Codg : An Adaptive Smart-Contract Blockchain

Abstract

Codg is a fairly-launched, actively-governed smart contract platform. It integrates advanced features like proof-of-stake validation, WebAssembly smart contracts, and an interface for on-chain identity and governance.

By distributing balances to other cryptocurrency holders who timelock their tokens, Codg secures an early community of stakeholders incentivized to participate in the network's governance.

Using a community-governed block reward and on-chain governance processes, stakeholders vote on upgrade proposals. This governance process is used to fund, manage, and deploy improvements to the network, creating a self-improving system that coordinates and supports its own development.

Contents

1 Introduction **1**

2 Technical Specification **1**

2.1 Modules 2

2.2 Contracts 3

2.3 Consensus 4

2.4 Staking 4

2.5 Balances and Accounts 4

2.6 `Codg` Token - `CODG` 5

2.7 Interoperability 5

3 Network Launch **5**

3.1 Lockdrop Contract Specifications 7

3.2 Lockdrop Participation 7

3.3 Genesis Block 9

4 Governance **9**

4.1 Governance Modules 10

4.1.1 Signaling 10

4.1.2 Identity 11

4.1.3 Democracy 13

4.1.4 Council 14

4.1.5 Treasury 14

4.2 Governance Interfaces 15

4.3 Governance Procedures 15

4.4 Experimentation Platform 16

5 Conclusion **16**

1 Introduction

Public blockchains have made it possible to create "unstoppable applications" that run in a shared computing environment, operate on open data, and cannot be halted by any central party.

Many exciting applications have been proposed to be built on this infrastructure, including forms of digital cash, financial instruments for lending and securitization, and marketplaces that minimize rent-seeking and maximize interoperability.

While their potential is exciting, blockchain applications are limited by a number of challenges. In particular, scaling blockchains to significant levels of throughput requires a large number of technical advances, including construction of efficient runtimes, rapid routing of blocks between nodes, moving computation off-chain through sidechains and state channels, and sharding across multiple chains.

Codg aims to solve the scalability problem by adopting a fundamentally different architecture. Codg tokenholders can vote to upgrade the network using on-chain voting, after which nodes automatically download a new version of the runtime. Critical decisions are made on-chain, creating a system with lower coordination overhead and a transparent process for deciding upon improvements.

2 Technical Specification

Codg is a smart contract blockchain that compiles to a client runtime, a blob of WebAssembly (Wasm) code that may be built and run natively or executed within a Wasm virtual machine. Either way, when an Codg native binary is compiled, it includes a Wasm virtual machine which can be used to execute later versions of the client runtime downloaded from the network.

The client runtime interfaces with networking code and other components provided by Parity Substrate. Substrate includes libp2p networking, PBFT consensus, and proof-of-stake block validation and finality [**substrate**]. Ultimately, the client is only responsible for downloading, executing, and validating blocks from the network.

A block is defined as follows:

- `Header := Parent + ExtrinsicsRoot + StorageRoot + Digest`
- `Block := Header + Extrinsics + Justifications`

At a glance, the block and header format resembles traditional blockchains; however, the notion of accounts and contracts are extended and abstracted into

an `extrinsic`. Roots within the header are stored in a Base-16 Modified Merkle Tree ("Trie") data structure.

Extrinsics define functions and data necessary to run a chain. They are analogous to transactions, except with a broader scope, so that any modification to the state of the chain is an extrinsic. This additional layer of abstraction allows individual transactions, batches of transactions, or other types of state changes to all be encoded as extrinsics.

Extrinsics are defined in modules, and any client may propose a valid extrinsic to be included in the next block.

2.1 Modules

Modules are written in Rust, compiled to Wasm, and linked as part of the client runtime. They define the core of Codg 's logic and have access to un -metered Turing-complete computation, so it is crucial that module code is carefully audited and tested. In particular, it is essential that no client can crash the chain by proposing an extrinsic that takes longer to execute than the blocktime. Otherwise, validators will be unable to call `author_block` in time to produce a valid block.

Codg includes these modules, this list is non exhaustive. Utility modules consumed by other modules as well as planned upgrades such as a `fees` module are not included:

Extrinsic	Description
`Balances`	Maintains balances for accounts.
`Consensus`	The set of authorities allowed to author blocks.
`Aura`	The set of authorities allowed to author blocks.
`Grandpa`	Ensures block finality in the face of misbehaving validators.
`Staking`	Handles staking required to become a validator.
`Session`	Rotates validators between sessions.
`Contract`	Allows the creation and execution of gas-metered smart contracts.
`Treasury`	On-chain treasury managed by governance.
`Signalling`	Allows tokenholders to use off-chain signalling to achieve informal consensus on proposals.
`Democracy`	Allows tokenholders to pass proposals through direct democracy.
`Council`	Allows a council elected by approval voting to pass proposals.
`Identity`	Simple primitives for on-chain identity.

2.2 Contracts

Codg smart contracts may be written in any Wasm-compatible higher level language. Already, there are well-tested toolchains for compiling C, C++, and Rust to Wasm, and independent work is in progress for compiling a subset of JavaScript to Wasm through the AssemblyScript project. Codg will also benefit from improvements in security, portability, and expressiveness as Wasm toolchains are further developed by other blockchain and web infrastructure projects.

Otherwise, contracts function much in the same way as they do on Ethereum or other EVM-compatible blockchains. Any user with an CODG balance may deploy a contract, and contracts can call other contracts as well. Gas, denominated in CODG , must be bought upfront and limits computation. Message calls use gas up to the transaction limit, with any excess refunded. If all gas is used, computation terminates, and effects are reverted.

Contracts will also be able to interact with other Codg modules through a limited set of whitelisted interfaces, in particular, to query accounts in the `Identity` and `Council` modules. Other improvements to the smart contract system, like parallelized transactions and state rent are expected to follow after the chain launches.

2.3 Consensus

Codg is launched with a nominated proof of stake Sybil-resistance mechanism. Rotating sets of validators are assigned to execute and seal blocks. Validators are initially chosen using a collective coin-flipping algorithm.

To avoid the nothing-at-stake problem, Codg uses the GRANDPA finality gadget to achieve block finality [**grandpa**]. On-chain governance gives Codg the ability to implement improved consensus mechanisms gracefully.

A five-second blocktime allows for sufficient network latency while still preserving high throughput.

2.4 Staking

Individual validators must bond their stake for three months. Codg maintains a validator set of **X** validators that retain their leadership position for the amount of time.

An Codg account may delegate their balance to other validators in a nominated proof-of-stake (NPoS) system to receive a portion of the block reward. In NPoS, block rewards are first split evenly between all active validators and then divided pro rata across the balances backing that validator. As a result, Codg accounts are incentivized to seek out trustworthy validators while avoiding excessive concentration.

2.5 Balances and Accounts

Balances are maintained in an account-based system, secured by ed25519 elliptic curve derived keys. Accounts and balance transfers must satisfy these constraints, maintained as global variables across the chain:

- `TotalIssuance`: The total amount of stake on the system.
- `ExistentialDeposit`: The minimum amount allowed to keep an account open.
- `TransferFee`: The fee required to make a transfer.
- `CreationFee`: The fee required to create an account. At least as big as ReclaimRebate.
- `FreeBalance`: The 'free' balance of a given account, used to determine the balance in the contract execution environment. When this balance falls below the value of 'ExistentialDeposits', then the 'current account' is deleted.
- `ReservedBalance`: The amount of the balance of a given account that is externally reserved, which can be slashed, but gets slashed last of all.

- `TransactionBaseFee`: The fee paid for making a transaction; the base portion.
- `TransactionByteFee`: The fee paid for making a transaction; the per-byte portion.

2.6 Codg Token - CODG

There are initially 5,000,000,000 (five billion) CODG tokens minted, divisible up to 18 decimal places. Initially, inflation is set to 158 CODG per block. This implies approximately 997,220,160 CODG in the first year, or just under 20%inflation.

The total amount of CODG minted will remain the same year after year, causing the percentage inflation to be disinflationary, with yearly infl ation falling to approximately 16.6% in the second year. Additionally, a system-wide vote may further increase or decrease inflation.

One of the first proposals on Codg will be to tie inflation to a specific participation or security rate, with the inflation floating up or down to reach a desired total stake. The specified proposal will be voted on and performed via a runtime upgrade modifying the inflation rate.

On Codg , CODG has multiple functions. CODG entitles holders both staking and voting rights, as well as the ability to delegate this right to another account. CODG
is also used to pay for state changes within the Wasm smart contract module.

2.7 Interoperability

Codg has a concrete roadmap towards cross-chain interoperability through Polkadot, a cross-chain bridging network that Parity is developing in tandem with Substrate [**polka**]. Polkadot will connect blockchains including Codg and Ethereum, allowing assets to be locked up on one network and managed on the other.

Over the longer term, users will be able to run Ethereum and Codg dapps alongside each other – perhaps using Ethereum as a stable network for high-stakes financial applications, and Codg as a progressive network for applications that require high throughput and first-class identity.

3 Network Launch

The Codg network will be launched with a 'lockdrop' of Codg tokens to Ether holders. A Lockdrop is an airdrop where token holders of one network lock

their tokens in a smart contract for a fixed or variable time period, the 'lock duration.' The purpose of this novel method of token distribution is to more fairly, securely and widely align incentives among the active participants[**lockdrop**], select for and incentivize committed initial participants of the network, and leverage existing token distributions in order to create relatively more diverse and wide distributions.

In the case of Codg , Ether holders will be able to participate through two interaction types: a 'Lock' where they send and make inaccessible their ETH tokens for an elected duration of 3, 6 or 12 months, or to 'Signal' their intent to participate in the Codg network.'Signaling,' is similar to an airdrop in that no lock duration is imposed and the participant's tokens remain accessible-however Signal participants receive a reduced allocation compared to a Lock interaction. For those that choose to Lock their tokens, longer lock duration corresponds to proportionally more Codg tokens received, and these are distributed at the launch of the network.

The Lockdrop process is a more fair and secure way to distribute tokens in a Proof-of-Stake network for the following reasons:

- **A non-zero opportunity cost**: To participate in the lockdrop, individuals are forgoing the opportunity cost of ETH for the duration of the lockdrop–e.g. being able to lend on Compound. For long-term ETH holders, this opportunity cost is irrelevant. For this reason, long-term ETH holders are most aligned with the CODG lockdrop. The minting of CODG tokens will allow token holders to participate in all the rights allocated to Codg participants–becoming a validator or voting on network proposals.
- **Downside protection**: At the end of the lockdrop, participants will have access to two productive utility tokens, ETH and CODG respectively. As Codg is a new and experimental chain, in the event of a malicious attack or exploit on the Codg network, lockdrop participants will still retain their ETH, eliminating the long-term risk of participation.
- **Easy, open and accessible process**: A single transaction that sends ETH to the Codg Master Lockdrop Contract allows one to receive CODG tokens. Any account can perform this from a hardware or software wallet (e.g., Trezor, Metamask, and more). Moreover, any ETH holder can participate.
- **Economic security**: Bootstrapping security on a new PoS chain. Ethereum is bootstrapping the Serenity release from an initial crowd-sale and PoW block reward. In the same way, Codg can use the already-wide distribution of Ethereum holders to bootstrap the economic security of the Codg chain.
- **Contract security**: When calling `lock` from the Master Lockdrop Contract, an individual Lockdrop User Contract is created, which actually holds the time-locked ETH of a participant. This significantly improves fund security, because value is not stored in one monolithic contract, but in an array of user-specific contracts.

3.1 Lockdrop Contract Specifications

The `lockdrop` contract is a simple two function smart contract. When the `function lock` is called with the interface specified below via the Master Contract Lockdrop, a new Lockdrop User Contract is created that holds the time-locked ETH.

Note, when `isValidator` is yes, the total allocated CODG may be staked at the time of distribution.

Notably, the Master Lockdrop Contract (MLC) itself does not hold the totality of locked ETH, instead, when calling `lock`, an individual Lockdrop User Contract (LUC) is created which holds the participant's time-locked ETH. This 2-step arrangement reduces the potential value and feasibility of a malicious attack. Further, the `Master Lockdrop` and `Lockdrop User` Contracts are extremely simple. In particular, the `LUC` contract is forty-five instructions until completing the call with no jumps. These have been audited by a third-party, Quantstamp. [**Quantstamp**]

Name	Data Type	Description
`term`	`Term`	An enum that specifies the lengths of time ETH may be locked.
Codg `Key`	`bytes`	The specified Codg address that, at genesis, will be allocated CODG to-kens.
`isValidator`	`bool`	Allows individuals to specify if this Codg address wants to be a validator at launch.

3.2 Lockdrop Participation

The Master Lockdrop Contract will accept "locks" and "signals" for a ninety-day Contribution Period. On Codg , lock interactions may have a duration of three months, six months, or 12 months with bonuses of 0%, 30%, and 120% respectively. In order to reduce centralization of power on Codg and increase the diversity of stakeholder voice, no single contributing ETH address or receiving CODG address will be able to obtain greater than or equal to 20% of the total CODG minted through the Lockdrop.

Length Lock	Method	Weight	CODG Received
`0 Months`	Signal	**0.20x**	25% at launch, 75% at 365 days after launch.
`3 Months`	Lock	**1.00x**	Upon network launch
`6 Months`	Lock	**1.30x**	Upon network launch
`12 Months`	Lock	**2.20x**	Upon network launch

Given the long duration of the Contribution Period, early participation for Lock

interactions exclusively will also be incentivized, in order to balance user interests in estimating in the outcomes of the lockdrop against the accessibility of participation. The Contribution Period is therefore divided into six 15-day bonus periods, starting June 1st 2019. When the next 15-day period begins, the early participation bonus decreases. This schedule is outlined below.

Date Range (2019)	**Bonus**	**ETH Cap**
`June 1 - June 15`	50%	**No Cap**
`June 16 - June 30`	35%	**No Cap**
`July 1 - July 15`	23%	**No Cap**
`July 16 - July 30`	14%	**No Cap**
`July 31 - August 14`	8%	**No Cap**
`August 15 - August 29`	5%	**No Cap**
`August 30 - August 31`	0%	**No Cap, End of Lockdrop**

Lock interactions will be accepted in parallel with Signal interactions, enabling participants to do both via the creation of multiple LUCs. In these Signal interactions, individuals will be able to receive CODG without locking ETH for any duration using a function call similar to the carbon vote. [**carbon**] Signals are open to both cold and hardware wallets. Note, if an individual participates by signaling, they will receive 25% of their CODG at network launch, and the remaining 75% of their CODG 365 days (twelve months) after the date of the network launch. Furthermore, for the ETH that is signaled from a single wallet that amount of CODG received at that time will be deducted by 80%.

The acceptance of signals supports the goal to involve long-term contributors to the Ethereum ecosystem in the launch of Codg . A significant fraction of tokens for these contributors remains on cold and hardware wallets, and an attempt to incentivize their participation must include non-locking interactions from both hot and cold wallets. In order to prevent Sybil-attacks or double Signal interaction attempts — where an individual may move their funds from one address to another to try and signal tokens twice, a snapshot of the main Ethereum chain is taken at the conclusion of the drop period, at 00:00 UTC of Sept 1 2019.

Further, there is a third type of interaction called a Generalized Lock, where a signal is awarded as a 3-month lock through the genesis specification. If an address that calls 'Signal' is unable to sign a transaction with `msg.value` greater than 0 for the entirety of the 12-month lock duration, that signal qualifies for Generalized Lock treatment. This category of interaction is designed to further involve committed participants who may be unable to participate normally but offer significant value to the Codg network as a whole.

During the duration that ETH tokens are locked,
individuals who receive CODG
at network launch will also be able to earn tokens by staking, and participating in other network activities. Finally, when the lock duration has ended for any user, they simply send a transaction of any kind to their LUC to trigger a return of their ETH.

We hypothesize that the proposed token distribution mechanisms should incentivize more active participation and widen distribution in a more fair manner than previous premines, initial coin offerings, or similar distribution schemes.

3.3 Genesis Block

After the launch, individuals will be able to individually validate the total amount of ETH locked or signaled by looking for both the `Locked` and the `Signaled Events` and calculating the corresponding calculating balances on Codg .

We expect to distribute 90% of the initial Codg tokens in the lockdrop.
A fair and broad distribution of tokens will be required to bootstrap governance and show that the consensus algorithm works under adversarial conditions. Codg is a project initially developed and governed by Commonwealth Labs. Other tokens will be distributed to Commonwealth Labs (4.5%), other partners assisting with the project such as Parity Technologies (3.0%), with an additional portion of tokens reserved for initial open-source contributors, community participants, and test-net users (2.5%), to be managed initially by Commonwealth Labs or through the network's Treasury module.

CODG holders or determinable but future CODG
holders may also vote to allocate
some additional issuance to other communities such as DOT holders.

4 Governance

A substantial majority (estimated 80%) of utility tokens and smart contract blockchains have indicated their intention to move to on-chain and decentralized governance. However, at the time of writing, there are few successful implementations of on-chain governance.

The few systems that have launched to date are primarily on-chain treasuries, delegated proof-of-stake (DPoS) systems, and direct token-weighted voting models. Directly token-weighted on-chain processes are generally easily captured by large economic holders in chains, such as block producers, miners, and exchanges [**pluto**]. Additionally, very few governance systems have any level of active use today. Communication, conversation, and voting are limited by the same unintuitive interfaces that hamper the growth of cryptocurrencies as a whole.

Codg aims to be a pioneer network in developing effective on-chain governance. By iterating upon product interfaces, voting systems, and other governance primitives, Codg can accelerate the implementation and deployment of core technologies like sharding, proof-of-stake, efficient SNARK implementations, and run-time changes, implementing technical advances at a faster pace than other blockchains. Codg network upgrades will be easier to coordinate and faster to deploy by using a clearly defined on-chain governance process.

4.1 Governance Modules

The core functionality of Codg 's governance is implemented in several extensible modules:

- **Signaling**: Non-binding polls are an important part of the governance process for existing blockchains. Users should also be able to create and distribute polls for signaling interest in different proposals, strategies, and directions.
- **Identity**: Codg requires a first-class identity primitive so people can identify each other in the governance process.
- **Democracy**: The democracy module initially restricts votes on proposals to binary votes. This module allows for delegated voting–improving the total stake allocated towards a vote. Future upgrades should allows users to choose between many different methods for voting on a proposal (e.g., binary or rank-choice).
- **Council**: Codg allows voters to nominate a set of accounts that hold certain rights over the network. Council members can bias proposals.
- **Treasury**: This module allows individuals to vote on the allocation of a portion of newly minted CODG .

In later iterations of Codg will include support for additional governance functionality. Development of additional features will be funded by the on-chain treasury, including:

- **Secure voting**: Users should be able to vote anonymously to prevent vote buying attacks, requiring the implementation of cryptographic primitives at the protocol layer.
- **DAOs**: As the Codg ecosystem grows, the need will arise for groups to coordinate outside of the network's initial governance structures. Over the long term, we expect this will happen through users building and managing DAOs, which allows for the exploration of alternative decision-making structures without alterations to core governance. For example, DAOs may follow decision-making processes including simple 1-person-1-vote, delegated voting, quadratic voting, and more complex models.

Note that further governance improvements may be implemented either at the `contract` or `module` level.

4.1.1 Signaling

The signalling module allows users to vote to create non-binding polls. Non-binding polls are an important part of the governance process for existing

blockchains and have been administered previously in an ad-hoc manner. Users should natively be able to create and distribute polls for signaling interest in different proposals, strategies, and directions.

Individuals can choose between many different methods for voting on a proposal (e.g., binary or rankchoice) and moreover allows for delegated voting–improving the total stake allocated towards a vote. Once a direction has been chosen, the council or the democracy module can create a binding on-chain referendum.

4.1.2 Identity

On the Internet, email addresses have emerged as a de-facto standard for user identity and authentication. They are universally interoperable and recognized across services. No such widely adopted identity system has emerged on the blockchain yet. Several areas of blockchain applications that have only recently been deployed in the real world benefit strongly from some form of persistent user identity, among these, are lending, protocol governance or curation networks. For lending protocols, identity is necessary as under-collateralized loans are infeasible and risky unless the lender or underwriter knows the identity of their counterparty. For protocol governance, identity is necessary to build social consensus.

While some individuals may choose to remain entirely pseudonymous, participants on Codg network may find it useful to link an identity or claim to a pseudonymous address. By building an opt-in identity solution into Codg as a first-class primitive, Codg becomes an ideal platform for decentralized finance, governance, and other dApps.

Ideally, an identity standard would serve the previously mentioned use case, while having the ease of use and universal addressability of email login. The Codg identity system implements this, detailed in the `IdentityRecord`:

Name	Data Type	Description
`account`	`AccountId`	The specific Codg account with which the identity is associated.
`identity`	`Vec<u8>`	The byte array corresponding to an identity. Can be encrypted or unencrypted.
`stage`	`IdentityStage`	An `enum` that shows a proposal's lifecycle–`Registered`, `Attested`, or `Verified`.
`expiration`	`BlockNumber`	The block at which an identity becomes invalid. This field is optional
`proof`	`Attestation`	Proof posted for an identity.
`verifications`	`<Vec<(AccountId, bool)>>`	An array, with corresponding tuple–account and bool. Shows whether third-party individuals can verify that the identity and account are indeed linked.
`metadata`	`MetadataRecord`	An optional field that allows people to add an avatar, tagline, or a display name on Codg . This allows for a human-readable identifier on the system.

Public functions in the identity module allow an account to register, attest, or verify that identity. Accounts first `register` an identity. Thereafter, individuals `attest` to this by linking to an external proof. Third parties may now `verify` that this linked identity is valid, voting whether or not the attestation is valid. Verifications must come from specifically selected verifiers, not necessarily just any third party individual. The initial set of verifiers are the validators on the Codg network. Work to validate an identity may be delegated to a daemon to manage this process.

The sender of both the registration and attestation must be the same, helping to mitigate false attestations. Therefore, the blockchain serves as the single source of truth for attestations on identity registrations. A verifier should only consult the proof stored in the chain to decide whether it is valid or not and not any other form or presentation of an attestation. Examples of attestations for different settings:

- **Github**: The identity is a Github `username`. Upon username registration, the registrar should publish a Github Gist proof under the `username`'s account with a link to the registration on Codg .
- **Ethereum**: The identity is an Ethereum public key or address. Upon username registration, the registrar should send a 0 value transaction to a burn address symbolizing the public key of their corresponding Codg account and submit the transaction hash as the attestation. Upon inspection, a verifier should have enough proof that if the owner of the

Ethereum account did not also own the recipient Codg account (represented as the target address on Ethereum), then they would not issue such a transaction.

Initially, the set of verifiers are the set of accounts that have a minimum of 0.1% of CODG tokens. The votes for identity verification are coin-weighted. Codg does this so that it is organically able to establish a 1p1v government for other areas of the system.

4.1.3 Democracy

The democracy module to create and move different binding referendum along the voting process. Referendums allows for the execution of an arbitrary extrinsic call on the chain as the root user. This allows for execution of rather significant actions (e.g. setting balance, modifying timestamps, etc) without performing a runtime upgrade. The module currently only supports binary supermajority-to-pass votes unless submitted by a council member (which also supports supermajority-to-fail and simple majority). Individuals vote on binary choice ballots that are tallied by coin-weight with lock-time adjustments.

At the outset, all votes will be restricted to a two-week fixed period. In the future, the democracy module will be extended to many `VotingTypes`, they are:

- **Binary**: a traditional yes-or-no vote
- **Multi-Option**: where individuals can select multiple choices
- **Commit-Reveal**: a scheme to commit to a specific vote by posting the hash of the vote

At the resolution of the vote the `TallyType`, changes the method by which votes are weighed. They are enumerated below:

- **Coin-Weight**: where votes that an individual account casts are weighed on a per coin–one coin is equivalent to one vote
- **Lock-time weight**: where an account votes with a specific lock duration, with longer times corresponding to more voting power. For example, a vote may be locked for a one or two week period, with the latter time corresponding to a two-times voting power increase.
- **One-person-one-vote**: where one account or identity can cast one vote. On Codg , *one-person-one-vote* may refer to a specific set of verified identities, such as verifiCODG ithub accounts.

Initially, all referendum on Codg will be cast with both coin-weighting and lock-time-weighting. An important note, for coin-Weighted voting, a user cannot do a "partial funds vote"–users can only vote and lock up all the funds for a given account for the lock period, or not vote at all. Lock-time-weighting allows individual accounts with a smaller token balance to exercise more power in the governance process. Following the resolution of a vote, there is a delay before implementing any change–initially set to two weeks.

Additionally, the democracy module allows for delegated voting, the middle point between direct and representative democracy. At any point before a vote ends, an account can cast a vote different than how the account to whom the account may have delegated has done. Voluntary delegation has the potential to increase political participation, reduce strategic incentives within the election process, and ameliorate the political principal-agent problem. Delegation can be recursive. Codg mitigates an attack where a malicious individual may manipulate the depth of a delegation tree to an extreme depth. Instead of forcing all nodes tallying votes to traverse a deep tree of delegates, Codg limits the delegation to five. Cyclic delegation additionally prevented.

Democracy will be extended by adding support for a different `VotingType` such as enabling rank-choice and anonymous voting, where accounts are directly linked to their cast ballot. Or by adding different `TallyingTypes` such as quadratic voting [**anon**].

4.1.4 Council

Decentralized systems present a problematic scenario for online and active participation, often leading to low turnout. Therefore, council members can bias the quorum, the number of votes needed and the relative difference between affirmative and non-affirmative votes. The Codg council module will eventually expand to allows 24 individual accounts to hold some exclusive rights over the network with council members having a fixed term of 12 months. At network launch, the number of council members will be six.

The council votes on a peer basis, that is, no coin-weighted votes. Quorum biasing allows the council to change the effective supermajority required to make it easier or more difficult for a proposal to pass in the case that there is no clear majority of voting power backing it or against it. If all council members vote for a proposal, then the required amount of non-council member votes is lessened. The reverse also applies. When the council votes and more than one member dissents, then the quorum is negatively biased.

4.1.5 Treasury

While validators are often the only stakeholder natively incentivized in other blockchains, Codg proposes to use the block reward to incentivize all stakeholders. Block inflation will be allocated token towards an on-chain treasury,

which will be allocated by specific treasury proposals.

Initially, the on-chain treasury will be funded by 50% of the block reward. Codg tokenholders may allocate funds to:

- **Core Technology**: Implementing of scaling technologies such as runtime improvements, sharding, and off-chain extensions.
- **Governance**: Governance systems, including on-chain identity as well as tools for organizing and coordinating the work of core developers.
- **Developer experience**: A mature toolchain for developing, debugging, and testing smart contracts.
- **User experience**: Wallets and user experience primitives (e.g., JavaScript libraries) to make decentralized apps simple and easy to use.
- **Ecosystem support**: Engaging developers, end users, and other stakeholders through in-person events.

4.2 Governance Interfaces

Commonwealth Labs has developed an interface for governing decentralized networks. [**cwi**] The interface will support discussing and voting on proposals, creating and managing identities linked to Web 2.0 social media accounts, and tracking discussions about Codg proposals across different channels of communication.

Commonwealth Labs may also operate a Slack, Riot, or Discord chat for participants in the Codg ecosystem to communicate with each other. [**cwhub**]

Since Codg follows the standard Parity Substrate API, users may alternatively use standard Substrate libraries to manually access the chain, read the state of governance proposals, and sign transactions locally to submit votes to the network.

4.3 Governance Procedures

Codg stakeholders will be able to collaborate on the roadmap of the network through on-chain signaling, by holding informal votes for the inclusion of specific items and proposals. This planning stage is a natural precursor to the on-chain governance process, and formal votes are held to fund and upon which agreed upon features are implemented.

Establishing robust procedures for conducting on-chain upgrades will be an essential step towards ensuring the security of the network. All network upgrades should be audited and tested by multiple independent parties, and additionally approved by a significant quorum of CODG holders.

4.4 Experimentation Platform

Over the long term, the evolution and success of Codg will require involving a large set of committed stakeholders, including validators, voting coinholders, application developers, and core tech developers. We anticipate that governance structures for coordination each of these classes of stakeholders will emerge organically over time.

However, to accelerate this process, Codg will provide a set of tools for groups of stakeholders to experiment with new forms of governance, such as different voting models or decision-making processes. Native governance tools allows DAOs to emerge organically to test the effectiveness of different governance systems.

For example, while individuals and off-chain companies will be the first recipients of treasury funds, DAOs may emerge to coordinate work between pseudonymous online participants, who can establish an identity and a track record on the Codg network. Alternatively, with the deployment of anonymous voting systems, it should become possible to poll groups of trusted developers, investors, or experts using linkable ring signatures, to establish sentiment around key network upgrades and governance decisions.

Successful patterns discovered through this experimentation process can be reused and improved upon in further iterations. They can be reused on Layer 2 protocols, public goods funding DAOs, and other projects incubated on the Codg chain itself.

5 Conclusion

Decentralized computation is early in its development right now, and should be developed with fast iteration cycles and ambitious roadmaps. Creating a new actively-governed chain with a progressive upgrade policy allows this to happen efficiently and with the right incentives.

As a recursively self-improving system, Codg can fund, develop, and deploy network upgrades that support the performance and usability of the network. This makes Codg a novel alternative to existing blockchains. It aligns incentives so that rapid development and open governance processes create a virtuous cycle, and early successes in governance are reflected in the value of the network, community, and ecosystem.