteria (majority rule, Condorcet winner and clone independence criteria) [22].

The analysis made in [27] has revealed several defects of preferential voting which were formulated as the following four paradoxes.

1. *No-show paradox.* The addition of identical ballots with candidate x ranked last may change the winner from another candidate to x.
2. *Thwarted-Majorities paradox.* A candidate who can defeat every other candidate in direct-comparison majority votes may lose the election.
3. *Multiple-Districts paradox.* A candidate can win in each district separately, yet lose the general election in the combined districts.
4. *More-is-less paradox.* If the winner were ranked higher by some voters, all else unchanged, then another candidate might have won.

Although the probabilities of these paradoxes seems to be very small, until their estimations received, a concern about preferential voting remains.

Furthermore, the Arrow's impossibility theorem [28,29] and the Gibbard–Satterthwaite theorem [30] prove that any useful single-winner voting system based on preference ranking is prone to some kind of manipulation. It has been shown by the Gibbard–Satterthwaite theorem that any ranked voting method which is not dictatorial must be susceptible to tactical voting.

Cardinal voting systems [20] are voting systems which allow a voter to give each candidate an independent rating or grade from among at least two levels of approval. The simplest possible cardinal system is approval voting, which allows only the two grades "approved" or "unapproved".

According to [20], under any cardinal system, if a candidate is given the maximal score for strategic reasons, the voter can still give that score also to all candidates whom she ranks higher. It is more sincere and if it has any effect at all, it will have contributed to the win of a candidate more preferred by the voter than her strategic choice. Cardinal systems are considered to be equally superior to plurality ones under both strategic and sincere voting.

Approval voting (AV) is a voting method that allows voters to approve any number of candidates [31]. The winner is the candidate(s) chosen by the largest number of voters. Approval voting is most often discussed in the context of single-winner elections, but variations using an approval-style ballot can also be applied to multi-winner (at-large) elections.

AV has a number of advantages [32]:

- simplicity;
- actual voting process is quick and easy;
- results are still easy to understand: a simple list of the candidates along with how many votes they received;
- tends to elect candidates who would beat all rivals head-to-head;

- tends to elect more moderate winners;
- alternate candidates get a more accurate measure of support;
- it is immune to Push-Over[6] and Burying[7].

Under AV a voter can make a decision about candidates to vote for in the two following ways [33]:

- he divides all candidates into two groups: those she approves and those she disapproves. There is no difference among candidates in each group (no one is superior to anyone in the group). Such voting is called dichotomous.
- she has some preference list *P*and she votes according to this list.

There is no full analysis of strategic voting under AV. Most of existed AV analysis is based on an assumption that the voter's preference is simply dichotomous. However, the dichotomous voting and voting according to an own preference list have quite different properties. The complexity of AV analysis is defined by quite large set of outcomes (it is larger than preferential voting one) [34]..

There is a number of AV variations. The most known are satisfaction approval voting and Yes-No voting.

Satisfaction approval voting (SAV) [35] is a voting system applicable to multi-winner elections. It uses an approval ballot, whereby voters can approve as many candidates as they like (no rankings). A voter's satisfaction score is the fraction of her approved candidates who are elected. If k candidates are to be elected, SAV chooses the set of k candidates that maximizes the sum of all voters' satisfaction scores.

In **Yes-No voting** each voter may vote "yes", abstain, or vote "no", the outcome is "yes" or "no," and all voters play interchangeable roles. There are many rules of elections outcome determination: absolute majority rule, qualified absolute majority rule, simple (or "relative") majority rule, voting using ratio and difference quotas.

4.6.2 A voting scheme for the treasury system

The analysis shows there is no voting system which satisfies all the criteria: while some of them does not satisfy IIA and Condorcet criterion, others - Pareto property or monotonicity, etc.

There is also no voting system which is completely immune to strategic voting. Cardinal voting systems are considered to be less vulnerable to strategic voting compared with plurality and preferential ones [20].

Preferential voting appeared in contrast to plurality voting, and allows voter to express her preference more accurately. However, the analysis of preferential voting systems has shown they have a number of paradoxes and are highly vulnerable to strategic voting and dictatorship.

AV has several advantages compared with the preferential one: simplicity, tendency to elect more moderate winners, more accurate measure of support for alternate candidates etc. There is

[6] Push-over (also called mischief voting) is a type of tactical voting in which a voter ranks a perceived weak alternative higher, but not in the hope of getting it elected.

[7] Burying is a type of tactical voting in which a voter insincerely ranks an alternative lower in the hope of defeating it.

no big reason to use range voting because of its complexity, but practically the same properties as AV. Furthermore, the most known cryptocurrency treasury (governance) systems use the AV modification – Yes-No-Abstain voting: Dash [6], The DAO [7], Fermat [36] etc. Thereby, we suppose that some of the approval voting options is the most suitable solution for the ETC treasury system.

We also propose to use Yes-No-Abstain voting, so in our model each voter may vote "yes", "no" or "abstain". Besides, we make voting deposit-weighted (the vote power is proportional to deposit voter made). Like in Dash, the election outcome is determined using difference quota: the alternative defeats the status quo if a number of "yes" votes exceeds a number of "no" votes by at least some specified absolute number, or difference quota, d.

Finally, the important difference between our voting system and classical Yes-No voting is that in our model (as in other cryptocurrency treasury voting systems) the number of winners is dynamic and limited by treasury funding reward. Actually, this number depends on the amount of coins allocated for the treasury funding and amounts required by each winning proposal. Obviously, such conditions should be taken into account in the formal analysis of the voting system. We propose a new formal voting model which consider all mentioned modifications and call it **Fuzzy Threshold Voting** (FTV). The Appendix A contains the description and analysis of the proposed model.

4.7 Computational and storage burden

It becomes very important for the developed system to estimate the introduced computational overhead. In this section we show that the proposed system does not have a significant impact on the general ETC performance.

First of all, we should mention that all ECTS transactions have a reasonable level of computational complexity (or gas consumption, if it would be implemented as a smart contract). They could be compared to the simple transfer transactions in Ethereum.

There are several types of transactions:

1. Deposit/RevokeDeposit - this type of transactions are not expected to be too frequent. Once a stakeholder made a deposit, she is able to participate in a voting procedure up until revocation, so she does not need to do this in each funding epoch.

2. Delegate/RevokeDelegate - this type of transaction is also expected to be rare. Once a voter decided on her delegation, she does not need to resubmit delegation transaction in each funding epoch.

3. Proposal ballot - a special transaction for project proposal submission. Some number of new proposals are expected in each funding epoch, but, from other cryptocurrencies experience, it is unlikely that the number of new proposals would be more than a couple of dozens.

4. Commitments/Voting ballots - these are the most frequent transactions because each voter will submit one commitment and the corresponding voting ballot in each funding epoch.

5. Payment - one-time transaction at the end of each epoch which pays money to the accepted proposals.

Thus it can be seen that the computational burden highly depends on the number of voters. From the previous section, we already know that the set of voters is self-balanced, so the stable

level of voter participation is expected during the normal operation. Given that there is an economic constraint for the total amount of the deposited stake, the maximal expected number of voters is estimated as $MENV = 8886$ (see section 4.5.1 "Expected number of voters").

Let us assume that we have 8886 voters and 100 proposals (it is unlikely to have such large number of proposals in the real world, but it is needed to estimate peak computational overhead). Given that each voter can submit only one deposit and delegation transaction per epoch, we can estimate the maximal number of transactions for a funding epoch (MNT):

$$MNT = 100 + 8886 \cdot 4 = 35644 \; transactions.$$

Given that Ethereum Classic can handle up to 15 tps on average which is approximately $38,8 \cdot 10^6$ transactions per funding epoch, we can calculate the transactions overhead (TO) introduced by the treasury system:

$$TO_{\%} = \frac{35644}{38,8 \cdot 10^6} \cdot 100\% = 0.09\%.$$

So it can be seen that even with an overestimated computational load, an impact of the treasury system on the Ethereum performance is negligible.

4.8 Directions for further development

In the process othe ECTS development, we took into account the experience of the Dash Governance System (DGS), the most advanced practically deployed system of such type, which effectively provides funding for Dash team since September 2015.

For the next research steps we plan to take into consideration gained experience on voting specifics to create an additional incentive for achieving consensus on treasury proposals. Besides it, we work on advanced cryptographic voting protocol that additionally improve theoretical properties of the voting scheme (that may be useful in the future, when system reaches high maturity level).

5 Attacks overview

This section gives a brief overview of potential attacks on the proposed treasury system. As mentioned above, the current version of the report does not provide strict formal analysis, game theoretic proofs, etc., leaving that for the further stages of research.

Attacks on the treasury system may have the following objectives:

1. To take full or partial control over treasury coins to own them (steal treasury coins).
2. To block treasury operation to prevent funding of cryptocurrency development (DoS attack).

Obviously, if the attacker takes the full control and steals treasury coins, she blocks treasury operation, so implementation of first type of the attack leads to automatic implementation of the second one.

But approaches may exist where an attacker cannot steal treasury funds, but for some purpose attempts to block cryptocurrency funding.

5.1 Taking control over treasury coins

5.1.1 Direct attacks on the voting procedure

All treasury information is included into the blockchain, so falsifying the winner selection procedure (see 4.2) must break underlying cryptocurrency consensus protocol.

5.1.2 Influence on the voting process

Proposals and a list of voters are fixed and cannot be changed during the treasury epoch (see 4.4.1).

As compared to different voting system, the chosen one (fuzzy threshold voting) has

- very good properties to express honest majority opinion (maximizes general satisfaction level);
- less manipulative strategies for an attacker (with no paradoxical cases happening, as e.g., in the preferential voting);
- easily understandable without deep learning of specialized voting literature (for more details see 4.6).

Ballots are committed secretly and opened when every voter has made their commitment, and that significantly limits attackers' strategies.

Thus, the voting scheme is transparent and easily verifiable. Some properties of the scheme are formally analyzed and proved in the Appendix A).

5.1.3 Taking control over a "treasury account"

The proposed treasury has no specific account for accumulating and redistributing of funds for core consensus implementation. Coins are sent directly to accounts of proposal owners. To organize a successful attack of this type, it is needed to take control over multiple accounts breaking underlying cryptocurrency.

5.1.4 Bribing influential part of voters

An attacker and bribed participants must have sufficient amount of stake, and successful attacks lead to decreased cryptocurrency exchange rate. Due to the deposit lock for 3 months (see 2.4), the attack is expected to have a negative influence on the adversarial stake (with respect to some fiat currency that has a constant value).

The monthly treasury budget is significantly lower than the stake needed to control it, so the expected losses are higher than attacker gains.

There is an economic incentive for an honest rational stakeholder to take part in the treasury voting that helps to have sufficiently large amount of honest voters (see 4.5.3).

5.1.5 Attack of 55%

For a guaranteed control over the voting process an attacker needs to have over 55% of the voting power, which is currently expected to be approximately 4.4% of the total stake in the system.

A successful attack decreases exchange rate, so expected losses of locked malicious deposits are higher than the attacker's gains (see 4.5.2 and 5.1.4).

5.2 Blocking the treasury operation

5.2.1 Issuing large amount of proposals

Each proposal has a fixed cost (5 coins in this proposal). Thus, to flood the system, an attacker must burn the stake proportionally to the amount of her proposals. Moreover, the voting client software can easily sorts requests according to the requested funding, so flood proposals won't interfere with honest proposals (or a DoS attack will require a budget comparable with a more expensive attack of taking control over all treasury coins).

5.2.2 Creation of excessive number of voting accounts

To become a voter, a stake hold deposit is needed (500 coins in this proposal). Considering a significant reserve of voting scheme performance, the stake required for this attack is higher than for taking control over all treasury coins.

5.2.3 Generation of many commitments and openings from each voting account

There is only one commitment per voter included into the blockchain (all the rest are discarded by miners). Only one opening that strictly corresponds to the commitment is included into the blockchain.

Thus, the treasury architecture accepts only one commitment and opening per voting account.

Together with this, there is a limited amount of delegation transactions (one per epoch) that also prevents an attacker from successful attack implementation.

6 Conclusions

The proposed Ethereum Classic Treasury System (ECTS) is intended to create community controlled decentralized process for funding of cryptocurrency development and further improvement.

ECTS has acceptable computation and blockchain space load for the system, having no significant influence on the ETC performance.

Development funding is provided via regular coin transfers to accounts associated with community approved proposals intended to solve tasks of cryptocurrency improvement and maintenance.

The system is open and verifiable, so anyone can provide her proposal and trace any decision made in the system, up to the full history of an ECTS operation.

Selection of winners for funding among all proposals are made by ETC community members, who locked their stake and incentivized for taking part in an honest treasury operation.

The voting scheme chosen for ECTS (fuzzy threshold voting, FTV) has advantages over other voting schemes. It is also introduced the formal mathematical model of FTV, and some of its properties were already proved.

Voters in ECTS have the ability to make delegation to trusted qualified participants (who also have their own incentive).

Initial analysis found no attack on ECTS that allows attacker to increase her stake (an attacker loses her funds even in a very unlikely case of taking full control over the treasury system). Honest community participants always have the ability to interfere for future attack prevention.

The detailed formal analysis and proofs are planned for the further stages of the research.

We expect that the ECTS implementation will increase ETC stability and future prospects, providing additional benefits both for stakeholders and miners from a more stable and predictable system.

Recall that all introduced system parameters (such as number of coins for the treasury fund or amount of voters reward) as well as architectural solutions are not set in stone and the subject for further analysis and discussion with the community members.

7 References

[1] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System.

[2] Crypto-Currency Market Capitalizations http://coinmarketcap.com/

[3] R3CEV blockchain consortium: present state. https://www.linkedin.com/pulse/r3cev-blockchain-consortium-present-state-carlo-r-w-de-meijer

[4] The Russian Government is Testing Blockchain for Document Storage. http://www.coindesk.com/ the-russian-government-is-testing-blockchain-for-document-storage/.

[5] Blockchain Tracker: Governments Trust In Blockchain. http://fintechranking.com/2017/02/04/ blockchain-tracker-governments-trust-in-blockchain/.

[6] E. Duffield, D. Diaz. Dash: A PrivacyÂCentric CryptoÂCurrency. https://www.dash.org/wp-content/uploads/2015/04/Dash-WhitepaperV1.pdf

[7] Whitepaper: DECENTRALIZED AUTONOMOUS ORGANIZATION TO AUTOMATE GOVERNANCE.

[8] D. Williams. The DFINITY "Blockchain nervous system". https://medium.com/dfinity-network-blog/the-dfinity-blockchain-nervous-system-a5dd1783288e#.pwji0ajtf

[9] Ethereum White Paper https://github.com/ethereum/wiki/wiki/White-Paper

[10] Gavin Wood. Ethereum: A Secure Decentralised Generalised Transaction Ledger.

[11] Dash Governance System: Analysis and Suggestions for Improvements. https://iohk.io/research/papers/ dash-governance-system-analysis-and-suggestions-for-improvement/

[12] Dash Governance System: Dash Core Team Response to the report "Dash Governance System: Analysis and Suggestions for Improvement" https://www.dash.org/wp-content/uploads/ 2016/12/Dash-Governance-System-Dash-Core-Team-Response.pdf

[13] Gilles Brassard, David Chaum, and Claude Crepeau, Minimum Disclosure Proofs of Knowledge, Journal of Computer and System Sciences, vol. 37, pp. 156–189, 1988.

[14] Stephanie Delaune, Steve Kremer, and Mark Ryan. Verifying privacy-type properties of electronic voting protocols. Journal of Computer Security, 17(4):435–487, 2009.

[15] L.M. Goodman. Tezos — a self-amending crypto-ledger. White paper.

[16] L.M. Goodman. Tezos: A Self-Amending Crypto-Ledger Position Paper.

[17] Svetlana Z. Lowry, Poorvi L. Vora. Desirable properties of voting systems. http://ws680.nist.gov/publication/ get_pdf.cfm?pub_id=903961.

[18] Plurality/majority systems. Access mode: https://www.mtholyoke.edu/acad /polit/damy/BeginnningReading/plurality.htm.

[19] Steven Hill. "Instant Runoff Voting". Access mode: https://static.newamerica.org/attachments/4119-instant-runoff-voting/NAF_10big_Ideas_9.677a9863789b4a23bc356ed7940b5cb8.pdf.

[20] Claude Hillinger. "The Case for Utilitarian Voting".

[21] Voting procedures and their properties. Access mode:

[22] Eric Erdmann. "Strengths and Drawbacks of Voting Methods for Political Elections".

[23] Peter Emerson. "The original Borda count and partial voting".

[24] Single Transferable Vote http://www.electoral-reform.org.uk/ single-transferable-vote

[25] Schulze, Markus. "A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method." Social Choice and Welfare 36.2 (2011): 267-303.

[26] Ronald L. Rivest and Emily Shen. "An Optimal Single-Winner Preferential Voting System Based on Game Theory".

[27] Peter C. Fishburn and Steven J. Brams. "Paradoxes of Preferential Voting".

[28] Arrow, Kenneth J. "A Difficulty in the Concept of Social Welfare". Access mode:

[29] Arrow, Kenneth J. "Handbook of Social Choice and Welfare".

[30] Andrew C. Eggers. "A Simple Proof of the Gibbard-Satterthwaite Theorem".

[31] Brams, Steven, Fishburn, Peter. "Approval Voting".

[32] https://electology.org/approval-voting.

[33] Niemi, R. (1984). The Problem of Strategic Behavior under Approval Voting. The American Political Science Review, 78(4), 952-958. doi:10.2307/1955800.

[34] Steven J. Brams, M. Remzi Sanver. "Critical Strategies Under Approval Voting: Who Gets Ruled In And Ruled Out".

[35] Steven J. Brams, D. Marc Kilgour. "Satisfaction Approval Voting".

[36] Fermat, the Internet of People and the Person to Person Economy. https://hackernoon.com/fermat-the-internet -of-people-and-the-person-to-person -economy-ce933865a0b0#.e07m1vd9b.

[37] ECIP-1017. Monetary Policy and Final Modification to the Ethereum Classic Emission Schedule. https://github.com/ethereumproject/ECIPs/blob/master/ECIPs/ECIP-1017.md

[38] A Joint Statement on Ethereum Classic's Monetary Policy. https://iohk.io/blog/ethereum-classic/a-joint-statement-on-ethereum-classics-monetary-policy/

A Appendix. Voting Model - Fuzzy Threshold Voting

A.1 Introduction

In this document, we introduce a mathematical model for a voting system which is used mainly in cryptocurrencies [1]. We call it Fuzzy Threshold Voting (FTV). The word "fuzzy" emphasizes that a set of winners in some cases may contain candidates with lower scores and may not contain candidates with higher scores. This property is a result of the fact that voting outcome depends not only on the scores of candidates (i.e. on their rankings in the voters' ballots) but also on other factors – budgets of corresponding projects (actually projects are the candidates).This is the main difference of our voting model in comparison with the AV (as other voting models) [2-11]. On one hand, this is also the main problem of the FTV model, because all existing theoretical analysis for voting models fails in this case. On the other hand, knowledge of projects' budgets gives voters some additional information which helps them to make more considerate decisions during the voting or to form more accurate preferences.

The proposed FTV voting model is considered to be some generalization of Yes-No voting model with expanded features.

Two-factor dependence of the voting procedure and its fuzzy threshold makes theoretical analysis of this model much more complicated. But in spite of this, we managed to obtain some results that may be considered analogies of the main results for the classical AV model [12, 13].

In this part, in addition to building of an adequate mathematical model for the new voting system, we also analyzed its main properties, such as optimality (in some sense) of the voting procedure, existence of admissible strategies, advantages of sincere voting and necessity of secret voting. We also show that one of the unquestionable advantages of this model is the Condorcet rule. It means that the Condorcet winner, or the obvious leader of voting, always belongs to the set of winners (except, maybe, some improbable case of specific tie voting).

A.2 Definition of FTV model

In what follows, we will mainly use terms and definitions from [12]. We denote by I the finite set of *voters* and by X the finite set of *projects* (*alternatives*, *candidates*) and for some $n \in N$: $L^{(n)} = \{-n, -(n-1), ..., -1, 0, 1, ..., n-1, n\}$. The set $L^{(n)}$ contains all possible rankings that voters may give to projects. We assume $\#I \geq 2$ and $\#X \geq 2$. Every voter $i \in I$ has a preference over X expressed by an utility function $u_i : X \longrightarrow R$. In some particular cases, we will assume that $u_i : X \longrightarrow L^{(n)}$, i.e. the set of possible preferences coincides with the set of projects' rankings. For example, Proposition (A.2). that shows the optimality of voting procedure in some sense, is proved just with such restrictions. But these restrictions are quite appropriate and much closer to reality than, for example, in the Satisfaction AV model [14] or in the Proportional AV model [15], because our model seriously takes into account the preference levels of all voters.

Given two projects $x, y \in X$, the voter i finds x at least *as good as* y if $u_i(x) \leq u_i(y)$. A project x is *high* in u_i if for all $y \in X$ $u_i(x) \geq u_i(y)$. We say that x is *low* in u_i if for all $y \in X$ $u_i(x) \leq u_i(y)$. We call u_i *null* whenever i is indifferent among all alternatives, i.e. $u_i(x) \geq u_i(y)$ for every $x, y \in X$. If u_i is null, then every project is both low and high

in u_i. If u_i is not null then the projects that are higher in u_i and those which are lower in u_i form disjoint sets. We assume that each project the voter likes has some positive "rating" in her preferences,and each project the voter dislikes has some negative "rating".

A *ballot* B_i of the voter i is any partition of the set X on $2n+1$ subsets $H_i^{(l)}$, $l \in L^{(n)}$, which satisfy the following conditions:

1. $H_i^{(l)} \cap H_i^{(k)} = \varnothing$ for any $l, k \in L^{(n)}$, $l \neq k$;
2. $\cup_{l \in S} H_i^{(l)} = X$.

Note that one or more of subsets $H_i^{(l)}$, $l \in L^{(n)}$, may be empty.

It follows immediately from this definition that the number of possible ballots (or the number of voting strategies) for each voter is equal to $(\#X)^{2n+1}$.

When the voter i, $i \in I$, casts ballot $B_i = (H_i^{(-n)}, ..., H_i^0, ..., H_i^{(n)})$, we say that i *approves* the projects in $H_i^{(l)}$, where $l > 0$, *with the (positive) score* l, *abstains* from the vote on the projects in H_i^0 and *rejects* the projects in $H_i^{(l)}$, where $l < 0$, *with the (negative) score* l.

For each voter i, $i \in I$, who casts ballot $B_i = (H_i^{(-n)}, ..., H_i^0, ..., H_i^{(n)})$, we define her *score function* $s_i : X \longrightarrow L$, where $s_i(x) = l$ if $x \in H_i^{(l)}$.

The FTV model is rather similar to the *Approval Voting* (AV) model, but also has some essential differences. The first difference is in the definition of a ballot. In the AV model ballot B_i consists of only approved candidates, i.e. candidates from $H_i^{(l)}$ for a positive l. And all these candidates have the same "rating" in the ballot in spite of a voter's preferences. We also may say that in the AV model $n = 1$ and $H_i^{(0)} = \varnothing$ for each $i \in I$. /The other differences we will see below in the definition of score function and in the definition of voting procedure.

For the AV model, the following natural restrictions are usually made [12, 13] to restrict the set of voting strategies that a voter is supposed to possibly use:

1. It is supposed that voters use *undominated* strategies that are often called "*admissible strategies*" and can be characterized as follows: if a voter's preference is strict, she approves her preferred candidate, she does not approve her worst candidate and no constraint is imposed to other, intermediate candidates.
2. The other restriction is *sincerity requirement*, which imposes that when the voter approves a candidate, she also approves all the candidates she strictly prefers to this one.

But our model is rather different from the AV. We won't restrict strategies in the same way, but we will use some definitions of *sincere voting* for FTV. Let the voter i have some definite preferences on the set X of all projects.

Definition A.2.1. Let $u_i : X \longrightarrow L^{(n)}$. We say that the voter i *votes sincerely* (in a narrow sense) if for any $x \in X$ the equality holds:

$$u_i(x) = \sum_{l \in L} l \cdot \delta(x, H_i^{(l)}),$$

where

$$\delta(a, A) = \begin{cases} 1, & \text{if } a \in A \\ 0, & \text{otherwise.} \end{cases}$$

Definition A.2.2. We say that the voter i *votes sincerely* (in a wide sense) *for two projects* $x, y \in X$, if the inequality

$$u_i(x) > u_i(y)$$

involves inequality $s_i(x) \geq s_i(y)$ (or involves $x \in H_i^{(l)}$, $y \in H_i^{(j)}$ for some $l \geq j$).

We say that the voter i *votes sincerely* (in a wide sense) if she votes sincerely for any $x, y \in X$.

Definition A.2.3. We will call the set $B = (B_i)_{i \in I}$ of all ballots *a ballot profile* and for all $i \in I$ define $B_{-i} := B \backslash B_i = (B_j)_{j \in I \backslash \{i\}}$. We will write $B = (B_i, B_{-i})$ whenever we wish to highlight the dependency of B with respect to i-th ballot. We refer to B_{-i} as a *ballot profile without* i.

Definition A.2.4. Given a profile B, *the score of project* $x \in X$ is

$$s(x, B) = \sum_{l \in L} l \cdot \#\{i \in I : x \in H_i^{(l)}\}.$$

Note that in the AV model the score of candidate $x \in X$ is defined as

$$s(x, B) = \#\{i \in I : x \in B_i\}$$

– the number of ballots that candidate x belongs to.

Definition A.2.5. Given a profile B, we say that the project $x \in X$ is *acceptable candidate*, if

$$s(x, B) \geq 0.1 \cdot |I|.$$

In this model only acceptable candidates may be the winners.

Definition A.2.6. We set some positive value M and will call this value *a common budget* given for realization of projects from the set X.

For each project $x \in X$ we set some positive value $m(x)$ which is *the budget of the project* x. In what follows we will assume that $\forall\, x \in X : m(x) \leq M$.

Definition A.2.7. We say that some subset $S \subset X$ *exhausts the common budget* M if the following conditions hold:

1. $\sum_{x \in S} m(x) \leq M$;
2. If $S \neq X$ then $\forall\, y \in X \setminus S : \sum_{x \in S} m(x) + m(y) > M$.

Given a ballot profile B, among all (acceptable) projects $X = \{x_i\}_{i \in I}$ we have to choose the set of *winners* (*winning projects*) $W(B) \subset X$. On one hand, the winners must be the projects with the highest scores; on the other hand, the set of winners must exhaust the common budget M. The latter restriction defines one more difference between AV and FTV models.

Definition A.2.8. Let $\#X = k$ and, given a ballot profile B, the sequences

$$x_1, ..., x_k, x_i \in X, \tag{A.1}$$

is organized in a such way that

$$s(x_1, B) \geq ... \geq s(x_k, B). \tag{A.2}$$

In what follows we will assume that, given a ballot profile B, conditions (A.1), (A.2) hold.

Definition A.2.9. (Voting procedure). To define the set $W(B) \subset X$ of *winners* (*winning projects*) we will use the following (iterative) *voting procedure*:

1. $$x_1 \in W(B); \tag{A.3}$$

2. $$for\ i = \underline{2,k} : x_i \in W(B) \Longleftrightarrow \sum_{j=1}^{i-1} m(x_j)\delta(x_j, W(B)) + m(x_i) \leq M.$$

We should emphasize that the FTV voting procedure (A.3) is the main difference of the FTV model from the AV model, because in the FTV case the set of winners depends not only on the "scores" of candidates, but also on their budgets, and the latter dependence is very essential in forming of the set of winners. This dependence involves a lot of difficulties in theoretical analysis of the FTV model. The main difference is that it is possible that there are two projects $x, y \in X$ such that under some profile B $s(x, B) > s(y, B)$ and $y \in W(B)$ but $x \notin W(B)$. This feature makes impossible to prove some statements which usually hold for "classical" voting models. For this reason, sometime we will use some simplification of the voting procedure to prove the main statements.

Definition A.2.10. (simplified voting procedure). Let (A.2), (A.1) hold. According to a simplified voting procedure, given a ballot profile B,

$$W(B) = \{x_1, x_2, ..., x_t\},$$

where

$$t = max\{l \geq 1 : \sum_{i=1}^{l} m(x_i) \leq M\}. \tag{A.4}$$

Under this simplification the set of winners $W(B)$ is such that if $x_i \in W(B)$ for some $i \geq 2$, then $x_j \in W(B)$ for all $j = \underline{1, i-1}$. But we should note that even in this case the threshold is still fuzzy and so is the number of winners. Also note that voting procedures A.3 and A.4 are identical if and only if the ballot profile B is such that for some t $(1 \leq t \leq n)$: $\{x_1, ..., x_t\}$ exhausts the common budget M.

Now proof some important properties of voting procedures A.3 and A.4 .

Proposition A.1. ***(i)*** *Under voting procedures A.3 and A.4 a Condorcet winner (if exists) always belongs to the set of winners.*

(ii) *Voting procedures A.3 and A.4 satisfy a monotone rule:*

a. *if $x \in W(B)$ and for some profile $\tilde{B}$: $s(x, \tilde{B}) > s(x, B)$ and $s(y, \tilde{B}) = s(y, B)$ for all $y \in X\backslash\{x\}$, then $x \in W(\tilde{B})$;*

b. *if $x \notin W(B)$ and for some profile $\tilde{B}$: $s(x, \tilde{B}) < s(x, B)$ and $s(y, \tilde{B}) = s(y, B)$ for all $y \in X\backslash\{x\}$, then $x \notin W(\tilde{B})$.*

Proof. Let $x \in X$ is such that

$$\forall i \in I \quad \forall y \in X\backslash\{x\} : \quad s_i(x) \geq s_i(y).$$

Then for any profile B and for any $y \in X \backslash \{x\}$

$$s(x, B) = \sum_{i \in I} s_i(x) \geq \sum_{i \in I} s_i(y) = s(y, B),$$

so $s(x, B) = max_{y \in X} s(y, B)$.

According to Definition A.2.6, $m(x) \leq M$. So under voting procedures A.3 or A.4 $x \in W(B)$ for any profile B. The first statement is proved.

2) Let for some profile B $x \in W(B)$ and ballot profile $\tilde{B}$ be such that

$$s(x, \tilde{B}) > s(x, B) \text{ and for any } y \in X \backslash \{x\}\text{: } s(y, B) = s(y, \tilde{B}).$$

Let projects $x_1, ..., x_n$ be written according to the definition A.2.8 . And let under ballot profile B $x = x_i$ for some $i = 1, n$. As $x \in W(B)$ then $\sum_{j=1}^{i} m(x_j) \leq M$. If $s(x, \tilde{B}) > s(x, B)$, then under ballot profile $\tilde{B}$ $x = x_{i-l}$ for some l such that $0 \leq l \leq i - 1$. Then $\sum_{j=1}^{i-l} m(x_j) \leq \sum_{j=1}^{i} m(x_j) \leq M$, so $x \in W(\tilde{B})$.

Part ***a*** of the second statement is proved.

To prove part ***b*** we can use the same considerations.

The proposition is proved. □

Note A.1. in the case of $\forall\, x, y \in X :\ s(x; B) \neq s(y; B)$ the conditions (A.3) define the set W uniquely. But in the opposite case (like in our model) a tie voting may happen. In what follows we assume that in the case of tie voting some fair procedure is used.

We suppose that voters vote secretly by casting ballots while the FTV model is used as the outcome function. So we consider a normal form game where strategy set for any voter i is the set of all possible ballots (partitions of X). Hence a ballot profile B is also a strategy profile and the outcome is the set of winning projects $W(B) \subset X$.

As $W(B)$ usually contains more than one project, our strategic analysis requires knowledge of voter's preferences over non-empty sets of X. We assume that ties over outcomes are broken according to the Note A.1, and that votes evaluate outcomes by expected Von-Neumann Morgenstern utilities. So the utility that the voter i attaches to a set S of winning projects is

$$u_i(S) = \frac{1}{\#S} \sum_{x \in S} u_i(x). \tag{A.5}$$

Ideally, we wished to choose the set $W(B)$ of winning projects in a such way that this set would maximize the average outcome

$$U(S) = \frac{1}{\#I} \sum_{i \in I} u_i(S). \tag{A.6}$$

But, generally speaking, it isn't so for our voting procedure.

Nevertheless, we can give some conditions that are sufficient for W to maximize the function (A.6).

Proposition A.2. *Let, given a ballot profile B and under conditions (A.1), (A.2), (A.4), $W = W(B) = \{x_1, ..., x_t\}$. Then under FTV sincere voting assumption (in the narrow sense):*

$$max_{S \subset X:\ |S| \geq t}\ U(S) = U(W),$$

i.e. the average (on all voters) value of satisfactory function, or average outcome, takes it's maximal value on the set of winners defined by the voting procedure (A.3).

Proof. Rewrite the function (A.6) using (A.5) in a such way:

$$U(S) = \frac{1}{\#I}\sum_{i \in I} u_i(S) = \frac{1}{\#I}\sum_{i \in I}(\frac{1}{\#S}\sum_{x \in S} u_i(x)) = \frac{1}{\#I}\frac{1}{\#S}\sum_{x \in S}(\sum_{i \in I} u_i(x)). \tag{A.7}$$

It's easy to see that under FTV sincere assumption and according to the definition of the score of project

$$\sum_{i \in I} u_i(x) = s(x, B).$$

So we can rewrite the expression (A.7) as

$$U(S) = \frac{1}{\#I}\frac{1}{\#S}\sum_{x \in S}(\sum_{i \in I} u_i(x)) = \frac{1}{\#I}\frac{1}{\#S}\sum_{x \in S} s(x; B).$$

Now it's sufficient to prove that for any $S \subset X$, $\#S = f \geq t$, $S = \{s_1, ..., s_f\}$:

$$U(S) \leq U(W).$$

According to the definition of function $U(S)$, this is the same that

$$\frac{\sum_{i=1}^{f} s(s_i, B)}{f} \leq \frac{\sum_{i=1}^{t} s(x_i, B)}{t}. \tag{A.8}$$

Define $T = S \cap W$, $\tilde{W} = W \backslash S$, $\tilde{S} = S \backslash W$, $f = t + v$ and

$$\Sigma_1 = \sum_{x \in T} s(x, B), \Sigma_2 = \sum_{x \in \tilde{W}} s(x, B), \Sigma_3 = \sum_{x \in \tilde{S}} s(x, B).$$

Let $|\tilde{W}| = z$, then $|\tilde{S}| = v + z$. Now we can rewrite (A.8) in an equivalent form:

$$\frac{\Sigma_1 + \Sigma_3}{t + v} \leq \frac{\Sigma_1 + \Sigma_2}{t},$$

or

$$t\Sigma_1 + t\Sigma_3 \leq t\Sigma_1 + v\Sigma_1 + t\Sigma_2 + v\Sigma_2,$$

or

$$t\Sigma_3 \leq v\Sigma_1 + t\Sigma_2 + v\Sigma_2. \tag{A.9}$$

According to the conditions of this proposition, $W = \{x_1, ..., x_t\}$, so $\tilde{S} \subset \{x_{t+1}, ..., x_k\}$, where $k = \#X$ and for any $x \in W$ and any $y \in \tilde{S}$: $s(x; B) < s(y; B)$. Also note that Σ_1 consists of $t - z$ items, Σ_2 consists of z items, and Σ_3 consists of $v + z$ items.

Then we can estimate the lefthand side of (A.9) as

$$t\Sigma_3 \leq t(v + z) \cdot max_{x \in \tilde{S}} s(x, B) \leq t(v + z) \cdot min_{x \in W} s(x, B).$$

Similarly, we can estimate the right part of (A.9) as

$$v\Sigma_1 + t\Sigma_2 + v\Sigma_2 \ \geq \ (tz + v(t-z) + vz) \cdot min_{x \in W} s(x; B) = t(z+v) \cdot min_{x \in W} s(x; B),$$

and (A.9) holds, then the equivalent inequality (A.8) also holds. □

Corollary A.1. *Let, given a ballot profile B and under conditions (A.1), (A.2), $W = W(B) = \{x_1, ..., x_t\}$.*

Let also for any $S \subset X$ the following condition hold: if S exhausts the common budget M, then $|S| \geq t$.

The, under the FTV sincere voting assumption (in the narrow sense):

$$max_{S \subset X, \ S \ exhausts \ M} \ U(S) = U(W).$$

This corollary means that under its conditions the set of winners chosen according to the voting model (A.3) is the best choice to "satisfy" in average all voters.

Note A.2. Proposition A.2 also holds when preferences of voters are proportional to the elements of L, i.e. when $\exists \ C > 0, \forall \ i \ \in \ I, \forall \ x \ \in \ X$: $u_i(x) \ \in \ C \cdot L = \{-Cn, ..., -C, 0, C, ..., Cn\}$.

A.3 Some properties of FTV model

Now we will discuss some properties of the AV model, that we'd like to be true (at least with some modification) for the FTV model. First we give some basic properties ofthe AV model and will show, using counterexample, that this properties do not run for the FTV model.

Then we will state some other properties, which may be considered, in some sense, as analogs of the AV model properties, and will prove these properties hold for the FTV model.

First we need some more definitions from [12].

Definition A.3.1. For any voter i with preference u_i, we say that *the ballot B_i (weakly) dominates the ballot $\widetilde{B_i}$* if and only if

$$u_i(W(B_i, B_{-i})) \ \geq \ u_i(W(\widetilde{B_i}, B_{-i}))$$

for all B_{-i} and

$$u_i(W(B_i, B_{-i})) > u_i(W(\widetilde{B_i}, B_{-i}))$$

for some B_{-i}.

A ballot is *undominated* if and only if it is dominated by no ballot.

Following [13], we also call undominated ballots *admissible* and use either word.

The following proposition characterizes admissible ballots for the AV model.

Proposition A.3. *(i) If u_i is null then all ballots are admissible for the voter i.*

(ii) Let the number of voters be at least three. If u_i is not null then the ballot B_i is admissible for the voter i if and only if B_i contains every candidate who is high in u_i and no candidate who is low in u_i.

Proof. The part (i) is trivial and follows directly from the definition.

The part (ii) is of great interest. Roughly speaking, it claims that it's better for a voter to vote "sincerely", or "according to her preferences", at least for the most preferable and for the least preferable projects.

Now we show that the part (ii), generally speaking, doesn't hold for the FTV model. □

Example A.1. Counterexample

Let the set of voters $I = \{1, ..., 10\}$, the set of projects $X = \{x_1, ..., x_7, y\}$, $L = \{-2, -1, 0, 1, 2\}$, and $\forall\ i\ \in\ I$: $u_i(x)\ \in\ L$.

Define the common budget as $M = 7$ and the projects' budgets as

$$m(x_i) = 1, i = \underline{1, 7}; m(y) = 3.$$

Table A.1: Preferences of the first voter

Project, x_i	x_1	x_2	x_3	x_4	x_5	x_6	x_7	y
$u_1(x_i)$	0	0	0	0	1	1	1	2

Below we consider two options of voting for the first voter. In the first option she votes "sincerely" for the most preferable project y with the ballot $\widetilde{B_1}$, i.e. the project y has the higher rating in $\widetilde{B_1}$ (though it doesn't matter, but without loss of generality we may assume that she also votes sincerely for other projects). In the second option she votes in an opposite manner with the ballot B_1, where all projects except y have the same rating as in ballot $\widetilde{B_1}$, and the project y has the smallest rating. And we will show that for some B_{-1}:

$$u_1(W(B_1, B_{-1})) > u_1(W(\widetilde{B_1}, B_{-1})).$$

Option 1: voter votes for y **according to her preferences.**

For ballot $\widetilde{B_1}$, $y\ \in\ H_1^{(2)}$.

Let the other votes form the profile B_{-1} be such that in profile $\tilde{B} = (\widetilde{B_1}, B_{-1})$ the scores of projects are distributed according to the Table A.2.

Table A.2: Scores of projects under the profile $\tilde{B} = (\widetilde{B_1}, B_{-1})$

Project, x_i	x_1	x_2	x_3	x_4	y	x_5	x_6	x_7
$s(x_i, \tilde{B})$	8	7	6	5	4	3	2	1

In this case $W(\tilde{B}) = \{x_1, ..., x_4, y\}$ and $u_1(W(\tilde{B})) = \frac{2}{5} = \frac{14}{35}$.

Option 2: voter votes for y **against her preferences.**

For ballot B_1, $y\ \in\ H_1^{(-2)}$.

Let the other voters vote as in the option 1 and their votes form the same profile B_{-1}. Then in the profile $B = (B_1, B_{-1})$ the scores of projects are distributed according to the Table A.3.

Table A.3: Scores of projects under the profile $B = (B_1, B_{-1})$

Project, x_i	x_1	x_2	x_3	x_4	x_5	x_6	x_7	y
$s(x_i, B)$	8	7	6	5	3	2	1	0

In this case $W(B) = \{x_1, ..., x_7\}$ and $u_1(W(B)) = \frac{3}{7} = \frac{15}{35}$, i.e. $u_1(W(B))>u_1(W(\tilde{B}))$.

After this example one may conclude that, roughly speaking, sincerity isn't the best strategy in this model. But below we will see that, nevertheless, there exists some sense to vote sincerely.

Now we prove some statements for the FTV model that are particularly similar to the properties of the AV model proved in Proposition (A.3).

Proposition A.4. *Under the voting procedure (A.4) the next statements hold:*

(i) If u_i is null then all ballots are admissible for the voter i.

(ii) If for some voter $i \in I$ and some project $y \in X$ the utility function u_i satisfies the inequality

$$u_i(y) \geq \sum_{x \in X \setminus \{y\}} |u_i(x)|,$$

then the ballot B_i is admissible for voter i only if $y \in H_i^{(n)}$.

(iii) If for some voter $i \in I$ and some project $y \in X$ utility function u_i satisfies the inequality

$$u_i(y) \leq - \sum_{x \in X \setminus \{y\}} |u_i(x)|,$$

then the ballot B_i is admissible for the voter i only if $y \in H_i^{(-n)}$.

Proof. (i) The part (i) follows directly from the definition.

(ii) Consider a ballot $B_i = \{H_i^{(-n)}, ..., H_i^{(n)}\}$ for which $y \notin H_i^{(n)}$. It means that for the ballot B_i, $y \in H_i^{(v)}$ for some $-n \leq v \leq n-1$. Let $\widetilde{B_i}$ be the ballot for which $\tilde{H}_i^{(v)} = H_i^{(v)} \setminus \{y\}$, $\tilde{H}_i^{(n)} = H_i^{(n)} \cup \{y\}$ and $\tilde{H}_i^{(j)} = H_i^{(j)}$ for all $j \in L \setminus \{v, n\}$. We will prove that $\widetilde{B_i}$ dominates B_i.

Given any B_{-i}, all candidates except y have the same scores at (B_i, B_{-i}) and $(\widetilde{B_i}, B_{-i})$ while the score of y is raised by $n - v$ units at the latter ballot profile. Therefore, regarding the sets of winning projects $Y = W(B_i, B_{-i})$ and $\tilde{Y} = W(\widetilde{B_i}, B_{-i})$, the following three cases are exhaustive:

1. $y \notin Y, y \notin \tilde{Y}$;
2. $y \in Y, y \in \widetilde{Y}$;
3. $y \notin Y, y \in \tilde{Y}$.

In the first two cases $Y = \tilde{Y}$ so $u_i(Y) = u_i(\tilde{Y})$.

In the third case, let $Y = \{x_1, ..., x_t\}$, then $\tilde{Y} = \{x_1, ..., y, ..., x_{t-l}\}$ for some $0 \leq l \leq t-1$. We show that in this case $u_i(\tilde{Y}) > u_i(Y)$. It's sufficient to show that for any l, $0 \leq l \leq t-1$,

$$\frac{\sum_{j=1}^{t} u_i(x_j)}{t} \leq \frac{\sum_{j=1}^{t-l} u_i(x_j) + u_i(y)}{t-l+1}. \tag{A.10}$$

Define

$$\Sigma_1 = \sum_{j=1}^{t-l} u_i(x_j), \Sigma_2 = \sum_{j=t-l+1}^{t} u_i(x_j), \Sigma = \sum_{j=1}^{t} u_i(x_j) = \Sigma_1 + \Sigma_2.$$

The inequality (A.10) is equivalent to the inequality

$$u_i(y) \ \geq \ \frac{\Sigma \cdot (t-l+1)}{t} - \Sigma_1 = \frac{\Sigma \cdot (t-l+1) - \Sigma_1 \cdot t}{t} = \frac{\Sigma_2 \cdot (t-l+1) - \Sigma_1 \cdot (l-1)}{t}. \quad \text{(A.11)}$$

We estimate the righthand part of inequality (A.11):

$$\frac{\Sigma_2 \cdot (t-l+1) - \Sigma_1 \cdot (l-1)}{t} \leq$$

$$\leq \frac{|\Sigma_2| \cdot (t-l+1) + |\Sigma_1| \cdot (l-1)}{t} \leq \frac{(t-l+1)\sum_{x \in X \setminus \{y\}} |u_i(x)| \ + \ (l-1)\sum_{x \in X \setminus \{y\}} |u_i(x)|}{t} =$$

$$= \sum_{x \in X \setminus \{y\}} |u_i(x)| \ \leq \ u_i(y),$$

where the latter inequality holds according to the conditions of the theorem. So inequality (A.11) holds and so does (A.10).

Now we have to show that there exists some profile B_{-i} that

$$u_i(W(\widetilde{B_i}, B_{-i})) > u_i(W(B_i, B_{-i})).$$

Let profile B_{-i} be such that exactly $\lceil 0.1 \cdot \#I \rceil - v - 1$ voters vote with $y \in H_1$ and $z \in H_0$ for any $z \in X \setminus \{y\}$. All the remaining voters vote $z \in H_0$ for any $z \in X$. Then for $B = (B_i, B_{-i})$:

$$s(y, B) = \lceil 0.1 \cdot \#I \rceil - v - 1 + v = \lceil 0.1 \cdot \#I \rceil - 1;$$

$$s(z, B) = 0, for \ any \ z \ \in \ X \setminus \{y\},$$

so $W(B) = \varnothing$ and $u_i(W(B)) = 0$.

At the same time, for profile $\tilde{B} = (\widetilde{B_i}, B_{-i})$:

$$s(y, \tilde{B}) = \lceil 0.1 \cdot \#I \rceil - v - 1 + n \ \geq \ \lceil 0.1 \cdot \#I \rceil \ because \ n \ \geq \ v - 1;$$

$$s(z, B) = 0, for \ any \ z \ \in \ X \setminus \{y\},$$

so $W(\tilde{B}) = \{y\}$ and $u_i(W(\tilde{B})) = u_i(y)$.

Then under profile B_{-i}, $u_i(W(\widetilde{B_i}, B_{-i})) > u_i(W(B_i, B_{-i}))$. Part (ii) is proved.

(iii) Consider a ballot $B_i = \{H_i^{(-n)}, ..., H_i^{(n)}\}$ for which $y \notin H_i^{(-n)}$. It means that for the ballot B_i, $y \in H_i^{(v)}$ for some $-(n-1) \leq v \leq n$. Let $\widetilde{B_i}$ be the ballot for which $\tilde{H}_i^{(v)} = H_i^{(v)} \setminus \{y\}$, $\tilde{H}_i^{(-n)} = H_i^{(-n)} \cup \{y\}$ and $\tilde{H}_i^{(j)} = H_i^{(j)}$ for all $j \in L \setminus \{v, -n\}$. We will prove that $\widetilde{B_i}$ dominates B_i.

Given any B_{-i}, all candidates except y have the same score at (B_i, B_{-i}) and $(\widetilde{B_i}, B_{-i})$ while the score of y is decreased by $n + v$ units at the latter ballot profile. Therefore, regarding the sets of winning projects $Y = W(B_i, B_{-i})$ and $\tilde{Y} = W(\widetilde{B_i}, B_{-i})$, the following three cases are exhaustive:

1. $y \notin Y, y \notin \tilde{Y}$;
2. $y \in Y, y \in \widetilde{Y}$;
3. $y \in Y, y \notin \tilde{Y}$.

In the first two cases $Y = \tilde{Y}$ so $u_i(Y) = u_i(\tilde{Y})$.

In the third case, let $\tilde{Y} = \{x_1, ..., x_t\}$, then $Y = \{x_1, ..., y, ..., x_{t-l}\}$ for some $0 < l \leq t-1$. Show that in this case $u_i(\tilde{Y}) \geq u_i(Y)$. It's enough to show that for any l, $0 < l \leq t-1$,

$$\frac{\sum_{j=1}^{t} u_i(x_j)}{t} \geq \frac{\sum_{j=1}^{t-l} u_i(x_j) + u_i(y)}{t-l+1}. \tag{A.12}$$

Define

$$\Sigma_1 = \sum_{j=1}^{t-l} u_i(x_j), \Sigma_2 = \sum_{j=t-l+1}^{t} u_i(x_j), \Sigma = \sum_{j=1}^{t} u_i(x_j) = \Sigma_1 + \Sigma_2.$$

The inequality (A.12) is equivalent to inequality

$$u_i(y) \geq \frac{\Sigma \cdot (t-l+1)}{t} - \Sigma_1 = \frac{\Sigma \cdot (t-l+1) - \Sigma_1 \cdot t}{t} = \frac{\Sigma_2 \cdot (t-l+1) - \Sigma_1 \cdot (l-1)}{t}. \tag{A.13}$$

Estimate the right part of inequality (A.13):

$$\frac{\Sigma_2 \cdot (t-l+1) - \Sigma_1 \cdot (l-1)}{t} \geq$$

$$\geq \frac{-|\Sigma_2| \cdot (t-l+1) - |\Sigma_1| \cdot (l-1)}{t} \geq \frac{-(t-l+1)\sum_{x \in X \setminus \{y\}} |u_i(x)| - (l-1)\sum_{x \in X \setminus \{y\}} |u_i(x)|}{t} =$$

$$= - \sum_{x \in X \setminus \{y\}} |u_i(x)| \geq u_i(y),$$

where the last inequality holds according to the conditions of the theorem.

So inequalities (A.12), (A.13) hold.

Now we have to show that there exists some profile B_{-i} that

$$u_i(W(\widetilde{B_i}, B_{-i})) > u_i(W(B_i, B_{-i})).$$

Let profile B_{-i} be such that exactly

$\lceil 0.1 \cdot \#I \rceil + n - 1$ voters vote with $y \in H_1$ and $z \in H_0$ for any $z \in X \setminus \{y\}$. All the remaining voters vote $z \in H_0$ for any $z \in X$.

It is easy to see that in this case $W(\tilde{B}) = \varnothing$ and $u_i(W(\tilde{B})) = 0$.

At the same time, $W(B) = \{y\}$ and $u_i(W(B)) = u_i(y)$, so

$$u_i(W(\widetilde{B_i}, B_{-i})) > u_i(W(B_i, B_{-i}))$$

and part (iii) is proved. □

Proposition A.5. *Let u_i be not null. Let for some $x, y \in X$ with $u_i(x) > u_i(y)$ the voter i vote sincerely with the ballot B_i such that $s_i(x) > s_i(y)$. Then:*

1) if $u_i(x) > 0$, then there exists a profile B_{-i} such that for any ballot $\widetilde{B_i}$ with $s'_i(x) < s'_i(y)$:

$$u_i(W(B_i, B_{-i})) > u_i(W(\widetilde{B_i}, B_{-i})); \tag{A.14}$$

2) if $u_i(x) \leq 0$, then there exists a profile B_{-i} such that for any ballot $\widetilde{B_i}$ with $s'_i(x) < s'_i(y)$:

$$u_i(W(B_i, B_{-i})) \geq u_i(W(\widetilde{B_i}, B_{-i})), \tag{A.15}$$

and for some $\widehat{B_i}$ with $s'_i(x) < s'_i(y)$:

$$u_i(W(B_i, B_{-i})) > u_i(W(\widehat{B_i}, B_{-i})). \tag{A.16}$$

Proof. Let in the ballot B_i: $s_i(x) = a$, $s_i(y) = b$ $(a > b)$.

1. Consider the case when $u_i(x) > 0$.

   Let profile B_{-i} be such that exactly $\lceil 0.1 \cdot \#I \rceil - a$ voters vote with $x, y \in H_1$ and $z \in H_0$ for any $z \in X \backslash \{x, y\}$. All the remaining voters vote $z \in H_0$ for any $z \in X$.

   Then for $B = (B_i, B_{-i})$:

   $$s(x, B) = \lceil 0.1 \cdot \#I \rceil - a + a = \lceil 0.1 \cdot \#I \rceil;$$

   $$s(y, B) = \lceil 0.1 \cdot \#I \rceil - a + b < \lceil 0.1 \cdot \#I \rceil,$$

   so

   $$W(B) = \{x\}$$

   and

   $$u_i(W(B)) = u_i(x).$$

   Let now $\widetilde{B_i}$ be any other ballot with $s'_i(x) < s'_i(y)$. Let $s'_i(x) = c$, $s'_i(y) = d$. Define $\tilde{B} = (\widetilde{B_i}, B_{-i})$.

   Then:

   $$W(\tilde{B}) = \{\{x, y\},\ if\ c \geq a;\ \{y\},\ if\ c < a\ and\ d \geq a;\ \{\varnothing\},\ if\ d < a\}. \tag{A.17}$$

   In all cases from (A.17), $u_i(W(\tilde{B})) < u_i(W(B))$, because of conditions $u_i(x) > 0$ and $u_i(x) > u_i(y)$ (we set $u_i(\varnothing) = 0$), and inequality (A.14) is proven.

2. Consider now the case when $u_i(x) \leq 0$.

   Let profile B_{-i} be such that exactly $\lceil 0.1 \cdot \#I \rceil - a - 1$ voters vote with $x, y \in H_1$ and $z \in H_0$ for any $z \in X \backslash \{x, y\}$. All the rest voters vote $z \in H_0$ for any $z \in X$.

   Then for $B = (B_i, B_{-i})$:

   $$s(x, B) = \lceil 0.1 \cdot \#I \rceil - a - 1 + a < \lceil 0.1 \cdot \#I \rceil;$$

   $$s(y, B) = \lceil 0.1 \cdot \#I \rceil - a - 1 + b < \lceil 0.1 \cdot \#I \rceil,$$

   so

   $$W(B) = \varnothing$$

and

$$u_i(W(B)) = 0.$$

Let now $\widetilde{B_i}$ be any other ballot with $s'_i(x) < s'_i(y)$ and $s'_i(x) = c$, $s'_i(y) = d$. Then for profile $\tilde{B} = (\widetilde{B_i}, B_{-i})$ there exist the following options:

$$W(\tilde{B}) = \{\{x, y\},\ if\ c > a;\ \{y\},\ if\ c \leq a\ and\ d > a;\ \{\varnothing\},\ if\ d \leq a\}. \tag{A.18}$$

In the first two cases from (A.18), $u_i(W(\tilde{B})) < u_i(W(B))$, and in the third case $u_i(W(\tilde{B})) = u_i(W(B))$. The inequalities (A.15), (A.16) are proven. □

A.4 The necessity of secret ballots

Now let us see how significant is to keep the results of the voting secret till the end of voting procedure. Below we describe the situation when the voter i, who votes the last, can fully change the set of winners, especially knowing the profile B_{-i} with all other votes. One can construct a lot of more complicated examples that involve the same result.

In what follows for some $x \in X$ we will denote $s(x, B_{-i})$ the score of project x under the profile B_{-i} and $W(B_{-i})$ the set of winners under profile B_{-i}, i.e. the set of projects that won according to the voting procedure A.3 when their scores are $s(x, B_{-i})$.

Proposition A.6. *There exists such profile B_{-i}, such ballot B_i and such budgets distribution $m = \{m(x_i),\ i \in I\}$, that*

$$W(B_{-i}) \cap W(B_i, B_{-i}) = \varnothing$$

(i.e. a voter i, voting with B_i, can completely change the set of winners).

Proof. To prove the statement it's sufficient give the relevant example. Let $l = 1$ and under profile B_{-i} the sequences of projects (in descending order by score) be

$$x_1, ..., x_n$$

and the budgets distribution be:

$$m(x_1) = 0,5 \cdot M, \quad m(x_2) = M, \quad m(x_3) + ... + m(x_k) = 0,5 \cdot M\ for\ some\ k \leq n.$$

Then under the assumption that all candidates $x_1, ..., x_k$ are acceptable and under the profile B_{-i} the set of winners is

$$W_1 = W(B_{-i}) = \{x_1, x_3, ..., x_k\}.$$

Assume that the ballot B_i is such that $H_i^{(1)} = \{x_2\}$, $H_i^{(-1)} = \{x_1\}$ and $H_i^{(0)} = X \backslash \{x_1, x_2\}$. Then $W_2 = W(B_i, B_{-i}) = \{x_2\}$.

These two sets of winners W_1 and W_2 are completely different that $W_1 \cap W_2 = \varnothing$ and the statement is proved. □

So even only one voter (which knows all other ballots) can cardinally change the set of winners. Of course it's much more easier to do this when she knows the rest of the voters' ballots.

A.5 Conclusions

We gave a formal description of the used voting system in ECTS as well as initial analysis of its properties. We call this system Fuzzy Threshold Voting. The main results are the following:

1. We presented mathematical model for the FTV voting system. The main difference of this system from the Yes-No-Abstain voting system is that the voting procedure depends not only on the score of candidates but also on some other factor. In our case this factor is an amount of money requested by each proposal.

2. We demonstrated that the presented FTV system satisfies some of the most important properties for the voting systems - the Condorcet criterion and monotone rule.

3. We proved several statements regarding possible voting strategies.

4. We analyzed the importance of secret voting and demonstrated that even a single voter can completely change the set of winners if she knows voting ballots of other voters.

A.6 References

[1] E. Duffield, D. Diaz. Dash: A Privacy Centric Crypto Currency. Whitepaper. https://www.dash.org/wp-content/uploads/2015/04/Dash-WhitepaperV1.pdf

[2] A. Gibbard. Manipulation of Voting Schemes: A General Result. http://courses.math.tufts.edu/math19/duchin/gibbard.pdf

[3] Brams, Fisburn. Going from Theory to Practice: The Mixed Success of Approval Voting. http://www.nyu.edu/gsas/dept/politics/faculty/brams/theory_to_practice.pdf

[4] J. Laslier. Strategic Voting in Multi-Winner Elections with Approval Balloting: A Theory for Large Electorates. http://www.barcelona-ipeg.eu/wp-content/uploads/2015/09/rationalcommitteeapproval_v2_2016_04_13.pdf

[5] Niemi, R. (1984). The Problem of Strategic Behavior under Approval Voting. The American Political Science Review, 78(4), 952-958. doi:10.2307/1955800

[6] Ottewell, Guy (2004) [1987]. "Arithmetic of Voting". Universal Workshop. Retrieved 2010-05-08. http://www.universalworkshop.com/ARVOfull.htm

[7] Remzi Sanver, Jean-FranÂ¸cois Laslier. The basic approval voting game. https://halshs.archives-ouvertes.fr/hal-00445835/document

[8] D. Baumeister et. al. Computational Aspects of Approval Voting. http://ftp.cs.rochester.edu/pub/papers/theory/09.tr944.Computational_Aspects_of_ - Approval_Voting.pdf

[9] E. Erdmann. Strengths and Drawbacks of Voting Methods for Political Elections. http://www.d.umn.edu/math/Technical%20Reports/Technical%20Reports%202007-/TR%202011/TR_2011_4.pdf

[10] S. Park, R. Rivest. Towards Secure Quadratic Voting. https://eprint.iacr.org/2016/400.pdf

[11] F. Maniquet, P. Mongin. Approval Voting and Arrow's Impossibility Theorem. http://www.isid.ac.in/~pu/seminar/30_11_2011_Paper.pdf

[12] Jean-FranÃ§ois Laslier, M. Remzi Sanver (2010) The Basic Approval Voting Game. Palaiseau, France

[13] Brams, SJ and PC. Fishburn (1983) Approval Voting. Boston: Birkhäuser.

[14] Brams, S. J., & Kilgour, D. M. (2014). Satisfaction approval voting. In Voting Power and Procedures (pp. 323-346). Springer International Publishing.

[15] https://arxiv.org/ftp/arxiv/papers/1602/1602.05248.pdf