Ternary Funding and Joint Tokens: A Trustless Approach to Public Goods Funding

Ajesiroo
ajesiroo@protonmail.com

July 14, 2021
Updated:
January 9, 2023

Abstract

A primary drawback of existing mechanisms for public goods funding is the degree of uncertainty—grants carry substantial risk for backers, while post factum rewards are subject to arbitrary disbursement. We propose the concept of ternary funding, a trustless mechanism that is contingent on pre-defined deliverables. In its simpler form, stablecoins are locked until attestation of an outcome by validators. In its more elaborate form, a non-fungible token (NFT) is procedurally generated by mapping the addresses of validators. Successful attestation transforms the latter into an implicit mapping of endorsements as its phenotype is dependent on the aforementioned addresses.

Contents

1 Introduction

The transfer of trust from off-chain decisions to on-chain mechanisms is a particularly difficult problem. Where subjectivity complicates attestation, the quandary becomes further apparent. Simply distributing the former process over many independent participants doesn't address it, even when assuming Sybil resistance and incentives to minimise collusion. In this scenario, trust remains on non-robust collective decision making rather than a reliable, on-chain system, and the advantage of a distributed ledger is questionable.

Even though absolute trustlessness whenever on-chain and off-chain components interface is impossible, trust in validators, as a practical concern, can nevertheless be minimised to the point where its significance is greatly diminished. Collusion is a primary consideration and can be largely mitigated by limiting attestation to a random subset and obfuscating individual attestations via zero-knowledge proofs. Of equal importance when designing decentralised oracles around trustlessness, however, is the nothing-at-stake problem. The latter can be addressed by implementing slashing conditions. Crucially, this is not an exhaustive description of the attack surface, but rather, a broad look at its two largest components [1][2][3].

For decentralised oracles that comprise subjectivity, the task of shifting trust to the mechanism itself is undoubtedly a complex one, but nevertheless represents significant upsides; where purely off-chain equivalents require the counterparty to trust collective determinations, trustlessness provides the counterparty with reliability, and in turn, a greater incentive to participate at the outset.

This paper makes several contributions. The core primitive is a decentralised oracle for inherently subjective outcomes, which is in turn harnessed to enable trustless mechanisms contingent on pre-defined criteria.[1] In the context of public goods, two such mechanisms are proposed; the first consists of locking stablecoins until attestation by the former. The second involves the use of validator addresses as seed inputs to procedurally generate an NFT that's subject to a similar attestation process. The mapping is contained within the token itself and its release to the counterparty reflects the cumulative endorsements of the validator pool. For both modes, an overarching DAO is not needed to manage successive rounds. A stigmergic approach can be utilised that enables the viability of each round to be contingent on the necessary threshold of validators being reached.

[1] In an extended version of this paper, the core primitive applied to a theoretical distributed ledger is described in Section 3.

1.1 Not-For-Profit

All mechanisms described herein are not-for-profit. There is no native token and the atypical staking pool described in Section 2.3 has no direct reward for validators.[2] In order to eliminate the volatility of Ether, a decentralised stablecoin is used for both the pool outlined in Section 2.2 and Section 2.3.

1.2 Public Goods

The term *public good* in the context of this paper refers to goods that are non-excludable, non-rivalrous and include the consideration of externalities.[3] The example provided in Section 1.3 is a public good in the form of open data.

1.3 Primary Example

The primary example of a public good that is referenced within this paper is the findings of a team of clinical researchers conducting a small study on the efficacy of a novel immunotherapy. This is used only for illustrative purposes, and the mechanisms described in Section 2 can be applied to any public good that meets the criteria in Section 1.2. The aforementioned example was chosen as it is small in scale and describes research for an underserved cohort that receives insufficient funding under traditional models.

[2]The social incentives for participating in such a staking pool are described in later subsections.

[3]See `https://web.archive.org/web/20210702173945/https://otherinter.net/research/positive-sum-worlds/` for discussion on externalities in the context of public goods.

2 Trustless Public Goods Funding

There is a current need to fund public goods that provide benefit to underserved communities such as those described in Section 1.3.

2.1 Ternary Funding

We introduce a funding mechanism consisting of three primary stages:

- Population of a staking pool, of which, a minimum number of validators are needed to progress to subsequent stages
- Population of a donation pool, or as described in Section 2.4, the generation of a joint token
- Attestation of the outcome by a random selection of validators, with release of the donation pool or joint token upon delivery of the public good

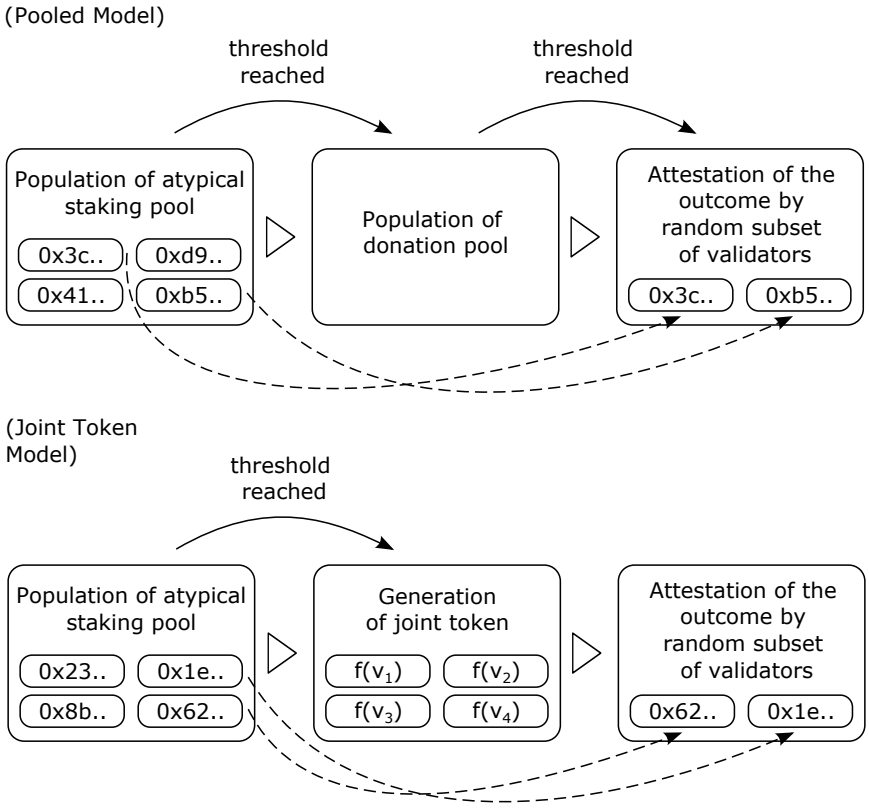

Figure 1

The cycle is renewed for each funding round and a DAO is not needed to manage subsequent rounds or changes to parameters (see Section 2.3.7).[4]

Critically, the staking pool must be populated before the donation pool. This is for several reasons:

- While validators that are randomly chosen to attest to the outcome are pseudonymous, all validators are initially identified (see Section 2.3). This, together with a minimum threshold of validators being reached, provides a degree of legitimacy to the funding round, which is the mechanism that incentivises contributions to the donation pool.
- The threshold of validators being reached is an implicit agreement of the parameters of the funding round, most notably, acceptance of the party developing the public good, and the specific criteria of what constitutes the materialisation of the public good (see Section 2.3.6).

As attestation by the staking pool is the mechanism that either releases or reverses the funds in the donation pool, the amount in the donation pool is capped in order to be within proportion to the staking pool. This is to prevent an attack where the cost of bribing validators, or validators otherwise colluding, is lower than the potential payout of the donation pool (see Section 2.3.2). As also explained in Section 2.3.2, the penalty for attesting outside of the three-fourths majority is the slashing of the entire stake.

Finally, while the contracts are ideally deployed to layer 1, gas fees are often prohibitively expensive, and EVM-compatible rollups must also be considered. At present, the only EVM-compatible layer 2 option that is *deployment-ready* (and provides the full security of the Mainnet) are optimistic rollups.[5]

2.2 Donation Pool

Once the threshold of validators in the staking pool described in Section 2.3 is reached, contributors can populate this pool up until the cap outlined in Section 2.1. It is important to note that contributors are able to participate in both pools.[6]

[4]The term "ternary" is reflective of the three stages associated with each round. It can alternatively be referred to as "sequent funding" given the similarity of the stages to sequent logic.

[5]At the time of publication, this includes Arbitrum and Optimism. EVM-compatible ZK-rollups are expected to be deployment-ready soon. While zkSync 2.0 has launched on the Mainnet, it is currently in a limited alpha phase.

[6]The pools are not merged as some may want to contribute without the additional burden of attesting.

2.3 Atypical Staking Pool

This is a novel decentralised oracle that determines whether the stablecoins locked in the donation pool, or the newly minted joint token described in Section 2.4, should be released or returned. In the context of a joint token, the latter would amount to it being burned.

Unlike other staking pools, there is no cryptoeconomic incentive to become a validator. The staked tokens are never converted into interest-bearing tokens[7] and the validator does not receive a direct reward once attestation is undertaken. On the other hand, there are costs involved to become a validator, thus the staking pool is itself a donation pool in the vein described in Section 2.2.

The staking pool is, in effect, donating the following:

- The opportunity cost represented by the risk-free rate (R_f)
- A small but theoretical risk that an honest validator's stake is slashed if it does not attest in line with the three-fourths majority
- The time taken to attest
- The gas fees involved in interacting with the staking contract

It is important to articulate the motivations for participating in such a pool. As described in the overview of Section 2.1, the participants of this pool are not anonymous. The reason for this is threefold. To minimise the risk of various attacks described in later subsections, to allow the donation pool to gauge the quality of the funding round, and most pertinently, to provide a social incentive for validators to participate. Validators can publicly post the transaction hash once they have staked the required amount. Thus the incentive is similar to other not-for-profit contributions. The main difference with a simple donation, however, is that an honest validator can expect to receive their staked tokens back,[8] minus R_f and gas fees.

2.3.1 Sybil Resistance

The inherent properties of the staking pool mitigates this attack. Validators are not anonymous (see Section 2.1) and only during attestation is a random selection of validators pseudonymous (see Section 2.3.5).

[7] The version the author is working to implement will not transfer tokens to a lending platform due to the increased counterparty and smart contract risk. Other implementations could use a third-party lending platform so long as all interest earned is provided to the party that delivered the public good (confirmed by attestation) rather than validators.

[8] Provided they attest in line with the three-fourths majority.

2.3.2 Collusion

This is a more complex problem in comparison to Sybil attacks [4, 5]. Nevertheless, this is unlikely to be successful given the inherent design of the various pools. As explained in the overview of Section 2.1, the total amount locked in the donation pool is capped in order to be kept in proportion with the staking pool. In the case of bribing validators, a bad actor would need to ensure that the total expenditure is less than the potential payout of the donation pool. As the random selection of validators that are chosen to attest does not occur until the final stage, and their addresses are obfuscated via zk-SNARKs (see Section 2.3.5), the bad actor would have to bribe the vast majority of validators in order to confidently meet the three-fourths needed to attest to a malicious outcome. If the bad actor was not able to persuade enough validators, the entire stake of all compromised validators is lost during attestation, making collusion prohibitively expensive.

Even in the scenario where the originator of a funding round (who can remain anonymous) is malicious and is associated with all validators in the staking pool, the donation pool would most likely not reach the minimum threshold required to progress to the final stage. As outlined in Section 2.1, the staking pool is populated before the donation pool precisely to allow participants in the latter to assess the quality of validators and the fund originator. An anonymous fund originator, in contrast to a known entity (see example provided in Section 1.3), is unlikely to receive enough funding to allow progression to the final stage. When the minimum threshold in either of the two pools is not reached, all funds are reversed. This also addresses validators simply colluding out of some common interest. In this circumstance, the donation pool can be expected to not receive the minimum threshold if most participants in the staking pool are not trusted.

In the case of a newly minted non-fungible joint token in lieu of a donation pool (described in Section 2.4), the joint token would have minimal value but still incur the costs outlined in Section 2.3.

2.3.3 Interval Before Attestation

There is a time-based interval between the delivery of the public good and attestation of the outcome. This is to provide enough time for validators to decide whether the criteria set at the beginning of the funding round have been met. For a typical funding round, this might range from weeks to months, depending on the nature of the funding. For the example outlined in Section 1.3, the interval should be roughly equivalent to the time required for peer review. It is important to note that the actual subset of validators that are randomly chosen to attest are not selected at the beginning of this time-period, but at the end.

2.3.4 Cycling of Validators

A random set of validators is selected to attest to the outcome. This occurs after the interval described in Section 2.3.3. As an anti-collusion measure, the window of time the validator has to attest is short, and if they are not available, a different validator is selected until enough have attested. If the required number of validators do not attest, all funds in the staking pool and donation pool are reverted, and in the case of a joint token, the token is burned.[9]

2.3.5 Attestation

Once a random set of validators is chosen,[10] a key is generated and each attestor submits a transaction containing the zk-SNARK of this key together with their attestation.[11] The zk-SNARK is needed to ensure only validators that are chosen to attest can call the necessary function. Once enough verified attestations occur, the stablecoins secured by the staking contract are either released or reverted. In the context of a joint token in lieu of a donation pool, the token is either released or burned.

The decision can be shown as:

$$\sum_{i=1}^{v} attestation_i \tag{1}$$

where v is the total amount of validators chosen to attest.

It is also theoretically possible to remove the need for an intermediary attestation contract which verifies the zk-SNARK of the key through the use of *private* zk-rollups[12]—that is, zk-rollups that use recursive proofs [6]. In this case, only a private transaction would need to be sent to the staking contract, and a proof of the rollup verifies the proof of the underlying transaction.

For a given zk-SNARK (G, P, V), where G is the generator algorithm that outputs the proving key G_p and verification key G_v, the output of recursive G is (G'_p, G'_v).

2.3.6 Subjectivity of the Outcome

Validators should be encouraged to only stake in pools where the funding originator has provided a short and highly specific criteria on what constitutes the delivery of the public good. This is to make it as easy as possible

[9] The funding round is considered cancelled, and all funds are returned to their originating addresses.

[10] For example, 16 out of a total pool of 64 validators.

[11] A zk-SNARK for each voting outcome (0,1) should also be considered, although this will use more gas.

[12] An example of a project currently working on this problem is Aztec.

to reach the three-fourths consensus during attestation. It is important to reiterate from Section 2.1 that all funds in the staking pool are reversed if the required threshold of validators is not reached to progress to the next stage, therefore it would require the minimum threshold of validators to make this oversight for the funding round to progress.

There will nevertheless be an inherent degree of subjectivity given the nature of the decentralised oracle. Those that attested in line with the three-fourths majority will always receive their stake back, regardless of the decision on the outcome. This also applies for circumstances where no three-fourths majority is reached.

2.3.7 Stigmergic DAO Paradigm

While a permanent DAO can be used to manage the parameters of each funding round, it is not necessary. A new lightweight contract can be deployed for each successive funding round and an interface can be defined that links to a common set of contracts in order to save on the gas fees of deployment. In the context of a joint token, the ERC-721 standard only requires that each address and `tokenId` form a unique pair, therefore it does not matter if continuity in `tokenId` is absent so long as a different contract address is used. The advantage of this stigmergic approach is that a governance token can be omitted, and acceptance of the new parameters is contingent on whether or not the new contract receives the necessary threshold of validators.[13] It also negates the ability of a party to spam trivial funding rounds.

2.3.8 Stablecoin

A decentralised stablecoin is used for both the staking pool and the donation pool. The primary requirements are that it is sufficiently decentralised and secure. At present, only several candidates meet these criteria, with Dai being the most notable.

2.3.9 Multiple Known Accounts

The preferred way to establish identity for the purpose of the staking pool is to adopt a proof of personhood system.[14] Given the infancy of major projects that address this need however, a temporary workaround involves validators posting the transaction hash of their stake from known accounts on multiple unrelated platforms. While this offers a degree of decentralisation, it can nevertheless result in less security than a proof of personhood system.

[13] This also addresses the funding allocation problem.

[14] Proof of Humanity and BrightID are two feasible options.

2.4 Joint Token

The donation pool described in Section 2.2 can be replaced entirely with a novel type of non-fungible token introduced herein as a joint token (JT). The token is generated pseudorandomly[15] based on the addresses of all validators, and can only be minted once the staking pool is populated.[16] Validators then decide whether to release or burn the token, in the way described in the overview of Section 2.3.

While an algorithm for generating the token is detailed in Section 2.4.4, it can be summarised as a mapping of each address to an entry, which then collectively form an n^2 grid representing all validators. In a primitive expression, each address can correspond to a colour.

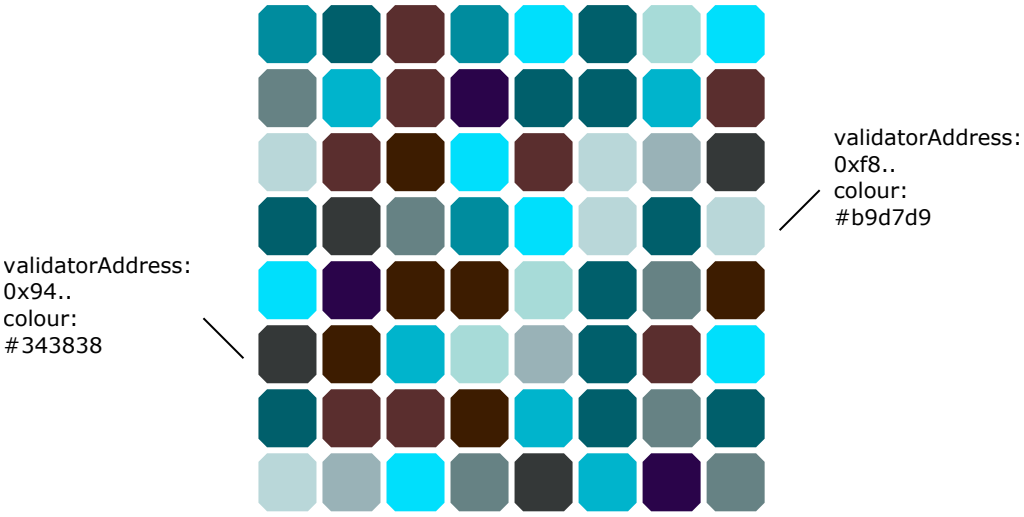

Figure 2

Crucially, all data relating to the token is stored on-chain. It primarily consists of an array of structs containing each address and corresponding payload. The figure shown above is simply the parsing of this data structure off-chain. The token conforms to the ERC-721 standard, and there is *no* image URI contained in the metadata schema, as only an array of structs containing each pair is necessary to develop a visual representation off-chain.

More complex mappings beyond colours can be used, with gas fees representing the primary bottleneck. For instance, each address can be mapped to an array of integers that define a markup element, culminating in a far more elaborate phenotype.

As only a data structure is stored within the token, a byproduct is composability. Its ultimate form when reconstructed in the browser is dependent

[15] To prevent coordination of its ultimate appearance.

[16] A hybrid funding round containing both a donation pool and joint token is technically possible.

on the way it is parsed and augmented. Zone-based interactivity can be enabled by abstracting away some values in tooltips, including the number of times a contributor has participated in previous rounds.[17]

2.4.1 Value

In the event of a successful funding round,[18] the token can be expected to have a degree of value:

- As its release to the counterparty is contingent on attestation of the public good, the data structure found within it is an implicit mapping of endorsements by validators
- In its visually parsed form, it's possible to ascertain the location of each validator address in the n^2 grid[19]
- Given that there is only one token associated with each round, and that the process through to attestation is designed to be non-trivial rather than efficient, dilution of endorsements is prevented by avoiding excessive minting
- Expenditures outlined in Section 2.3 are also encapsulated as a result of the lack of cryptoeconomic incentives to participate

Ultimately, the released token represents delivery of the public good, endorsement by a set of validators and a series of expenditures. The primary component that contributes to its value is the cumulative endorsements.

The recipient can opt to either keep it or trade it on any ERC-721 marketplace.[20]

2.4.2 JT-Specific Incentives

Similar to the staking pool described in Section 2.3, the incentive to generate a joint token is social rather than cryptoeconomic. The validator can publicly post their transaction hash, in the same way an entity can publicly announce any other not-for-profit contribution. As the staking pool can be capped, an additional incentive is scarcity—if the grid is a mapping of 64 validators,[21] then a given address can only be associated with the token if the contribution is made before the cap is reached.

[17] In this example, data on previous contributions is derived off-chain.

[18] Where the token is released to the party that delivered the public good.

[19] One way to show each validator address is via tooltips.

[20] The recipient is always the party that delivered the public good.

[21] In practice, the value of this parameter should be highly dependent on the funding round.

2.4.3 Default Funding Vehicle

Given the reduced risk due to the minimisation or absence of the donation pool described in Section 2.2, a joint token should be the default mechanism for most rounds. This can take a basic form, consisting of only a joint token, or a hybrid form, consisting of both the former and a reduced cap donation pool.

2.4.4 Procedural Generation

The following is a truncated algorithm in order to illustrate the key components needed to generate a primitive, colour-based joint token.

We first define a struct that will hold the address and colour pair.

```
struct Validator {
    address validatorAddress;
    string colour;
}
```

The string `colour` corresponds to the hexadecimal code that is ultimately selected, e.g `#008b46`. For efficiency purposes, this can be represented as a set of integers that form the RGB equivalent, i.e. `0, 139, 70`. In order to keep the truncated algorithm simple however, we will use a predefined palette that is represented as an array of strings.

We then initialise an integer representing the total number of validators, and an array of structs, validator addresses and a predefined palette.

```
uint validatorCount;

Validator[] coloursArray;
address[] validators; // [0x7a.., 0x6B.., 0xb77.., ..]
string[] palette; // ["#ece5ce", "#424242", "#78c0a8", ..]
```

We define a function that pseudorandomly selects a colour based on the validator address and the current block timestamp. Note that a third-party oracle is not needed as we do not require a more secure source of randomness for this specific application. A hash function that accepts the validator address and `block.timestamp` is sufficient.

```
function rand(address _validator) internal view returns(uint256) {
    uint256 seed = uint256(keccak256(abi.encodePacked
    (_validator, block.timestamp)));
return seed % validatorCount;
}
```

We populate the array of structs that contains each pair.

```
function setColoursArray() internal {
    for(uint i=0; i<validatorCount; i++){
        coloursArray.push(
            Validator({
                validatorAddress: validators[i],
                colour: palette[rand(validators[i])]
            })
        );
    }
}
```

We define a function that gets all pairs.

```
function getColoursArray() public view returns (Validator[] memory){
    return coloursArray;
}
```

Table 1: Output

Index	validatorAddress	colour
0	0x3a..	#78c0a8
1	0xc20..	#ece5ce
..	..	..

As each funding round is associated with a single token, `coloursArray` is the token itself, not an array of tokens. `coloursArray` needs to be transferred to an address, per the ERC-721 standard, and the `tokenId` is the index of the array containing all `coloursArray`'s.

Once the array is parsed off-chain, we can build the grid that can be presented in the browser.[22]

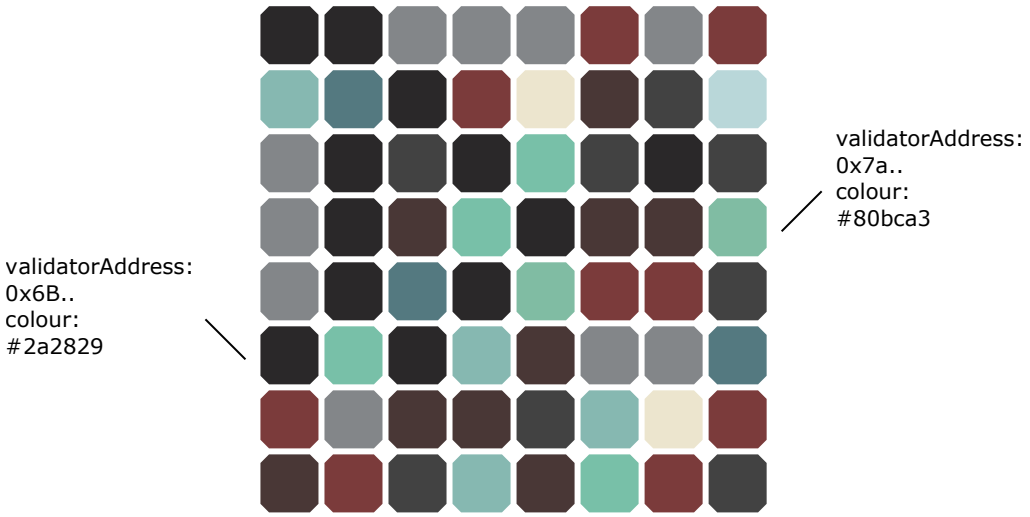

Figure 3

2.4.5 Colour Permutations

For simplicity, we assume a funding round of 64 validators v and a palette of 64 colours c. When colours are able to be reused in the same funding round, we observe $c^v = 3.94 \times 10^{115}$ permutations. In the case that colours can only be selected once, we observe $\frac{c!}{(c-v)!} = 1.27 \times 10^{89}$ permutations. With either approach, it is highly unlikely that any two parsed tokens will have the same presentation.

[22]ENS name can also be displayed if reverse resolution is enabled.

References

[1] H. Al-Breiki, M. H. U. Rehman, K. Salah and D. Svetinovic. 2020. *Trustworthy Blockchain Oracles: Review, Comparison, and Open Research Challenges*. In IEEE Access, vol. 8, pp. 85675-85685. DOI: https://doi.org/10.1109/ACCESS.2020.2992698

[2] Rebello, G.A.F., Camilo, G.F., Guimarães, L.C.B. et al. 2022. *A Security and Performance Analysis of Proof-based Consensus Protocols*. In Ann. Telecommun. 77, 517–537. DOI: https://doi.org/10.1007/s12243-021-00896-2

[3] Li, W., Andreina, S., Bohli, JM., Karame, G. 2017. *Securing Proof-of-Stake Blockchain Protocols*. In Garcia-Alfaro, J., Navarro-Arribas, G., Hartenstein, H., Herrera-Joancomartí, J. (eds) Data Privacy Management, Cryptocurrencies and Blockchain Technology. DPM CBT 2017. Lecture Notes in Computer Science, vol 10436. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-67816-0_17

[4] Vitalik Buterin. 2019. *On Collusion*. Retrieved Jun 17, 2021 from https://vitalik.ca/general/2019/04/03/collusion.html

[5] Siddarth Divya, Ivliev Sergey, Siri Santiago and Berman Paula. 2020. *Who Watches the Watchmen? A Review of Subjective Approaches for Sybil-Resistance in Proof of Personhood Protocols*. In Frontiers in Blockchain, Volume 3. DOI: https://doi.org/10.3389/fbloc.2020.590171

[6] Juan A. Garay and Rosario Gennaro. 2014. *Advances in Cryptology—CRYPTO 2014*. 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II. Germany, Springer Berlin Heidelberg.