On the Robustness of Blockchain-Based Oracle Mechanisms

Manuel Mueller-Frank*, Siyang Xiong†

February 13, 2025

Abstract

The game theoretic analysis of blockchain oracle mechanism so far has explicitly (or implicitly) assumed that reporting agents commonly know the realized state of the world. We investigate the robustness of oracle mechanisms to perturbations to common knowledge. A Bayes Nash equilibrium is *continuous* if for any sequence of types converging to common knowledge, the equilibrium strategies converge to the equilibrium strategies at complete information. An oracle mechanism satisfies *strong continuous implementation* if all of its continuous BNE implement the true state. We show that *any* oracle mechanism that does not provide payments to reporting agents in an oracle native coin, fails strong continuous implementation. We further show that any (simultaneous-report) oracle mechanism that satisfies strong continuous implementation is either dictatorial or vulnerable to *single deviations*, where a single agent can (probabilistically) prevent the implementation of the true state. To address this weakness, we propose a sequential direct voting mechanism. We show that this sequential mechanism satisfies *strong sequential-continuous implementation* and is not vulnerable to single deviations.

*IESE Business School, mmuellerfrank@iese.edu
†UC Riverside, siyang.xiong@ucr.edu

1 Introduction

Satoshi Nakamoto authored the Bitcoin whitepaper ([8]) introducing a digital money that functions without a central entity controlling the monetary base and intermediating transactions. At the core of his invention is blockchain, a decentralized distributed ledger that records transactions in a secure, transparent and censorship resistant manner. While considered a monetary and technological breakthrough, Bitcoins functionality is limited to simple transactions.

This limitation gave rise to general purpose blockchains that can host and execute complex computer programs or smart contracts, Ethereum ([3]) being the most prominent among them. One usecase of such general purpose blockchains and smart contracts is DeFi[1]. The vision of DeFi is to replace a significant part of financial services provided by financial institutions with decentralized applications deployed on blockchain as smart contracts.[2] Such decentralized applications automatically and deterministically execute agreements based on predefined conditions. For DeFi to become significant, agreements must be able to integrate real-world events that are not native to blockchain. For example, blockchain-based prediction markets trade contracts on election outcomes; for the betting contract to be executed the decentralized application requires the information who won the election.

This is the role of blockchain oracles that provide real-world data (henceforth real-world state or state) to blockchain-based smart contracts. In line with the core feature of blockchain such oracles are decentralized, i.e., there are many different entities that report. The reports are submitted to an oracle smart contract which provides an aggregated state given the reports. As such the oracle can be seen as a mechanism. It must incentivize reporting agents to report the state truthfully in order to assure that the correct state is aggregated (henceforth implemented). The formal analysis of blockchain oracles so far has explicitly or implicitly assumed that the state of the world is commonly known among agents. Common knowledge of the state requires that all players know the state, all players know that all players know the state, all players know that all players know that all players know the state, and so on ad infinitum.

[1] Short for Decentralized Finance.

[2] Examples of financial services rendered by blockchain-based decentralized applications are payments, exchanges, lending, and insurance.

The goal of this paper is to understand how general oracle mechanisms perform when the state is not commonly known but agents are 'close' to common knowledge. We believe our analysis to be important for two reasons. First, as it requires an infinite knowledge hierarchy it is hard to argue that common knowledge of the state, for example the election winner, is indeed commonly known among reporting agents close to the realization of the event. Second, it is conceivable that malicious agents with the aim to induce the incorrect state being implemented launch an attack that creates doubt about the realized state or about the knowledge of others about the realized state etc. To preview and summarize our results, we will establish that for a simultaneous-report oracle mechanism to implement the true state in a neighborhood of common knowledge requires either a dictatorial agent who solely determines the state or giving implementation veto power to a single agent. Neither of these two types of mechanisms are particularly attractive when seen through the lense of decentralization. However, there exist a sequential oracle mechanism with 'attractive' properties that is robust to perturbations to common knowledge and is neither dictatorial nor assigns implementation veto power.

We next describe the model.

Model Primitives and Oracle Mechanisms

We consider a finite state space Θ and a finite set N of reporting agents. An oracle mechanism $\mathcal{O}$ is described by the tuple (M, g, b) where M is a set of message profiles, g is the implementation function that assigns one state to each message profile, and b is a payment function that assigns a payment for every agent to each message profile.[3] As oracle mechanisms operate on blockchains as smart contracts, payments are made in a blockchain native coin. The utility of agents is equal to his payment multiplied by the fiat price of the coin payments are denominated in. We consider generalized direct mechanisms where the states are viable messages, $\Theta^n \subset M$. The oracle mechanism is deterministic if the implementation and payment functions are as well. Otherwise, the oracle mechanism is random, i.e., it involves lotteries over the implemented state and/or payments.

Part of our analysis focuses on oracle mechanisms that satisfy the properties of *monotonicity* and *label-neutrality*. Monotonicity implies that if state θ is implemented for a

[3]Agents submit reports simultaneously.

message profile m, then θ is still implemented if any agent where to switch his message m_i to θ. Label-neutrality requires that the implementation function is invariant in the label assigned to states. One natural class of mechanisms that satisfy both properties are weighted voting functions, where the state with the largest weighted vote count is implemented.

Information Structure, Equilibrium Concept and Strong Continuous Implementation

We use the standard formulation of incomplete information introduced in [6] and developed in [7]. The baseline model is complete information where all agents commonly know the realized state. To capture deviations from complete information, we consider a an epistemic model (or type space) that embeds the baseline model (complete information type space). For such a type space, we say that a Bayes Nash equilibrium (henceforth BNE) is *continuous* if for any sequence of type profiles that converge to the complete information type profile, the corresponding sequence of equilibrium strategy profiles converges to the strategy profile at complete information.[4] We say that an oracle mechanism satisfies *strong continuous implementation* if all continuous BNE implement the truth at complete information and there exists a continuous BNE σ^* such that the complete information strategy is to report the realized state. Ours is a variant of the notion of continuous implementation in Oury and Tercieux [9] and the notion of truthful continuous implementation in Chen, Mueller-Frank and Pai [4].[5] In other words, we require the equilibrium strategies to be continuous in types close to complete information.

Main Results

Our first result, Theorem 1, establishes that if the oracle payment coin is *implementation invariant*, where the fiat price of the coin does not vary in the relation between realized and implemented state, then there exists *no* oracle mechanism that satisfies strong continuous implementation. One take-away of this result is that oracles should denominate payments to reporting entities in its own oracle native token rather than a stablecoin or another cryptocurrency unrelated to the performance of the oracle.

[4] We use the product metric on types but our our results carry over to other metrics, for example the uniform-weak metric.

[5] We defer a discussion to the Conclusion.

We next show that there exists a direct, monotone and label-neutral oracle mechanism[6] that satisfies strong continuous implementation (Proposition 1). This mechanism is *dictatorial*, i.e., it selects one agent as dictator and implements the state the dictator reports, and thus ignoring the reports of all other agents. Clearly, a dictatorial mechanism stands in direct contrast with blockchain's ideal and reality of decentralization, and is thus undesirable. Theorem 2 considers a binary state space and establishes that if a deterministic monotone oracle mechanism satisfies strong continuous implementation then it is dictatorial. For a general state space, Theorem 3 establishes that if a deterministic monotone and label-neutral oracle mechanism satisfies strong continuous implementation then it it dictatorial. The inability for deterministic mechanisms with common and desirable properties to satisfy strong continuous implementation motivates our analysis of random oracle mechanisms.

To contrast dictatorial mechanisms, we say that a mechanism is *democratic* if both the payment and implementation function are invariant in the identities of reporting agents. Theorem 4 constructs a direct random mechanism that is monotone, label-neutral and democratic and satisfies strong continuous implementation. This appears like a substantial improvement relative to deterministic mechanisms. However, Proposition 2 establishes that any oracle mechanism that satisfies strong continuous implementation has the *single deviation property*, where a single agent deviating from his equilibrium message prevents the true state being implemented with probability one. In summary, all 'desirable' simultaneous oracle mechanisms that satisfy strong continuous implementation are either dictatorial or assign probabilistic veto power to at least one agent.

To address this, we analyze sequential oracle mechanisms. We introduce the notion of *strong continuous-sequential implementation* and show that there exists a deterministic direct mechanism that is monotone, label-neutral and non-dictatorial, and does satisfy strong sequential-continuous implementation. In distinction to the random mechanism of Theorem 4, it is not subject to single deviations.

We believe that there are important implications of our results for the design of blockchain oracles that are novel. First, an oracle native token is necessary for robustness against (infinitessimal) deviations from common knowledge. Second, any simultaneous mechanism that achieves the desired robustness is subject to single deviations, i.e.,

[6]For a direct mechanism we have $M = \Theta^n$.

where there exists at least one agent who unilaterally can prevent the true state being implemented (with probability one). Third, sequential oracle mechanisms can achieve robustness without the vulnerability of a single agent preventing the implementation of the true state.

The rest of the paper is organized as follows. Section 2 introduces the model. Section 3 provides the results for deterministic oracle mechanisms. Section 4 presents the results for random oracle mechanisms. Section 5 considers sequential oracle mechanisms and provides the corresponding result. Section 6 provides a discussion of the results in context of the relevant literature and concludes. The Appendix presents some proofs omitted from the main text.

2 The model

There is a finite set of players $N = \{1, ..., n\}$ and a finite set of states, denoted Θ. The players participate in providing real world data onto the blockchain and a state $\theta \in \Theta$ corresponds to the realized real-world data, such as, for example the winner of a presidential election. An *oracle mechanism* elicits messages from all players and assigns to each message profile a state and payment to each player. Let M_i denote the set of player i's messages and let $b_i(m) \in \mathbb{R}$ denote agent i's payment given message profile $m = (m_1, ..., m_n)$. Formally, we have the following:

Definition 1. A *deterministic oracle mechanism* $\mathcal{O}$ is described by a message space $M \equiv \times_{i \in N} M_i$ such that $\Theta \subset M_i$ for all $i \in N$, a *implementation function* $g : M \longrightarrow \Theta$ and a *payment function* $b \equiv (b_i : M \longrightarrow \mathbb{R})_{i \in N}$. A *random oracle mechanism* $\mathcal{O} = (M, g, b)$ allows for random implementation and payment functions, i.e., $g : M \longrightarrow \Delta(\Theta)$ and $b : M \longrightarrow \Delta(\mathbb{R}^n)$.

We analyze (generalized) direct mechanisms, as each state is a feasible message. We will consider several different properties of an oracle mechanism. These properties are realistic in the sense that they are typically satisfied by oracle mechanism that are implemented on blockchains.

Definition 2. A deterministic oracle mechanism $\mathcal{O} = (M, g, b)$ is *monotone* if

$$g(m) = \theta \Longrightarrow g\left(m_i' = \theta, m_{-i}\right) = \theta, \, \forall \left[i, \theta, m\right] \in N \times \Theta \times M.$$

Monotonicity is a common property for finite state space oracle mechanisms that are implemented on blockchains. It requires that if state θ is implemented for a message profile m, then it is also implemented for any message profile m' that differs from m only in that some players switch their message to θ.

Definition 3. A deterministic oracle mechanism $\mathcal{O} = (M, g, b)$ is *label-neutral* if for any bijection $\gamma : \Theta \longrightarrow \Theta$, there exists a profile of bijections $(\Gamma_j : M_j \longrightarrow M_j)_{j \in N}$ such that

$$\Gamma_j(\theta) = \gamma(\theta), \forall (\theta, j) \in \Theta \times N,$$

and

$$g\left(\left[\Gamma_j(m_j)\right]_{j \in N}\right) = \gamma\left(g\left[(m_j)_{j \in N}\right]\right), \forall (m_j)_{j \in N} \in \Theta.$$

Label-neutrality requires that the implemented outcome is invariant in the label assigned to states. This is a natural property in so far as it does not allow for an implementation bias towards one particular state. One example of a natural oracle mechanism is a direct voting mechanism, where the reports are equal to states, and the state with the highest vote count is implemented.[7] Such a voting mechanism satisfies both monotonicity and label-neutrality.

More generally, many consensus protocols and smart contracts, including oracles, deployed on blockchains rely on *weighted voting functions*. In the context of an oracle mechanism, this implies that each reporting agent i has an assigned weight $w_i \in [0, 1]$, $\sum_{i=1}^{n} w_i = 1$, and state θ is implemented if and only if

$$\sum_{i=1}^{n}(w_i \times 1_{[m_i=\theta]}) > \sum_{i=1}^{n}(w_i \times 1_{[m_i=\theta']}) \tag{1}$$

for all $\theta' \neq \theta$. Note that weighted voting mechanisms also satisfy both monotonicity and label-neutrality.[8] As such, the two properties we described are arguably natural and realistic.

Definition 4. In a deterministic oracle mechanism $\mathcal{O} = (M, g, b)$, an agent $i \in N$ is a *dictator* if

$$m_i = \theta \Longrightarrow g(m_i, m_{-i}) = \theta, \forall \left[\theta, (m_j)_{j \in N}\right] \in \Theta \times M.$$

[7] Assume that there is an order over the agents such that ties are resolved in favor of the highest ranked among the tied agents.

[8] For generic weights ties do not occur.

A mechanism $\mathcal{O}$ is dictatorial if a dictator exists, and it is non-dictatorial otherwise.

All existing oracle mechanisms are non-dictatorial, i.e., they have no reporting player that alone dictates the implemented state.

As oracle mechanisms are implemented as smart contracts (or programs) on a blockchain, the payments to reporting agents are made in a blockchain native coin. Let $p_{\theta'\theta}$ denote the fiat price of the coin when state θ' is implemented while the true state is θ. The following two properties of the fiat price of the oracle payment coin will be of fundamental importance:

Definition 5. The oracle payment coin is *implementation invariant* if

$$p_{\theta\theta} = p_{\theta'\theta} > 0, \forall (\theta, \theta') \in \Theta \times \Theta.$$

The oracle payment coin satisfies *positive implementation impact* if

$$p_{\theta\theta} > p_{\theta'\theta}, \forall (\theta \neq \theta') \in \Theta \times \Theta.$$

2.1 Incomplete Information and Type Space

We are interested in the incomplete information case, where the realized state is not common knowledge among the agents. Common knowledge of the state requires that all players know the state, all players know that all players know the state, all players know that all players know that all players know the state, and so on ad infinitum. When common knowledge of the state fails, agents need to form beliefs over the state space and beliefs over the beliefs of others. This is formalized via a model (or type space). A type space $\mathcal{T}$ consists of a tuple (T, κ) with $T = T_1 \times ... \times T_n$, and $\kappa_{t_i} \in \Delta(\Theta \times T_{-i})$ denotes the associated beliefs of type t_i, and $\kappa \equiv [(\kappa_{t_i} \in \triangle (\Theta \times T_{-i})_{t_i \in T_i}]_{i \in N}$. For each agent $i \in I$, a type $t_i \in T_i$ induces a hierarchy of higher-order beliefs. Specifically, the first-order belief of t_i, denoted by t_i^1, is the marginal distribution of κ_{t_i} on θ, and the second-order belief of t_i, denoted by t_i^2, is the marginal distribution of κ_{t_i} on $\left(\theta, t_{-i}^1\right)$, or more precisely,

$$t_i^2 (E) = \kappa_{t_i} \left[\left\{ (\theta, t_{-i}) \in \Theta \times T_{-i} : \left(\theta, t_i^1, t_{-i}^1\right) \in E \right\} \right], \forall E \subset \Theta \times \left(\triangle (\Theta)\right)^N.$$

Similarly, we can define t_i's third-order, fourth-order,.... beliefs iteratively. In order to analyze the equilibrium behavior of close-by types we rely on the standard metric applied to the universal type space. Define $X^0 \equiv \Theta$ and $X^k \equiv X^{k-1} \times \left(\triangle\left(X^{k-1}\right)\right)^N$ for any integer $k \geq 1$. Thus, $t_i^{k+1} \in \triangle\left(X^k\right)$ for any $k \geq 0$. Let d^0 denote the discrete metric on Θ, and let d^1 denote the Prohorov metric on $\triangle(\Theta)$. Inductively, for any $k \geq 1$, let $\widehat{d}^k$ denote the super-metric induced by d^0,..., d^k, and let d^{k+1} denote the Prohorov metric on $\triangle\left(X^k\right)$, where X^k is endowed with $\widehat{d}^k$. For an agent i, we now define a metric on types in T_i: for any $(t_i, t_i') \in T_i \times T_i$, define

$$d_i\left(t_i, t_i'\right) \;=\; \sum_{k \in \mathbb{N}} \left(\frac{1}{2}\right)^k d^k\left(t_i^k, t_i^{k\prime}\right),$$

i.e., d_i is the classic product metric on higher-order beliefs.[9] For type profiles in T we define the following metric

$$d(t, t') = \max_{i \in N} d_i(t_i, t_i').$$

Our benchmark model is that of complete information where there is common knowledge of the realized state. The complete information type space $\overline{\mathcal{T}}$ is formalized as follows: :

$$\overline{\mathcal{T}} \equiv \left(\overline{T} \equiv \times_{i \in N} \overline{T}_i = \left\{t_i^\theta : \theta \in \Theta\right\},\ \overline{\kappa} \equiv \left[\left(\overline{\kappa}_{t_i} \in \triangle\left(\Theta \times \overline{T}_{-i}\right)\right)_{t_i \in \overline{T}_i}\right]_{i \in N}\right)$$
$$\text{such that } \overline{\kappa}_{t_i^\theta}\left[\left\{\left(\theta, \left(t_j^\theta\right)_{j \in N \setminus \{i\}}\right)\right\}\right] = 1,\ \forall\,(\theta, i) \in \Theta \times N,$$

i.e., type t_i^θ has common knowledge of θ.

In order to consider deviations from complete information, we need to enrich the complete information model with a larger model that includes it. Formally, given two models $\mathcal{T} \equiv (T, \kappa)$ and $\mathcal{T}' \equiv (T', \kappa')$, we follow [9] to define

$$\mathcal{T} \supset \mathcal{T}' \Longleftrightarrow \left(\begin{array}{c} T \supset T' \text{ and} \\ \text{for any } i \in I, \text{ any } t_i \in T_i' \text{ and any measurable } E \subset \Theta \times T_{-i}, \\ \kappa_{t_i}(E) = \kappa'_{t_i}\left(E \cap \left[\Theta \times T'_{-i}\right]\right) \end{array}\right).$$

[9] All of our results also hold for other metrics on higher-order beliefs, e.g., the uniform-weak metric in [5].

2.2 The Bayesian Game and Equilibrium Concept

We assume that the preferences of agents are strictly monotone in the fiat payment agents receive. Let $u_i : M \times \Theta \longrightarrow \mathbb{R}$ denote player i's utility function. We have

$$u_i(m, \theta) = p_{g(m)\theta} \times b_i(m). \tag{2}$$

Having defined the agents' preferences, an oracle mechanism $\mathcal{O}$ and a type space $\mathcal{T}$ define an incomplete information game. Given $(\mathcal{O}, \mathcal{T})$, let $\sigma \equiv (\sigma_i : T_i \longrightarrow \triangle(M_i))_{i \in I}$ denote a strategy profile, and as usual, σ is a Bayesian Nash equilibrium (hereafter, BNE) if and only if for any $i \in I$, any $t_i \in T_i$, every m_i on the support of σ_i is a best reply to σ_{-i}. We are interested in oracle mechanisms that 1) implement the truth at complete information, i.e., where the BNE

$$\sigma(t_\theta) = (\sigma_1(t_1^\theta), ..., \sigma_n(t_n^\theta)) \tag{3}$$

satisfies $g(\sigma(t_\theta) = \theta$ for all $\theta \in \Theta$, and 2) players' strategies remain "close" to the complete information Nash equilibrium strategies for types close to complete information. We formalize this in the two definitions below.

Definition 6. Given $(\mathcal{O}, \mathcal{T})$ with $\mathcal{T}(\supset \overline{\mathcal{T}})$, a BNE σ is *continuous* if for any sequence $\{t_n\} \subset T$ with $d(t_n, t_\theta) \to 0$, we have $\sigma(t_n) \to \sigma(t_\theta)$.

A *continuous* BNE has the property that the strategies for types close to complete information are close to the strategies at complete information.

Definition 7. We say that an oracle mechanism $\mathcal{O}$ satisfies *strong continuous implementation* if for any Harsanyi type space $\mathcal{T}(\supset \overline{\mathcal{T}})$, we have 1) *all* continuous BNE σ satisfy $g(\sigma((t_\theta)) = \theta$ for all $\theta \in \Theta$, and 2) there exists a continuous BNE σ^* such that $\sigma_i^*(t_\theta) = \theta$ for all agents $i \in N$.

Strong continuous implementation requires that *all* continuous BNE implement the true state at complete information, and that there exist one truth-telling continuous BNE where all agents truthfully report the realized state. For our purposes, we require a stronger notion than standard continuous implementation (see Oury and Tercieux [9], and Chen, Mueller-Frank and Pai[4]) which requires the existence of *one* continuous BNE

that implements the truth at complete information. One trivial example highlighting why continuous implementation is insufficient for oracle mechanisms is that an oracle mechanism with a plurality-based implementation function and a zero constant payment function continuously implements the truth.[10]

3 Strong Continuous implementation for Deterministic Oracle Mechanisms

To the best of our knowledge, oracles that are implemented in the real world are all deterministic. Thus, we shall analyze the case of deterministic oracles first. The first result focuses on the role of the oracle payment coin.

Theorem 1. If the oracle payment coin is implementation invariant, then every oracle mechanism $\mathcal{O}$ fails strong continuous implementation.

One interpretation of the result is that blockchain oracle providers should issue their own coin and denominate oracle payments in their own coin.[11] Please see below for the proof of Theorem 1.

Proof. We proceed with a proof by contradiction. Suppose that the oracle mechanism $\mathcal{O}$ satisfies strong continuous implementation and the corresponding oracle payment coin is implementation invariant. Consider any type space T with $\mathcal{T} \supset \overline{\mathcal{T}}$. As $\mathcal{O}$ satisfies strong continuous implementation, there exists a BNE σ and for each $\theta \in \Theta$ a strategy profile m^*_θ such that $g(m^*_\theta) = \theta$, and $\sigma(t^\theta) = m^*_\theta$. Pick any $i \in I$ and any $m'_i \in M_i$, define

$$\theta' \equiv g(m'_i, m^*_{\theta-i}).$$

Since σ is a BNE, we have

$$b_i(m^*_\theta) \times p_{\theta^*\theta^*} \geq b_i(m'_i, m^*_{\theta-i}) \times p_{\theta'\theta^*}, \, \forall i \in N, \, \forall m'_i \in M_i,$$

[10] To see this, note that here all strategy profiles are payoff equivalent and hence all strategy profiles constitute a Bayes-Nash equilibrium. Thus, one can construct a continuous BNE where each player for every type t_i close to the complete information type t^θ plays the strategy θ.

[11] This is the case for most oracle providers, for example Chainlink's \$LINK coin and Pyth Network's \$Pyth coin. Two notable exceptions are Chronicle Protocol and Redstone Oracles which both do not have their own coin.

which, together with the implementation invariant oracle payment coin condition, implies

$$b_i(m^*_\theta) \times p_{\theta^*\widetilde{\theta}} \geq b_i(m'_i, m^*_{\theta-i}) \times p_{\theta'\widetilde{\theta}}, \, \forall i \in N, \, \forall m'_i \in M_i, \, \forall \widetilde{\theta} \in \Theta.$$

This implies that $\sigma(t) = m^*_\theta$ for all $t \in \mathcal{T}$ is a continuous BNE. However, this contradicts strong continuous implementation as for $\theta'' \neq \theta$ we have $g(m^*_\theta) \neq \theta''$. □

For the remainder of the paper, we will restrict attention to the case of oracle payment coin that satisfy positive implementation impact. We next show that this condition is sufficient to ensure the existence of an oracle mechanism that satisfies strong continuous implementation.

Proposition 1. If the oracle payment coin satisfies positive implementation impact, then there exists a deterministic oracle mechanism that satisfies strong continuous implementation.

Proof. Consider the following direct mechanism $\mathcal{O} = (M, g, b)$ where $M = \Theta^n$ and $b_i(m) = 1$ for all $i \in N$ and $m \in M$. That is, the payment function is constant and equal to 1. For the implementation function, select an agent $i^* \in N$ and set $g(m) = m^*_i$ for all $m \in M$. In other words, i^* is a dictator. Note that any strategy profile σ such that

$$\sigma_{i^*}(t_{i^*}) \in \arg\max_{m_i \in \Theta} \mathbb{E}[\sum_{\theta \in \Theta} 1_{[m_i = \theta]} \times p_{m_i\theta} \mid t^1_{i^*}] \tag{4}$$

constitutes a continuous BNE and all such strategy profiles implement the true state. □

Proposition 1 establishes that strong continuous implementation is achievable via a simple deterministic mechanism. The dictatorial design of the mechanism translates a multi-player game into a single agent decision problem, that of the dictator. This in turn implies that higher-order beliefs are irrelevant.

One potential downside of the mechanism for which strong continuous implementation holds, is that it features a dictator who unilaterally determines the implemented state. This raises the question whether this dependence on a single agent is a general feature of deterministic oracle mechanisms that satisfy strong continuous implementation. We will answer this questions with two results that we present in the following.

Theorem 2. Consider a binary state space, $|\Theta| = 2$. If a deterministic monotone oracle mechanism satisfies strong continuous implementation, then it is dictatorial.

Thus, if one desires the oracle mechanism to satisfy monotonicity, then strong continuous implementation requires the reliance on a dictator.[12] We next consider a general finite state space and focus on oracle mechanisms that satisfy monotonicity and label neutrality. We have the following result.[13]

Theorem 3. Consider a deterministic oracle mechanism $\mathcal{O}$ that satisfies strong continuous implementation. If $\mathcal{O}$ satisfies monotonicity and label-neutrality, then it is dictatorial.

To summarize, the analysis of deterministic oracle mechanisms has shown that an oracle native coin is necessary for strong continuous implementation. Further, achieving robustness to deviations from complete information comes at the cost of reducing the oracle mechanism to a single agent decision problem where a dictator determines the implemented state. As one of the main points of blockchain technology is decentralization and the removal central intermediaries, this might be considered a disappointment. In the following section, we shall analyze how allowing for randomization impacts the ability to create oracle mechanisms that satisfy strong continuous implementation.

4 Strong Continuous Implementation for Random Oracle Mechanisms

Recall that a random oracle mechanism allows the implementation function and payment function to be lotteries over the implemented state and payments, respectively. Monotonicity is generalized to random mechanisms as follows. Let $g(m)_\theta$ denote the probability that θ is implemented given message profile m. A random oracle mechanism O is monotone if $g(m_i = \theta, m_{-i})_\theta \geq g(m)_\theta$, for all $m \in M$ and all $i \in N$.

We next turn to the analysis. First note that Theorem 1, the impossibility of strong continuous implementation carries forward to random oracle mechanisms. Thus we again focus on the case of positive implementation impact oracle coins. The results so far have

[12] The proof of Theorem 2 is relegated to the Appendix.
[13] The proof of Theorem 3 is relegated to the Appendix.

shown that strong continuous implementation in deterministic mechanisms with desirable properties such as monotonicity requires a dictator. That is, all the implementation power is concentrated on one reporting agent. If dicatorship is one end of the spectrum, democracy is the other end. To contrast dicatorial mechanisms, we introduce the following property of an oracle mechanism.

Definition 8. A (random) oracle mechanism $\mathcal{O} = (M, g, b)$ is *democratic* if for any permutation function of the players $\Gamma : N \longrightarrow N$, any $m = (m_j)_{j \in N} \in M$ and any $m' = (m_j)_{j \in N} \in M$, we have

$$m_j = m'_{\Gamma(j)}, \forall j \in N \Longrightarrow g(m) = g(m') \text{ and } b(m) = (m').$$

Thus a democratic oracle mechanism treats the messages of all agents symmetrically. The implemented (lottery on) outcome and payments depend only on the distribution of reported messages. We have the following result.

Theorem 4. There exists a direct, $M = \Theta^n$, monotone, label-neutral, and democratic random oracle mechanism $\mathcal{O}^R$ that satisfies strong continuous implementation.

Theorem 4 establishes that one can construct a direct random oracle mechanism that satisfies all the desired properties. The fact that this oracle mechanism is direct has additional benefits due its simplicity; the set of messages is equal to the set of states. Please see below for the proof which relies on the construction of a dominant strategy mechanism.

Proof. Consider a direct oracle mechanism $\mathcal{O}^R = (\Theta, g, b)$ with a constant payment function, $b_i^*(m) = 1$ for all $m \subset \Theta^n$ and for all $i \in N$, and the following random implementation function $g^* : \Theta^n \longrightarrow \Delta(\Theta)$

$$\Pr[g^*(m) = \theta] = \frac{\sum_{i=1}^{n} 1_{[m_i = \theta]}}{n} \tag{5}$$

That is, state θ is implemented with the probability equal to its reported proportion. Note that this mechanism satisfies label neutrality, monotonicity and is democratic.
Consider the realized state θ, a player i and a message profile $m_i \in \Theta^{n-1}$ of the remaining players $j \neq i$. An equivalent way of representing the random implementation function is

to say that with uniform probability, $\frac{1}{n}$, one player is selected as dictator and the state reported by the chosen dictator is implemented deterministically. As the payment function is constant and the oracle payment coin satisfies positive implementation impact, i.e., $p_{\theta\theta} > p_{\theta\theta'}$ for all $\theta \neq \theta'$, it follows that it is a dominant strategy for player i to report the true state. He has no impact if not chosen as dictator but if chosen, reporting the truth results in strictly higher payoff. □

There are two features of the constructed mechanism that are worthwhile to discuss. First, for each player $i \in N$ and complete information type t_i^θ, truthtelling, i.e., $\sigma_i(t_i^\theta) = \theta$ is a dominant strategy. Second, the mechanism does not robustly implement the true state θ with probability one if even a single agent deviates from the truthtelling message. We formalize this property as follows.

Definition 9. A random oracle mechanism $\mathcal{O}$ is subject to *single deviation* if for all $\theta \in \Theta$

$$g(\boldsymbol{\theta}) = \theta \neq g(m_i', \boldsymbol{\theta_{-i}})$$

for some agent $i \in N$ and message $m_i' \in M$.

We have the following result.[14]

Proposition 2. If a (random) oracle mechanism satisfies strong continuous implementation then it is subject to single deviation.

Thus, any oracle mechanism that satisfies strong continuous implementation is vulnerable to single deviations. Theorem 4 has shown that while allowing for random implementation functions enables equal sharing of implementation power among reporting agents, still a single agent can veto the implemented state with positive probability.

There is an important implication of the results we have established so far. The main concern of blockchain oracle providers are so called "bribery attacks", where a malicious entity bribes reporting agents to misreport the true state. One aspect of the implemented solution is to decentralize, i.e., have many agents report to the oracle mechanism.[15] Our results have shown that protection against a "common knowledge attack" requires that the mechanism is subject to single deviations, where a single reporting agent can prevent

[14]The proof of Proposition 2 is relegated to the Appendix.
[15]See the "Chainlink 2.0" whitepaper[2] for example.

the true state being implemented with probability one. This motivates our analysis of sequential oracle mechanisms which we pursue in the next section.

5 Strong Continuous Implementation for Sequential Oracle Mechanisms

In this section, we identify a sequential-voting oracle mechanism which achieves strong truthful continuous implementation. Meanwhile, this oracle mechanism satsifies monotonicity, label-neutrality, is non-dictatorial and not subject to single deviations.

5.1 The Sequential-Voting Mechanism

Fix any order of agents, i.e.,

$$N = \{i_1, i_2, ..., i_n\}.$$

We consider a sequential-voting oracle mechanism

$$\mathcal{O}^S = \left(M^* \equiv \Theta^N, \ \left[g^S : \Theta^N \longrightarrow \Theta, b^S \equiv \left(b_i^S : \Theta^N \longrightarrow \mathbb{R}\right)_{i \in N}\right]\right),$$

which is defined as follows. We consider a perfect-information n-period voting game: at each period $k \in \{1, ..., n\}$, agent i_k vote for one outcome in Θ, and in particular, agent i_k knows the voting outcomes in the previous $(k-1)$ periods when he casts his vote. Let $(v_{i_1}, ..., v_{i_n}) \in \Theta^N$ denote a voting profile, where $v_k \in \Theta$ in $(v_{i_1}, ..., v_{i_n})$ represents agent i_k voting for v_{i_k} at period k. For notational ease, we will denote the set of voting profiles Θ with V, i.e., $V \equiv \Theta^N$.

We adopt a weighted voting rule, i.e., for any voting profile $(v_{i_1}, ..., v_{i_n}) \in V$, we have

$$g^S(v_{i_1}, ..., v_{i_n}) = \theta \Longleftrightarrow \sum_{k=1}^{n} (w_{i_k} \times 1_{[v_{i_k}=\theta]}) > \sum_{k=1}^{n} (w_{i_k} \times 1_{[v_{i_k}=\theta']}), \ \forall (\theta, \theta') \in \Theta \times \Theta.$$

$(w_{i_k})_{i_k \in N}$ are the weights for agents, and we choose $(w_{i_k})_{i_k \in N}$ such that $w_{i_k} \approx \frac{1}{n}$ and ties never occur. It is straightforward to show that this voting mechanism satisfies

monotonicity and label-neutrality.[16] Finally, define

$$b_{i_k}^S\left(v_{i_1},...,v_{i_n}\right)\equiv\begin{cases}1, & \text{if } v_{i_k}=g\left(v_{i_1},...,v_{i_n}\right),\\ 0, & \text{if } v_{i_k}\neq g\left(v_{i_1},...,v_{i_n}\right),\end{cases}$$

i.e., agent i_k receives a payment of 1 if and only if her votes matches the outcome chosen by the oracle mechanism (i.e., g^S).

5.2 The Incomplete-Information Game

Since we consider a sequential-move game, we adopt a different solution concept: perfect Bayesian equilibrium (PBE), which is defined as follows. Define

$$H_{i_1}\equiv\{\emptyset\},\ H_{i_k}\equiv\Theta^{(k-1)},$$

i.e., H_{i_k} is the history set of agent i_k, and $\emptyset$ in H_{i_1} represents the initial history of agent i_1 (which contains no information). Define $H\equiv\times_{i\in N}H_i$.

Given any Harsanyi type space, $\mathcal{T}\equiv(T,\kappa)$, we use

$$\sigma\equiv\left(\sigma_{i_k}:T_{i_k}\times H_{i_k}\longrightarrow\triangle\left(\Theta\right)\right)_{k\in\{1,2,3,...,n\}}$$

to denote a strategy profile. For any $t_{i_k}\in T_{i_k}$, we use

$$\mu_{t_{i_k}}^{\sigma}\in\triangle\left[\Theta\times T_{-i}\times\Theta^N\right]$$

to denote the distribution induced by both $\kappa_{t_i}\in\Delta(\Theta\times T_{-i})$ and σ. Precisely, for any $E\subset\Theta\times T_{-i}$ and any $v\equiv(v_{i_1},...,v_{i_n})$, we have

$$\mu_{t_{i_k}}^{\sigma}\left[E\times\{v\}\right]=\int\limits_{E\subset\Theta\times T_{-i}}\left[\sigma_{i_1}\left[t_{i_1},h_{i_1}=\emptyset\right]\left(v_{i_1}\right)\times\prod_{k'=2}^{n}\sigma_{i_{k'}}\left[t_{i_{k'}},h_{i_{k'}}=\left(v_{i_1},...,v_{i_{k'-1}}\right)\right]\left(v_{i_{k'}}\right)\right]d\kappa_{t_i},$$

where $\sigma_i\left[t_i,h_i\right]\left(v_i\right)$ denotes the probability assigned to v_i by $\sigma_i\left[t_i,h_i\right]$. Finally, for any

[16] This voting mechanism is almost democratic. Since individual weights are almost identical between agents, for $|\Theta|=2$ and an odd number of agents, it does satisfy democracy.

$h_{i_k} \in H_{i_k}$,

$$\mu_{t_{i_k}}^{\sigma-h_{i_k}} \in \triangle\left[\Theta \times T_{-i} \times \Theta^N\right]$$

denotes the conditional probability of $\mu_{t_{i_k}}^{\sigma}$ given the history h_{i_k}.[17]

We now define perfect Bayesian equilibrium (PBE): σ is a PBE on $\mathcal{T} \equiv (T, \kappa)$, if and only if for any agent $i_k \in N$ and any $h_{i_k} \in H_{i_k}$, we have

$$\int\limits_{(\theta,t_{-i},v)\in\Theta\times T_{-i}\times V} \left[b_{i_k}^S(v) \times p_{g(v)\theta}^S\right] \mu_{t_{i_k}}^{\sigma-h_{i_k}} \geq \int\limits_{(\theta,t_{-i},v)\in\Theta\times T_{-i}\times V} \left[b_{i_k}^S(v) \times p_{g(v)\theta}^S\right] \mu_{t_{i_k}}^{\left(\sigma'_{i_k},\sigma_{-i_k}\right)-h_{i_k}}$$

for any agent i_k's strategy σ'_{i_k}.

5.3 Interim Beliefs and Sequential Rationality

The Harsanyi type space $\mathcal{T} \equiv (T, \kappa)$ describes players' beliefs before they play the sequential-voting game. Given an equilibrium σ, as the vote game progresses, players observe other players' votes following σ, which reveal other players' private information, and as a result, players update their information before they cast their votes. In particular, different histories may leads to different updated beliefs for every players. Therefore, $\mathcal{T} \equiv (T, \kappa)$ and σ induces new beliefs and a new interim Harsanyi type space, which is denoted by

$$\widehat{\mathcal{T}}^{\sigma} \equiv (\widehat{T} \equiv \times_{i_k\in N}\left(\widehat{T}_{i_k} \equiv T_{i_k} \times H_{i_k}\right), \widehat{\kappa}^{\sigma}).$$

Each history $h_{i_k} \in H_{i_k}$ leads each original type $t_{i_k} \in T_{i_k}$ to a different updated belief, and hence, a new type is $\widehat{t}_{i_k} \equiv (t_{i_k}, h_{i_k}) \in \widehat{T}_{i_k}$. For any $t_{i_k} \in T_{i_k}$ and any $h_{i_k} = \left(v_{i_1}, ..., v_{i_{k-1}}\right)$, we define $\widehat{\kappa}^{\sigma}_{\widehat{t}_{i_k}\equiv\left(t_{i_k},h_{i_k}\right)} \in \triangle\left(\Theta \times \widehat{\mathcal{T}}^{\sigma}_{-i_k}\right)$ as follows. For any $E \subset \Theta \times \widehat{\mathcal{T}}^{\sigma}_{-i_k}$, we have

$$\widehat{\kappa}^{\sigma}_{\widehat{t}_{i_k}}(E) \equiv \mu_{t_{i_k}}^{\sigma-h_{i_k}}\left[\left\{(\theta, t_{-i}, v') \in \Theta \times T_{-i} \times V : \left(\theta, \left[t_{i_{\widetilde{k}}}, \left(v'_{i_{\widetilde{k}}}, ..., v'_{i_{\widetilde{k}-1}}\right)\right]_{i_{\widetilde{k}}\in N\setminus\{i_k\}}\right) \in E\right\}\right].$$

[17] That is, $\mu_{t_{i_k}}^{\sigma-h_{i_k}}\left[\Theta \times T_{-i} \times \{h_{i_k}\} \times \Theta^{\{i_k,...,i_n\}}\right] = 1$ and

$$\mu_{t_{i_k}}^{\sigma}\left[\Theta \times T_{-i} \times \{h_{i_k}\} \times \Theta^{\{i_k,...,i_n\}}\right] > 0$$

$$\Longrightarrow \mu_{t_{i_k}}^{\sigma-h_{i_k}}\left[E \times \{(v_{i_1}, ..., v_{i_n})\}\right] = \begin{cases} \frac{\mu_{t_{i_k}}^{\sigma}\left[E\times\left\{\left(v_{i_1},...,v_{i_n}\right)\right\}\right]}{\mu_{t_{i_k}}^{\sigma}\left[\Theta\times T_{-i}\times\left\{h_{i_k}\right\}\times\Theta^{\{i_k,...,i_n\}}\right]}, & \text{if } h_{i_k} = \left(v_{i_1}, ..., v_{i_{k-1}}\right), \\ 0, & \text{if } h_{i_k} \neq \left(v_{i_1}, ..., v_{i_{k-1}}\right). \end{cases}$$

Based on the new Harsanyi type space $\widehat{\mathcal{T}}^{\sigma} \equiv (\widehat{T}, \widehat{\kappa}^{\sigma})$ defined above, we can derive higher order beliefs of each new type $\widehat{t}_{i_k} \equiv (t_{i_k}, h_{i_k}) \in \widehat{T}_{i_k}$.

Definition 10. Given $(\mathcal{O}, \mathcal{T})$ with $\mathcal{T}(\supset \overline{\mathcal{T}})$, a PBE σ is *continuous* if any $h \in H$ and any sequence $\{t_n\} \subset T$ such that $d\left((t_n, h), (t_\theta, h)\right) \to 0$, we have $\sigma\left(t_n, h\right) \to \sigma(t_\theta, h)$.

Continuous PBE has a similar flavor as continuous BNE: the strategies for types close to complete information are close to the strategies at complete information. Different from continuous BNE, continuous PBE is defined based on convergence of types' interim beliefs (i.e., $d\left((t_n, h), (t_\theta, h)\right) \to 0$), rather than their original beliefs (i.e., $d\left(t_n, t_\theta\right) \to 0$). The rationale is that players choose their best replies in a PBE based on their interim beliefs rather than their original beliefs. Or equivalently, continuous PBE is based on sequential rationality (induced by interim beliefs), while continuous BNE does not. Finally, we define strong sequential-continuous implementation.

Given any $\theta \in \Theta$ and any PBE σ, we define the equilibrium path for t_θ:

$$h_{i_1}^{\sigma-\theta} \equiv \emptyset,\ h_{i_2}^{\sigma-\theta} \equiv \left(\sigma_{i_1}^{*}(t_\theta, h_{i_1}^{\sigma-\theta})\right), ..., \ h_{i_n}^{\sigma-\theta} \equiv \left(\sigma_{i_1}^{*}(t_\theta, h_{i_1}^{\sigma-\theta}), ..., \sigma_{i_{n-1}}^{*}(t_\theta, h_{i_{n-1}}^{\sigma-\theta})\right),$$
$$\text{and } h^{\sigma-\theta} \equiv \left(h_i^{\sigma-\theta}\right)_{i \in N}.$$

Definition 11. We say that an oracle mechanism $\mathcal{O}$ satisfies *strong sequential-continuous implementation* if 1) *all* continuous PBE σ satisfy $g(\sigma((t_\theta, h^{\sigma-\theta})) = \theta$ for all $\theta \in \Theta$, and 2) there exists a continuous PBE σ^* such that $\sigma_i^*(t_\theta, h_i^{\sigma-\theta}) = \theta$ for all $\theta \in \Theta$ and all agents $i \in N$.

5.4 Achieving Strong Sequential-Continuous Implementation

We now present the main result of this section.

Theorem 5. The deterministic direct sequential oracle mechanism $\mathcal{O}^S$, which is monotone, non-dictatorial and label-neutral, satisfies strong sequential-continuous implementation.

The similarity between the mechanisms $\mathcal{O}^R$ (Theorem 4) and $\mathcal{O}^S$ (Theorem 5) is that both are monotone, label neutral, and non-dicatorial, and satisfy strong (sequential-)continuous implementation. The oracle mechanism $\mathcal{O}^R$ achieves this via randomization

and $\mathcal{O}^S$ through is sequential extensive form. The crucial difference between them, however, is that the sequential voting mechanism $\mathcal{O}^S$ is not subject to single deviation.[18] That is, a sequential mechanism with the desired properties can achieve strong sequential-continuous implementation and is not subject to individual (probabilistic) veto power.

6 Conclusion

We analyzed the robustness of oracle mechanisms to deviations from common knowledge and highlight several vulnerabilities. First, a blockchain native coin is necessary for strong continuous implementation. This is relevant as there are several blockchain providers that do not have their own native coin. Two notable examples are Redstone Oracles and Chronicle Protocols whose oracles secure over 7 billions US dollars of value locked into smart contracts.[19] Second, any oracle mechanism with simultaneous reporting that does satisfy strong continuous implementation is either dictatorial or subject to single deviations, i.e., a single agent having the ability to probabilistically veto the implementation of the true state. Third, there exists a direct sequential voting mechanism that achieves strong sequential-continuous implementation and is not vulnerable to single deviations.

While not the primary goal of the analysis, our results also provide some relevant insights to the implementation literature. It follows from results in Chen, Mueller-Frank and Pai [4] that (strong) continuous implementation in the product topology requires an oracle mechanism that at complete information implements the true state through uniquely rationalizable strategies. Proposition 2 of Bergemann, Morris and Tercieux [1] provide sufficient conditions for a social choice function, in our case $f(\theta) = \theta$, to be implemented through rationalizable strategies. Bergemann et al.'s [1] proof constructs a general, random mechanism with an infinite message space that achieves rationalizable implementation. Thus their mechanism is highly complex and arguably difficult to implement in the real world. An interesting feature of Theorem 4 is that, within the relevant setting of oracle mechanisms, there exists a *direct* mechanism that achieves implementation in dominant strategies.

[18] We omit a formal statement and the proof. The result follows trivially from the plurality rule that determines the implemented state and the weights being approximately equal across agents.

[19] Data from February 10th, 2025. See https://defillama.com/oracles.

To the best of our knowledge, there have been no previous attempts to generalize the notion of continuous implementation for general Harsanyi type spaces to sequential games. Our positive result on the sequential mechanism highlights the difference between simultaneous and sequential mechanisms in their ability to achieve strong (sequential-)continuous implementation. Finally, note that all our results for simultaneous mechanisms hold if you replace strong continuous implementation with the notion of strict continuous implementation introduced by Chen et al.[4].

References

[1] Dirk Bergemann, Stephen Morris, and Olivier Tercieux. Rationalizable implementation. *Journal of Economic Theory*, 146:1253–1274, 2011.

[2] Lorenz Breidenbach, Christian Cachin, Benedict Chan, Alex Coventry, Steve Ellis, Ari Juels, Farinaz Koushanfar, Andrew Miller, Brendan Magauran, Daniel Moroz, Sergey Nazarov, Alexandru Topliceanu, Fan Zhang, and Florian Tramèr. Chainlink 2.0: Next steps in the evolution of decentralized oracle networks. *Chainlink Whitepaper*, April 2021. Version 1.0.

[3] Vitalik Buterin. Ethereum: A next-generation smart contract and decentralized applicaiton platform. *Ethereum Foundation White Paper*, 2014.

[4] Yi-Chun Chen, Manuel Mueller-Frank, and Mallesh Pai. Continuous implementation with direct revelation mechanisms. *Journal of Economic Theory*, 201(105422), 2022.

[5] Yi-Chun Chen, Alfredo Di Tillio, Eduardo Faingold, and Siyang Xiong. Uniform topologies on types. *Theoretical Economics*, pages 445–478, 2010.

[6] John C. Harsanyi. Games with incomplete information played by "bayesian" players, i–iii part i. the basic model. *Management Science*, 14:127–261, 1967.

[7] Jean François Mertens and Shmuel Zamir. Formulation of bayesian analysis for games with incomplete information. *International Journal of Game Theory*, 14:1–29, 1985.

[8] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *White Paper*, 2008.

[9] Marion Oury and Olivier Tercieux. Continuous implementation. *Econometrica*, 80(4):1605–1637, 2012.

A Theorem 2: Auxiliary Lemma and Proof

The proof of Theorem 2 relies on the following lemma.

Lemma 1. Fix a metric topology on the universal type space, an oracle mechanism $\mathcal{O}$ and a continuous BNE σ. For every state θ, let $\sigma(t^\theta) = m^*_\theta \in M$. If there exists a state θ' such that for all agents $i \in N$ and messages $m'_i \in M_i$ we have

$$g(m^*_{\theta'}) = g(m'_i, m^*_{\theta' -i}) \tag{6}$$

then strong continuous implementation fails.

Proof. Fix the equilibrium σ and assume that there exists a state θ' such that

$$g(m^*_{\theta'}) = g(m'_i, m^*_{\theta' -i}) = \theta' \tag{7}$$

for all agents $i \in N$ and messages $m'_i \in M_i$. σ being a BNE implies that at the complete information type profile $t^{\theta'}$

$$b_i(m^*_{\theta'}) \times p_{g(m^*_{\theta'})\theta'} \geq b_i(m'_i, m^*_{\theta' -i}) \times p_{g(m'_i, m^*_{\theta' -i})\theta'} \tag{8}$$

By assumption we have $g(m^*_\theta) = g(m'_i, m^*_{\theta -i}) = \theta'$ for all m'_i which in turn implies

$$b_i(m^*_{\theta'}) \geq b_i(m'_i, m^*_{\theta -i}) \tag{9}$$

It follows that $\sigma(t) = m^*_{\theta'}$ for all $t \in \mathcal{T}$ is a continuous BNE. Since $g(m^*_{\theta'}) \neq \theta$ for all $\theta \neq \theta'$ strong continuous implementation fails. □

We can now present the proof of Theorem 2.

A.1 Proof of Theorem 2

Proof. Suppose that $\mathcal{O}$ satisfies strong continuous implementation. Denote by σ^* the truthful continuous BNE with $\sigma^*_i(t^\theta) = \theta$ for all complete information type profiles, and for all agents $i \in N$. Let $\Theta = 0, 1$. Thus we have $\sigma^*_i(t^1) = 1$ and $\sigma^*_i(t^0) = 0$ for all $i \in N$. Denote by $\mathbf{1}$ and $\mathbf{0}$ the profile where all agents report 1 respectively 0. By Lemma 1,

it follows that for the continuous BNE σ^* there exists an agent i_1 respectively i_0 and messages m_{i_1} respectively m_{i_0} such that

$$\begin{aligned} g(m_{i_1}, \mathbf{1}_{-i_1}) &\neq 1 \\ g(m_{i_0}, \mathbf{0}_{-i_0}) &\neq 0 \end{aligned} \tag{10}$$

As the state space is binary we have

$$\begin{aligned} g(m_{i_1}, \mathbf{1}_{-i_1}) &= 0 \\ g(m_{i_0}, \mathbf{0}_{-i_0}) &= 1 \end{aligned} \tag{11}$$

Monotonicty implies

$$\begin{aligned} g(0, \mathbf{1}_{-i_1}) &= 0 \\ g(1, \mathbf{0}_{-i_0}) &= 1 \end{aligned} \tag{12}$$

Consider a message profile m' such that $m'_{i_0} = 1$ and $m'_{i_1} = 0$ and $m'_j \neq m'_k$ for two agents $j, k \in N \setminus \{i_0, i_1\}$. Without loss of generality assume that $g(m') = 0$. Consider a profile m'' such that $m''_j = 0$ for all $j \neq i_0$ and $m''_{i_0} = 1$. Monotonicty implies $g(m'') = 0$ establishing a contradiction with equation (9). This implies that $i_0 = i_1 = i^*$. To show that agent i^* is a dictator, assume that there exists a message profile m''' such that $g(m''') \neq m'''_{i^*}$. Without loss of generality let $g(m''') = 0$. Now consider a profile $\overline{m}$ such that $\overline{m}_j = 0$ for all $j \neq i^*$ and $\overline{m}_{i^*} = 1$. Monotonicity and $g(m''') = 0$ imply that $g(\overline{m}) = 0$ which contradicts equation (9). □

B Proof of Theorem 3

Proof. We will prove Theorem 3 in three steps.

1. As $\mathcal{O}$ satisfies strong continuous implementation for all θ we have $g(\boldsymbol{\theta}) = \theta$.

2. By Lemma 1, we have that for each state and corresponding consensus report profile $\boldsymbol{\theta}$ there exists an agent i_θ and message m_{i_θ} such that $g(m_{i_\theta}, \boldsymbol{\theta}_{-i_\theta}) \neq \theta$. Fix a state θ' and consider agent $i_{\theta'}$. Let

$$g(m_{i_{\theta'}}, \boldsymbol{\theta'}_{-i_{\theta'}}) = \theta'' \tag{13}$$

Label neutrality then implies that for all profiles m with $m_{i_{\theta'}} \in \Theta$ and $m_{-i_{\theta'}} = \boldsymbol{\theta}, \theta \in \Theta$ we have

$$g(m_{i_{\theta'}}, m_{-i_{\theta'}}) = m_{i_{\theta'}} \tag{14}$$

3. We show that agent $i_{\theta'}$ is a dictator. Suppose not, i.e., there exist a profile m' such that $g(m') = \theta'' \neq m'_{i_{\theta'}}$. By monotonicity it follows that for profile m'' with $m''_{-i_{\theta'}} = \boldsymbol{\theta''}$ and $m''_{i_{\theta'}} = m'_{i_{\theta'}}$ we have $g(m'') = \theta''$ contradicting step 2.

□

C Proof of Proposition 2

Proof. Assume that $\mathcal{O}$ satisfies strong continuous implementation. By definition, there exists a "truth-telling" BNE σ^* such that $\sigma^*(t^\theta) = \boldsymbol{\theta}$ for all $\theta \in \Theta$. By Lemma 1 it follows that if

$$g(\boldsymbol{\theta}) = g(m'_i, \boldsymbol{\theta}_{-i}) \tag{15}$$

for all $i \in N$ and for all $m'_i \in M$ then strong continuous implementation fails, establishing the result. □

D Proof of Theorem 5

Proof. We consider the sequential-voting oracle mechanism $\mathcal{O}^S$ defined above. By two steps, we prove that $\mathcal{O}^S$ achieves strong sequential-continuous implementation. First, we show that, in any PBE σ, we have $\sigma_i(t_\theta, h_i^{\sigma-\theta}) = \theta$ for every $\theta \in \Theta$ and every agent $i \in N$, and as a result, we have $g(\sigma((t_\theta, h^{\sigma-\theta})) = \theta$ for every $\theta \in \Theta$. Second, we prove existence of a continuous PBE, and therefore, $\mathcal{O}^*$ achieves strong sequential-continuous implementation.

Fix any PBE σ and any state $\theta \in \Theta$. We show that type t_θ of every agent vote for θ in the equilibrium path of σ, i.e., $\sigma_i(t_\theta, h_i^{\sigma-\theta}) = \theta$ for every $i \in N$. There are three possible payoffs for each agent: (i) an agent votes for a state which is not chosen by $\mathcal{O}^*$, (ii) an agent votes for $\theta' \neq \theta$ and θ' is chosen by $\mathcal{O}^*$, and (iii) an agent votes for θ and θ is chosen by $\mathcal{O}^*$, with payoffs 0, $p_{\theta'\theta}$, $p_{\theta\theta}$, respectively, and $0 < p_{\theta'\theta} < p_{\theta\theta}$. Consider the path along which type t_θ of every agent votes for θ. We apply backward induction:

at period n, given that all the other agents have voted for θ, it is a unique best reply for agent i_n to vote for θ and gets $p_{\theta,\theta}$, because, by deviating to vote for $\theta' \neq \theta$, i_n gets either $p_{\theta',\theta}$ or 0; ...; at each period $k \in \{1, 2, ..., n-1\}$, given that agents $i_1, ..., i_{k-1}$ have voted for θ, it is a unique best reply for agent i_k to vote for θ and gets $p_{\theta,\theta}$, because, by deviating to vote for $\theta' \neq \theta$, i_k gets either $p_{\theta',\theta}$ or 0. This completes Step 1.

Second, we construct a continuous PBE σ^*. For each agent $i_k \in N$, define

$$\widetilde{H}_{i_k}^{\theta} \equiv \left\{ h_{i_k} \in H_{i_k} : g\left[h_{i_k}, \left(v_{i_{\widetilde{k}}} = \theta \right)_{\widetilde{k} \in \{k, k+1, ..., n\}} \right] = \theta \right\},$$

i.e., $\widetilde{H}_{i_k}$ is the set of i_k's histories such that θ could still be chosen by $\mathcal{O}^*$. Furthermore, for each $h_{i_k} = \left(v_{i_1}, ..., v_{i_{k-1}} \right) \in H_{i_k}$, define

$$\Theta^{h_{i_k}} \equiv \left\{ \widetilde{\theta} \in \Theta : \begin{array}{l} \forall \theta' \in \Theta, \\ \left| \left\{ \widetilde{k} \in \{1, ..., k-1\} : v_{i_{\widetilde{k}}} = \widetilde{\theta} \right\} \right| \geq \left| \left\{ \widetilde{k} \in \{1, ..., k-1\} : v_{i_k} = \theta' \right\} \right| \end{array} \right\},$$

i.e., $\Theta^{h_{i_k}}$ is the set of states which get most votes under h_{i_k}. The following is a PBE for complete information.

$$\widetilde{\sigma}_{i_k}(t_{i_k} = t_\theta, h_{i_k}) = \left\{ \begin{array}{ll} \theta, & \text{if } h_{i_k} \in \widetilde{H}_{i_k}^{\theta}, \\ \varphi\left(\Theta^{h_{i_k}} \right), & \text{if } h_{i_k} \notin \widetilde{H}_{i_k}^{\theta}, \end{array} \right. , \forall i_k \in N, \forall h_{i_k} \in H_{i_k},$$

where φ is any selection function, i.e., $\varphi : 2^{\Theta} \diagdown \{\varnothing\} \longrightarrow \Theta$ with $\varphi(E) \in E$ for any $E \in 2^{\Theta} \diagdown \{\varnothing\}$.

By the same backward induction argument as above, $\widetilde{\sigma} \equiv \left(\widetilde{\sigma}_{i_k} \right)_{i_k \in N}$ is a PBE for complete information, and $\widetilde{\sigma}_{i_k}(t_\theta, h_{i_k})$ is a strict best reply given type t_θ of the other players following $\widetilde{\sigma}$. When we apply a backward induction argument, the strict best replies remain best replies when we perturb types slightly. Specifically, define

$$\begin{aligned} T_{i_n}^{\theta\text{-}\varepsilon} &\equiv \left\{ t_{i_n} \in T_{i_n} : d^1\left(t_{i_n}, t_\theta \right) < \varepsilon \right\}, \\ T_{i_{n-1}}^{\theta\text{-}\varepsilon} &\equiv \left\{ t_{i_{n-1}} \in T_{i_{n-1}} : d^2\left(t_{i_{n-1}}, t_\theta \right) < \varepsilon \right\}, \\ &... \\ T_{i_1}^{\theta\text{-}\varepsilon} &\equiv \left\{ t_{i_1} \in T_{i_1} : d^n\left(t_{i_1}, t_\theta \right) < \varepsilon \right\}, \end{aligned}$$

and

$$\widetilde{\sigma}_{i_k}(t_{i_k}, h_{i_k}) = \begin{cases} \theta, & \text{if } h_{i_k} \in \widetilde{H}^{\theta}_{i_k}, \\ \varphi\left(\Theta^{h_{i_k}}\right), & \text{if } h_{i_k} \notin \widetilde{H}^{\theta}_{i_k}, \end{cases} \quad \forall i_k \in N, \forall h_{i_k} \in H_{i_k}, \forall t_{i_k} \in T^{\theta\text{-}\varepsilon}_{i_k}.$$

By the same backward induction argument as above, we can find sufficiently small $\varepsilon > 0$, such that $\widetilde{\sigma}_{i_k}(t_{i_k}, h_{i_k})$ is a best reply for $(t_{i_k}, h_{i_k}) \in T^{\theta\text{-}\varepsilon}_{i_k} \times H_{i_k}$, if types in $\times_{i \in N} T^{\theta\text{-}\varepsilon}_{i}$ follow $\widetilde{\sigma}$.

Finally, we consider an auxiliary game, in which types types in $\times_{i \in N} T^{\theta\text{-}\varepsilon}_{i}$ follow $\widetilde{\sigma}$, and types not in $\times_{i \in N} T^{\theta\text{-}\varepsilon}_{i}$ play the game defined by $\mathcal{O}^S$. Since this is a finite-action game, a PBE exists, which is denoted by $\widehat{\sigma}$. We then define σ^* on the Harsanyi type space $\mathcal{T} \equiv (T, \kappa)$ as follows.

$$\sigma^*_{i_k}(t_{i_k}, h_{i_k}) = \begin{cases} \widetilde{\sigma}_{i_k}(t_{i_k}, h_{i_k}), & \text{if } t_{i_k} \in T^{\theta\text{-}\varepsilon}_{i_k}, \\ \widehat{\sigma}_{i_k}(t_{i_k}, h_{i_k}), & \text{if } t_{i_k} \in T_{i_k} \diagdown T^{\theta\text{-}\varepsilon}_{i_k}, \end{cases} \quad \forall i_k \in N, \forall h_{i_k} \in H_{i_k}.$$

With $t_{i_k} \in T^{\theta\text{-}\varepsilon}_{i_k}$, incentive compatibility of $\sigma^*_{i_k}(t_{i_k}, h_{i_k})$ follows from the backward induction argument above, and with $t_{i_k} \in T_{i_k} \diagdown T^{\theta\text{-}\varepsilon}_{i_k}$, incentive compatibility of $\sigma^*_{i_k}(t_{i_k}, h_{i_k})$ follows from $\widetilde{\sigma}$ being a PBE of the auxiliary game. Therefore, $\sigma^* \equiv \left(\sigma^*_{i_k}\right)_{i_k \in N}$ is a continuous PBE. □