Blockchain Economics: Online Appendix*

Joseph Abadi† Markus Brunnermeier‡

January 3, 2022

Abstract

In the Online Appendix, we discuss our model's assumptions (in Appendix D), prove the existence result in the Blockchain Trilemma (in Appendix G), and prove results in Sections 5.2 and 6.2 (in Appendices H and I, respectively). Mostly, we use the formal framework provided in Appendix A of the paper. Given that we prove some results in extensions of our model, though, we first slightly generalize our framework in Appendix E. Then, we present the consensus algorithms that we use in our proofs in Appendix F. We then proceed with our proofs.

*__Disclaimer__: The views expressed in this paper are solely those of the authors and do not necessarily reflect the views of the Federal Reserve Bank of Philadelphia or the Federal Reserve System. Any errors or omissions are the responsibility of the authors.

†Federal Reserve Bank of Philadelphia

‡Princeton University

Contents

D Discussion of model assumptions

In this section, we provide a more thorough discussion of our model's assumptions, which has been postponed up until this point. We discuss three key issues: the model's payoff environment and assumptions on preferences, our interpretation of faulty agents, and the need for the specific equilibrium concept that we use.

D.1 Assumptions on payoffs and preferences

In understanding how our static model with quasilinear payoffs maps to record-keeping, it is useful to think of agents' ledger balances as payoffs in a continuation game (as in Abreu, Pearce, and Stacchetti, 1990). When agents reach consensus on an outcome, they each have a piece of information (an update to the ledger) informing them of how the continuation game is to be played. That is, the process of reaching consensus can be thought of as a "pre-play" communication phase in a dynamic game (e.g. Aumann and Hart, 2004). It may seem odd, then, that we restrict to a quasilinear environment in which the Pareto frontier of payoffs in this continuation game is a simplex,

$$\mathcal{V} = \{\tilde{\mathbf{v}} \in \mathbb{N}^{\mathcal{N}} \mid \sum_{n \in \mathcal{N}} \tilde{v}_n \leq \sum_{n \in \mathcal{N}} v_n\}.$$

Typically the Pareto frontier is not restricted to be of this form. We make the assumption of quasilinear preferences only for notational simplicity, though. All of our results extend to a setting in which agents' preferences over ledger balances are instead described by increasing functions $g_n(v_n + t_n)$, so that the Pareto frontier of ledger payoffs can take an arbitrary form.

In Section 2.4, we also impose additional assumptions on preferences. Assumption 2 should be thought of as effectively restricting to environments in which agents engage in mutually beneficial transactions. The first condition in that assumption implies that all possible transfers of value are in fact part of an individually rational outcome under some conditions. This gives the designer a reason to attempt to implement all possible transfers of value among agents. More generally, the designer may not know precisely which class of mechanisms the record-keeping system will implement. In order to render the record-keeping system as useful as possible, the designer must ensure that any possible transfer of value can be achieved. The need for the second condition in Assumption 2 can be understood through the double-spend lemma. It implies that there exists a state of the world in which the deviating coalition has beneficial transactions that can be realized with two distinct subsets of agents. Otherwise, it could be that a double-spend is never beneficial for any agent, neutralizing the need to provide incentives for honesty.

We proceed with two examples in order to illustrate the content of this assumptions: one in which Assumption 2 is satisfied, and one in which Assumption 2 (and, indeed, the Blockchain Trilemma) fails.

Example 1: A production economy. Suppose that each agent n produces a differentiated good at a unit cost. A transaction y is a sale of j units of a good from a seller n to a buyer n' and a

transfer of balances $t(y) = j$ from the buyer to the seller. Each agent n has a demand for $J_{n,n'}$ units of the good produced by agent n', where $J_{n,n'}$ is an arbitrary integer. When an agent's demand for good n' is $J_{n,n'}$, that agent receives marginal utility $u > 1$ for the first $J_{n,n'}$ units consumed and zero for each additional unit in excess of $J_{n,n'}$. Hence, agent n's preferences can be described by a type $\theta_n = \{J_{n,n'}\}_{n' \in \mathcal{N}}$, and the state of the world is $\theta = \{\theta_n\}_{n \in \mathcal{N}}$. Sellers produce differentiated goods (indexed by the seller's identity n) at a unit cost. It is easy to check that Assumption 2 is satisfied: for each possible transfer $\mathbf{t}$, there is an outcome in which that transfer is realized: the quantity $j_{n,n'}$ of good n' bought by agent n simply needs to be set equal to the required transfer from n to n'. Furthermore, for each possible set of transactions, there is a state in which precisely those transactions are individually rational. Therefore, the Blockchain Trilemma holds in this environment.

Example 2: The Byzantine Generals problem. Suppose that the agents form an army that wishes to successfully attack an enemy encampment, which lies somewhere in a set Y of possible locations. One of the agents is a general, and the others are lieutenants ($\theta_n = L$). Only the general knows the location of the enemy encampment, so the general's type is $\theta_n = (G, y)$ for some $y \in Y$. All agents can choose to either attack a particular location $y \in Y$ or retreat. A "transaction" in this environment is therefore a pair (S, y_n), where $y \in Y \cup \{r\}$ denotes a location that the set of agents S agreed to attack (or $y = r$ if agents in S retreated). If at least J agents attack the location of the enemy encampment, all the attackers receive utility $u > 0$. An agent who chooses to retreat receives a utility of 0. An agent who attacks an incorrect location, or a correct location that fewer than $J - 1$ others attacked, suffers a utility loss of $\ell > 0$.

This environment is similar to that in the Byzantine Generals problem that originally appeared in Lamport, Shostak, and Pease (1982), which was also analyzed in a game-theoretic context by Rubinstein (1989). Transfers of ledger balances are not needed in this environment to achieve efficient outcomes, so Assumption 2 fails. It is indeed possible to design a consensus algorithm that achieves fault-tolerance, resource-efficiency, and full transferability. Intuitively, agents play a coordination game, so they would never want to deceive one another. Therefore, there is no need to provide incentives for agents to communicate honestly.

D.2 Faulty agents and communication frictions

The two key communication frictions in our model are (1) the possibility of delays in communication, and (2) the presence of faulty nodes. In reality, of course, all communication networks are subject to lags in communication, so the first friction in Assumption 1 is relatively innocuous. Our assumptions about faultiness are, in some cases, more substantial.

We state all of our benchmark results in a setting where faulty nodes do not communicate (Assumption 1), so faulty nodes can be interpreted as computers that are offline or have suffered a crash. However, we also extend the model to account for arbitrary behavior by faulty nodes (Assumption 1'). A richer set of deviations from the communication protocol could represent glitches or even a situation in which a malicious attacker has taken control of some of the network's

computers. Both the assumptions of "shutdown" faults and arbitrary faults are typical in the computer science literature.

In our setting, the interpretation of faults as crashes, glitches, or malicious attacks are all useful. It is obvious that any sensible communication protocol should be required to be robust to crashes or shutdowns. It is reasonable (and necessary) to assume that in large networks (e.g. the internet) participants will occasionally shut down their devices. A communication protocol that is not robust to shutdowns would almost certainly fail in any large-scale setting. Moreover, a node's failure to communicate could also reflect a situation in which a potential user of the network has not yet joined, which must also be accommodated (unless the set of network participants is to remain fixed after its inception).

The assumptions that glitches or malicious attacks may occur are stronger. A glitch can be interpreted as a situation in which a node sends random messages, without any particular intent to crash the network. Imposing this assumption also seems reasonable in a setting in which programming errors may cause some computers in the network to act incorrectly. Intuitively, other network participants who program their devices correctly should not be punished too severely (through a breakdown in consensus) if some other participants make a programming mistake.

Accounting for the possibility of a malicious attack is the strongest robustness assumption that can be made. In an economic model, it may be desirable to account for maliciousness as a way of expressing the idea that some agents' preferences over outcomes may not actually be known. An agent could, for example, make a bet with others outside of the record-keeping system that the digital ledger will cease to function. This would provide that agent with an incentive to collude with others in order to destroy the network, and the fault-tolerance requirement in this case represents the desire to have a record-keeping system that is robust to such side bets.

Importantly, our impossibility result rests only on the weak assumption that computers may fail to communicate. By contrast, we are able to prove our existence results (in Propositions 1 and 2) under the much stronger assumption that attacks on the network may be malicious. Therefore, in making these assumptions, we actually prove our results under the most stringent possible conditions.

D.3 Equilibrium concept

While our equilibrium concept may seem extremely strong, it is necessary to understand the difficulties in digital record-keeping. It is easy to design communication protocols that entirely prevent coalitional deviations when faults do not have to be considered. This is proven by the existence result of the Blockchain Trilemma: in the absence of a fault-tolerance requirement, it is possible to achieve full transferability and resource-efficiency.

Additionally, we have argued that in a digital environment, it is crucial to consider the possibility of faults, since it is impossible to guarantee that all computers in a network will operate correctly. As described previously, our equilibrium concept also permits agents to entertain heterogeneous beliefs about the identities of faulty agents. This requirement is also natural: in reality, it would be

unreasonable to require that agents have common information about who is offline and who is not. It would be quite difficult for the designer to determine the set of possible beliefs held by agents, so a robust approach (rather than a Bayesian equilibrium concept) is appropriate.

Likewise, if we drop the coalition-proofness requirement, it becomes easy to design fault-tolerant communication protocols. For example, consider the following system: there are two trusted record-keepers, n_1^* and n_2^*. All non-faulty agents initially send a message to both record-keepers in order to indicate they are non-faulty. Then, n_1^* and n_2^* communicate with each other to reach agreement on an outcome. For example, they could agree on the outcome $x = F(\theta, S)$, where S represents the set of agents who reported to both record-keepers. Finally, an agent decides on outcome x only if n_1^* and n_2^* report the same outcome x.

In this type of record-keeping system, there is no unilateral deviation that will permit n_1^* or n_2^* to do anything other than prevent consensus from ever taking place. There is no way for either record-keeper to double-spend, since a deviation by one record-keeper to attempt to generate consensus on two different outcomes will be foiled by the other record-keeper, who will not go along with the deviation. It is therefore trivially easy to design a consensus algorithm that entirely prevents dishonesty if we drop the coalition-proofness requirement.

Given the efforts put into the design of elaborate consensus algorithms in reality, it would seem that at least the designers of such systems bear in mind the possibility of coalitional deviations. Bitcoin's consensus algorithm, for example, is designed to dissuade "51% attacks" in which one entity comes to be in control of a majority of the network's computing power. Proof-of-stake consensus algorithms, similarly, are designed to prevent deviations by two-thirds of "validators," who are the agents responsible for approving blocks added to the blockchain.

E Preliminaries

We begin by setting up some formal details and notation that we will use in the constructions of consensus algorithms and the proofs. First, we describe the types of information that agents might have. The state of the world is $\theta = (\tilde{\theta}_1, \ldots, \tilde{\theta}_n)$, where $\tilde{\theta}_n \in \tilde{\Theta}_n$ (where $\tilde{\Theta}_n$ is finite) describes agent n's preferences. In our benchmark model, θ is public knowledge, whereas in our extensions, we permit each agent n to have private information about $\tilde{\theta}_n$. The designer may also use verifiable private randomization (in the form of signals ϕ_n in a finite set Φ of arbitrary size), so agent n's information is described by $\theta_n = (\tilde{\theta}_n, \phi_n)$.[1] The state of the world is distributed according to a full-support distribution $G_\Theta \in \Delta\Theta = \prod_{n \in \mathcal{N}} \tilde{\Theta}_n$, and the distribution $G_\Phi \in \Delta\Phi$ of the private signals is chosen by the designer. We let $\Theta_n = \tilde{\Theta}_n \times \Phi$ denote the possible initial information sets of agent n. Given this definition of agents' information θ_n, we can then apply all of the definitions in Appendix A of the paper for nodes' behaviors, equilibrium, consensus sets, and the desired features

[1] Feldman and Micali (1988) demonstrate how the designer could provide agents with a communication protocol that permits them to replicate verifiable randomization, even if agents themselves do not have access to a source of public randomization.

of a consensus algorithm.

We also define a notion of a social choice function.

Definition E.1. *A **social choice function** $\mathcal{F}$ is a collection of maps $F_S : \prod_{n \in S} \Theta_n \to \mathcal{X}_S$ for each $S \subset \mathcal{N}$.*

We must specify a map from information to outcomes for each subset of agents S because faulty agents might fail to share their information, so when a set of agents S are non-faulty, the outcome on which they reach consensus should not depend on the information held by faulty agents $\mathcal{N} - S$. Given that we allow for public randomization whose inputs is incorporated in the outcome specified by the social choice function, our notion of a social choice function is actually similar to the usual notion of a social choice correspondence, since a single profile of preferences can map to multiple different outcomes.

In what follows, it will occasionally be useful to define an auxiliary type θ^F for each agent n, corresponding to a situation in which n is faulty. We then denote the set of possible states, when this auxiliary type is added, by $\tilde{\Theta} = \prod_{n \in \mathcal{N}} (\Theta_n \cup \{\theta^F\})$. A profile of types will be denoted $\boldsymbol{\theta} \in \tilde{\Theta}$, the restriction of a profile $\boldsymbol{\theta}$ to agents in S will be denoted $\boldsymbol{\theta}_S = \{\theta_n\}_{n \in S}$, and it will be generally understood that

$$F(\boldsymbol{\theta}) \equiv F_S(\boldsymbol{\theta}_S) \text{ if } \theta_n = \theta^F \ \forall \ n \notin S.$$

We denote the set of agents n such that $\boldsymbol{\theta}_n \neq \theta^F$ by $S(\boldsymbol{\theta})$. The problem of reaching consensus is simply the problem of reaching an agreement on some value $\boldsymbol{\theta} \in \tilde{\Theta}$. Our consensus algorithms will specify how agents are to reach agreement on $\boldsymbol{\theta}$ without specifying the social choice function, which will be specific to each of our proofs.

Finally, it will sometimes be convenient for us to denote a vector of ones with length $|\mathcal{N}|$ by $\mathbf{1}^{\mathcal{N}}$ and a vector of length $|\mathcal{N}|$ in which all entries are equal to θ^F by $\boldsymbol{\theta}^F = \theta^F \cdot \mathbf{1}^{\mathcal{N}}$.

F Consensus algorithms in the Trilemma

This section describes consensus algorithms used in our benchmark result and in the proofs related to the Trilemma. The constructions are involved but well-known in the computer science literature. We additionally prove the algorithms' correctness and characterize the types of deviations that are possible for coalitions of strategic agents.

F.1 The shutdown fault algorithm

In this section, we present an asynchronous consensus algorithm that is appropriate for our benchmark model, in which faulty nodes simply do not communicate. The algorithm is based loosely on the filter algorithm presented in Toueg (1983) and the asynchronous consensus algorithm presented in Section 2 of Bracha and Toueg (1985). However, since we later adapt the consensus

algorithm for the case of malicious faults in Bracha and Toueg (1985), we do not call the algorithm in this section the Bracha-Toueg algorithm.

The algorithm consists of two consecutive sub-algorithms: the filter algorithm and the majority vote algorithm, which we describe below. Nodes will internally keep track of *values* $\hat{\boldsymbol{\theta}}^n \in \tilde{\Theta}$. The purpose of the filter algorithm is to narrow the possible range of outcomes available to nodes, so that in the majority vote algorithm, each makes only a binary choice. After both algorithms are completed, each node will have a terminal value $\hat{\boldsymbol{\theta}}^n$. In this section, we will prove properties of nodes' terminal values under this algorithm. In the proof of the existence result of the Blockchain Trilemma, we will describe how a node's terminal value translates into a decision made by that node.

The filter algorithm: The message vocabulary for the filter algorithm has three types of messages: type broadcasts of the form $n : \theta$, indicating that agent n declares her type to be θ, profile broadcasts of the form $n : \hat{\boldsymbol{\theta}}$, indicating that agent n declares her node's current value to be the profile of types $\hat{\boldsymbol{\theta}}$, and echoes, which have the same form as profile broadcasts. The space of messages for the filter algorithm is therefore

$$\mathcal{M}^F = \mathcal{M}^T \cup \mathcal{M}^P \cup \mathcal{M}^E,$$

$$\mathcal{M}^T = \{n : \theta \mid n \in \mathcal{N}, \theta \in \Theta_n\}, \mathcal{M}^P = \{n : \hat{\boldsymbol{\theta}} \mid n \in \mathcal{N}, \hat{\boldsymbol{\theta}} \in \tilde{\Theta}\},\ \mathcal{M}^E = \{e : m \mid m \in \mathcal{M}^P\}$$

We also need to specify the messages that nodes are permitted to send at each information set (the set $M_n(H_{nk})$). A node may not send a type or profile broadcast with a signature different from its own. Furthermore, node n may only send a profile broadcast $\hat{\boldsymbol{\theta}} = \{\hat{\theta}_{n'}\}_{n' \in \mathcal{N}}$ if, for all n' such that $\hat{\theta_{n'}} \neq \theta^F$, node n previously received a type broadcast $n' : \hat{\theta}_n$. The requirements to send an echo are precisely the same as those to send the corresponding profile broadcast.

We outline the algorithm below.

Algorithm F.1 (Filter algorithm (J) in the shutdown case)**.** *Each node n initializes its value to $\hat{\theta}^n_{n'} = \theta^F$ for all n' and communicates using the following protocol.*

1. *Node n sends a type broadcast $n : \theta_n$ to all other nodes.*

   - *Whenever n receives a type broadcast $n' : \theta_{n'}$, it updates its value by setting $\hat{\theta}^n_{n'} = \theta'_n$. It ignores any subsequent type broadcasts from the same node.*
   - *Node n moves to the next step after it has received at least $N - J$ type broadcasts.*

2. *Node n sends a profile broadcast $n : \hat{\boldsymbol{\theta}}^n$ to all other nodes.*

   - *After node n receives profile broadcasts from at least $N - J$ other nodes (of the form $n' : \hat{\boldsymbol{\theta}}^{n'}$), it sets*

$$\hat{\boldsymbol{\theta}}^n = \begin{cases} \hat{\boldsymbol{\theta}}^* & \text{a majority of received profile broadcasts are equal to some } \hat{\boldsymbol{\theta}}^* \\ \theta^F \cdot \mathbf{1}^{\mathcal{N}} & \text{there is no majority value} \end{cases}$$

(where $\mathbf{1}^{\mathcal{N}}$ denotes a vector of N ones) and moves to the next step.

3. *Node n sends an echo $e : n : \hat{\boldsymbol{\theta}}^n$ to all other nodes.*

- *After receiving echoes from at least $N - J$ other nodes, node n sets a final value*

$$\hat{\boldsymbol{\theta}}^n = \begin{cases} \hat{\boldsymbol{\theta}}^* & \text{all received echoes are equal to some } \hat{\boldsymbol{\theta}}^* \\ \theta^F \cdot \mathbf{1}^{\mathcal{N}} & \text{there is no unanimous value} \end{cases}$$

We now prove that after running the filter algorithm, when there are at most $J \leq \lfloor \frac{N}{2} \rfloor$ faulty nodes, then each node either has some value $\hat{\boldsymbol{\theta}}^* \in \tilde{\Theta}$ or the value $\theta^F \cdot \mathbf{1}^{\mathcal{N}}$.

Lemma F.1. *Suppose Assumption 1 holds and that there are at most $J \leq \lfloor \frac{N}{2} \rfloor$ faulty nodes. Then if all non-faulty nodes follow the communication protocol in Algorithm F.1, (1) the algorithm eventually terminates, and (2) there exists $\hat{\boldsymbol{\theta}}^* \in \prod_{n \in \mathcal{N}} \tilde{\Theta}_n$ such that $\hat{\boldsymbol{\theta}}^n \in \{\theta^F \cdot \mathbf{1}^{\mathcal{N}}, \hat{\boldsymbol{\theta}}^*\}$ for all non-faulty nodes n.*

Furthermore, for all n such that $\hat{\boldsymbol{\theta}}^* \neq \theta^F$, $\hat{\boldsymbol{\theta}}^* = \theta_n$, and any such value $\hat{\boldsymbol{\theta}}^*$ with at most J entries equal to θ^F is reached with positive probability.

Proof. To see that the algorithm eventually terminates, note that in each step, a node must receive messages from $N - J$ nodes in order to proceed to the next step. Since at most J nodes are faulty, and messages are delivered with some maximum lag Δ, a node will always terminate the protocol so long as all $N - J$ non-faulty nodes communicate according to it.

Next, suppose that two non-faulty nodes n, n' have distinct values $\hat{\boldsymbol{\theta}}^n \neq \hat{\boldsymbol{\theta}}^{n'}$ that are not equal to $\theta^F \cdot \mathbf{1}^{\mathcal{N}}$. This implies that in the third step of Algorithm F.1, n received at least $N - J$ echoes of $\hat{\boldsymbol{\theta}}^n$, while n' received at least $N - J$ echoes of $\hat{\boldsymbol{\theta}}^{n'}$. Since $2(N - J) > N$, it must be that some non-faulty node sent n and n' different echoes, which is impossible if all non-faulty nodes follow the protocol.

To prove the second part of the lemma, first note that the instructions in the first step imply that no matter what node n's value $\hat{\boldsymbol{\theta}}^n$ is, it will be the case that $\hat{\boldsymbol{\theta}}^n_{n'} \in \{\theta_{n'}, \theta^F\}$ for each n'. Then, note that with positive probability, the following happens: some subset of nodes S with $|S| \geq N - J$ send each other type messages that are received by all other nodes in S in the second round, but no node in S receives a message from a node not in S, so that all nodes in $n \in S$ have a value $\hat{\boldsymbol{\theta}}^*$ such that $\hat{\boldsymbol{\theta}}^n_{n'} = \theta_{n'}$ for all $n' \in S$ and $\hat{\boldsymbol{\theta}}^n_{n'} = \theta^F$ for $n' \notin S$. Since $N - J > \frac{N}{2}$, it must be that in step 2, *any* node following the protocol will terminate the second step with value $\hat{\boldsymbol{\theta}}^*$, meaning that this value will be chosen by all nodes that follow the protocol in Step 3. □

The majority rule algorithm: In the majority rule algorithm, each node keeps track of a value $\hat{\boldsymbol{\theta}}^n \in \tilde{\Theta}$, a *phase number* t, and a *cardinality* q (which is a positive integer). Each message also has the same form: a message is $m = t : q : \hat{\boldsymbol{\theta}}$, where q is the message's cardinality, t is its phase number, and $\hat{\boldsymbol{\theta}}$ is its value. Hence, the message space is

$$\mathcal{M}^{\text{maj}} = \{t : q : \hat{\boldsymbol{\theta}} \mid t \in \mathbb{N}, q \in \mathbb{N}, \hat{\boldsymbol{\theta}} \in \tilde{\Theta}\}.$$

A node can send a message $m = t : q : \hat{\boldsymbol{\theta}}_n$ only if it received at least q messages of the form $t' : q' : \hat{\boldsymbol{\theta}}$ previously. We proceed with the description of the algorithm.

Algorithm F.2 (Majority rule algorithm (J))**.** *Nodes begin with the value $\hat{\boldsymbol{\theta}}^n$ that they computed in the Filter (J) algorithm, phase $t = 0$, and cardinality $q = 1$. They communicate according to the following protocol.*

1. *Node n sends message $t : q : \hat{\boldsymbol{\theta}}^n$ to all other nodes, where t represents its current phase, q represents its current cardinality, and $\hat{\boldsymbol{\theta}}^n$ is its current value.*

   - *Once n has received $N - J$ messages during its current phase, if it received a message for a value $\hat{\boldsymbol{\theta}}^n$ with cardinality $q > \frac{N}{2}$ (called a **witness** for $\hat{\boldsymbol{\theta}}$), and it only received such a message for one value, it updates its value to $\hat{\boldsymbol{\theta}}^n$.*
   - *Otherwise, it updates its value $\hat{\boldsymbol{\theta}}^n$ to the majority among those values (if one exists) or $\theta^F \cdot \mathbf{1}^{\mathcal{N}}$ (if not).*
   - *Node n then updates its phase to $t + 1$ and sets its cardinality q equal to the number of messages with value $\hat{\boldsymbol{\theta}}^n$ that it received during phase t.*

2. *If, in phase t, node n received more than J witnesses for a value $\hat{\boldsymbol{\theta}}$, it stops updating its phase, value and cardinality (but continues to send messages). Otherwise, it goes back to the beginning of step 1.*

We now prove that (1) when all non-faulty nodes follow the protocol, and there are at most J faults, it terminates in finite time, (2) all nodes that follow the protocol reach agreement on a value consistent with their types, and (3) all such values are reached with positive probability.

Proposition F.1. *Suppose that Assumption 1 holds and that there are at most J faulty nodes. Then if all non-faulty nodes follow Algorithm F.1 followed by Algorithm F.2, the communication protocol terminates in finite time, all non-faulty nodes terminate with a value consistent with their types, and all such values are reached with positive probability.*

Proof. This proof is adapted from Bracha and Toueg (1985). We first define some terms for what follows. A message of the form $t : q : \hat{\boldsymbol{\theta}}$ is called a t-message for $\hat{\boldsymbol{\theta}}$, and if $q > \frac{N}{2}$, it is called a t-witness. A node is said to *decide* in phase t if its phase is t when it finishes Step 2 of Algorithm F.2.

We prove the theorem by showing the protocol's consistency, then show that non-faulty nodes reach agreement on all possible values consistent with their types. Then we prove two properties (deadlock-freedom and convergence) that together prove the algorithm terminates in finite time.

Consistency: Let t be the smallest phase in which a node decides. We prove that in phase t, there cannot be two nodes n and n' that each have witnesses for different values. We proceed by contradiction. Suppose that n received a witness in t for $\hat{\boldsymbol{\theta}}^*$, while n' received a witness in t for $\boldsymbol{\theta}^F \equiv \theta^F \cdot \mathbf{1}^{\mathcal{N}}$. This means that n received a $t-1$-witness from some n'' for $\hat{\boldsymbol{\theta}}^*$, while q

received a $t-1$-witness for $\boldsymbol{\theta}^F$ from some n'''. In turn, this implies that n received more than $\frac{N}{2}$ $t-2$-messages for $\hat{\boldsymbol{\theta}}^*$, while n''' received more than $\frac{N}{2}$ $t-2$-messages for $\boldsymbol{\theta}^F$. This is impossible unless some node sent conflicting $t-2$-messages, contrary to our assumption that all non-faulty nodes behave honestly.

Now suppose that node n decides (for example) $\hat{\boldsymbol{\theta}}^*$ in phase t. We prove that no node n' can ever decide $\boldsymbol{\theta}^F$. If n decides in phase t, it must have more than J witnesses for $\hat{\boldsymbol{\theta}}^*$. By the claim we proved above, it is therefore not possible that n' has any witnesses for $\boldsymbol{\theta}^F$. Hence, if n' also decides in phase t, it must decide $\hat{\boldsymbol{\theta}}^*$.

Next, we show that all t-messages must have value $\hat{\boldsymbol{\theta}}^*$. Since n decides at t, it must have received more than J $t-1$-witnesses. Any other node that sends a t-message receives at least $N-J$ $t-1$-messages, so one of those messages must be a $t-1$-witness. By the protocol definition, any such node must therefore set its value to $\hat{\boldsymbol{\theta}}^*$ at t, so all t-messages have the desired form. This argument implies that any node n' that decides in $t+1$ must also decide $\hat{\boldsymbol{\theta}}^*$.

Given that all t-messages have value $\hat{\boldsymbol{\theta}}^*$, it must be that all $t+1$-messages are of the form $t+1:q:\hat{\boldsymbol{\theta}}^*$ for $q \geq N-J$. Therefore, all these messages are witnesses for $\hat{\boldsymbol{\theta}}^*$, meaning that any node deciding at $t+2$ will decide $\hat{\boldsymbol{\theta}}^*$. Moreover, any node that reaches phase $t+2$ will decide $\hat{\boldsymbol{\theta}}^*$ in that phase, so no node ever enters phase $t+3$.

It is clear that nodes may eventually agree on either $\hat{\boldsymbol{\theta}}^*$ or $\boldsymbol{\theta}^F$. If all nodes begin with the same value, then by the proof above, all nodes will decide on that value in two phases.

Deadlock-freedom: We prove that no node ever remains in a phase t permanently without advancing. Suppose, for the sake of contradiction, that a set D of nodes are deadlocked (in phases t_n^d for $n \in D$). Let $t_0 = \min_{n \in D} t_n^d$ and let n be a node that is deadlocked in phase t_0. Take some arbitrary set of non-faulty nodes S with $|S| = N-J$. There are two cases.

Case 1. No node decides in a phase t with $t \leq t_0 - 2$. Then, since t_0 is the minimal phase at which a node gets deadlocked, all nodes in S must decide in $t_0 - 1$, decide in t_0, or reach phase t_0 without deciding. In all of these cases, nodes in S will send t_0-messages at some point to all other nodes. Therefore, every node will eventually receive at least $|S| = N-J$ t_0-messages, so no node will ever be deadlocked in t_0, a contradiction.

Case 2. Some node decides in $t \leq t_0 - 2$. In the consistency proof, we showed that if a node decides at t, then no node ever reaches a phase greater than $t+2$, and all nodes that reach $t+2$ decide. Therefore, all nodes that are deadlocked must be at t_0, and they all must eventually decide, a contradiction.

Convergence: Now we prove that every node decides in finite time with probability one, no matter how many phases have already elapsed. Let S be the set of non-faulty nodes, and suppose that no $n \in S$ decides before some round k_0. We show that there exists a positive constant in (0,1) such that with probability at least ρ, all nodes in S decide by round $k_0 + 2$.

With some probability ρ, all nodes in S receive messages from all other nodes in S in round $k_0 + 1$. Then, with the same probability, the same thing happens in rounds $k_0 + 2$ and $k_0 + 3$. By our proof of consistency, if this occurs, then all nodes in S will decide by round $k_0 + 3$ with at least

probability ρ^3. $\square$

F.2 The revision algorithm

In this section, we outline a "revision algorithm" that we will sometimes use as an extension to the shutdown fault algorithm presented in the previous section. The revision algorithm, which uses proof-of-work, effectively consists of a repetition of the shutdown fault algorithm.

In the revision algorithm, all nodes keep track of a phase number t and a value $\hat{\boldsymbol{\theta}}^n$. There are two types of messages: costly broadcasts and free broadcasts.

A costly broadcast is of the form $n : 0 : \hat{\boldsymbol{\theta}}$ for some $\hat{\boldsymbol{\theta}} \in \tilde{\Theta}$, and n denotes the signature of the node sending the broadcast. A free broadcast is of the form $n : t : \hat{\boldsymbol{\theta}}$ for $t \geq 1$. The spaces of costly and free broadcasts are therefore

$$\mathcal{M}^C = \{n : 0 : \hat{\boldsymbol{\theta}} \mid n \in \mathcal{N}, \hat{\boldsymbol{\theta}} \in \tilde{\Theta}\}, \mathcal{M}^F = \{t : \hat{\boldsymbol{\theta}} \mid t \geq 1, \hat{\boldsymbol{\theta}} \in \tilde{\Theta}, n \in \mathcal{N}\}$$

A node cannot send a costly broadcast $n : 0 : \hat{\boldsymbol{\theta}}$ unless, while the shutdown fault algorithm was running, it received at least $N - J$ witnesses for $\hat{\boldsymbol{\theta}}$ (see Section F.1 for definitions). A node cannot send a free broadcast $1 : \hat{\boldsymbol{\theta}}$ unless it received a costly broadcast for $\hat{\boldsymbol{\theta}}$ for all n such that $\hat{\theta}_n \neq \theta^F$. Likewise, a node cannot send a free broadcast $n : t : \hat{\boldsymbol{\theta}}$ for $t > 1$ unless it received a free broadcast $n : t - 1 : \hat{\boldsymbol{\theta}}$ from all n such that $\hat{\theta}_n \neq \theta^F$. The one exception occurs if $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^F$ (as in the previous section): broadcasts for $\boldsymbol{\theta}^F$ can be sent by any node at any time. Additionally, of course, a node cannot send any type of broadcast bearing a different node's signature. Finally, a costly broadcast $n : 0 : \hat{\boldsymbol{\theta}}$ requires that the sender pay a cost $\kappa(n, \hat{\boldsymbol{\theta}})$, which we specify in the proof of the existence result in Proposition 3.

We next describe the algorithm. This algorithm is meant to run after the two algorithms described in Section F.1.

Algorithm F.3 (Revision algorithm (T^*))**.** *Each node n enters with a value $\hat{\boldsymbol{\theta}}^n$ computed in the Majority Rule algorithm (Algorithm F.2) and a phase number of $t = 0$. Nodes communicate according to the following protocol.*

1. *When a node n is in phase $t = 0$, it sends a costly broadcast $n : 0 : \hat{\boldsymbol{\theta}}^n$ to all other nodes.*
   - *If n receives a costly broadcast of the same value $\hat{\boldsymbol{\theta}}^n$ from each n' such that $\hat{\theta}^n_{n'} \neq \theta^F$ before receiving a costly broadcast for any other value, then it updates its phase to $t = 1$ and moves to the next step.*
   - *If n receives a costly broadcast for some value $\hat{\boldsymbol{\theta}}' \neq \hat{\boldsymbol{\theta}}^n$, it updates its value $\hat{\boldsymbol{\theta}}^n$ to $\boldsymbol{\theta}^F$, updates its phase to $t = 1$, and moves to the next step.*
2. *When node n is in phase $t > 0$, it sends a free broadcast of its value $n : t : \hat{\boldsymbol{\theta}}^n$ to all other nodes.*

- *If n receives a free broadcast of the same value, $n' : t : \hat{\boldsymbol{\theta}}^n$, from each n' such that $\hat{\theta}^n_{n'} \neq \theta^F$ before receiving a costly broadcast for any other value, then if its phase is T^*, it permanently stops updating its phase and value (although it continues to broadcast). Otherwise, it updates its phase to $t+1$ and moves to the next step.*
- *If n receives a free broadcast $n' : t : \hat{\boldsymbol{\theta}}'$ for some value other than $\hat{\boldsymbol{\theta}}^n$, then n updates its value to $\hat{\boldsymbol{\theta}}^n = \boldsymbol{\theta}^F$ and updates its phase to $t+1$.*

We now establish some properties of the revision algorithm. We will not be interested in convergence on a single value or the algorithm's termination. The revision algorithm will be used to dissuade dishonest, off-equilibrium deviations. Hence, we focus on characterizing the probability with which nodes can draw different conclusions under the revision algorithm (considering the possibility that in the filter and majority rule algorithms, some nodes acted dishonestly).

Proposition F.2. *Consider a set of nodes S that communicate honestly according to Filter algorithm (J) (Algorithm F.1), Majority Rule algorithm (J) (Algorithm F.2), and the Revision algorithm (T^*) (Algorithm F.3). Then the probability that two nodes $n, n' \in S$ decide on different values $\hat{\boldsymbol{\theta}}$ (such that for a majority of entries n, $\hat{\theta}_n \neq \theta^F$) is at most $1 - \left(\frac{T^*}{1+T^*}\right)^{|S|}$.*

Proof. Let k_0 be the first round in which some node $n \in S$ receives a costly broadcast for some value $\hat{\boldsymbol{\theta}}$ (such that for more than $\frac{N}{2}$ entries n', $\hat{\theta}_{n'} \neq \theta^F$). Now suppose that in some round $k \geq k_0$, another honest node $n \in S$ receives a different costly broadcast for such a value $\hat{\boldsymbol{\theta}}'$. Note that, since each of $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\theta}}'$ have a majority of entries not equal to θ^F, there must be some $n'' \in \mathcal{N}$ that sent a costly broadcast of $\hat{\boldsymbol{\theta}}$ to n and a costly broadcast of $\hat{\boldsymbol{\theta}}'$ to n'.

If n eventually accepts $\hat{\boldsymbol{\theta}}$ and n' eventually accepts $\hat{\boldsymbol{\theta}}'$, then by the protocol definition in Algorithm F.3, it must be that n delivers at least T^* messages to n'' before n delivers a single message to n'. This is because for each phase t of the revision algorithm, when n'' sends a message to n, n'' must provide proof-of-receipt that n delivered a message to n'' in phase $t-1$.

Let d_0 be the time it takes for the message sent by n in round k_0 to reach n', and let $d_1, \ldots, d_{T^*}$ be the delivery times for the messages from n to n'' (starting in round k_0). The probability that n' accepts $\hat{\boldsymbol{\theta}}'$ is at most $\Pr(d_0 \leq \sum_{k=1}^{T^*} d_k)$. Message delivery times are IID according to a distribution G, so by symmetry, this probability is exactly equal to $\Pr(d_k \leq d_0 + d_1 + \cdots + d_{k-1} + d_{k+1} + \cdots + d_{T^*})$. There are $1 + T^*$ such probabilities, so it must be that $\Pr(d_0 \leq \sum_{k=1}^{T^*} d_k) \leq \frac{1}{1+T^*}$. Therefore, the probability that n and n' decide on different outcomes can be no larger than $\frac{1}{1+T^*}$.

Consider the probability

$$P^* \equiv \Pr(\nexists\, \tilde{n} \in S : \tilde{n} \text{ decides on a value different from } n).$$

We have shown that a necessary condition for any $\tilde{n}$ to decide on a different value from n is for $\tilde{n}$ to deliver T^* messages to some dishonest node before n delivers a single message to $\tilde{n}$. Let $P_{n,\tilde{n}}$ denote the probability that this occurs. Using the independence of message delivery times, P^* then

satisfies

$$P^* \geq \prod_{\tilde{n} \in S} \left(1 - P_{n,\tilde{n}}\right) \geq \left(1 - \frac{1}{1+T^*}\right)^{|S|} = \left(\frac{T^*}{1+T^*}\right)^{|S|}.$$

□

F.3 The Bracha and Toueg (1985) algorithm

The algorithm we present in this section does not require the assumption of synchronous communication– it can be implemented under asynchronous communication as well (Assumption 4). Our algorithm is adapted from the one first derived in Section 3.3 of Bracha and Toueg (1985).[2] We will show that when at most one-third of nodes behave dishonestly, it is possible for others to achieve consensus, no matter how malicious the dishonest nodes are.

Just as in the shutdown fault algorithm, nodes will internally keep track of values $\hat{\boldsymbol{\theta}}^n \in \tilde{\Theta}$ and a phase number t. The value held by node n in phase t is denoted $\hat{\boldsymbol{\theta}}^n_t$.

The message vocabulary consists of three types of messages: initial messages, broadcasts, and echoes. An initial message is of the form $m = (n : \theta, t)$ for some $n \in \mathcal{N}$, $\theta \in \Theta_n$, denoting a message signed by n declaring that its input is $\theta_n = \theta$. A broadcast is of the form $m = (n : \hat{\boldsymbol{\theta}}, t)$ for some $n \in \mathcal{N}$, $\hat{\boldsymbol{\theta}} \in \tilde{\Theta}$, and $t \in \mathbb{N}$. In a broadcast, a node signs a statement declaring that its current value is $\hat{\boldsymbol{\theta}}^n_t = \hat{\boldsymbol{\theta}}$ and that it is in phase t. An echo is of the form $m = n : (n' : \hat{\boldsymbol{\theta}}, t)$, denoting a message by n declaring that it received broadcast $n' : \hat{\boldsymbol{\theta}}$ from a node n' in phase t. The full set of messages is $\mathcal{M} = \mathcal{M}^I \cup \mathcal{M}^B \cup \mathcal{M}^E$, where

$$\mathcal{M}^I = \{n : \theta \mid n \in \mathcal{N}, \theta \in \Theta_n\}$$

$$\mathcal{M}^B = \{(n : \hat{\boldsymbol{\theta}}, t) \mid n \in \mathcal{N},\ \hat{\boldsymbol{\theta}} \in \tilde{\Theta}, t \in \mathbb{N}\},\ \mathcal{M}^E = \{n : m \mid n \in \mathcal{N},\ m \in \mathcal{M}^B\}$$

Nodes cannot forge others' signatures, so they are only permitted to send messages with their own signatures. Additionally, when sending an echo, a node must provide a proof-of-receipt, so a node cannot send echo $n : n' : \hat{\boldsymbol{\theta}}$ unless it previously received broadcast $n : n' : \hat{\boldsymbol{\theta}}$. Finally, a node cannot send a broadcast of an initial input it never received. If a node sends the broadcast $n : \hat{\boldsymbol{\theta}}$, then for all n' such that $\hat{\boldsymbol{\theta}}_{n'} \in \Theta$, node n must have received the initial message $n' : \theta$.

We outline the algorithm below. It consists of two parts: the "filter algorithm" and the algorithm outlined in Bracha and Toueg (1985). The filter algorithm, as we show, reduces the set of values that nodes may start with in the Bracha-Toueg algorithm, which applies only to binary values.

Algorithm F.4 (Filter algorithm (J))**.** *Nodes initialize their values to $\hat{\boldsymbol{\theta}}^n_{n'} = \theta^F$ for all n' and communicate using the following protocol.*

Initial stage

1. *Node n sends the initial message $n : \theta_n$ to all other nodes.*
2. *When a node receives a message $n' : \theta$, it sets $\hat{\boldsymbol{\theta}}^n_{n'} = \theta$.*

[2] Our adaptation of their proof also draws elements from the *filter algorithm* described in Toueg (1984, Figure 4).

3. *Once a node has received $N - J$ such messages, it moves to the second stage.*

Second stage

1. *Node n sends the broadcast message $n : \hat{\boldsymbol{\theta}}$ to all other nodes.*
2. *Once a node has received a set H of $N - J$ such messages, it updates each n'-th component of $\hat{\boldsymbol{\theta}}^n$ to either*
   - *The most frequent input θ among the values $\hat{\boldsymbol{\theta}}_{n'}$ that it received, if that value occurred at least $N - 2J$ times;*
   - *θ^F, if no such value exists.*

   Then, node n moves to the final stage.

Final stage

1. *Node n sends a message $m = n : (\hat{\boldsymbol{\theta}}, H)$, including a proof-of-receipt of the set of messages H that it received in the previous phase.*
2. *Once n receives $N - J$ such (valid) echo messages, it updates each component n' of $\hat{\boldsymbol{\theta}}^n$ as follows:*
   - *If there exists θ such that for all accepted echoes $\hat{\boldsymbol{\theta}}_{n'} = \theta$, then $\hat{\boldsymbol{\theta}}^n_{n'}$ is set equal to θ;*
   - *Otherwise, $\hat{\boldsymbol{\theta}}^n_{n'}$ is set equal to θ^F.*
3. *Node n terminates the filter algorithm and initiates the BT(J) algorithm.*

Lemma F.2. *Filter algorithm (J) eventually terminates so long as at least $N - J$ nodes follow the protocol honestly, as long as $J < \frac{N}{3}$. When the algorithm terminates, for each $n \in \mathcal{N}$, there exists $\theta^*_n \in \Theta$ such that all honest nodes n' have $\hat{\boldsymbol{\theta}}^{n'}_n \in \{\theta^*_n, \theta^F\}$ (where $\theta^*_n = \theta_n$ for honest nodes).*

Furthermore, for all n such that $\hat{\boldsymbol{\theta}}^* \neq \theta^F$, $\hat{\boldsymbol{\theta}}^* = \theta_n$, and any such value $\hat{\boldsymbol{\theta}}^*$ with at most J entries equal to θ^F is reached with positive probability.

Proof. The proof that the filter algorithm terminates is identical to the proof that the BT algorithm terminates, so we postpone it to the proof of Proposition F.3 below.

Suppose that two honest nodes n, n' terminate with values of $\tilde{n}$'s input $\hat{\boldsymbol{\theta}}^n_{\tilde{n}}, \hat{\boldsymbol{\theta}}^{n'}_{\tilde{n}} \in \tilde{\Theta}$ (distinct from θ^F). Let these values be denoted θ and θ'. Denote by S the set of $N - J$ nodes from which n accepted messages (all of whom sent value θ). At least $N - 2J$ nodes in S must also have sent messages that were accepted by n'. The first part of the consistency proof in Proposition F.3 shows that the values accepted by n and n' must be identical, so n' accepts at least $N - 2J$ values equal to θ. But $N - 2J > \frac{N-J}{2}$, so the majority of values accepted by n' are equal to θ, meaning either $\hat{\boldsymbol{\theta}}^{n'}_{\tilde{n}} = \theta$ or $\hat{\boldsymbol{\theta}}^{n'}_{\tilde{n}} = \theta^F$. Hence, $\theta^*_{\tilde{n}} = \theta$.

We now show that the values $\theta^*_{\tilde{n}}$ are consistent with nodes' initial inputs. Note that in order for a node to accept a value $\theta_{\tilde{n}}$, it must be broadcast with an appropriate proof that $\tilde{n}$ sent that value in the initial phase. Therefore, $\theta^*_{\tilde{n}} = \theta_{\tilde{n}}$ for all honest nodes $\tilde{n}$.

To prove the second part of the lemma, first note that the instructions in the first step imply that no matter what node n's value $\hat{\boldsymbol{\theta}}^n$ is, it will be the case that $\hat{\boldsymbol{\theta}}^n_{n'} \in \{\theta_{n'}, \theta^F\}$ for each n'. Then, note that with positive probability, the following happens: some subset of nodes S with $|S| \geq N - J$ send each other type messages that are received by all other nodes in S in the second round, but no node in S receives a message from a node not in S, so that all nodes in $n \in S$ have a value $\hat{\boldsymbol{\theta}}^*$ such that $\hat{\boldsymbol{\theta}}^n_{n'} = \theta_{n'}$ for all $n' \in S$ and $\hat{\boldsymbol{\theta}}^n_{n'} = \theta^F$ for $n' \notin S$. Since $N - J > N - 2J$, it must be that in the second phase, *any* node following the protocol will terminate the second step with value $\hat{\boldsymbol{\theta}}^*$, meaning that this value will be chosen by all nodes that follow the protocol in the third phase. □

Algorithm F.5 (Algorithm BT(J))**.** *Nodes use the values $\hat{\boldsymbol{\theta}}^n$ computed in filter algorithm (J) and communicate using the following protocol.*

1. *If node n is in phase t, it sends the broadcast $m = (n : \hat{\boldsymbol{\theta}}^n_{n'}, t)$ to all other nodes.*
2. *The first time a node n receives a broadcast $(n' : \hat{\boldsymbol{\theta}}, t)$ from each other node n', it sends an echo $n : (n' : \hat{\boldsymbol{\theta}}, t)$ to all other nodes.*
3. *Node n accepts value $\hat{\boldsymbol{\theta}}$ from node n' if it receives more than $\frac{N+J}{2}$ echoes of the form $n'' : (n' : \hat{\boldsymbol{\theta}}, t)$.*
4. *Once n accepts at least $N - J$ values, it updates its value $\hat{\boldsymbol{\theta}}$ and enters main phase $t + 1$.*
   - *For each component n' of $\hat{\boldsymbol{\theta}}^n_{n',t+1}$ is updated to either (1) whichever value was in the majority of the newly accepted values $\hat{\boldsymbol{\theta}}_{n'}$, if such a majority exists, or (2) θ^F, if there is no (strict) majority among the newly accepted values.*
   - *If n accepted at least $\frac{N+J}{2}$ identical values $\hat{\boldsymbol{\theta}}$ and has not decided on a value yet, then n decides $\hat{\boldsymbol{\theta}}$.*

We can now prove that this algorithm permits nodes to come to an agreement when at most one-third act dishonestly.

Proposition F.3. *Algorithm BT(J), for $J \leq \frac{\lfloor N-1 \rfloor}{3}$, achieves consensus among all honest nodes if at most J nodes act dishonestly, and the consensus value is consistent with the inputs of honest nodes. If all non-faulty nodes initiate the protocol with the same value, they reach consensus on that value.*

Proof. **Deadlock-freedom**: First, we show that the consensus algorithm does not deadlock. Consider an honest node n that has repeated the main phase t times, where t is the lowest number of times the main phase has been repeated by any honest node. Then, all other honest nodes n' have already sent broadcasts in main phase t, meaning that eventually, those honest nodes will

also receive those broadcasts and send echoes. Since there are at least $N - J$ honest nodes and $N - J > \frac{N+J}{2}$, n will eventually receive at least $\frac{N+J}{2}$ echoes of the value sent by n' and accept it.

Consistency: Now we prove that if two honest nodes n and n' accept a value from some node n'' in phase t, then those values must be equal. Suppose, for the sake of contradiction, that n accepts a value $\hat{\boldsymbol{\theta}}$ and n' accepts $\hat{\boldsymbol{\theta}}'$. Then, node n received more than $\frac{N+J}{2}$ echoes of $\hat{\boldsymbol{\theta}}$ from n'', and node n received more than $\frac{N+J}{2}$ echoes of $\hat{\boldsymbol{\theta}}'$ from n''. This implies that more than J nodes echoed both $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\theta}}'$. This is not possible, though, because we assume at most J nodes act dishonestly, and by the protocol definition, an honest node would never echo both values.

We must also show that when an honest node decides, then all other honest nodes eventually decide on the same value. Suppose that node n is the first to decide (in some phase t). If its terminal value $\hat{\boldsymbol{\theta}}^n$ satisfies $\hat{\boldsymbol{\theta}}^n_{\tilde{n}} = \theta^*_{\tilde{n}} \in \Theta$, it needs to be shown that eventually, all other honest nodes n' will decide $\hat{\boldsymbol{\theta}}^{n'}_{\tilde{n}} = \theta$. Given that n decides on $\hat{\boldsymbol{\theta}}^n_{\tilde{n}} = \theta$, it must be that in phase t, n received values satisfying $\hat{\boldsymbol{\theta}}_{\tilde{n}}$ from at least $\frac{N+J}{2}$ other nodes (say, in a set S).

By the deadlock-freedom property, in phase t, all other honest nodes n' will accept at least $N - J$ values. Given that $N - J > \frac{N+J}{2}$, it must be that node n' accepts messages from more than $\frac{N+J}{2} - J = \frac{N-J}{2}$ *honest* nodes in the set S, who must have sent the same values to n and n'. Therefore, a majority of the messages accepted by node n' have value θ for component $\tilde{n}$, meaning in phase t, n' updates its value so that its $\tilde{n}$-th component is θ. This argument implies that at phase $t + 1$, all honest nodes have value θ for node $\tilde{n}$. Thereafter, no honest node will ever change the $\tilde{n}$-th component of its value to anything other than θ. A node needs to receive at least $\frac{N-J}{2}$ echoes in any phase in order to decide against θ, but given that there are at most $J < \frac{N-J}{2}$ dishonest nodes, and all honest nodes from that point forward have value θ, it is not possible for this to ever happen.

Now, suppose that node n decides on a value such that $\hat{\boldsymbol{\theta}}^n_{\tilde{n}} = \theta^F$. A parallel argument applies, so that all other honest nodes must end phase t with value θ^F and must keep that value thereafter. Therefore, after some honest node n decides, all other honest nodes n' have values $\hat{\boldsymbol{\theta}}^{n'}$ that are equal to the value held by n and are consistent with (at least) the initial inputs of honest nodes (since for all honest nodes, $\theta^*_n = \theta_n$).

Finally, note that if all non-faulty nodes start the protocol with the same value, they must eventually agree on that value. Nodes enter with one of two values, $\hat{\boldsymbol{\theta}}^*$ or $\boldsymbol{\theta}^F = \theta^F \cdot \mathbf{1}^{\mathcal{N}}$ (by Lemma F.2). If all non-faulty nodes start with value $\hat{\boldsymbol{\theta}}^*$, then since $N - J > \frac{N+J}{2}$, no non-faulty node can ever update its value to $\boldsymbol{\theta}^F$, meaning they must reach agreement on $\hat{\boldsymbol{\theta}}^*$. The same argument applies if all non-faulty nodes enter with value $\boldsymbol{\theta}^F$.

Convergence: After some history of communication, let S be the subset of honest nodes that have not yet decided. Suppose that no such node has decided in any phase $t < t_0$ (for some t_0). All nodes in S eventually reach phase t_0 with probability one. Note that in phase t_0, with positive probability, nodes in S only communicate with each other and other honest nodes that have already decided. Then, all nodes in S must exit phase t_0 with the same value. In phase $t_0 + 1$, again with positive probability, all nodes in S may communicate only with each other and with honest nodes

that have decided. Agreement on a value is unanimous, so all nodes in S decide in phase t_0+1. □

Having proved the correctness of the algorithm, we now prove a proposition describing the range of deviations that are possible under this consensus algorithm. We will say there is *agreement* on a value $\boldsymbol{\theta} \in \tilde{\Theta}$ if, at some point during the execution of the algorithm, all $n \in S(\boldsymbol{\theta})$ decide $\boldsymbol{\theta}$.

Proposition F.4. *Fix a type profile $\boldsymbol{\theta}$ and let S (with $|S| > \frac{N}{3}$) be a coalition of agents. Then any deviation from honesty by agents in S must lead to a set of agreements on type profiles $\hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_K$ with $S_1 = S(\hat{\boldsymbol{\theta}}_1), \ldots, S_K = S(\hat{\boldsymbol{\theta}}_K)$ such that*

- *$S_k \cap S_{k'} \subset S$ for all $k \neq k'$;*
- *$\hat{\boldsymbol{\theta}}_{k,n} = \theta_n$ for $n \notin S$.*

Then any deviation from the BT($\lfloor \frac{N-1}{3} \rfloor$) algorithm for agents in S must lead to a set of agreements of this form.

Proof. The content of the proposition is that if agents in S deviate from honest behavior, no agent $n \notin S$ will (1) decide on multiple different values or (2) decide on a value $\tilde{\boldsymbol{\theta}}$ such that $\tilde{\theta}_n \neq \theta_n$ for some $n \notin S$.

The first point is guaranteed by the fact that under the BT algorithm, no node ever decides more than once. As for the second point, note that in the second phase of the filter algorithm that precedes the BT algorithm, if n behaves honestly, a node n' cannot broadcast a profile $\hat{\boldsymbol{\theta}}$ with a component $\hat{\boldsymbol{\theta}}_n \notin \{\theta_n, \theta^F\}$. This is because sending the broadcast in the second phase requires a proof-of-receipt that n actually sent the message θ_n in the first phase. In the final phase of the filter algorithm, the final value held by a node must be an echo of some valid broadcast sent in the second round, so it is impossible for the final value of a node n', $\hat{\boldsymbol{\theta}}^{n'}$, to have a component $\hat{\boldsymbol{\theta}}^{n'}_n \notin \{\theta_n, \theta^F\}$.

If agents coordinate on a decision $\hat{\boldsymbol{\theta}}$ after running the BT algorithm, it must be of the form $\hat{\theta}_n = \theta_n$ if $n \in S$ and $\hat{\theta}_n = \theta^F$ otherwise (for some consensus set S). Therefore, agents cannot coordinate on a decision with $\hat{\boldsymbol{\theta}}_n \neq \theta_n$.

□

F.4 The Lamport, Shostak, and Pease (1982) algorithm

We outline an analogue the algorithm derived by Lamport, Shostak, and Pease (1982) to reach consensus in a synchronous environment when signed messages are available. We define the message vocabulary as chains of signed messages. There is a base vocabulary $\mathcal{M}_0 = \bigcup_{n \in \mathcal{N}} \Theta_n$, which includes a message for each possible type θ_n of agent n. Then, we recursively define the message spaces

$$\mathcal{M}_j = \{n : m \mid n \in \mathcal{N}, m \in \mathcal{M}_{j-1}\}$$

for $j = 1, 2, \ldots$. The message $n : m$ denotes message m with node n's signature appended. In general, then, a message will take the form

$$m = n_1 : n_2 : n_3 : \cdots : n_J : \theta,$$

denoting that n_J sent message θ, which was then received and signed by n_{J-1}, which was then received and signed by n_{J-2}, and so on. A node n may send message $n : m \in \mathcal{M}_j$ only if it has received message $m \in \mathcal{M}_j$ in the past. Additionally, a node is not permitted to send any message of the form $n' : m$ for $n' \neq n$. Hence, a node cannot forge another's signature.

We now describe the consensus algorithm. Under Assumption 4′.1, the designer is permitted to make nodes' strategies depend on the maximum message lag Δ. Nodes are instructed communicate only in rounds that are integer multiples of Δ, i.e., $k = 1, 1 + \Delta, 1 + 2\Delta, \ldots$. This ensures that if a message is sent in round $1 + j\Delta$, it will be received by all others with certainty before round $1 + (j + 1)\Delta$. For simplicity, we will henceforth refer to round $1 + j\Delta$ simply as round j.

In order to reach consensus, nodes need to reach an agreement on a value $\boldsymbol{\theta} \in \tilde{\Theta}$. Thus, each node maintains an internal state $\hat{\boldsymbol{\theta}}^n \in \tilde{\Theta}$. The value of each element n', $\hat{\boldsymbol{\theta}}^n_{n'}$, is initialized to θ^F. Faulty nodes may not communicate, or they could communicate different values to different non-faulty nodes, so it is not possible in general for non-faulty nodes to come to an agreement on their initial inputs. However, it will be possible for them to deduce the identities of faulty nodes. The consensus algorithm must permit honest nodes to come to an agreement on their values of $\hat{\boldsymbol{\theta}}^n$. When non-faulty nodes agree on some $\hat{\boldsymbol{\theta}}$ with $S(\hat{\boldsymbol{\theta}}) = S$, that means they agree on the inputs of nodes in S.

We say a message m received in round j constitutes a *properly signed value θ by node n* if that message is of the form

$$m = n_1 : \cdots : n_j : \theta,$$

where the signatures $n_1, \ldots, n_j$ are distinct and $n_j = n$. Such a message constitutes evidence that, at some point, node n send the message θ, and that this signed value was seen by at least j distinct nodes.

The communication protocol, outlined below, specifies which messages a node should send at each information set and the point at which a node should recommend an action to its owner. Under this communication protocol, nodes are instructed to send the same set of messages to all others in each round, so the instruction "send message m" should be understood as shorthand for "send message m to all other nodes n'."

Algorithm F.6 (Algorithm LSP(J))**.** *Nodes communicate using the following protocol.*

1. *In the first round, a node n with input θ_n sends message $n : \theta_n$.*

   - *Upon receiving a message $n' : \theta_{n'}$ from another node n', node n sets $\hat{\boldsymbol{\theta}}^n_{n'} = \theta_{n'}$.*
   - *If a node n receives multiple messages from another node n', or no message at all, the value of $\hat{\boldsymbol{\theta}}^n_{n'}$ remains θ^F.*

2. *In any round $j > 0$, for each message m received by node n in the previous round $j-1$, node n sends message $n : m$.*

   - *If node n receives a properly signed value $\theta_{n'}$ by node n', and had not previously received a properly signed value, then n sets $\hat{\boldsymbol{\theta}}^n_{n'} = \theta_{n'}$.*
   - *If node n receives a properly signed value $\theta_{n'}$ by node n', and had previously received a properly signed value, then n sets $\hat{\boldsymbol{\theta}}^n_{n'} = \theta^F$.*

3. *Communication terminates in round J. Node n decides on value $\hat{\boldsymbol{\theta}}^n$.*

We now prove that the algorithm LSP(J), which terminates in J rounds of communication, achieves consensus among all honest nodes so long as at most $J-1$ nodes behave dishonestly.

Proposition F.5. *Algorithm LSP(J) achieves consensus among all honest nodes if at most $J-1$ nodes act dishonestly, and the consensus value is consistent with the inputs of honest nodes.*

Proof. We need to show that when at most $J-1$ nodes fail to follow the communication protocol, then at the time of termination, all nodes that follow it have the same values of $\hat{\boldsymbol{\theta}}^n$, and, moreover, $\hat{\boldsymbol{\theta}}^n_{n'} = \theta_{n'}$ if n and n' acted honestly throughout the execution of the algorithm. We will call nodes that follow the communication protocol *honest nodes*.

First, note that if a node n follows the protocol honestly, then all other nodes n' that follow the protocol will terminate communication with the same value of $\hat{\boldsymbol{\theta}}^{n'}_n = \theta_n$. This is because in the first round, given that n is honest, it sends all other nodes a properly signed value of θ_n. Then, all other nodes that follow the protocol update $\hat{\boldsymbol{\theta}}^{n'}_n$ to θ_n in the first round. According to the protocol, this value is reverted to θ^F only if n' later receives a properly signed value $\theta'_n \neq \theta_n$ by node n. Under the assumption that n communicates honestly, however, n never sends another message of the form $n : \theta'_n$, so by the assumption that signatures are unforgeable, it is not possible for n' to receive such a message.

Now we need to prove that honest nodes have the same value of $\hat{\boldsymbol{\theta}}_{\tilde{n}}$ for nodes $\tilde{n}$ that acted dishonestly at some point. It suffices to show that if an honest node n ever received a properly signed value $\theta_{\tilde{n}}$ by $\tilde{n}$, then all other honest nodes n' also received the same value. If all honest nodes received multiple properly signed values by $\tilde{n}$ (or no value), then they will agree on $\hat{\boldsymbol{\theta}}_{\tilde{n}} = \theta^F$. Otherwise, if they all received a single value, they will agree on it.

Suppose that in some round $j < J$, an honest node n received a properly signed value θ by $\tilde{n}$. Then in round $j+1$, node n would have sent that properly signed value to all other nodes, guaranteeing that they receive it before round J. Now suppose that n receives a properly signed value θ by $\tilde{n}$ for the first time in round J. The corresponding message is of the form

$$n_1 : n_2 : \cdots : n_{J-1} : \tilde{n} : \theta.$$

The identities of the nodes $n_1, \ldots, n_{J-1}$ are distinct from each other and from $\tilde{n}$, by definition. Now, note that at least one of the nodes $n_1, \ldots, n_{J-1}$ must be honest: if all of them are dishonest,

then $n_1, \ldots, n_{J-1}$ and $\tilde{n}$ must be dishonest, contradicting the assumption that at most $J-1$ nodes acted dishonestly. Hence, some honest node had seen the properly signed value θ by $\tilde{n}$ in some round $j < J$, meaning that in round $j+1 \leq J$, it had sent that value to all other honest nodes (by Step 2 of the protocol). Therefore, all other honest nodes have received value θ properly signed by $\tilde{n}$ by the time the protocol terminates. This concludes the proof, showing that all honest nodes possess the same set of properly signed values, and therefore agree on a value of $\hat{\boldsymbol{\theta}}$. □

From this proposition, we can derive conclusions about the ways in which coalitions of nodes can strategically deviate to affect outcomes.

Corollary 1. *Consider a game $\mathcal{G}_S(\tilde{\sigma}_F)$ in which under honest communication, all non-faulty nodes reach agreement on some value $\hat{\boldsymbol{\theta}}$. Under Algorithm LSP(N), any deviation from honesty by a coalition of nodes $S' \subset S$ results in consensus among honest nodes on a value $\hat{\boldsymbol{\theta}}$ in the set*

$$\Gamma(\hat{\boldsymbol{\theta}}|S') = \{\tilde{\boldsymbol{\theta}} \in (\Theta \cup \{\theta^F\})^{\mathcal{N}} : \ \tilde{\theta}_n = \hat{\theta}_n \ \forall \ n \notin S'\}.$$

Proof. It is clear from the statement of Proposition F.5 that under algorithm LSP(N) all honest nodes must agree on a value $\hat{\boldsymbol{\theta}}_n$, since if at least one node is honest, at most $N-1$ nodes are dishonest. Furthermore, in the proof of Proposition F.5, we showed that honest nodes agree on the true inputs θ_n of all other honest nodes n. Hence, the consensus value $\hat{\boldsymbol{\theta}}$ must have $\hat{\theta}_n = \theta_n$ for all honest n.

Clearly, for each $\hat{\boldsymbol{\theta}}' \in \Gamma(\hat{\boldsymbol{\theta}}|S)$, there exists a deviation for agents in S results in a consensus on the value $\hat{\boldsymbol{\theta}}'$. Each $n \in S'$ simply has to send all others the message $n : \hat{\theta}'_n$ in the first round and then behave exactly as the communication protocol specifies. It then remains to prove that these are the only outcomes achievable through deviations by S'. As already stated, it is not possible that by deviating, agents in S' could achieve an outcome in which consensus is reached on some $\hat{\boldsymbol{\theta}}'$ such that $\hat{\boldsymbol{\theta}}'_n \neq \theta_n$ for a non-faulty agent $n \notin S$.

Furthermore, under the LSP algorithm, the value of $\hat{\theta}_n$ for a faulty node n accepted by any non-faulty agent $n' \notin S'$ must be properly signed. This means that there must have been a message of the form $n : \hat{\theta}_n$ sent by n in the first round. The message sent by n in the first round cannot depend on the behaviors of nodes in the deviating coalition, so it is not possible for agents in S' to jointly deviate and reach an outcome $\hat{\boldsymbol{\theta}}'$ such that $\hat{\theta}'_n \neq \hat{\theta}_n$ for a faulty node n. □

This corollary implies that under the LSP algorithm, a coalition of dishonest nodes can at most misreport their types, or they can report nothing (resulting in all honest nodes imputing the fictitious value θ^F for their inputs). These are precisely the same deviations that are possible for coalitions of strategic agents under the mediated consensus algorithm.

G Proof of the existence result

In this section, we prove the existence result in the Blockchain Trilemma. The result actually requires three separate proofs: one showing that there exists a consensus algorithm that gives up

only fault-tolerance (i.e., is resource-efficient and achieves full transferability), one showing the existence of a consensus algorithm that gives up only full transferability, and one showing that there exists a consensus algorithm that gives up only resource-efficiency. For short, we call these the fault-tolerance proof, the full transferability proof, and the resource-efficiency proof.

G.1 Fault-tolerance proof

Proposition G.1. *Under Assumptions 1-4, there exists a consensus algorithm $(\mathcal{G}, \mathcal{C})$ that is resource-efficient and achieves full transferability.*

Proof. **The social choice function**: We begin by defining a social choice function that the consensus algorithm will implement. We assume that the designer uses the following private randomization scheme. Agents draw signals ϕ_n of the form (j, g), where j is an integer drawn uniformly from $\{1, \ldots, N\}$, and $g : \Theta \to \mathcal{X}$ is a map from states of the world Θ to a Pareto-efficient and individually rational outcome for each state (drawn uniformly from the set of such maps, which is finite). Hence, $\Phi = \{1, \ldots, J\} \times \Omega_g$, where

$$\Omega_g = \{g : \Theta \to \mathcal{X} : \ g(\theta) \text{ is individually rational } \ \forall\, \theta \in \Theta\}.$$

We define the *highest-ranking signal* $\phi_{\max}(\boldsymbol{\theta})$ to be

- The ϕ_n with the highest index j, if a unique maximum exists;
- If there is a tie between multiple values $\phi_{n_1}, \ldots, \phi_{n_k}$, then the tie is broken by choosing the agent with the lowest index (n_1 here).

Then, we use the following social choice function. If $\phi_{\max}(\boldsymbol{\theta})(j_{\max}, g_{\max})$ is the highest-ranking signal, and θ is the profile of preferences given the profile of types $\boldsymbol{\theta}$, then

$$F(\boldsymbol{\theta}) = g_{\max}(\theta).$$

If the consensus algorithm succeeds in implementing this social choice function, then it will accomplish the goal of full transferability. By construction, in state θ, each Pareto-efficient and individually rational outcome $x \in \mathcal{X}^{\text{IR}}(\theta)$ is chosen with positive probability.

The consensus algorithm: The message vocabulary that agents will use in this consensus algorithm has two types of messages: broadcasts and echoes. A broadcast is a signed message of the form $n : \theta$ for some $n \in \mathcal{N}$ and $\theta \in \Theta_n$. A broadcast $n : \theta$ is a declaration by n that their type is θ. An echo is a signed message of the form $n : \hat{\boldsymbol{\theta}}$ for $n \in \mathcal{N}$ and $\hat{\boldsymbol{\theta}} \in \tilde{\Theta}$. An echo indicates that n received the types $\hat{\theta}_{n'}$ from each $n' \in \mathcal{N}$. The spaces of broadcasts and echoes are then

$$\mathcal{M}^B = \{n : \theta \mid n \in \mathcal{N}, \theta \in \Theta\}, \ \mathcal{M}^E = \{n : \hat{\boldsymbol{\theta}} \mid n \in \mathcal{N}, \hat{\boldsymbol{\theta}} \in \prod_{n \in \mathcal{N}} \Theta_n\}.$$

Nodes are required to provide proofs-of-identity when they send any message, meaning node n may send only messages of the form $n : \theta$ and $n : \hat{\boldsymbol{\theta}}$. Additionally, nodes are required to provide

proofs-of-receipt when they send echoes. Hence, a node is not permitted to send an echo $n : \hat{\boldsymbol{\theta}}$ unless it has received a message $n' : \hat{\theta}_{n'}$ from every other node $n' \in \mathcal{N}$. This is where we use Assumption 3. In this particular case, nodes are permitted to decide on any value in any round (so $D_{nk} = \mathcal{X}$).

The communication protocol specifies that if it has not already done so, a node n should send broadcast $n : \theta$ to all other nodes n' (where θ is the type of agent n). Then node n should wait until it receives exactly one broadcast from all other nodes. Once it has received a broadcast $n' : \hat{\theta}_{n'}$ from every other node, it sends the echo $n : \hat{\boldsymbol{\theta}}^n$ to all others, where $\hat{\boldsymbol{\theta}}^n_{n'} = \hat{\theta}_{n'}$. Finally, node n waits until it has received exactly one echo $n' : \hat{\boldsymbol{\theta}}^{n'}$ from all other nodes. If all of the values $\hat{\boldsymbol{\theta}}^{n'}$ in those echoes are equal to each other and to $\hat{\boldsymbol{\theta}}^n$, node n decides on the outcome $F(\hat{\boldsymbol{\theta}}^n)$. Otherwise, the node does not decide. If a node n receives more than one message from another n' in the broadcast phase or the echo phase, it ceases communicating entirely. Note that this communication protocol is feasible under Assumption 4: no part of it makes reference to the number of rounds that have elapsed nor to the maximum message delay Δ.

Incentive-compatibility: We must check that honest communication according to the consensus algorithm is an equilibrium whenever there are no faulty nodes. We verify that no coalition $S \subset \mathcal{N}$ has an incentive to deviate from the protocol. There are two cases: the case in which the deviating coalition is a subset of $\mathcal{N}$ and the case in which the deviating coalition consists of all agents in $\mathcal{N}$. In both cases, we will use the fact that the outcome $x = F(\boldsymbol{\theta})$ at which agents arrive by following the protocol is also an equilibrium outcome under the mediated consensus algorithm (by construction).

Case 1: $S \subsetneq \mathcal{N}$. Observe that under this consensus algorithm, it is never possible for two agents n, n' whose nodes follow the protocol to decide on different outcomes. Nodes that behave according to the communication protocol wait until they receive confirmation that all others have learned the same value $\hat{\boldsymbol{\theta}}$ (and therefore will take the same action). Hence, it is not possible for two agents whose nodes behave honestly to decide on different outcomes.

Furthermore, note that if n, n' are agents whose nodes behave honestly, node n' can never decide on an outcome $F(\hat{\boldsymbol{\theta}})$ such that $\hat{\theta}_n \neq \theta_n$. The final consensus outcome (if any) will always be one decided by some honest node. Therefore, the final consensus outcome must satisfy

$$\hat{x} = F(\boldsymbol{\theta}') \text{ s.t. } \theta'_n = \tilde{\theta}_n \ \forall \ n \notin S, \phi'_n = \phi_n \ \forall \ n \in \mathcal{N},$$

(where we use the notation $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n) = ((\tilde{\theta}_1, \phi_1), \ldots, (\tilde{\theta}_n, \phi_n))$). Of course, we have $\phi'_n = \phi_n$ for all n because the random signal is verifiable.

Hence, the deviations available to agents in S, when they *only* deviate to alter the final consensus outcome, are those that would be available if they lied about their preferences under the mediated consensus algorithm. On the other hand, if the coalition S does not deviate, the outcome is $x^* = F(\boldsymbol{\theta})$, which is the equilibrium outcome with a trusted mediator. Recalling that under the mediated consensus algorithm agents have no incentive to lie about their types (as demonstrated by Lemma 1), it must be incentive-compatible for agents in the deviating coalition S to report their

types honestly to agents $n \notin S$.[3]

Case 2: $S = \mathcal{N}$. In this case, the set of outcomes that occur can be completely arbitrary. However, recall that the equilibrium prescribed by the mediated consensus algorithm is robust to coalitional deviations, and the coalitional deviations available under this consensus algorithm are precisely the same as those that can be achieved under the mediated consensus algorithm.

Full transferability: If nodes behave according to the communication protocol and there are no faults, the outcome will be $F(\boldsymbol{\theta})$ when the profile of types is $\boldsymbol{\theta}$. By assumption, for each $x \in \mathcal{X}^{\text{IR}}_{\mathcal{N}}(\theta)$, there exists a profile of types $\boldsymbol{\theta}$ such that $F(\boldsymbol{\theta}) = x$, meaning the goal of allocative efficiency is achieved.

Resource-efficiency: In its definition, the message space does not require that agents provide proof-of-work when sending any message, so the consensus algorithm is resource-efficient.

□

G.2 Resource-efficiency proof

Proposition G.2. *Under Assumptions 1-4, there exists a consensus algorithm* $(\mathcal{G}, \mathcal{C})$ *that is fault-tolerant and achieves full transferability.*

Proof. **The consensus algorithm**: We use the same social choice function as in Proposition G.1.

The message vocabulary $\mathcal{M}$ is comprised of the vocabulary used in the Filter($\lfloor \frac{N-1}{2} \rfloor$) algorithm $\mathcal{M}^{\text{filt}}$ (described in Algorithm F.1), the vocabulary used in the Majority Rule($\lfloor \frac{N-1}{2} \rfloor$) algorithm $\mathcal{M}^{\text{maj}}$ (described in Algorithm F.2), and the vocabulary used in the Revision(T^*) algorithm $\mathcal{M}^{\text{rev}}$ (described in Algorithm F.3), where the constant T^* will be specified later. Hence,

$$\mathcal{M} = \mathcal{M}^{\text{filt}} \cup \mathcal{M}^{\text{maj}} \cup \mathcal{M}^{\text{rev}}.$$

The only costly messages are the costly broadcasts in $\mathcal{M}^{\text{rev}}$. For all $m \in \mathcal{M}^{\text{filt}} \cup \mathcal{M}^{\text{maj}}$, then, $\kappa(m) = 0$. The cost to node n of sending a costly broadcast $n : t : \hat{\boldsymbol{\theta}}$, if $\hat{\theta}_{n'} \neq \theta^F$ for all $n' \in S$, is

$$\kappa(n : 0 : \hat{\boldsymbol{\theta}}) = \sum_{y \in \mathbf{y}(F(\hat{\boldsymbol{\theta}}))} u_n(y|\hat{\theta}_n) + t_n(F(\hat{\boldsymbol{\theta}})),$$

which is positive, because by construction, the chosen outcome is individually rational. The permissible set of messages that a node may send at each information set (H, θ) is specified in Algorithms F.1-F.3. Finally, we must specify the set of outcomes D_{nk} on which a node n may decide in round k. We impose that a node n can decide on $x = F(\boldsymbol{\theta})$ only if it has received a costly broadcast of x from all n' such that $\theta_{S,n'} \neq \theta^F$ as well as T^* free broadcasts from each such node, where T^* is specified below.

The communication protocol dictates that nodes follow the communication strategy outlined in Algorithm F.1, followed by the strategy in Algorithm F.2, followed by the strategy in Algorithm

[3]For the case in which preferences are common knowledge, it is not necessary to even consider the possibility that agents misreport their types.

F.3. Let $U_{\max} = \max_{n \in \mathcal{N}} \max_{\mathbf{y} \subset \bigcup_{S \ni n} \mathcal{Y}_S} \max_{\theta \in \tilde{\Theta}_n} \sum_{y \in \mathbf{y}} u_n(y|\theta) + \sum_{n' \in \mathcal{N}} v_{n'}$ and $\kappa_{\min} = \min_{m:\kappa(m)>0} \kappa(m)$. The constant T^* specifying the number of phases of communication in Algorithm F.3 is set such that

$$1 - \left(\frac{T^*}{1+T^*}\right)^N = \frac{\kappa_{\min}}{U_{\max}}.$$

When nodes reach the end of the Revision algorithm, they decide on whatever value $\hat{\boldsymbol{\theta}}$ they currently possess, choosing outcome $x = F(\hat{\boldsymbol{\theta}})$ if $\hat{\boldsymbol{\theta}} \neq \theta^F \cdot \mathbf{1}^{\mathcal{N}}$ and an outcome x with $\mathbf{y} = \emptyset$ and $\Delta\mathbf{v} = 0$ otherwise.

Full transferability: We defined the collection of social choice functions $\mathcal{F}$ so that for each individually rational outcome $x \in \mathcal{X}_S^{\text{IR}}(\theta)$, in state θ, there exists a profile of types $\boldsymbol{\theta} \in \tilde{\Theta}$ such that $x = F(\boldsymbol{\theta})$.

Fault-tolerance: We must show that any subset of agents $S \subset \mathcal{N}$ with $|S| > \frac{N}{2}$ is a consensus set. This requires that we show that (1) for any such S, the behaviors specified by the communication protocol are a fault-tolerant equilibrium, (2) whenever such a set S of nodes communicate according to the protocol, they come to a consensus on an outcome consistent with their types, and (3) when nodes in such a set S communicate according to the protocol, they achieve all outcomes in $F(\boldsymbol{\theta})$ (for some $\boldsymbol{\theta} \in \tilde{\Theta}$) with positive probability.

Points (2) and (3), corresponding to Condition 1 in Definition A.6 and the full transferability condition condition, are proven by Propositions F.1 and F.1. Both propositions show that if a majority of nodes follow the communication protocol, then they will eventually reach agreement on a value consistent with their initial inputs θ. Proposition F.1 shows that when the Majority Rule algorithm (Algorithm F.2) begins, the range of possible values considered by each node is narrowed to just two values, $\hat{\boldsymbol{\theta}}^*$ or $\boldsymbol{\theta}^F \equiv \theta^F \cdot \mathbf{1}^{\mathcal{N}}$, where $\hat{\boldsymbol{\theta}}^*$ takes all values in

$$\Gamma(\boldsymbol{\theta}, S) = \{\hat{\boldsymbol{\theta}} \in \tilde{\Theta} : \hat{\theta}_n \in \{\theta_n, \theta^F\} \ \forall\ n, \exists\ S' \subset S \text{ s.t. } \hat{\theta}_n = \theta_n \ \forall\ n \in S', |S'| > \frac{N}{2}\}$$

with positive probability. Then, Proposition F.1 shows that with positive probability, nodes agree on the value $\hat{\boldsymbol{\theta}}^*$ derived from the Filter algorithm. Hence, the required condition holds.

We are left with proving that the strategies specified by the communication protocol constitute a fault-tolerant equilibrium. There are two types of deviations that a coalition S could undertake: deviations in which they cause multiple outcomes to occur (with positive probability) and deviations in which only one outcome occurs, but agents in S misreport their types. First consider the possibility of multiple outcomes occurring. By Proposition F.2, no matter what deviation is attempted by agents in S, the probability that multiple outcomes actually occur is at most $1 - \left(\frac{T^*}{1+T^*}\right)^{|S|} \leq 1 - \left(\frac{T^*}{1+T^*}\right)^N = \frac{\kappa_{\min}}{U_{\max}}$. By construction, the maximum utility that an agent in S could derive from such a deviation is $U_{\max}$. The cost incurred by an agent in S who engages in this deviation is at least $\kappa_{\min}$. Therefore, the expected value of engaging in the deviation is negative for all agents in S.

Now suppose that agents in S simply misreport their types. If $F_{S'}(\boldsymbol{\theta}_{S'})$ is the outcome that

would have been realized under honest behavior by agents in S (for some $S' \supset S$), then the set of type profiles that could be realized under such a deviation is

$$\Gamma(\boldsymbol{\theta}, S|S') = \{\hat{\boldsymbol{\theta}} \in \tilde{\Theta} : \hat{\theta}_n = \theta_n, n \in S' \backslash S\}.$$

The payoff to an agent in S from such a deviation is

$$U_n(\{F_{S'}(\hat{\boldsymbol{\theta}})\}|\theta_n) - \kappa(n:0:\hat{\boldsymbol{\theta}}) = \sum_{y \in \mathbf{y}(F_{S'}(\hat{\boldsymbol{\theta}})} u_n(y|\theta_n) + t_n(F_{S'}(\hat{\boldsymbol{\theta}})) - \kappa(n:0:\hat{\boldsymbol{\theta}})$$

On the other hand, under honest behavior, agents in S receive a payoff of zero by construction. Therefore, if it is the case that it is profitable for coalition S to deviate, it must be that

$$U_n(\{F_{S'}(\hat{\boldsymbol{\theta}})\}|\theta_n) - \kappa(n:0:\hat{\boldsymbol{\theta}}) \geq 0 \Leftrightarrow \sum_{y \in \mathbf{y}(F_{S'}(\hat{\boldsymbol{\theta}})} u_n(y|\theta_n) + t_n(F_{S'}(\hat{\boldsymbol{\theta}})) \geq \sum_{y \in \mathbf{y}(F_{S'}(\boldsymbol{\theta})} u_n(y|\theta_n) + t_n(F_{S'}(\boldsymbol{\theta}))$$

for all $n \in S$, with strict inequality for some n. Note that if this condition holds, then there is a profitable deviation for coalition S under the mediated consensus algorithm: they could misreport their types in the exact same way so that the mediator receives reports $\hat{\boldsymbol{\theta}}$ instead of $\boldsymbol{\theta}$. By Lemma 1, there is no profitable deviation for S under the mediated consensus algorithm, so this deviation cannot be profitable either. Hence, no coalition has a profitable deviation from honesty.

□

G.3 Full transferability proof

Proposition G.3. *Under Assumptions 1-4, there exists a consensus algorithm $(\mathcal{G}, \mathcal{C})$ that is fault-tolerant and resource-efficient.*

Proof. **The consensus algorithm**: For each subset of agents S, there is a feasible outcome x_S^0 such that $\mathbf{y}(x_S^0) = \emptyset$ and $\mathbf{t}(x_S^0) = 0$. We choose a collection of social choice functions such that $F_S(\boldsymbol{\theta}_S) = x_S^0$ for all $\boldsymbol{\theta}_S \in \prod_{n \in S} \Theta_n$.

We assume that the message vocabulary consists of the set of messages $\mathcal{M}^{\text{filt}}$ used in the Filter($\lfloor \frac{N-1}{2} \rfloor$) algorithm for the shutdown case (described in Algorithm F.1) and the messages $\mathcal{M}^{\text{maj}}$ used in the Majority Rule($\lfloor \frac{N-1}{2} \rfloor$) algorithm (described in Algorithm F.2), so the full message vocabulary is

$$\mathcal{M} = \mathcal{M}^{\text{filt}} \cup \mathcal{M}^{\text{maj}}.$$

All messages $m \in \mathcal{M}$ have a cost of zero, $\kappa(m) = 0$. The set of permissible messages M_{nk} that node n can send in round k is precisely as described in Algorithms F.1 and F.2. The set of permissible values D_{nk} on which node n may decide in round k is $D_{nk} = \{x : \exists\ S \subset \mathcal{N}, x = x_S^0\}$.

The communication protocol dictates that agents should communicate as described in Algorithms F.1 and F.2.

Resource-efficiency: The communication game $\mathcal{G}$ does not make use of costly messages, so this consensus algorithm is resource-efficient.

Fault-tolerance: As usual, we must show that any subset of agents $S \subset \mathcal{N}$ with $|S| > \frac{N}{2}$ is a consensus set. This requires that we establish that (1) no sub-coalition of any such subset S has a profitable deviation from honest behavior, (2) when all nodes in S follow the protocol, they eventually reach agreement on an outcome, and (3) all outcomes specified by F_S occur with positive probability (taking as given the identity of faulty nodes, $\mathcal{N} - S$).

The implication of Proposition F.1 is that when all nodes follow the filter protocol, the value held by each node when it terminates is some profile of types that implies an outcome $x^0_{S'}$ for some $S' \subset S$. If the value $\hat{\boldsymbol{\theta}}^*$ given in the statement of Proposition F.1 has $\hat{\theta}^*_n \neq \theta^F$ for $n \in S'$, then the corresponding outcome that will be learned by all nodes is $x^0_{S'}$. Then, after running the majority rule algorithm, all nodes will have the same value, as proven in Proposition F.1, so they will all agree on some such outcome. This argument proves points (2) and (3).

To see that behavior according to the communication protocol is a fault-tolerant equilibrium, note that no matter what outcome is realized, all agents receive a utility of zero, by construction (both from transactions and from balance transfers). Therefore, no deviation could possibly be profitable for a coalition of agents $S' \subset S$, since no matter how many outcomes are realized, those agents would receive no utility. Hence, any majority of agents constitutes a consensus set. □

H Proof of Proposition 1

In this section, we prove Proposition 1, which extends all of our main results to the case in which Assumption 1' holds instead of Assumption 1. The proofs of Lemma 1 and the impossibility result in Proposition 3 carry through without modification. Note that in our proof of Lemma 1, we do not impose that faulty nodes necessarily do not communicate – our result holds regardless of how they behave. Similarly, for the double-spend lemma (Lemma 2) to hold, all we needed to assume about faulty nodes' behavior is that it is *possible* for them to not communicate at all, but that need not be the only behavior that faulty nodes exhibit. Proposition 3 follows from the double-spend lemma and Assumption 2, so it continues to hold in under Assumption 1' as well.

We are then left with proving the existence result in Proposition 3 under Assumption 1'. We will have to construct new consensus algorithms that accommodate the possibility of arbitrary faults, so we will need a new proof for each of the three cases presented in Section G.

H.1 Fault-tolerance proof

Proposition H.1. *Under Assumptions 1' and 2 - 4, there exists a consensus algorithm $(\mathcal{G}, \mathcal{C})$ that is resource-efficient and allocation-efficient.*

Proof. We must show that there exists a consensus algorithm with at least one consensus set that implements all Pareto-efficient outcomes and does not make use of costly messages.

In the proof of Proposition G.1, we showed that there exists a consensus algorithm $(\mathcal{G}, \mathcal{C})$ that achieves both resource-efficiency and full implementation. The consensus algorithm, by construction, had only a single consensus set: the full set of agents $\mathcal{N}$. Whether $\mathcal{N}$ is a consensus set is independent of our assumptions about the behavior of faulty nodes, since by definition, $\mathcal{N}$ is a consensus set if the communication protocol $\mathcal{C}$ is a fault-tolerant equilibrium in $\mathcal{G}_{\mathcal{N}} = \mathcal{G}$ (in which no node is faulty). Hence, $\mathcal{N}$ is also a consensus set of this consensus algorithm under Assumption 1' in place of Assumption 1. Therefore, $(\mathcal{G}, \mathcal{C})$ achieves resource-efficiency and full implementation under Assumption 1' as well. □

H.2 Resource-efficiency proof

Proposition H.2. *Under Assumptions 1' and 2-4, there exists a consensus algorithm $(\mathcal{G}, \mathcal{C})$ that is strongly fault-tolerant and allocation-efficient.*

Proof. **The consensus algorithm**: We use a construction analogous to that in the proof of Proposition G.2. The social choice function is the same as in the previous proof.

The message vocabulary $\mathcal{M}$ is comprised of the vocabulary used in the Filter($\lfloor \frac{N-1}{3} \rfloor$) algorithm $\mathcal{M}^{\text{filt}}$ (described in Algorithm F.4), the vocabulary used in the BT($\lfloor \frac{N-1}{3} \rfloor$) algorithm $\mathcal{M}^{\text{BT}}$ (described in Algorithm F.5), and the vocabulary used in the Revision(T^*) algorithm $\mathcal{M}^{\text{rev}}$ (described in Algorithm F.3), where the constant T^* will be specified later. Hence,

$$\mathcal{M} = \mathcal{M}^{\text{filt}} \cup \mathcal{M}^{\text{BT}} \cup \mathcal{M}^{\text{rev}}.$$

As in the proof of Proposition G.2, the only costly messages are the costly broadcasts in $\mathcal{M}^{\text{rev}}$. For all $m \in \mathcal{M}^{\text{filt}} \cup \mathcal{M}^{\text{maj}}$, then, $\kappa(m) = 0$. The cost to node n of sending a costly broadcast is precisely the same as in Proposition G.2,

$$\kappa(n : 0 : \hat{\boldsymbol{\theta}}) = \sum_{y \in \mathbf{y}(F(\hat{\boldsymbol{\theta}}))} u_n(y|\hat{\theta}_n) + t_n(F(\hat{\boldsymbol{\theta}})).$$

which is positive, because the chosen outcome is assumed to be individually rational. The permissible set of messages M_{nk} that a node may send, and the permissible outcomes D_{nk} on which a node may decide, at each information set (H, θ) is exactly the same as in the proof of Proposition G.2.

The communication protocol specifies that nodes should follow the Filter($\lfloor \frac{N-1}{3} \rfloor$) algorithm (Algorithm F.4), followed by the BT($\lfloor \frac{N-1}{3} \rfloor$) algorithm (Algorithm F.5), and then, once the BT($\lfloor \frac{N-1}{3} \rfloor$) algorithm terminates, they communicate according to the Revision(T^*) algorithm (Algorithm F.3). Once a node has finished the Revision(T^*) algorithm, if its terminal value is $\hat{\boldsymbol{\theta}} \in \prod_{n \in \mathcal{N}} \tilde{\Theta}_n$ (and $\hat{\theta}_n \in \Theta_n$ for $n \in S$), it decides on outcome $x = F_S(\hat{\boldsymbol{\theta}})$. Just as in the proof of Proposition G.2, we define $U_{\max}$ and $\kappa_{\min}$. The constant T^* specifying the number of phases of communication in

Algorithm F.3 is set such that

$$1 - \left(\frac{T^*}{1+T^*}\right)^N = \frac{\kappa_{\min}}{U_{\max}}.$$

Full transferability: By construction, the social choice function implements all individually rational outcomes in each state.

Fault-tolerance: We must show that for any $S \subset \mathcal{N}$ with $|S| > \frac{2}{3}N$, the behaviors specified by the communication protocol (1) are a fault-tolerant equilibrium, (2) allow non-faulty agents to come to a consensus on an outcome consistent with their types, and (3) allow non-faulty agents to come to a consensus on every outcome consistent with their types (with positive probability) as long as they communicate among themselves.

Point (2) follows from the proof of correctness of Algorithm F.5 (Lemma F.2 and Proposition F.3). By the final stage of the algorithm, all agents who follow the protocol have already learned an outcome that is consistent with the types of non-faulty agents. During the revision algorithm, as long as all nodes behave according to the communication protocol, no node changes its value. In the final stage, agents must take the action corresponding to the outcome they agreed upon, so the outcome under this modified algorithm is also consistent with non-faulty agents' types. Point (3) follows from Lemma F.2 and Proposition F.3 as well: Lemma F.2 proves that non-faulty nodes may enter the BT($\lfloor\frac{N-1}{3}\rfloor$) algorithm with any value consistent with their types (with positive probability), and Proposition F.3 shows that if all non-faulty nodes enter the BT($\lfloor\frac{N-1}{3}\rfloor$) with the same value, they ultimately agree on that value.

We next must prove that the prescribed behaviors constitute a fault-tolerant equilibrium for any set S consisting of more than two-thirds of agents. The proof follows along exactly the same lines as the argument in the proof of Proposition G.2. Any deviation that could be pursued by a sub-coalition of non-faulty agents $S' \subset S$, aiming to induce consensus on multiple outcomes, succeeds with probability at most $1 - \left(\frac{T^*}{1+T^*}\right)^N$ (by Proposition F.2). The benefit that an agent in S' would derive from such a deviation is at most $U_{\max}$, and the cost of engaging in the deviation is at least $\kappa_{\min}$. We chose T^* so that

$$\left(1 - \left(\frac{T^*}{1+T^*}\right)^N\right) U_{\max} \leq \kappa_{\min},$$

so it is never profitable for a coalition to engage in a deviation. We can also prove in the exact same way as in Proposition G.2 that no such coalition S' would ever engage in a deviation to induce consensus on a single outcome different from the one that would have occurred if they had behaved honestly.

□

H.3 Full transferability proof

Proposition H.3. *Under Assumptions 1' and ??-4, there exists a consensus algorithm $(\mathcal{F}, \mathcal{G}, \mathcal{C})$ that is strongly fault-tolerant and resource-efficient.*

Proof. This proof will follow along the same lines as the proof of Proposition G.3. The only difference is that we must show non-faulty nodes can reach agreement when faulty nodes are permitted to behave in arbitrary ways (rather than simply not communicating).

The consensus algorithm: The collection of social choice functions is precisely the same as in the proof of Proposition G.3.

The message vocabulary consists of the messages $\mathcal{M}^{\text{filt}}$ used in the Filter($\lfloor \frac{N-1}{3} \rfloor$) algorithm (Algorithm F.4) and the messages $\mathcal{M}^{\text{BT}}$ used in the BT($\lfloor \frac{N-1}{3} \rfloor$) algorithm (Algorithm F.5).

The communication protocol dictates that nodes should communicate according to the Filter($\lfloor \frac{N-1}{3} \rfloor$) algorithm, and when that algorithm terminates, they communicate according to the BT($\lfloor \frac{N-1}{3} \rfloor$) algorithm. Once they finish the BT($\lfloor \frac{N-1}{3} \rfloor$) algorithm, if they have a value $\hat{\boldsymbol{\theta}} \in \tilde{\Theta}$ such that $\hat{\theta}_n \neq \theta^F$ for all $n \in S$, they decide on outcome $x = F(\hat{\boldsymbol{\theta}})$.

Resource-efficiency: Resource costs are not used anywhere in the definition of the message space for the BT($\lfloor \frac{N+1}{3} \rfloor$) communication protocol, so the consensus algorithm is resource-efficient.

Fault-tolerance: We must show that any subset of agents $S \subset \mathcal{N}$ with $|S| > \frac{N}{2}$ is a consensus set. This requires that we show that (1) for any such S, the behaviors specified by the communication protocol are a fault-tolerant equilibrium, (2) whenever such a set S of nodes communicate according to the protocol, they come to a consensus on an outcome consistent with their types, and (3) when nodes in such a set S communicate according to the protocol, they achieve all outcomes in $F_S(\boldsymbol{\theta}_S)$ (for some $\boldsymbol{\theta}_S \in \prod_{n \in S} \Theta_n$) with positive probability.

Points (2) and (3) follow by exactly the same logic as in the proof of Proposition H.2: relative to Proposition G.3, the only new requirement is that non-faulty agents be able to reach consensus no matter how faulty nodes behave, and Proposition H.2 shows that if non-faulty nodes follow the Filter($\lfloor \frac{N-1}{3} \rfloor$) and BT($\lfloor \frac{N-1}{3} \rfloor$) algorithms, they will do so.

It then remains to show that the behaviors prescribed by the communication protocol are a strongly fault-tolerant equilibrium. The proof of this point is exactly as in the proof of Proposition G.3: even if a sub-coalition of non-faulty agents $S' \subset S$ deviates, their payoffs are invariant to the outcome that is realized, so there is no reason for a deviation to occur.

□

I Proofs of results in synchronous settings (Section 6.2)

I.1 Proof of Proposition 2

We first prove our benchmark result, Proposition 2.

Proof. **The consensus algorithm**: Lemma 1 implies that we are given a collection of social choice functions $\mathcal{F}$ that achieve full transferability. Agents use the LSP($\lfloor \frac{N-1}{3} \rfloor$) consensus algorithm (Algorithm F.6, defined in Section F.4) to implement this collection of social choice functions.

Fault-tolerance: We must show that any set of agents S with $|S| > \frac{2}{3}N$ is a consensus set. According to Definition 3', this requires that (1) behavior according to the communication

protocol is a coalition-proof equilibrium in the game $\mathcal{G}_S(\tilde{\boldsymbol{\sigma}}^F)$ for any set of behaviors of faulty nodes $\tilde{\boldsymbol{\sigma}}^F \in \prod_{n \notin S} \Sigma_n$, (2) the consensus reached is consistent with non-faulty agents' initial types, and (3) all outcomes specified by the social choice function F_S are achieved with positive probability (for *all* possible behaviors of faulty nodes).

In the proof of correctness of the LSP algorithm (Proposition F.5), we show that each component $\hat{\theta}_n$ of the final consensus value $\hat{\boldsymbol{\theta}}$ must be consistent with a message $n : \theta$ sent by agent n in the first round. The communication protocol instructs agents to send messages containing their types, so the final consensus value must be consistent with agents' types. Furthermore, if agents in a consensus set S communicate among themselves, the final consensus value will be equal to $F_S(\boldsymbol{\theta}_S)$, where $\boldsymbol{\theta}_S$ denotes the profile of their types.

Note that under the mediated consensus algorithm (Section B.1) and the LSP algorithm, the outcome is a deterministic function of nodes' behavior, since in both cases the algorithm is constructed so that there is no randomness in the order in which messages are received. Furthermore, under both algorithms, no two agents who behave according to the protocol will decide on different outcomes, so no matter how a coalition deviates, only one outcome can occur. Finally, in both cases, if a set of agents S are non-faulty, then the outcome must always be an element of $\mathcal{X}_{\tilde{S}}$ for some $\tilde{S} \supset S$.

Suppose that in the game $\mathcal{G}_S(\tilde{\boldsymbol{\sigma}}^F)$, when the profile of types is $\boldsymbol{\theta} \in \prod_{n \in \mathcal{N}} \Theta_n$, the outcome is $x \in \mathcal{X}_{\tilde{S}}$ under honest behavior. We show that if there exists a deviation from the LSP protocol for a coalition S' under which the outcome is instead x', then under the mediated consensus algorithm, there also exists a game $\mathcal{G}_S(\tilde{\boldsymbol{\sigma}}^{F,M})$ and a deviation for S' that yields outcome x'. Since the deviation is not profitable for S' under the mediated consensus algorithm (by Lemma 1), then it is also unprofitable under the LSP algorithm. In turn, this argument will imply that no coalition has a profitable deviation from the communication protocol.

Consider a game $\mathcal{G}_S(\tilde{\boldsymbol{\sigma}}^F)$ in which, when agents in S behave according to the LSP algorithm and the profile of types is $\boldsymbol{\theta}$, the outcome is $x = F_{\tilde{S}}(\hat{\boldsymbol{\theta}})$ (where $\tilde{S} = \{n : \hat{\boldsymbol{\theta}}'_n \neq \theta^F\}$). We show in the proof of correctness of the LSP algorithm that we must have $\hat{\boldsymbol{\theta}}_n = \theta_n$ for all $n \in S$. Proposition 1 shows that any deviation for a sub-coalition $S' \subset S$ must yield an outcome x' in the set

$$\Gamma(\hat{\boldsymbol{\theta}}|S') = \{\hat{\boldsymbol{\theta}}' : \ \hat{\boldsymbol{\theta}}'_n = \hat{\boldsymbol{\theta}}_n \ \forall \ n \notin S\}.$$

Under the mediated consensus algorithm, agents in S' are free to misreport their initial types θ_n, but they cannot influence the type reported by any other agent. Consider a game $\mathcal{G}_S(\tilde{\boldsymbol{\sigma}}^{F,M})$ in which the outcome would have been $x = F_{\tilde{S}}(\hat{\boldsymbol{\theta}})$ under honest behavior (where $\hat{\boldsymbol{\theta}} \in \prod_{n \in \mathcal{N}} \tilde{\Theta}_n \equiv \prod_{n \in \mathcal{N}} (\Theta_n \cup \{\theta^F\})$). Then for any $\hat{\boldsymbol{\theta}}'$ in the set $\Gamma(\hat{\boldsymbol{\theta}}|S)$ there exists a deviation for coalition S' yielding outcome $x' = F_{\tilde{S}}(\hat{\boldsymbol{\theta}}')$. It is easy to find such a game: set $\theta_n = \hat{\theta}_n$ for $n \in S$ (it does not matter what the types of faulty agents are), and assume that faulty nodes n follow a behavior in which they send message $\hat{\theta}_n$ in the first round of communication.

Therefore, the deviations that are possible under the LSP algorithm are precisely the same as those that are possible under the mediated consensus algorithm. Honest communication according to the LSP algorithm for all $n \in S$ must then be a coalition-proof equilibrium in all games $\mathcal{G}_S(\tilde{\boldsymbol{\sigma}}_F)$.

Resource-efficiency and full transferability: The LSP algorithm (Algorithm F.6) does not make use of resource costs. We also assumed that the algorithm is used to implement a correlated action profile $\mathcal{F}$ given in Lemma 1, which achieves full transferability.

□

I.2 Proof of Proposition 3

We next prove the result on scalability: Proposition 3.

Proof. The argument proceeds in three steps.

1. If a consensus algorithm guarantees that all honest nodes reach agreement regardless of what others do, it must achieve common knowledge of a value $\hat{\boldsymbol{\theta}}$ among all nodes.

2. If k nodes act dishonestly in arbitrary ways, it is generally impossible to guarantee common knowledge in k rounds of communication.

3. If at most k nodes act dishonestly, then common knowledge is achievable among honest nodes in $k+1$ rounds.

The first step shows that the problem of finding a consensus algorithm guaranteeing agreement among honest nodes is equivalent to finding a way for them to achieve common knowledge about a value. The second and third steps show that any such consensus algorithm must achieve agreement in precisely $O(N\Delta)$ rounds of communication when there are N agents.

Preliminaries: First note that under the assumption of synchronous communication (Assumption 4'), we may restrict attention to the case in which $\Delta = 1$, so that all messages are delivered in exactly one round. If $\Delta > 1$, the communication protocol can be designed so that nodes send a message no more than once every Δ rounds, so that messages from the same (honest) node n are never received in different orders by others (say n' and n'').

Consider the following fictitious communication protocol, which resembles a revelation mechanism.

- In the first round, each node n sends every other node a vote $n : \theta_n$, which denotes type θ_n signed by vote n.

- In each subsequent round k, each node n sends every other node the signed message $n : H_{n,k-1}$, where $H_{n,k-1}$ denotes the full history of messages received by n through round $k-1$ (where each message in $H_{n,k-1}$ includes its original signature as well).

Of course, a node n should not be permitted to send a message signed by a different node n', and when it sends a message $n : H_{n,k-1}$, it should be required to provide a proof that every message in $H_{n,k-1}$ was actually received.

If nodes follow this protocol, they reveal all of their information to each other in each round. We can assume without loss of generality that they use this protocol. If nodes used any other protocol, they would necessarily receive coarser information about what others know regarding the final consensus value. Note that this is almost exactly the same as the consensus algorithm in Lamport, Shostak, and Pease (1982), which we describe in Algorithm F.6.

Step 1: A communication protocol $\mathcal{C}$ specifies a rule $d_n(H, \theta)$ mapping a node's information to a decision on an outcome. If the consensus algorithm permits all non-faulty nodes to come to agreement no matter what other nodes do, it must be that

$$d_n(H_{nk}, \theta_n) = x \in \mathcal{X} \;\Leftrightarrow\; d_{n'}(H_{n'k}, \theta_{n'}) = x \;\forall \text{ non-faulty } n'.$$

Let K denote the knowledge operator, so that $K_n(I)$ denotes that n knows event I has occurred. Then $d_n(H_{nk}, \theta_n) = x \Rightarrow K_n(d_{n'}(H_{n'k}, \theta_{n'})) = x$ for all non-faulty n', but then

$$K_{n_1}(K_{n_2}(K_{n_3} \dots (K_{n_m}(d_{n_m}(H_{n_m,k}, \theta_{n_m})) = x))) \dots)$$

for all sequences of non-faulty agents. Therefore, when one non-faulty node knows outcome x, then it is common knowledge among all non-faulty nodes.

Step 2: First, we state what we need to prove more precisely. We want to show that if k nodes act dishonestly, then there exists a strategy for those nodes such that

- Up until round k, no node is known to be faulty.
- All non-faulty nodes know a profile of types $\hat{\boldsymbol{\theta}}$ up to k-th order,

$$K_{n_1}(K_{n_2}, \dots (K_{n_k}(\hat{\boldsymbol{\theta}}))) \;\forall \text{ non-faulty } \{n_1, \dots, n_k\}.$$

- In round $k+1$, agreement on a profile of types $\hat{\boldsymbol{\theta}}$ can be broken at order $k+1$ (along any arbitrary subset of non-faulty nodes). That is, for some non-faulty nodes n,

$$K_n(K_{n_1}, \dots (K_{n_k}(\hat{\boldsymbol{\theta}}))) \;\forall \text{ non-faulty } \{n_1, \dots, n_k\},$$

  whereas for others this does not hold.

We start with the base case. If a node n acts dishonestly in the first round, it can send different messages to different agents. Suppose it sends distinct values θ_n and θ'_n to honest nodes. For a node that receives a profile of types $\hat{\boldsymbol{\theta}}$, the situation is indistinguishable from one in which n actually acted honestly.

Now assume that the statement holds up to k deviating nodes, and consider the case of $k+1$ deviating nodes $n_1, \dots, n_{k+1}$. Consider the following strategy for deviating nodes. Under the inductive assumption, there exists a strategy for nodes $n_1, \dots, n_k$ such that (1) all nodes $n \notin \{n_1, \dots, n_k\}$ know a profile of types $\hat{\boldsymbol{\theta}}$ up to k-th order, (2) there is a set of messages $\{m_{n_1,n_{k+1}}, \dots, m_{n_k,n_{k+1}}\}$

that $\{n_1, \ldots, n_k\}$ could send to n_{k+1} so that n_{k+1} knows $\hat{\boldsymbol{\theta}}$ up to $k+1$-th order, and there is a different set of messages $\{m'_{n_1,n_{k+1}}, \ldots, m'_{n_k,n_{k+1}}\}$ that $\{n_1, \ldots, n_k\}$ could send to n_{k+1} so that n_{k+1} does not know $\hat{\boldsymbol{\theta}}$ at $k+1$-th order. That is, by sending messages $\{m_{n_1,n_{k+1}}, \ldots, m_{n_k,n_{k+1}}\}$ in round $k+1$, n_{k+1} would have a history $H_{n_{k+1},k+1}$ of messages such that $K_{n_{k+1}}(K_{j_1}, \ldots (K_{j_k}(\hat{\boldsymbol{\theta}})))$ for all non-faulty $j_1, \ldots, j_k$, and by sending messages $\{m'_{n_1,n_{k+1}}, \ldots, m'_{n_k,n_{k+1}}\}$, n_{k+1} would have a history $H'_{n_{k+1},k+1}$ of messages such that $\sim K_{n_{k+1}}(K_{j_1}, \ldots (K_{j_k}(\hat{\boldsymbol{\theta}})))$ for all non-faulty $j_1, \ldots, j_k$.

In round $k+1$, nodes $n_1, \ldots, n_k$ send all non-faulty nodes, as well as node n_{k+1}, messages $\{m_{n_1,n_{k+1}}, \ldots, m_{n_k,n_{k+1}}\}$. Hence, all non-faulty nodes know $\hat{\boldsymbol{\theta}}$ up to order $k+1$. However, they also send node n_{k+1} the messages $\{m'_{n_1,n_{k+1}}, \ldots, m'_{n_k,n_{k+1}}\}$. they would have sent to get that node to agree on $\mathbf{a}$ at order $k+1$ as well as the messages that would result in that node disagreeing with $\mathbf{a}$ at order $k+1$. Denote these histories of messages by $H_{n_{k+1},k+1}$ and $H'_{n_{k+1},k+1}$. Note that up until round $k+1$, node n_{k+1} has not yet deviated. Furthermore, no node in $\{n_1, \ldots, n_k\}$ has communicated inconsistent messages to any non-faulty node, so by the beginning of round $k+2$, no faulty nodes have been identified. Therefore, the first two points hold.

In round $k+2$, node n_{k+1} may send some non-faulty nodes the history $H_{n_{k+1},k+1}$ and others the history $H'_{n_{k+1},k+1}$. Any node that receives $H_{n_{k+1},k+1}$ will know $\hat{\boldsymbol{\theta}}$ up to order $k+2$, since all nodes receiving that message will know that any non-faulty node knows $\hat{\boldsymbol{\theta}}$ up until level $k+1$. However, honest nodes that receive history $H'_{n_{k+1},k+1}$ will not know $\hat{\boldsymbol{\theta}}$ up to level $k+2$. They do not yet have evidence that n_{k+1} is faulty, since that node acted honestly up until round $k+1$ but sent a message with a history $H'_{n_{k+1},k+1}$ indicating that it did not know $\hat{\boldsymbol{\theta}}$ up to level $k+1$. This proves the third point. Therefore, it is impossible for honest nodes to achieve common knowledge on a set of votes in $k+2$ rounds.

Step 3: Now we show that the proposed communication protocol achieves common knowledge in $k+1$ rounds when there are k faulty nodes. Consider any two non-faulty nodes n and n'. It follows from Proposition F.5 that any two non-faulty nodes will reach agreement on a value consistent with the types of all non-faulty nodes after $k+1$ rounds of communication. Hence, the value held by a non-faulty node in round $k+1$ is actually common knowledge among all non-faulty nodes.

Proof of scalability property: Having proven Steps 1-3, we can now show that, given any consensus algorithm that permits non-faulty nodes to come to a consensus no matter how others behave, that algorithm must take $|N|\Delta$ rounds of communication. Step 1 establishes that a consensus algorithm must establish common knowledge about a value $\hat{\boldsymbol{\theta}}$. Step 2 shows that if there are at most k dishonest nodes, then it is impossible to achieve common knowledge before round $k+1$. This implies that when the consensus algorithm is required to tolerate faults by up to $\lfloor \frac{N-1}{3} \rfloor$ nodes (as required by the strong fault-tolerance property), common knowledge cannot be achieved before round $|\lfloor \frac{N-1}{3} \rfloor|\Delta$. Finally, Step 3 shows that there exists a consensus algorithm that achieves common knowledge in round $|\lceil \frac{N-1}{3} \rceil|\Delta$.

□

References

Aumann, R., and S. Hart (2003): "Long Cheap Talk," *Econometrica*, 71(6), 1619–1660.

Bracha, G., and S. Toueg (1985): "Asynchronous Consensus and Broadcast Protocols," *Journal of the ACM*, 32(4), 824–840.

Feldman, P., and S. Micali (1997): "An Optimal Probabilistic Protocol for Synchronous Byzantine Agreement," *SIAM Journal on Computing*, 26(4), 873–933.

Lamport, L., D. Shostak, and M. Pease (1980): "Reaching Agreement in the Presence of Faults," *Journal of the ACM*, 27(2), 228–234.

——— (1982): "The Byzantine Generals Problem," *ACM Transactions on Programming Languages and Systems*, 4(3), 382–401.

Linial, N. (1994): "Game-Theoretic Aspects of Computing," in *Handook of Game Theory with Economic Applications*, ed. by R. Aumann, and S. Hart, pp. 1339–1395. Elsevier.