Blockchain Economics*†

Joseph Abadi‡ Markus Brunnermeier§

May 2022

Abstract

The fundamental problem in digital record-keeping is establishing *consensus* on an update to a ledger, e.g., a payment. Consensus must be achieved in the presence of *faults*—situations in which some computers are offline or fail to function appropriately. Traditional centralized record-keeping systems rely on *trust* in a single entity to achieve consensus. Blockchains decentralize record-keeping, dispensing with the need for trust in a single entity, but some instead build a consensus based on the wasteful expenditure of computational resources (proof-of-work). An ideal method of consensus would be tolerant to faults, avoid the waste of computational resources, and implement efficient transfers of value among agents. We prove a Blockchain Trilemma: any method of consensus, be it centralized or decentralized, must give up (i) fault-tolerance, (ii) resource-efficiency, or (iii) allocative efficiency.

Keywords: Blockchain, Consensus, Cryptocurrency, Mechanism Design, Payments, FinTech
JEL Codes: D80, E42, G00

*We are grateful for insightful discussions by Gur Huberman and Will Cong, helpful comments from Zhiguo He, Rohit Lamba, Stephen Morris, Ulrich Müller, Arvind Narayanan, Jonathan Payne, Wolfgang Pesendorfer, Fahad Saleh, Raphael Auer, conference participants at the SF Fed Fintech Conference, the Cambridge University Conference on the Economics of Distributed Ledger Technology, the San Francisco Blockchain Week CESC, the CEBRA Annual Meeting, the UCLA Anderson Fink Center Conference on Financial Markets, the Harvard CMSA Blockchain Conference, the AFA Annual Meeting, the NYU Five Star Conference, the University of Chicago Cryptocurrencies and Blockchains Conference, the St. Louis Fed, and the NYU Conference on Financial Intermediation, and seminar participants at FRB Philadelphia, the SEC, Yale University, the Wharton School at the University of Pennsylvania, the Princeton University Department of Computer Science, the Princeton University Department of Economics, and the BIS.

†**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 (joseph.abadi@phil.frb.org)
§Princeton University (markus@princeton.edu)

1 Introduction

Trust is of central importance in record-keeping. A ledger represents an agreed-upon history of events and must therefore remain free of tampering or fraud. However, record-keepers are rarely inherently trustworthy, and they often face the temptation to profit by altering the ledger. For record-keepers to be trusted, then, they must be provided with incentives to behave honestly.

Traditionally, record-keeping has been centralized: a single entity is given full power to update the ledger however it desires, and it derives some long-term benefit from its privileged position (e.g., by extracting rents). The record-keeper can be trusted to communicate honestly with the ledger's users, ensuring they remain in agreement about the ledger's contents, only if these long-run incentives are strong enough. Blockchain technology has provided a radical decentralized alternative to record information. Public blockchains seek to minimize reliance on centralized trust: there is no single authority who keeps records. For instance, Bitcoin uses a voting method based on proof-of-work (PoW), in which power to update the ledger is allocated based on the expenditure of computational resources, whereas other blockchains use proof-of-stake (PoS), which instead allocates voting power to token holders (called "validators"). Although there is no centralized entity who can extract rents, these decentralized systems often have costs of their own: large energy expenditures are required to maintain PoW blockchains, and many PoS systems restrict validators' transactions by requiring them to lock up their tokens as collateral. These costs are typically motivated as *necessary* features of mechanisms that incentivize honest record-keeping.[1] Hence, the advent of blockchains raises important new questions regarding the fundamental tradeoffs in the design of digital record-keeping systems.

We take the view that the central problem in digital record-keeping is ensuring agents reach a consensus on updates to a ledger in the presence of faults – situations in which some communication devices, e.g., computers, are offline or do not function appropriately. We therefore study the design of record-keeping protocols that permit agents with non-faulty communication devices to keep a record of a sequence of transactions. Crucially, in an economic environment, there is no *inherently* trustworthy agent who can be relied upon to truthfully relay information to others: all agents must be incentivized to follow the prescribed protocol. From a design perspective, two main questions arise: In the absence of an inherently trustworthy record-keeper, how can record-keeping be designed to incentivize honest reporting? What types of constraints does a lack of trust impose on the types of outcomes that can be implemented?

We posit three ideal features of a record-keeping protocol. First, such a protocol should be *fault-tolerant*: it should allow agents who communicate honestly to reach agreement even when some fraction of agents have faulty devices that fail to communicate (or send messages erratically). Faults are fundamental to digital communication: computers may be temporarily disconnected from the

[1] Budish (2018) argues that PoW blockchains are secure only if record-keepers' computational expenditures are sufficiently large. Gans and Gandal (2019) observe that the same is true for PoS systems with free entry of validators: the blockchain will be secure only if validators are required to stake large amounts of collateral. Indeed, they demonstrate that the costs of staking collateral in PoS systems should not be expected to be lower than those of PoW.

network or suffer from programming errors, so any digital record-keeping system must be robust to these considerations. Second, a record-keeping protocol should achieve *allocative efficiency*: it should allow agents to implement a Pareto-efficient allocation (i.e., sequence of transactions). Third, a record-keeping protocol should ideally be *resource-efficient*, meaning it does not force agents to solve intrinsically useless computational problems and incur a deadweight loss of resources through proof-of-work. We investigate the conditions under which these three ideal features can be achieved by a protocol that incentivizes honest behavior.

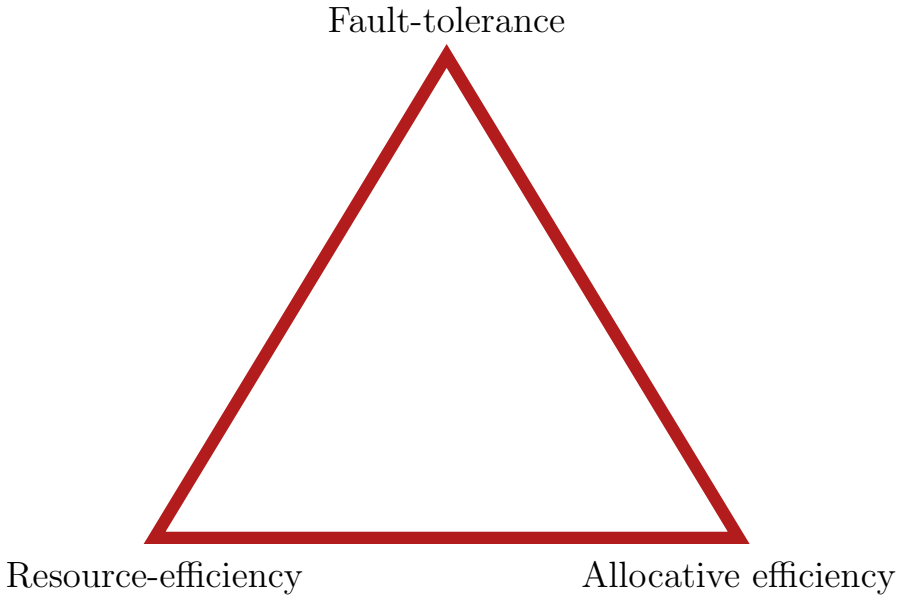

Figure 1: The Blockchain Trilemma.

We begin by adapting the theory of distributed consensus from the computer science literature (e.g., Lamport, Shostak, and Pease, 1980) to an economic setting. We impose typical assumptions on agents' communication technology and the behavior of faulty devices. Unlike previous work in computer science, though, we do not assume the existence of any agent who will automatically act honestly: the provision of incentives is key to our theory. Our consensus framework applies to a broad variety of digital record-keeping arrangements: it can accommodate centralized ledgers as well as PoW and PoS blockchains.

Our main contribution is a Blockchain Trilemma (Figure 1): we prove that under our model's assumptions, no communication protocol can simultaneously achieve fault-tolerance, resource-efficiency, and allocative efficiency. We also prove a converse: any two of the desired properties are achievable by *some* communication protocol. Hence, there is a sense in which our framework allows us to tightly characterize the role of trust, which is one of the main questions that originally motivated the development of public blockchains (Nakamoto, 2008). When no agent can be trusted to behave honestly, it is necessary to either give up fault-tolerance (which is usually infeasible in digital systems) or sacrifice one of two types of efficiency: allocative efficiency or resource-efficiency. As we will show, giving up fault-tolerance renders dishonest behavior detectable, whereas incentives

for honesty can be provided by giving up resource-efficiency or allocative efficiency.

1.1 The distributed record-keeping problem

We begin with a brief overview of the classical treatment of the distributed record-keeping problem in computer science and then describe how we adapt it to an economic environment.

In the classical treatment, there are N computers (called "nodes"). Each node n possesses a "ledger," which consists of a sequence of entries $\{y_{n1}, \ldots, y_{nK}\}$ from some set $\mathcal{Y}$. The objective is to design a communication protocol that permits nodes to (1) propose new entries in the ledger, and (2) update their ledgers while ensuring that they all remain consistent with one another.

One naïve solution is the following: nodes should simply add entries to their ledgers in the order that they are received. If nodes are guaranteed to receive proposals in the same order, this solution clearly achieves the objective. A second naïve solution is based on unanimous voting. A node n should add an entry y to its ledger in position $K+1$ only if it has received confirmation from *all* other nodes that they will do the same. Under this protocol, the ledgers of two well-functioning (honest) nodes can never conflict.

In practical settings, however, these simple solutions are rendered infeasible by the presence of communication frictions. First, nodes are typically assumed to send each other bilateral, private messages that are delivered with a random time lag. The first naïve solution then no longer guarantees consistency: two honest nodes might not receive proposals in the same order. Second, at each point in time, some subset of nodes are *faulty*. Faulty nodes do not follow the prescribed communication protocol: some models assume they are assumed to be incapable of communicating, and other models assume that they may behave in arbitrary and erratic ways (called *Byzantine* faults). The presence of faulty nodes rules out a unanimous voting scheme, since such nodes might not vote. Finally, the literature often assumes that communication among nodes is *asynchronous*, meaning that nodes do not have access to synchronized clocks. For subtle reasons that we will discuss later on, this seemingly innocuous friction renders distributed record-keeping substantially more difficult.

Given these main ingredients, the distributed record-keeping problem can be formulated as exemplified by Lamport (1998): the objective is to find a record-keeping protocol such that

1. (**Eventual consensus**) Every entry that is added to the ledger of one honest node is eventually added to the ledgers of all honest nodes;

2. (**Fault-tolerance**) If sufficiently many nodes are honest (i.e., non-faulty), then each honest node is eventually able to update its ledger (with probability one).

Importantly, note that in this formulation, non-faulty nodes are assumed to automatically follow the protocol.

We adapt this problem to capture distributed record-keeping in an economic environment. Our model features N players in a continuous-time setting. Over time, players communicate with one another to propose and reach agreement on a sequence of transactions y in a set $\mathcal{Y}$, each of which

may have a different set of participants. The game ends when *all* players have reached agreement on a *terminal state*, which is a split of a fixed surplus. Agreement on a terminal state can be thought of as an eventual consensus on a state of a ledger, where the split of surplus represents players' final "balances" on the ledger. A *record-keeping protocol* in our environment is a communication protocol specifying how these balances should be allocated after each history of play.

Our model incorporates the communication frictions typically used in the distributed record-keeping literature. Players send one another bilateral, private messages that are subject to random (but bounded) lags. A minority of players may be faulty, meaning that they are incapable of communicating or agreeing to transactions during the course of play – faulty players may only participate at the end of the game, by agreeing to a terminal state. Thus, faulty players can be viewed as those who are offline during the stage of play when transactions take place. Finally, players communicate in an asynchronous environment, which we capture by assuming that they lack common knowledge of the current time t. In addition to these frictions, we also allow for the possibility that some messages are costly to send. This permits our model to capture record-keeping systems such as proof-of-work, in which participants must perform costly computations to keep records.

Differently from the computer science literature, non-faulty players in our environment cannot simply be assumed to follow any prescribed record-keeping protocol, since they may profit by deviating from it. Players will follow a record-keeping protocol only if the strategies it prescribes constitute an equilibrium. Given that designers of distributed record-keeping systems in practice are often concerned about attacks by coalitions that control a substantial fraction of the network, we use a coalition-proof solution concept.

With this game-theoretic framework in hand, we define the desired properties of a record-keeping protocol. The notion of eventual consensus from the computer science literature is inherent in our definition of a record-keeping protocol, which guarantees players will eventually reach agreement on a terminal state if they behave according to the prescribed strategies. Fault-tolerance, on the other hand, is a desideratum in our environment: a record-keeping protocol is fault-tolerant if the prescribed strategies constitute an equilibrium whenever *some* majority of players are non-faulty. To this standard condition, we add two additional desired features: resource-efficiency, meaning that the protocol does not make use of costly messages, and allocative efficiency, meaning that the protocol implements efficient allocations (with positive probability).

We make a final assumption on preferences, which implies that any restriction on transfers of ledger balances among players necessarily precludes allocative efficiency. This logic is in line with that typically used in monetary economics, in which, e.g., restrictions on the quantity of money agents can transfer to one another will prevent certain transactions from taking place. Under this assumption, we obtain the two results constituting the Trilemma. First, there does not exist a record-keeping protocol achieving fault-tolerance, resource-efficiency, and allocative efficiency. Second, for any two of the three desired properties, there exists a record-keeping protocol achieving both. Importantly, this result implies a role for communication costs: if these costs are calibrated

correctly, it is possible to design a record-keeping protocol that achieves both fault-tolerance and allocative efficiency.

1.2 Related literature

Our paper is mainly related to three strands of the academic literature. First, our paper follows a long tradition in economics that studies communication in games with and without trusted mediators. Forges (1986) and Myerson (1986) illustrate how a revelation principle applies to communication mechanisms in games with trusted mediators. The literature on implementation theory (e.g., Maskin, 1999, among many others) studies how games with trusted mediators can be designed to implement desirable social outcomes. Eliaz (2002) extends this work to a setting in which some agents may be "faulty" and communicate in unpredictable ways, like ours. Our paper's main point of departure from this extensive literature is the assumption that agents lack a trusted mediator.

Of course, there are other papers in economics that study unmediated communication in games. Forges (2020) provides an extensive treatment of the subject.[2] Relative to the literature on unmediated communication, we differ in our assumptions regarding communication frictions – namely, that agents may have faulty communication devices and that communication takes time. As our results show, these realistic assumptions are consequential for our conclusions.

Second, our paper is related to the literature on distributed consensus in computer science. Lamport, Shostak, and Pease (1980) is the seminal paper in the study of synchronous consensus algorithms that tolerate "Byzantine" faults. We focus on a setting with asynchronous communication, studied by Bracha and Toueg (1985), Fischer, Lynch, and Paterson (1985), and Castro and Liskov (1999), among others. In the existence result of our Trilemma, we build on the consensus algorithms derived by this earlier literature.[3] Our main contribution in this area is to adapt the distributed consensus literature to an economic setting. We require that consensus algorithms be designed not only to protect against non-strategic faults in communication, but also to disincentivize coordinated deviations by strategic agents. Concurrently with our work, Halaburda, He, and Li (2021) provide an economic analysis of equilibrium multiplicity under the Byzantine Fault Tolerance (BFT) algorithm developed by Castro and Liskov (1999).

Finally, our work is related to the emergent literature on the economic properties of blockchains and cryptocurrencies. Auer, Monnet, and Shin (2021) study the optimal design of fees for permissioned ledgers in a framework where verification of transactions is a common good. Budish (2018) and Gans and Gandal (2019) study the costs of incentivizing honesty for cryptocurrency blockchains in isolation. Biais et al. (2019) propose a game-theoretic model of Bitcoin mining and show that while the strategy of mining the longest chain proposed by Nakamoto (2008) is in fact an equilibrium, there are other equilibria in which the blockchain forks, as observed empirically. Chiu and Koeppl (2019) derive optimal incentive-compatible blockchain updating protocols for proof-

[2] See also Aumann and Hart (2003), Ben-Porath (1998, 2003), and Gerardi (2004), which study the correlated equilibria that may arise.

[3] The impossibility result of Fischer, Lynch, and Paterson (1985) does not apply to our setting because we assume the lags in message delivery are random and cannot be manipulated by malicious attackers.

of-work-based systems in a model of monetary exchange. Saleh (2020) shows how proof-of-stake blockchains may guarantee security through fluctuations in token prices. Relative to the previous literature, we study a unified record-keeping framework and outline the requirements that must be satisfied by *any* record-keeping system in order to ensure security.

1.3 Outline

The remainder of the paper is structured as follows. We present our model of digital record-keeping in Section 2. In the model, we describe a class of communication games in which agents attempt to reach consensus on updates to a ledger. We introduce our concept of a consensus algorithm and define the fault-tolerance, resource-efficiency, and full transferability properties. Then, we outline our model's main assumptions, allowing us to prove the Trilemma in Section 3. Section 4 discusses our main assumptions on the communication technology and formally extend the model to incorporate relaxations of those assumptions. Section 5 concludes.

2 Model

2.1 Environment

The model is populated by a finite set of players $\mathcal{N} = \{1, \ldots, N\}$ who do not discount payoffs. Time t is continuous, running from zero until players reach agreement on a *terminal state*, representing a split of a fixed surplus $V > 0$ among them. A terminal state is thus $\mathbf{v} = (v_1, \ldots, v_N) \in \mathbb{Z}_+^N$ with $\sum_{n=1}^{N} v_n \leq V$.[4] There is a finite set of *transactions* $y \in \mathcal{Y}$ that can occur before a terminal state is reached. A transaction y is associated with a set of *participants* $S(y) \subset \mathcal{N}$ and a payoff for each participant, $u_n(y)$ for $n \in S(y)$.

At $t = 0$, Nature randomly draws a subset $\mathcal{Y}^F \subset \mathcal{Y}$ of *feasible transactions* and a subset $\mathcal{F} \subset \mathcal{N}$ of *faulty players*, whose role will be fully defined momentarily.[5] We will refer to $(\mathcal{Y}^F, \mathcal{F})$ as an *instance* of the game among players.[6] A *feasible allocation* is a sequence $\{y_{t_1}, y_{t_2}, \ldots, y_{t_K}, \mathbf{v}_T\}$ of feasible transactions with non-faulty participants occurring at times $t_1 \leq t_2 \leq \cdots \leq t_K$, with a terminal state $\mathbf{v}_T$ being realized at T.

Agents would like to decide on an allocation, but there is no central mediator who can choose one for them. Transactions and terminal states are realized as part of a *communication game*. At each instant t, players perform two types of actions. First, they may send bilateral, private messages to others. Some messages may be costly to send, as is common in distributed record-keeping. Second, players can *agree* to an outcome or a terminal state. When an outcome y is agreed to by all participants $n \in S(y)$, then that outcome (and the payoffs $u_n(y)$) are realized. A terminal state $\mathbf{v}$, by contrast, is realized only when *all* agents agree to it, ending the game.

[4] Given that we consider a finite set of terminal states, it is without much loss of generality to assume the entries of $\mathbf{v}$ are integers.

[5] We will use a robust equilibrium concept that will not require us to specify a distribution over these draws.

[6] We use this terminology rather than "state of the world" to avoid confusion with "terminal state."

Communication is hindered by the presence of two frictions, as summarized below.

Assumption 1. *We assume the following two communication frictions.*

1. *Messages are delivered with an I.I.D.* ***random lag*** *of maximum length* Δ*;*
2. ***Faulty players*** *cannot communicate or agree to transactions.*

Faulty players may only participate in agreeing to a terminal state – they cannot send messages or agree to transactions occurring during the course of the game. Thus, faulty players can be interpreted as users of a ledger who are presently offline but will participate in the future after learning the state of the ledger. When some players are faulty, it will be infeasible for non-faulty players to wait for unanimous acknowledgment of a transaction before agreeing to it. Delays in message delivery, in turn, will make it difficult for non-faulty players to determine who is faulty.

It remains to fully specify the set of histories in this game and players' information. A *history* h_t consists of the $t = 0$ draw of $(\mathcal{Y}^F, \mathcal{F})$ and the ordered sequence of actions taken by agents and messages sent through time t. The set of feasible transactions $\mathcal{Y}^F$ is common knowledge. A player n knows whether she is faulty and has perfect recall of her *local history*: the sequence of her own past actions, the messages she has received, and the realized transactions y in which she participated (i.e., $n \in S(y)$). However, players do not know whether others are faulty, and they do not observe messages sent to others or transactions in which they do not participate. We make a final key assumption about players' information: they do not have synchronized clocks, as formalized below.

Assumption 2. *Players know neither the times at which events in their local history occurred nor the current time* t*.*

In the theory of distributed consensus, this assumption is often referred to as *asynchronicity*. As we will discuss, agreement in an asynchronous communication environment is substantially more difficult than consensus in a synchronous one (in which players know t).

2.2 Communication game and equilibrium

Now we turn to a more formal account of the *communication game* $\mathcal{G}$, which consists of

1. A message vocabulary $\mathcal{M}$ as well as a cost $\kappa(m) \geq 0$ for each $m \in \mathcal{M}$;
2. A set of messages $M_{nt} \subset \mathcal{M}$ that player n is permitted to send at time t, and a set of outcomes $Y_{nt} \subset \mathcal{Y}$ and terminal states V_{nt} to which player n is permitted to agree at time t, which may depend on the set of messages received by n through time t;
3. Payoff functions
$$U_n = \sum_{k=1}^{K} u_n(y_{t_k}) + v_{nT} - \sum_{m \in \hat{M}_n^S} \kappa(m) \tag{1}$$
where $\{y_{t_1}, \ldots, y_{t_k}, \mathbf{v}_T\}$ is the realized allocation and $\hat{M}_n^S$ denotes the set of messages sent by n over the course of the game.

A (pure) strategy σ_n for an agent n consists of a plan of action at each local history, specifying which message to send to each other player n' and whether to agree to an outcome or terminal state. A profile of strategies for all players is denoted $\boldsymbol{\sigma}$. For our solution concept, we use a coalition-proof notion of equilibrium. In particular, we impose the Strong Nash Equilibrium concept defined in Aumann (1959), which permits deviating players to make binding agreements with one another.

Definition 1. *A profile of strategies $\boldsymbol{\sigma}$ is a **equilibrium in instance** $(\mathcal{Y}^F, \mathcal{F})$ if for all coalitions of non-faulty players, $S \subset \mathcal{N} \backslash \mathcal{F}$, there does not exist an alternative profile of strategies $\tilde{\boldsymbol{\sigma}}_S$ s.t.*

$$\begin{aligned} \textit{In instance } (\mathcal{Y}^F, \mathcal{F}): \quad \mathbb{E}[U_n | \tilde{\boldsymbol{\sigma}}_S, \boldsymbol{\sigma}_{-S}] &\geq \mathbb{E}[U_n | \boldsymbol{\sigma}] \ \forall \ n \in S \\ &> \mathbb{E}[U_n | \boldsymbol{\sigma}] \textit{ for some } n \in S. \end{aligned}$$

Coalition-proofness is an important requirement in our setting. Typically, designers of distributed ledgers have in mind attacks by groups of bad actors attempting to subvert the system and design security measures accordingly.[7] If coalitions were not permitted to form, it would be trivial to design a communication game that would render any deviation from honesty unprofitable. We use Strong Nash Equilibrium rather than the more usual Coalition-Proof Equilibrium of Bernheim, Peleg, and Whinston (1987) because designers of digital record-keeping systems often have in mind situations in which the deviating coalition consists of agents under the control of a single overarching entity, e.g. a large group of miners (a "mining pool") attempting to conduct an attack on a proof-of-work blockchain.

2.3 A restriction on the class of communication games

Note that, according to our definitions, the set of actions that a player may take at time t depends on their identity as well as the history of previous messages received. For instance, a message m that can only be sent by player n could be interpreted as one that requires n's (unforgeable) cryptographic signature. Similarly, it may be the case that a player cannot send some message m before receiving a different message m'. This permits agents to *prove* that others have previously sent certain messages to them. Additionally, a player might not be able to agree to a transaction y before receiving a message m, since m might constitute a proposal in favor of a particular transaction that must be received prior to agreement. This framework is flexible enough to capture all of the systems of proofs used by distributed record-keeping systems in practice.

We now impose some additional structure on these requirements.

Assumption 3. *Let R_{nt} be the set of messages received by n through time t. Then the permissible message sets M_{nt} take the form $M(n, R_{nt})$, where $M(n, R) \subset M(n, R')$ if $R \subset R'$. The same restriction applies to the sets of transactions (terminal states) Y_{nt} (V_{nt}) to which n may agree at t.*

[7] For instance, the well-known "51% attack" faced by proof-of-work blockchains requires that the attacking coalition control the majority of the network's computing power. The "long-range attack" that is possible in proof-of-stake systems requires collusion among users who held a supermajority of voting power at some point in the past.

Under this assumption, players are always free to engage in lies of omission: they can pretend not to have received messages that were sent to them in the past.

3 The Blockchain Trilemma

In what follows, we will be interested in the following question: which allocations can arise in an equilibrium $\boldsymbol{\sigma}$ of *some* communication game $\mathcal{G}$? We will be interested in studying the case where $\boldsymbol{\sigma}$ is a *record-keeping protocol*, under which agents are guaranteed to receive a promised value that is updated in response to events occurring over the course of the communication game.

Definition 2. *Strategy profile $\boldsymbol{\sigma}$ is a **record-keeping protocol** of game $\mathcal{G}$ if for each history of events h_t, there exist values $\boldsymbol{v}(h_t)$ summing to V such that when players act according to $\boldsymbol{\sigma}$, then*

$$\forall\ (\mathcal{Y}^F, \mathcal{F})\ \textit{s.t.}\ \boldsymbol{\sigma}\ \ \textit{is an equilibrium}: \sum_{t_k > t} u_n(y_{t_k}) + v_{nT} \geq v_n(h_t)\ \forall\ n\ \textit{almost surely.} \qquad \text{(PK)}$$

A record-keeping protocol can be viewed as a process for updating a ledger shared among players. The ledger records the values v_{nt} promised to players at each moment in time, which can be viewed as their "balances" on the ledger. As events unfold, balances are transferred among the players. Note that when agents behave according to a record-keeping protocol, then each transaction y is associated with a transfer $\mathbf{t} = (t_1, \ldots, t_N)$ with entries summing to zero: if players were promised values $\mathbf{v}$ before transaction y is realized, then after y occurs, their balances are updated to $\mathbf{v} + \mathbf{t}$.

We view the communication game $\mathcal{G}$ as an object that can be designed: for instance, the designer of a blockchain-based record-keeping system can require that users provide certain types of proofs when sending transaction requests or approving new additions to the ledger. Then, a record-keeping protocol $\boldsymbol{\sigma}$ of the game could be viewed as a communication protocol that the game's designer recommends to users of the system. We introduce three desiderata.

Definition 3. *A game $\mathcal{G}$ and record-keeping protocol $\boldsymbol{\sigma}$ are said to achieve:*

1. ***Fault-tolerance** if $\boldsymbol{\sigma}$ is an equilibrium of $\mathcal{G}$ in all instances $(\mathcal{Y}^F, \mathcal{F})$ such that a majority of players are non-faulty;*
2. ***Resource-efficiency** if the strategy profile $\boldsymbol{\sigma}$ does not involve the use of costly messages m with $\kappa(m) > 0$;*
3. ***Allocative efficiency** if, in any instance $(\mathcal{Y}^F, \mathcal{F})$ such that $\boldsymbol{\sigma}$ is an equilibrium, then a Pareto-efficient allocation[8] is realized with positive probability.*

The fault-tolerance requirement implies that as long as all players n believe a majority of the others to be non-faulty, then they will have no incentive to deviate from the prescribed communication

[8]Formally, a feasible allocation is *Pareto-efficient* in instance $(\mathcal{Y}^F, \mathcal{F})$ if there does not exist another feasible allocation that (1) is weakly preferred by all players, and (2) is strictly preferred by some player.

protocol $\boldsymbol{\sigma}$. The protocol therefore represents a robust equilibrium in the sense that players can entertain different beliefs about who, specifically, may be faulty among the other players. Resource-efficiency is clearly desirable because the costs paid to send messages in this model represent a pure deadweight loss. Finally, players would always like to implement a Pareto-efficient allocation, hence our allocative efficiency criterion.

Our final assumption will allow us to derive our main result. We define a *transfer* $\mathbf{t} = (t_1, \ldots, t_N)$ among a subset of players S to be a vector of the form $\mathbf{v} - \mathbf{v}'$, where $\mathbf{v}$ and $\mathbf{v}'$ are terminal states such that $v_n = v'_n$ for all $n \notin S$. We assume that for any two subsets of players S and S', given a transfer among those players, there exists a mutually beneficial transaction y among them such that the disutility incurred by each $n \in S'$ is exactly offset by the transfer t_n.

Assumption 4. *Let S, S' be disjoint subsets of players, and let $\boldsymbol{t} \in \mathbb{Z}^N$ be a transfer among $S \cup S'$. Then there exists a transaction y with participants $S \cup S'$ such that (1) $u_n(y) \geq -t_n$ for all $n \in S \cup S'$ (strictly for some n), and (2) $u_n(y) = -t_n$ for $n \in S'$.*

Under our interpretation of record-keeping as the process of updating a ledger, Assumption 4 implies that if transfers of value on the ledger are restricted in some way, it is necessarily the case that some mutually beneficial transactions will not be realized in equilibrium. Hence, restrictions on transfers will imply a violation of the allocative efficiency property.

We can now state our main result, the Blockchain Trilemma.

Theorem 1 (Blockchain Trilemma)**.** *Under Assumptions 1-4, the following hold:*

- ***(Impossibility)** There does not exist a communication game $\mathcal{G}$ and a record-keeping protocol $\boldsymbol{\sigma}$ satisfying fault-tolerance, resource-efficiency, and full transferability.*
- ***(Existence)** For any two of the desired properties in Definition 3, there exists a communication game $\mathcal{G}$ and a record-keeping protocol $\boldsymbol{\sigma}$ achieving both properties.*

We will sketch an argument for the impossibility result before moving on to its proof below. The constructions involved in the proof of the existence result are quite complex (but well-known in the computer science literature), so we postpone a proof to the Appendix.

The Trilemma effectively states that, when restrictions of transfers of value among players impede efficiency (Assumption 4), then a record-keeping system with the communication technology specified in Assumptions 1-3 must either give up fault-tolerance or one of two types of efficiency: resource efficiency (a waste of resources in updating the ledger) or allocative efficiency (restrictions on the transactions that can be implemented in equilibrium).

The impossibility result: Our argument for the impossibility result proceeds by contradiction. We assume the existence of a communication game and a record-keeping protocol achieving fault-tolerance, resource-efficiency, and allocative efficiency. Fix two majorities of players S and S', and let $A = S \cap S'$, $B = S \backslash S'$, and $C = S' \backslash S$. Assumption 4 guarantees the existence of a transaction y among $A \cup B$ for which players in A must transfer their initial promised value to

players in B, and a corresponding transaction y' among $A \cup C$ for which players in A must transfer their entire promised value to players in C.

Fault-tolerance implies that playing according to the protocol must be an equilibrium when players in C are faulty, since $A \cup B$ is a majority. Allocative efficiency implies that when players in $A \cup B$ are non-faulty and only transactions y and y' are feasible, then $A \cup B$ must agree to transaction y with positive probability, and all of A's promised value is transferred to B. Similarly, fault-tolerance and allocative efficiency imply that when players in $A \cup C$ are non-faulty, $A \cup C$ must agree to y' with positive probability. The key intermediate step in our argument is to show that under Assumptions 1-3, there exists a deviation for A such that when $A \cup B \cup C$ are non-faulty, then *both* y and y' are realized with positive probability. Players in A effectively spend their promised value twice: they send it to players in B, and then again to players in C. This deviation is intimately related to the "double-spend problem" faced by distributed ledgers in practice, whose relation to our model we discuss later on.

Therefore, if the profile $\boldsymbol{\sigma}$ is indeed an equilibrium, incentives must be provided against this type of "double-spending." There are two ways to provide incentives: players in A can face an ex-ante cost for engaging in a deviation, or they can face an ex-post punishment once their dishonesty is uncovered. There are ex-ante costs for deviation if messages are costly to send. However, since we have assumed resource-efficiency, it is not possible that players in A face this type of cost. Ex-post punishments are also impossible, since players in A agreed to transfer away their entire promised value in transaction y. We then derive a contradiction by assuming fault-tolerance, resource-efficiency, and allocative efficiency.

The existence result: This argument illustrates the logic behind the existence result as well. In a fault-tolerant system, incentives against dishonest behavior can be provided either by imposing costs in communication or by imposing ex-post punishments, reducing the terminal payoff of players who misbehave. Heuristically, for honesty to be incentive-compatible for a coalition of agents $n \in A$ who stand to gain utility U_n^D through double-spending, it must be that

$$U_n^D \leq \underbrace{\kappa_n^D}_{\text{Ex-ante cost}} + \underbrace{v_n^D - v_n^H}_{\text{Ex-post punishment}} \quad \text{for some } n \in A, \tag{2}$$

where κ_n^D is the communication cost incurred by n during the deviation and v_n^D (v_n^H) represents n' share of the final surplus when acting dishonestly (honestly). Imposing ex-ante communication costs requires giving up resource-efficiency, of course. It is possible to punish agents ex-post only if they are unable to transfer away their entire promised value to others during the course of the game ($v_n^H > 0$). Such restrictions on transfers of value among agents result in a loss of allocative efficiency by Assumption 4. On the other hand, if the fault-tolerance requirement is dropped, it is possible to design a communication protocol that entirely prevents profitable dishonest behavior ($U_n^D = 0$), so it is not necessary to give up resource-efficiency or allocative efficiency.

In our proof of the existence result, we use this logic to construct communication games and record-keeping protocols satisfying each pair of desired properties. By doing so, we uncover a role

for communication costs. Our impossibility result demonstrates that in the absence of such costs, there exist pairs of Pareto-efficient allocations that cannot both be implemented in equilibrium, since that would allow some coalition to profitably deviate from honest behavior. By introducing communication costs, such deviations are rendered unprofitable, thereby allowing a greater range of allocations to be implemented in equilibrium.

3.1 The key lemma

We begin our proof with the key intermediate step, which is a lemma demonstrating the scope for dishonest behavior under a fault-tolerant record-keeping protocol. In terms of our ledger analogy, we demonstrate the existence of a deviation that allows a coalition A to "double-spend" their balances on the ledger: they can be sent once to some group of players B and once to another group C.

More specifically, the lemma takes as given two distinct majorities of players S and S' and a set of feasible transactions $\mathcal{Y}^F$. We show that if sequence of transactions $s = \{y_{t_1}, \ldots, y_{t_K}\}$ is realized with positive probability when players in S are non-faulty, and if sequence $s' = \{y'_{t'_1}, \ldots, y'_{t'_{K'}}\}$ is implemented with positive probability when players in S' are non-faulty, then there exists a deviation for players $A = S \cap S'$ such that the transactions in *both* s and s' are realized (with positive probability). A formal statement of the lemma follows.

Lemma 1. *Fix $\mathcal{Y}^F \subset \mathcal{Y}$ and distinct majorities of players S, S'. Suppose that under a strategy profile $\boldsymbol{\sigma}$, allocation $\{y_{t_1}, \ldots, y_{t_k}, \boldsymbol{v_T}\}$ (resp. $\{y'_{t'_1}, \ldots, y'_{t'_{K'}}, \boldsymbol{v'_{T'}}\}$) is realized with positive probability in instance $(\mathcal{Y}^F, \mathcal{N}\backslash S)$ (resp. instance $(\mathcal{Y}^F, \mathcal{N}\backslash S')$).*

Then under Assumptions 1-3, there exists a joint deviation from $\boldsymbol{\sigma}$ for players $S \cap S'$ such that in instance $(\mathcal{Y}^F, \mathcal{N}\backslash(S \cup S'))$, transactions $y_{t_1}, \ldots, y_{t_k}$ are realized, followed by transactions $y'_{t'_1}, \ldots, y'_{t'_{K'}}$ (with probability $p \in (0, 1)$).

Proof sketch. We sketch the proof here, leaving some technical details to the Appendix. We will denote $A = S \cap S'$, $B = S\backslash S'$, and $C = S'\backslash S$. Throughout, we will denote a history of events by h, and a player's local history will be denoted h_n.

Under Assumption 2, players do not know the current time or the time at which any events in their local history occurred. Hence, players' actions are determined purely by the ordered sequence of events they have observed (i.e., messages they have sent, messages they have received, and the realization of transactions in which they have participated). In particular, when players behave according to $\boldsymbol{\sigma}$, the set of realized transactions depends only on the order in which messages are delivered among players. Assumption 1 specifies that messages are delivered with an I.I.D. lag with full support on $[0, \Delta]$, so *any* order of message deliveries among non-faulty players is possible.

Our hypotheses imply that when players behave according to $\boldsymbol{\sigma}$ and players in $A \cup B$ are non-faulty, there exists an order of message deliveries among $A \cup B$ that leads to the realization of the sequence of transactions $s = \{y_{t_1}, \ldots, y_{t_k}\}$. The argument above implies that when players in $A \cup B \cup C$ are non-faulty, it is possible for these messages to be delivered in precisely the same order

among players in $A \cup B$, *before* any messages are delivered to or from players in C. Let $q \in (0,1)$ denote the probability with which this occurs.

By the same token, when players $A \cup B \cup C$ are non-faulty and behave according to $\boldsymbol{\sigma}$, there exists an order in which messages can be delivered among $A \cup C$ (*before* any messages are delivered to or from B) such that the sequence of transactions $s' = \{y'_{t'_1}, \ldots, y'_{t'_{K'}}\}$ is realized. Let $q' \in (0,1)$ denote this probability.

We will consider the following deviation for players $n \in A$:

- If the sequence of transactions $s = \{y_{t_1}, \ldots, y_{t_k}\}$ has not occurred, play according to $\sigma_n(h_n)$.
- After the sequence of transactions s has occurred, play according to $\sigma(\tilde{h}_n)$, where $\tilde{h}_n$ is the sub-sequence of events in h_n that
  1. Omits all events that occurred before the realization of the sequence of transactions s;
  2. Omits all messages received from players in B.

That is, players in A behave honestly until sequence s is realized. Then, they communicate with C as if they have never communicated with players in B. By Assumption 3, this deviation is feasible for players in A. The assumption implies that the allowable set of actions is increasing in the set of messages previously received. Here, players in A simply omit received messages.

With probability q, sequence s is realized and players in A engage in the deviation described above. At this point, players in C are unable to distinguish the situation from one in which players in B are faulty, since they have not yet received any messages from those players. Players in A begin to communicate with players in C as if they, too, have never received a message from B. Thus, until players in C receive a message from B, communication between players in $A \cup C$ from this point forward proceeds precisely as it would if both were following $\boldsymbol{\sigma}$ and had not yet heard from B. Our argument above implies that under this behavior, the sequence s' of transactions is realized with probability q'.

Sequence s followed by sequence s' is therefore realized with probability $p = qq' > 0$.

□

3.2 The impossibility result

We can now prove the Trilemma's impossibility result. The key lemma will imply that under a fault-tolerant record-keeping protocol, there will exist a coalition of agents who can spend their balances on the ledger multiple times. When the record-keeping protocol achieves resource-efficiency, these agents do not incur a cost for deviating ex-ante. Allocative efficiency will imply that they can initially transfer their entire balance on the ledger away, so it will not be possible to punish them ex-post.

Proof of the impossibility result. We proceed by contradiction. Suppose that there exists a communication game $\mathcal{G}$ with a record-keeping protocol $\boldsymbol{\sigma}$ satisfying fault-tolerance, resource-efficiency,

and allocative efficiency. Let S, S' be two distinct majorities of players, and denote $A = S \cap S'$, $B = S - S'$ and $C = S' - S$. Since S and S' are majorities, the fault-tolerance property implies the protocol $\boldsymbol{\sigma}$ is an equilibrium in any instance such that the set of non-faulty agents is S (resp. S').

Suppose that under protocol $\boldsymbol{\sigma}$, the initial values promised to players are given by a vector $\mathbf{v}^0 = (v_1^0, \ldots, v_N^0)$. Let $\mathbf{t}$ be a transfer among players in S such that A gives their entire promised value to B, $t_n = -v_n^0$ for all $n \in A$. Similarly, let $\mathbf{t}'$ be a transfer among players in S' such that A gives their entire promised value to C. Assumption 4 implies the existence of a transaction y among players in S such that $u_n(y) \geq t_n$ for all $n \in A$ (strictly for some n) and $u_n(y) = -t_n$ for $n \in B$. There also exists an analogous transaction y' among players in S'.

Step 1. *Consider the instance $(\{y, y'\}, \mathcal{N} \backslash S)$ in which only transactions y and y' are feasible and only players in S are non-faulty. Then allocation $\{y, \boldsymbol{v}^0 + \boldsymbol{t}\}$ is realized with positive probability (and likewise for $\{y', \boldsymbol{v}^0 + \boldsymbol{t}'\}$ in instance $(\{y, y'\}, \mathcal{N} \backslash S')$).*

In instance $(\{y, y'\}, \mathcal{N} \backslash S)$, allocation $\{y, \mathbf{v}^0 + \mathbf{t}\}$ is Pareto-efficient and satisfies the promise-keeping constraint on payoffs implied by Definition 2. Suppose that $\{y, \mathbf{v}^0 + \tilde{\mathbf{t}}\}$ is another such allocation, where $\tilde{\mathbf{t}}$ is a transfer among players in S. We must have $\tilde{t}_n \geq t_n$ for all $n \in B$, since by assumption $u_n(y) = -t_n$ for $n \in B$. Furthermore, $\tilde{t}_n \geq t_n$ for $n \in A$, since players in A must receive non-negative surplus $v_n^0 + \tilde{t}_n$ at the end of the game (and $t_n = -v_n^0$ for $n \in A$). Therefore, $\tilde{\mathbf{t}}$ must be equal to $\mathbf{t}$, so $\{y, \mathbf{v}^0 + \tilde{\mathbf{t}}\}$ is the only Pareto-efficient allocation that respects the promise-keeping constraint on payoffs. Given that $\boldsymbol{\sigma}$ achieves allocative efficiency, then in this instance, that allocation is realized with positive probability. By the same token, in instance $(\{y, y'\}, \mathcal{N} \backslash S')$, allocation $\{y', \mathbf{v}^0 + \mathbf{t}'\}$ is realized with positive probability.

This result allows us to prove the next key step in our argument, where we make use of our assumptions on the communication technology available to agents.

Step 2. *Suppose Assumptions 1-3 hold, as well as the result of Step 1. In instance $(\{y, y'\}, \mathcal{N} \backslash (S \cup S'))$, there exists a deviation from the protocol $\boldsymbol{\sigma}$ for players in A such that*

- *If transaction y is realized (which happens with positive probability q), then transaction y' is also realized with positive probability q'.*
- *If transaction y is not realized, then players in A receive exactly the same payoffs they would have if all players had behaved according to $\boldsymbol{\sigma}$.*

These claims follow directly from lemma 1. We have shown that there exists a deviation for A leading to the realization of both y and y' with positive probability. The proof of the lemma also specifies that under this deviation, if y is not realized, then players in A behave according to the prescribed profile $\boldsymbol{\sigma}$, so their payoffs are identical to those under honest behavior.

Step 3. *Transactions y and y' cannot both be realized on the equilibrium path.*

Consider some history h_t such that that y and y' have already occurred. The total utility realized by players in B (C) up until that point is $u_n(y) = -t_n$ ($u_n(y') = -t_{n'}$). Thus, in order for the

promise-keeping constraint to be satisfied, it must be that in the terminal state $\mathbf{v}$, $u_n(y) + v_n \geq v_n^0$ for all $n \in B$ (and similarly, $u_n(y') + v_n \geq v_n^0$ for all $n \in C$). But this implies:

$$\begin{aligned}\sum_{n \in B \cup C} v_n &\geq \sum_{n \in B \cup C} (v_n^0 - u_n(y)) \\ &= \sum_{n \in B} (v_n^0 + t_n) + \sum_{n \in C} (v_n^0 + t'_n) \\ &= \left(\sum_{n \in B} v_n^0 + \sum_{n \in A} v_n^0 \right) + \left(\sum_{n \in C} v_n^0 + \sum_{n \in A} v_n^0 \right) \\ &= V + \sum_{n \in A} v_n^0 > V.\end{aligned}$$

The third line uses the fact that the entries of $\mathbf{t}$ and $\mathbf{t}'$ sum to zero, so $\sum_{n \in B} t_n = -\sum_{n \in A} t_n = \sum_{n \in A} v_n^0$ (and similarly for $\sum_{n \in C} t_n$). Thus, the amount of surplus that would have to be provided to $B \cup C$ in the terminal state exceeds the total surplus V.

Step 4. *In instance $(\{y, y'\}, \mathcal{N} \backslash (S \cup S'))$, coalition A receives a higher payoff in (in expectation) by pursuing the deviation described in Step 2 rather than the protocol $\boldsymbol{\sigma}$.*

Given that the record-keeping protocol $\boldsymbol{\sigma}$ is assumed to be resource-efficient, players in A do not have to incur any cost in sending the messages required to pursue the deviation described above. Hence, each $n \in A$ must receive utility of at least $u_n(y) + u_n(y')$. On the other hand, if players in A behave honestly, then after the realization of transaction y, some terminal state $\mathbf{v}$ must be realized such that $v_n^0 \leq u_n(y) + v_n$ for all $n \in A \cup B$ and $v_n^0 \leq v_n$ for $n \in C$. Recalling that $\sum_{n \in B} u_n(y) = -\sum_{n \in A} v_n^0$, it must be that $\sum_{n \in B} v_n \geq \sum_{n \in A \cup B} v_n^0$. Thus, $\sum_{B \in C} v_n \geq V$, so $v_n = 0$ for all $n \in A$. By succeeding in their deviation, players in A get payoff $u_n(y) + u_n(y') > u_n(y)$, which is the payoff they would have received under honest behavior.[9]

By assuming the existence of a communication game $\mathcal{G}$ and a communication protocol $\boldsymbol{\sigma}$ satisfying fault-tolerance, resource-efficiency, and allocative efficiency, we have derived a deviation for coalition A that performs better than $\boldsymbol{\sigma}$, a contradiction. Therefore, it is not possible for all three desiderata to be achieved simultaneously.

□

4 Discussion

In this section, we discuss our model's key elements and assumptions as well as their connection to distributed record-keeping in practice.

[9] Note that $u_n(y') \geq -t'_n$ and $t'_n < 0$ for each $n \in A$ implies $u_n(y') > 0$ for all $n \in A$.

4.1 The environment and record-keeping protocols

Our environment features a set of players who must come to an agreement on a sequence of transactions, which can be thought of as analogous to the addition of entries to a ledger in the distributed record-keeping problem. We are interested in record-keeping protocols that *guarantee* payoffs to each player after any history of messages and transactions. In this sense, when players behave according to the protocol, they are jointly updating a ledger: anyone who knows the full history of events knows exactly the value promised to each player. Differently from the classic distributed record-keeping problem, however, the full vector of promised payoffs is not necessarily known by any individual. For instance, it is possible that a player knows only her own promised payoff but not those of other players.

The game ends when players reach an agreement on a terminal state $\mathbf{v}$, which in our environment is a division of surplus. This terminal state could also represent an agreement on a vector of payoffs in some continuation game: the game ends when players attain common knowledge of the continuation equilibrium that will be played and its payoffs $\mathbf{v}$. Therefore, $\mathbf{v}$ can alternatively be thought of as final state of the ledger (hence our terminology).

Our environment places two key restrictions on payoffs. The most significant is Assumption 4, which is key to our results. It effectively states that *any* restriction on the terminal states that can be reached necessarily implies that some efficient allocations cannot be implemented in equilibrium. Second, we restrict the Pareto frontier of terminal payoffs to take the form of a simplex $\mathcal{V} = \{\mathbf{v} \in \mathbb{Z}_+^N : \sum_{n=1}^{N} v_n \leq V\}$. For other types of (compact) Pareto frontiers, our results would continue to hold if Assumption 4 were suitably adjusted.

4.2 Communication

Our model imposes four assumptions about the communication technology: the assumption of bilateral, private messages and Assumptions 1-3. All four assumptions are typical in the literature on distributed consensus.

Assumption 1 implies that (1) messages are delivered with a bounded, I.I.D. lag, and (2) there are some faulty players who cannot communicate or agree to transactions. The substantive restrictions implied by our first assumption are that messages are eventually delivered and that there is some randomness in message delivery. Of course, eventual message delivery is necessary if we are to obtain our existence result (which relies on eventual agreement among non-faulty players). More subtly, randomness is necessary to circumvent the impossibility result of Fisher, Lynch, and Paterson (1985), which demonstrates that in an asynchronous *deterministic* system, agreement on an update to a ledger is impossible.

Assumption 2 is central to our results insofar as players do not have *perfectly* synchronized clocks, i.e., t is not common knowledge. We could, however, allow for players to observe "local clocks" that are not fully synchronized. When players have perfectly synchronized clocks, though, it is indeed possible to design a record-keeping protocol that achieves fault-tolerance, resource-

efficiency, and allocative efficiency. In the computer science literature, asynchrony is considered the most appropriate assumption in the case of large-scale public networks (e.g., the internet).

Finally, the logic embedded by Assumption 3 is that players cannot prove they *have not* received certain messages: they are always free to engage in lies in omission. Nevertheless, Assumption 3 allows for a rich enough class of communication games to capture the types of record-keeping systems used in practice. Players can be required to sign messages with unforgeable signatures (corresponding, e.g., to the "public key" cryptography used by blockchain systems). They can also be required to provide a proof that they *have* received certain messages: this allows players to potentially prove the dishonest behavior of others, as is typical in some proof-of-stake systems. We explicitly allow for proof-of-work (as in Bitcoin) by allowing for costs in communication.

4.3 The solution concept

Our solution concept incorporates two important considerations: a notion of robustness to the presence of faulty players and a notion of coalition-proofness. Recall that players are not assumed to know who is faulty, so a strategy profile $\boldsymbol{\sigma}$ cannot specify actions that are contingent on the set of faulty players. If $\boldsymbol{\sigma}$ is a fault-tolerant strategy profile, though, the actions it prescribes must be optimal as long as *any* majority of players are non-faulty. Thus, equilibrium is robust to the possibility that players entertain different beliefs, or have different information, about the set of faulty players: they have incentives to play according to $\boldsymbol{\sigma}$ as long as there is common knowledge that a majority are non-faulty.

In the context of distributed record-keeping, it is imperative to consider some notion of coalition-proofness. Indeed, the most important security concerns faced by distributed record-keeping systems typically involve collusion among a significant fraction of network participants. For instance, proof-of-work blockchain protocols are constructed to discourage "51%" attacks, in which a coalition of users who control the majority of the network's computing power collude to double-spend their tokens. Proof-of-stake blockchains must typically put incentives in place against a situation in which a supermajority of "validators" (i.e., record-keepers) cooperate in order to double-spend. Importantly, our equilibrium concept permits players in the deviating coalition to make binding commitments to a particular deviation. This assumption is appropriate in the context we consider because, in distributed record-keeping systems, it is not atypical for many of the network's users to be acting on behalf of a single entity. For example, Bitcoin miners are often employed by "pools" that can comprise a significant fraction of the network's computing power.

5 Conclusion

The fundamental problem in digital record-keeping is to develop a method by which agents can reach an agreement on an update to a ledger in the presence of faults. When an inherently trustworthy mediator is available, this problem is easy to solve: the mediator can report updates to the ledger to all of its users, guaranteeing they remain in agreement about the ledger's contents.

When record-keepers are self-interested, however, they have incentives to equivocate and report mutually inconsistent outcomes to different sets of agents. In an economic environment, then, the provision of incentives for honesty is central to the digital record-keeping problem.

We have developed a framework to study the design of record-keeping protocols in a setting without trustworthy agents (in contrast to the computer science literature, which assumes certain nodes will act honestly). Our main result is a Blockchain Trilemma, which proves that any record-keeping protocol must give up fault-tolerance, resource-efficiency, or allocative efficiency. The central tension faced by the designer is between fault-tolerance and incentive provision. The designer can eliminate the need for incentives by giving up on the fault-tolerance requirement, but in many important digital record-keeping applications, it is unrealistic to assume that agents' computers will always function properly. Hence, the designer must provide either ex-ante incentives for agents to behave honestly (by giving up resource-efficiency) or impose ex-post punishments on dishonest agents (by giving up allocative efficiency).

There are two promising directions future research might take. First, we have not characterized constrained-optimal record-keeping protocols. The nature of such a protocol would likely be quite specific to the relevant application, so we have not analyzed that problem in our general model. Second, in mechanism design, it is typical to study implementation protocols that guarantee *all* equilibria of a given mechanism yield a desirable outcome, while in our analysis, the designer is content with specifying a single equilibrium (i.e., communication protocol) that yields a desirable outcome. Future work could study the problem of designing communication games without "bad" equilibria, as implementation theory has done.

Bibliography

Abreu, D., D. Pearce, and E. Stacchetti (1990): "Toward a Theory of Discounted Repeated Games with Imperfect Monitoring," *Econometrica*, 58(5), 1041–1063.

Auer, R., C. Monnet, and H. Shin (2021): "Permissioned Distributed Ledgers and the Governance of Money," *Working Paper*.

Aumann, R. (1959): "Acceptable Points in General Cooperative n-Person Games," in *Contributions to the Theory of Games IV*, pp. 321–353. Princeton University Press.

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

Ben-Or, M. (1983): "Another Advantage of Free Choice: Completely Asynchronous Agreement Protocols," in *Proceedings of the Second Annual ACM Symposium on Principles of Distributed Computing*, pp. 27–30.

Ben-Porath, E. (1998): "Correlation without Mediation: Expanding the Set of Equilibrium Outcomes by 'Cheap' Pre-Play Procedures," *Journal of Economic Theory*, 80(1), 108–122.

——— (2003): "Cheap Talk in Games with Incomplete Information," *Journal of Economic Theory*, 108(1), 45–71.

Bernheim, B. D., B. Peleg, and M. Whinston (1987): "Coalition-Proof Nash Equilibria I. Concepts," *Journal of Economic Theory*, 42(1), 1–12.

Biais, B., C. Bisière, M. Bouvard, and C. Casamatta (2019): "The Blockchain Folk Theorem," *Review of Financial Studies*, 32(5), 1662–1715.

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

Budish, E. (2018): "The Economic Limits of Bitcoin and the Blockchain," *Working paper*.

Castro, M., and B. Liskov (1999): "Practical Byzantine Fault Tolerance," *OSDI*, pp. 173–186.

Chiu, J., and T. Koeppl (2019): "The Economics of Cryptocurrencies – Bitcoin and Beyond," *Bank of Canada Staff Working Paper 2019-40*.

Cong, L. W., and Z. He (2019): "Blockchain Disruption and Smart Contracts," *Review of Financial Studies*, 32(5), 1754–1797.

Cong, L. W., Y. Li, and N. Wang (2021): "Tokenomics: Dynamic Adoption and Valuation," *Review of Financial Studies*, 34(3), 1105–1155.

Easley, D., M. O'Hara, and S. Basu (2019): "From Mining to Markets: The Evolution of Bitcoin Transaction Fees," *Journal of Financial Economics*, 134(1), 91–109.

Eliaz, K. (2002): "Fault Tolerant Implementation," *Review of Economic Studies*, 69(3), 589–610.

Fischer, M., N. Lynch, and M. Paterson (1985): "Impossibility of Distributed Consensus with One Faulty Process," *Journal of the ACM (JACM)*, 32(2), 374–382.

Forges, F. (1986): "An Approach to Communication Equilibria," *Econometrica*, 54(6), 1375–1385.

——— (2020): "Correlated Equilibria and Communication in Games," in *Complex Social and Behavioral Systems: Game Theory and Agent-Based Models*, pp. 107–118.

Gans, J., and N. Gandal (2019): "More (or Less) Economics Limits of the Blockchain," *NBER Working Paper 26534*.

Garay, J., and A. Kiayias (2020): "SoK: A Consensus Taxonomy for the Blockchain Era," in *Cryptographers' Track at the RSA Conference*, ed. by J. S. Springer.

Gerardi, D. (2004): "Unmediated Communication in Games with Complete and Incomplete Information," *Journal of Economic Theory*, 114(1), 104–131.

Gilad, Y., R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich (2017): "Algorand: Scaling Byzantine Agreements for Cryptocurrencies," in *SOSP '17: Proceedings of the 26th Symposium of Operating Systems Principles*, pp. 51–68.

Halaburda, H., Z. He, and J. Li (2021): "An Economic Model of Consensus on Digital Ledgers," *NBER Working Paper 29515*.

Huberman, G., J. Leshno, and C. Moallemi (2021): "Monopoly without a Monopolist: An Economic Analysis of the Bitcoin Payment System," *Review of Economic Studies*, 88(6), 3011–3040.

Lamport, L., D. Shostak, and M. Pease (1980): "Reaching Agreement in the Presence of Faults," *Journal of the ACM (JACM)*, 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 *Handbook of Game Theory with Economic Applications*, ed. by R. Aumann, and S. Hart, pp. 1339–1395. Elsevier.

Maskin, E. (1978): "Implementation and Strong Nash Equilibrium," *Unpublished manuscript*.

——— (1999): "Nash Equilibrium and Welfare Optimality," *Review of Economic Studies*, 66(1), 23–38.

Myerson, R. (1986): "Multistage Games with Communication," *Econometrica*, 54(2), 323–358.

Nakamoto, S. (2008): "Bitcoin: A Peer-to-Peer Electronic Cash System," *Whitepaper*.

Narayanan, A., J. Bonneau, E. Felten, A. Miller, and S. Goldfeder (2016): *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press.

Pagnotta, E. (2022): "Decentralizing Money: Bitcoin Prices and Blockchain Security," *Review of Financial Studies*, 35(2), 866–907.

Rubinstein, A. (1989): "The Electronic Mail Game: Strategic Behavior Under 'Almost Common Knowledge'," *American Economic Review*, 79(3), 385–391.

Saleh, F. (2020): "Blockchain without Waste: Proof-of-Stake," *Review of Financial Studies*, 34(3), 1156–1190.

Schilling, L., and H. Uhlig (2019): "Some Simple Bitcoin Economics," *Journal of Monetary Economics*, 106, 16–26.

Schneider, F. (1990): "Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial," *ACM Computing Surveys (CSUR)*, 22(4), 299–319.

Sockin, M., and W. Xiong (2021): "A Model of Cryptocurrencies," *NBER Working Paper 26816*.

Toueg, S. (1984): "Randomized Byzantine Agreements," in *PODC '84: Proceedings of the Third Annual ACM Symposium on Principles of Distributed Computing*, pp. 163–178.

A Proof of the double-spend lemma

A.1 Preliminaries

First, we prove some preliminary results on communication in an asynchronous environment (Assumption 2). We will define an *order of message deliveries* to be a sequence $o = \{(n_1, n_1'), (n_2, n_2'), \ldots, (n_K, n_K')\}$ of pairs in $\mathcal{N}^2$, denoting a sequence of events in which a message sent by n_1 was received by n_1', and then a message from n_2 was received by n_2', and so on.

Lemma 2. *Suppose players behave according to a strategy profile $\boldsymbol{\sigma}$. Under Assumption 2, the realized history of events h_t at time t depends only on the order of message deliveries prior to t.*

Proof. Assumption 2 imposes that players do not know the current time or the time at which events in their local histories occurred. Therefore, players' local histories h_{nt}, which determine their behavior at time t, are simply ordered sequences of events (consisting of messages sent, messages received, and transactions realized). A player's actions (messages sent and transactions agreed to) then depend only on the order in which messages were received. Hence, the entire realized history of events depends only on the order of message delivery. □

A consequence of the previous lemma is that any history of events that can be realized when a subset S of players are non-faulty can also be realized when a subset $S' \supset S$ are non-faulty.

Lemma 3. *Let h be a history that is realized with positive probability in instance $(\mathcal{Y}^F, \mathcal{N}\backslash S)$ when players follow strategy profile $\boldsymbol{\sigma}$. Under Assumptions 1 and 2, in any instance $(\mathcal{Y}^F, \mathcal{N}\backslash S')$ for $S' \supset S$, agents $n \in S$ reach local histories h_n with positive probability.*

Proof. By Lemma 3, in any instance with feasible transactions $\mathcal{Y}^F$, the local histories h_n of players in S through time t depend only on the order in which messages are delivered before t. Let o be an order of message deliveries (in instance $(\mathcal{Y}^F, \mathcal{N}\backslash S)$) resulting in local histories h_n for each player $n \in S$. Assumption 1, which states that message deliveries are I.I.D. according to a distribution with full support on $[0, \Delta]$, implies that with positive probability, order o is realized in instance $(\mathcal{Y}^F, \mathcal{N}\backslash S')$. Thus players in S reach local histories h_n in that instance with positive probability. □

A.2 Proof of the lemma

Lemma 4. *Fix $\mathcal{Y}^F \subset \mathcal{Y}$ and distinct majorities of players S, S'. Suppose that under a fault-tolerant profile $\boldsymbol{\sigma}$, allocation $\{y_{t_1}, \ldots, y_{t_k}, \boldsymbol{v_T}\}$ (resp. $\{y'_{t'_1}, \ldots, y'_{t'_{K'}}, \boldsymbol{v'_{T'}}\}$) is realized with positive probability in instance $(\mathcal{Y}^F, \mathcal{N}\backslash S)$ (resp. instance $(\mathcal{Y}^F, \mathcal{N}\backslash S')$).*

Then under Assumptions 1-3, there exists a joint deviation from $\boldsymbol{\sigma}$ for players $S \cap S'$ such that

- *With probability $p \in (0, 1)$, transactions $y_{t_1}, \ldots, y_{t_k}$ are realized, followed by transactions $y'_{t_1}, \ldots, y'_{t_k}$ (with probability $p \in (0, 1)$).*
- *With probability $1 - p$, the payoffs of players in A are precisely those received under profile $\boldsymbol{\sigma}$.*

Proof. Denote $A = S \cap S'$, $B = S \backslash S'$, and $C = S' \backslash S$. Since S and S' are distinct majorities of players, all three sets are nonempty. Furthermore, by the assumption that $\boldsymbol{\sigma}$ is fault-tolerant, it must be an equilibrium both when agents in S are non-faulty and when agents in S' are non-faulty.

In what follows, we will denote a history of the game at time t by h and a player n's local history by h_n. By Assumption 2, agents do not know the time at which events occurred, so their local histories are simply ordered sequences of events (each of which is the sending of a message, the receipt of a message, or the realization of a transaction).

Suppose that, in instance $(\mathcal{Y}^F, \mathcal{N} \backslash S)$, the sequence of transactions $s = \{y_{t_1}, \ldots, y_{t_K}\}$ are realized after a history of events $h(s)$, corresponding to local histories $h_n(s)$ for each $n \in S$. We will assume that players $n \in A$ behave according to the following strategy:

- If local history $h_n(s)$ has not been realized, behave according to $\sigma_n(h_n)$, where h_n is n's local history.
- If local history $h_n(s)$ has been realized, behave according to $\sigma_n(\tilde{h}_n)$, where $\tilde{h}_n$ denotes the sub-history of h_n that
  1. Excludes all events in $h_n(s)$;
  2. Excludes all messages received from players in B.

The strategy described above is obtained by *ignoring* certain events that occurred in the past. Hence, it is feasible for agents in A by Assumption 3, which states that the set of feasible messages and decisions for an agent in A is increasing in the set of messages previously received.

We will show that, by following this strategy, players in A can cause the sequence of transactions s to be realized, followed by the sequence of transactions $s' = \{y'_{t'_1}, \ldots, y'_{t'_{K'}}\}$. Note that lemma 3 immediately implies that under this deviation, the sequence of transactions s is realized with positive probability q (say, finishing at some time t_s). In particular, this can occur if no message sent by or to a player in C is delivered prior to t_s.

After time t_s, players in A behave as if they have not previously received any messages, and they ignore any messages received from B. By assumption, players in C have not received any messages either. Hence, players in $A \cup C$ behave as they would at the beginning of the game, before the delivery of any messages. Lemma 3 then implies that, with positive probability p', an order of message deliveries among $A \cup C$ is realized such that the sequence of transactions s' takes place.

Then the probability that both sequences of transactions s and s' are realized is $pp' > 0$, as desired. □

B Proof of the existence result

In this section, we describe the record-keeping protocols prescribed in our proof of the existence result. The record-keeping protocols we construct will all share a common simplifying

property: in the strategies described, players will first learn an entire individually rational allocation $\{y_1, \ldots, y_K, \mathbf{v}\}$ (with $\sum_{n=1}^{N} v_n = V$ and $\sum_{n=1}^{N} \sum_{k' \geq k} u_n(y_{k'}) \geq 0$ for all k)[10] such that for all n and $k \leq K$,

$$\sum_{k' \geq k} u_n(y_k) + v_n \geq 0.$$

Then, after learning this allocation, players agree to transactions in sequence. First, all $n \in S(y_1)$ agree to y_1, then $n \in S(y_2)$ agree to y_2, and so on.

Note, further, that any such strategy profile constitutes a record-keeping protocol, so long as (1) every player eventually learns the same allocation x, and (2) any allocation $x = \{y_1, \ldots, y_K, \mathbf{v}\}$ to which players may agree satisfies the property

$$\sum_{n \in S(y_k)} u_n(y_k) + v_n \geq v_{0n}$$

for some $\mathbf{v}_0$ with $\sum_{n=1}^{N} v_{0n} = V$. After the realization of y_k, the value that players will receive in aggregate is

$$\sum_{k' \geq k} \sum_{n \in S(y_k)} u_n(y_k) + V \geq V$$

by our assumptions above. Hence, since the allocation is individually rational, there exist non-negative promised values v_{kn} with $\sum_{n=1}^{N} v_{kn} = V$ that can be promised to players after this history.

Our proof of the existence result consists of the construction of three different communication games and record-keeping protocols: one that achieves resource-efficiency and allocative efficiency (but not fault-tolerance), one that achieves fault-tolerance and resource-efficiency (but not allocative efficiency), and one that achieves fault-tolerance and allocative efficiency (but not resource-efficiency). In the first case, we will prove that the record-keeping protocol is an equilibrium in any instance such that no player is faulty, meaning the result is not vacuous. In the second and third cases, we will show that the prescribed protocol is an equilibrium in any instance such that a strict majority of players are non-faulty.

First, however, we outline the communication protocol that agents will follow after learning an allocation, since this procedure will be the same across our three proofs.

---

[10] Observe that all Pareto-efficient and individually rational allocations satisfy this property. If $\sum_{n=1}^{N} \sum_{k' \geq k} u_n(y_{k'}) < 0$, then there exists an individually rational allocation making all players better off. The transactions $\{y_{k'}\}_{k' \geq k}$ are removed from the allocation, and the terminal payoffs v_n of players n with $\sum_{k' \geq k} u_n(y_{k'}) < 0$ are reduced to $v_n + \sum_{k' \geq k} u_n(y_{k'})$ (which is greater than zero by individual rationality). The additional surplus $-\sum_{k' \geq k} u_n(y_{k'})$ can be split among the remaining players so that they all prefer the resulting allocation (since $\sum_{n=1}^{N} \sum_{k' \geq k} u_n(y_{k'}) < 0$).

B.1 The quorum algorithm

We define the message space and communication protocol used in the *quorum algorithm*, which is the procedure by which players implement an allocation after they have learned which allocation will be implemented. A player n begins with knowledge of an allocation $x = \{y_1, \ldots, y_K, \mathbf{v}\}$. To each allocation x is associated a *quorum* $Q_x \subset \mathcal{N}$. The quorum Q_x corresponding to an allocation x will be defined separately in each of our three cases, so for now we permit it to be arbitrary.

The message space: There are types of messages. First, there are "allocation messages" of the form (n, x), where $n \in \mathcal{N}$ is a player and $x = \{y_1, \ldots, y_K, \mathbf{v}\}$ is an allocation. These messages should be interpreted as signed proposals from n to implement allocation x. Second, there are "transaction messages" of the form (n, y) for $n \in \mathcal{N}$ and $y \in \mathcal{Y}$, which should be interpreted as a message signed by n saying that she has accepted transaction y. Third, there are "echoes" of the form (n, n', y) for $n, n' \in \mathcal{N}$ and $y \in \mathcal{Y}$, which should be interpreted as a message from n stating that she has received message (n', y) from n'. The full message space is denoted $\mathcal{M}$, and the cost $\kappa(m)$ of each message $m \in \mathcal{M}$ is set to zero.

We now describe which actions a player is permitted to take at each history. A player n may send any message of the form (n, x) or (n, y) after any history. Player n may send echo (n, n', y) after any history in which she has previously received message (n', y) from n'. Finally, a player is permitted to agree to transaction y if

1. She has previously received a message of the form (n', x) from all $n' \in Q_x$, with $y = y_k$ for some k;
2. If $y = y_k$ for $k > 1$, she has previously received an echo of the form (n', n'', y_{k-1}) from all $n' \in Q_x, n'' \in S(y_{k-1})$.

The conditions under which a player may accept a terminal state $\mathbf{v}$ are analogous, except she must have received an echo (n', n'', y_K) from all $n' \in Q_x$, $n'' \in S(y_K)$.

The record-keeping protocol: The record-keeping protocol $\boldsymbol{\sigma}$ prescribes the following strategies to each player n at a local history h_n:

1. Send an allocation message (n, x) to all other players n'.
2. Move to the next step after receiving the same allocation message (n', x) from all other $n' \in Q_x$.
   - If any other allocation message (n', x') is received for $x' \neq x$, cease communication.
3. Let k be the smallest index such that n has not received an echo (n', n'', y_k) from all $n' \in Q_x$, $n'' \in S(y_k)$. (Move to Step 5 if there is no such k.) Then
   - If $n \in S(y_k)$, send message (n, y_k) to all other n (if this action has not already been taken);
   - After receiving a message (n', y_k) from any other n', send echo (n, n', y_k) to all other n.

4. After receiving echoes (n', n'', y_k) from all $n' \in Q_x$, $n'' \in S(y_k)$, agree to y_k if $n \in S(y_k)$. Go back to Step 3.

5. If all transactions y_k have been realized, agree to $\mathbf{v}$ after receiving all echoes of the form (n', n'', y_K) for $n' \in \mathcal{N}$, $n'' \in S(y_K)$.

Lemma 5. *If all players begin with knowledge of the same allocation x and all players $n \in Q_x$ are non-faulty, the allocation x is realized when players follow the quorum algorithm.*

Proof. This result is immediate from the fact that messages sent by non-faulty players are delivered in bounded time (Assumption 1). □

In the previous section, we further proved that these strategies constitute a record-keeping protocol.

B.2 First case: Resource-efficiency and allocative efficiency

Proof. In each instance $(\mathcal{Y}^F, \emptyset)$ with no faulty players, fix a Pareto-efficient and individually rational allocation $x = \{y_1, \ldots, y_K, \mathbf{v}\}$. The set of feasible transactions is common knowledge at the start of the game, so x is therefore common knowledge as well.

The communication game: The message space and set of allowable actions are precisely as in the quorum algorithm of Section B.1. The quorum for each allocation x is set to the full set of players $\mathcal{N}$, $Q_x = \mathcal{N}$ for each allocation x.

Incentive-compatibility: We first consider deviations by a sub-coalition $S \subsetneq \mathcal{N}$. Note that by construction, such a coalition cannot engage in any deviation that causes some transaction y with $S(y) \not\subset S$ to be realized if $y \notin \{y_1, \ldots, y_K\}$, nor can S engage in any deviation that changes the order of the realization of $\{y_1, \ldots, y_K\}$. At most, players in S can (1) prevent further transactions (or the terminal state) from being realized, (2) realize transactions y such that $S(y) \subset S$, or (3) omit transactions $y_k \in \{y_1, \ldots, y_K\}$ such that $S(y) \subset S$. The individual rationality of x implies (1) is not profitable. Pareto-efficiency implies (2) and (3) are not profitable either: if players in S were made better off by including additional transactions, or through the omission of transactions, then x would not be Pareto-efficient.

We can then consider deviations by the grand coalition $\mathcal{N}$. Again, though, Pareto-efficiency implies that no profitable deviation is possible.

Therefore, the prescribed record-keeping protocol $\boldsymbol{\sigma}$ is an equilibrium of the communication game $\mathcal{G}$ in all instances with no faulty players. The pair $(\mathcal{G}, \boldsymbol{\sigma})$ achieves both resource-efficiency and allocative efficiency by construction.

□

B.3 Second case: Fault-tolerance and resource-efficiency

Proof. **The communication game**: For the purposes of this proof, it is sufficient to consider a communication game in which players are not permitted to agree to any transaction y after any

(local) history. However, they are permitted to agree to any terminal state $\mathbf{v}$ after any history. In this case, the strategy profiles will be simple enough that we do not even have to define a message vocabulary $\mathcal{M}$ (i.e., we can take the message vocabulary to be empty).

The record-keeping protocol: Fix some terminal state $\mathbf{v}^*$ with $\sum_{n=1}^{N} v_n^* = V$. Players are recommended to agree to $\mathbf{v}^*$ immediately after any history (if they have not already done so). Clearly, this strategy profile constitutes a record-keeping protocol: players are promised the payoffs v_n^* after any history.

Incentive-compatibility: Given any instance in which the majority of players are non-faulty, there are no profitable deviations for coalitions $S \subsetneq \mathcal{N}$. If players in S agree to a terminal state $\mathbf{v} \neq \mathbf{v}^*$, the remaining players not in S still will not do so, so $\mathbf{v}$ will never be realized. Furthermore, members of S cannot profitably deviate by refusing to accept $\mathbf{v}^*$, since that can only delay their non-negative payoff when the game ends.

We are therefore left to consider deviations by the grand coalition $\mathcal{N}$. The split of surplus $\mathbf{v}^*$ is Pareto-efficient (among the set of terminal states), so there is no deviation for the grand coalition that results in a weakly greater payoff for all players with a strictly greater payoff for some player.

Thus, the communication game $\mathcal{G}$ and the record-keeping protocol $\boldsymbol{\sigma}$ achieve fault-tolerance. The message vocabulary is empty, so they satisfy resource-efficiency as well, as desired. □

B.4 Third case: Fault-tolerance and allocative efficiency

The construction that we use in this section is the most involved. There are two phases to the record-keeping protocol. The final phase is similar to the record-keeping protocol in Section B.2: non-faulty players begin with a known allocation and then send messages and echoes to implement the corresponding sequence of transactions and terminal state. The protocol must be altered due to the fact that some players are faulty.

In the first phase, players communicate to reach agreement on an allocation that will be implemented in the second phase. This phase consists of two sub-phases, which we term (1) the majority rule algorithm and (2) the revision algorithm. We begin by fixing a vector of promised payoffs $\mathbf{v}_0$ (with $\sum_{n=1}^{N} v_{0n} = V$) and then choosing two allocations for each instance $\iota = (\mathcal{Y}_\iota^F, \mathcal{F}_\iota)$: a Pareto-efficient and individually rational allocation x_ι and a "reference" allocation $\underline{x} = \{\underline{v}\}$. The allocation x_ι can be chosen arbitrarily.

In the majority rule algorithm, players narrow the range of possible allocations to a binary choice between some x_ι and x_0. The revision algorithm imposes communication costs and serves to provide incentives for players to behave honestly. If all players act honestly, then allocation x_ι is implemented, but if some are dishonest, then $\underline{x}$ might be implemented instead. Dishonest players stand a chance of being discovered during the course of the revision algorithm and losing out on payoffs to be received later. We describe each of these in turn.

B.4.1 The majority rule algorithm

Note that in order to decide on an allocation x_ι, it suffices for players to reach agreement on a subset $S \subset \mathcal{N}$ of non-faulty players. Once all players have reached agreement on S, they will have learned that allocation x_ι is to be implemented, where $\iota = (\mathcal{Y}^F, \mathcal{N} \backslash S)$. A subset S of players, in turn, can be identified with a vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_N) \in \{0,1\}^N$, where $z_n = 1$ denotes that n is non-faulty.

The majority rule algorithm, adapted from Bracha and Toueg (1985), is a communication protocol through which players can reach agreement on a binary variable. This algorithm can be applied to each element of $\boldsymbol{\theta}$ so that players can reach agreement on a subset S.

Preliminaries: There is one player n^* who is termed the "leader," who may be faulty or non-faulty. All players track a *value* $\hat{\theta}_n \in \{0,1\}$, initialized to zero, and two additional numbers: a *phase number* z_n and a *cardinality* q_n (also initialized to zero).

The message space: A message is $m = (n, z, \hat{\theta}, q)$, consisting of the identity n of some player, a phase number z, a value $\hat{\theta}$, and a cardinality q. We will term any message with cardinality $q > \frac{N}{2}$ and value $\hat{\theta}$ a *witness* for $\hat{\theta}$. For each player n, there is also a special message m_n^* that can be sent only by that player (after any history).

A player n can send message $(n, 0, 1, 1)$ after (1) receiving message $m_{n^*}^*$ from the leader, and (2) receiving at least $\frac{N}{2}$ other messages $m_{n'}^*$. Message $(n, 0, 0, 1)$ can be sent after any history. A player n can send message $(n, z, \hat{\theta}, q)$ (for $z > 1$) after any history such that (1) more than $\frac{N}{2}$ phase $z-1$ messages were received, (2) either a $\hat{\theta}$-witness was received or a majority of the messages had value $\hat{\theta}$, and (3) at least q of the received messages had value $\hat{\theta}$.

The communication protocol: Player n is instructed to communicate according to the following protocol.

1. Send message m_n^* to all other players n'.

2. Once more than $\frac{N}{2}$ messages $m_{n'}^*$ have been received, set $\hat{\theta} = 1$ if a message was received from the leader n^* and $\hat{\theta}_n = 0$ otherwise. Then, phase $z_n = 0$ begins.

3. At the beginning of phase z_n, send message $(n, z_n, \hat{\theta}_n, q_n)$ to all other players. Once more than $\frac{N}{2}$ phase-z_n messages have been received from other players, change value $\hat{\theta}_n$ to $\hat{\theta}$ if a single $\hat{\theta}$-witnesses was received. If two distinct $\hat{\theta}$-witnesses were received, set $\hat{\theta}_n = 0$. If no $\hat{\theta}$-witnesses were received, set $\hat{\theta}_n$ equal to the majority value among the received messages (setting $\hat{\theta} = 1$ in case of a tie). Update cardinality q_n to the number of messages received with value $\hat{\theta}_n$, and increase phase z_n by one.

4. Once $\lfloor \frac{N-1}{2} \rfloor$ witnesses for a value $\hat{\theta}$ have been received, update value $\hat{\theta}_n$ to $\hat{\theta}$, cardinality q_n to the number of messages with value $\hat{\theta}$ received, and increment phase z_n by one. Send a final message $(n, z_n, \hat{\theta}_n, q_n)$ before ceasing communication.

Now we show that when non-faulty players follow this communication protocol, they eventually reach agreement on a value $\hat{\theta}$ with probability one.

Proposition 1. *If all non-faulty players follow the majority rule algorithm, then communication terminates with probability one, and all non-faulty players end with the same value $\hat{\theta}_n$. Furthermore, when the leader is non-faulty, $\hat{\theta}_n = 1$ for all non-faulty n with positive probability.*

Proof. The proof is adapted from Bracha and Toueg (1985). To prove the first part of the proposition, we must demonstrate the protocol's *consistency*: any two non-faulty players end with the same value. Then we show its *deadlock-freedom*: no non-faulty player ever remains in a particular phase t indefinitely. Finally, we show its *convergence*: all non-faulty players terminate communication with probability one.

Consistency: Let z be the smallest phase in which some player terminates. We claim that it is not possible for any two players n and n' to receive witnesses for different values before phase z. Suppose that in phase z, player n has already received a witness for value $\hat{\theta}$ (in phase $z-1$, from some player n''). This implies that player n'' must have received more than $\frac{N}{2}$ messages with value $\hat{\theta}$ in phase $z-2$. Thus, if player n' received a $z-1$-witness for value $\hat{\theta}' \neq \hat{\theta}$, it must be that some player sent messages with two different values in phase $z-2$ (since there are only N players). As an implication, a player n can never receive witnesses for two different values in phase $z-1$.

Suppose that player n terminates with value 0 in phase z. We now show that no other player n' can terminate with value 1. If n' terminates in phase z, he must terminate with value 0, since player n must have at least $\frac{N-1}{2}$ witnesses for value 0 in phase z.

Now, we can show that all messages sent in phase z are of the form $(\tilde{n}, z, 0, q)$ for some $\tilde{n}, q$. Since n terminated in phase z, n must have received at least $\lfloor \frac{N-1}{2} \rfloor$ $z-1$-witnesses for 0. Consider a player n' who sends a z-message. That player must have received more than $\frac{N}{2}$ $z-1$-messages, at least one of which must have been a witness for value 0. Thus, player n' sends a message of the specified form.

Suppose that player n'' terminates in phase $z+1$. The above argument proves that n'' must terminate with value 0, since all messages sent in phase $z+1$ have value zero. Furthermore, observe that all messages sent in phase $z+1$ must be of the form $(\tilde{n}, z+1, 0, N-\lfloor \frac{N-1}{2} \rfloor)$, since all z-messages have value 0.

A player n'' who reaches phase $z+2$ must terminate with value 0, since that player receives only witnesses for 0 in phase $z+1$. No player ever reaches phase $z+3$, then. Hence, the consistency property is proven.

Deadlock-freedom: Since players wait for each others' messages, it is possible that some players would become deadlocked and are unable to proceed with communication. We now prove this is not the case.

Suppose by way of contradiction that D is the set of deadlocked players, with each $n \in D$ being deadlocked in phase z_n. Let $z_0 = \min_{n \in D} z_n$, with n_0 being a player who is deadlocked in phase z_0. Now let S be a set of more than $\frac{N}{2}$ non-faulty players. There are two cases to consider.

1. No player in S terminates in phase $z \leq z_0 - 2$. By the minimality of z_0, every player in S either terminates at $z_0 - 1$, terminates at z_0, or gets deadlocked in some phase $z \geq z_0$. In

any case, each player in S will send z_0-messages to all other players, meaning that n_0 receives more than $\frac{N}{2}$ messages and can remain deadlocked in phase z_0.

2. Some player in S terminates in phase $z < z_0 - 2$. We previously proved that if this occurs, then all players must terminate before phase $z + 3$, a contradiction.

Since no non-faulty player remains deadlocked, faulty players do not deadlock either. In each phase, a non-faulty player will receive messages from more than $\frac{N}{2}$ non-faulty players.

Convergence: Finally, we must show that communication terminates in finite time with probability one. Let S be a set of more than $\frac{N}{2}$ non-faulty players, and suppose that no player in S terminates before phase z_0. We show that with positive probability, all players in S terminate in phase $z_0 + 2$.

Deadlock-freedom implies that ever player in S will reach phase z_0. With positive probability, the following occurs: in phase z_0, players in S receive z_0-messages from each other before receiving a message from any other player, and then the same happens in phase $z_0 + 1$. When this occurs, from our definition of the protocol, all players in S will terminate in phase $z_0 + 2$.

Having proven those three properties, we must now prove the final part of the proposition: that if the leader is non-faulty, all players will terminate with value 1 with positive probability. Note that all players will begin with value 1 if they receive a message from the leader before receiving more than $\frac{N}{2}$ other messages (which occurs with positive probability). Hence, the claim is proven. □

B.4.2 The revision algorithm

Now we define our revision algorithm and derive its key properties. A player begins the revision algorithm once he has terminated communication in the majority rule algorithm, yielding a vector of values $\hat{\boldsymbol{\theta}}^n \in \{0,1\}^N$. Player n also keeps track of a phase number z_n in the revision algorithm, initialized to zero.

The message space: There are two types of messages: free broadcasts and costly broadcasts. A free broadcast is of the form $(n, z, \hat{\boldsymbol{\theta}})$, where $z \geq 1$ is a phase number and $\hat{\boldsymbol{\theta}} \in \{0,1\}^N$ is a vector. This type of message should be interpreted as a signed message from n stating that his current phase is z and that he believes all n' such that $\hat{\theta}^n_{n'} = 1$ are non-faulty. A costly broadcast is a message of the form $(n, 0, \hat{\boldsymbol{\theta}})$. The cost of a free broadcast is set to zero, whereas a costly broadcast $(n, 0, \hat{\boldsymbol{\theta}})$ will require the sender to pay a positive cost $\kappa(n, \hat{\boldsymbol{\theta}})$, specified later.

A player n can send a costly broadcast $(n, 0, \hat{\boldsymbol{\theta}})$ if, for each component $\hat{\theta}_{n'}$, that player received $\lfloor \frac{N-1}{2} \rfloor$ witnesses for $\hat{\theta}_{n'}$ during the majority rule algorithm. Player n can send a free broadcast $(n, z, \hat{\boldsymbol{\theta}})$ if (1) n is permitted to send a costly broadcast of $\hat{\boldsymbol{\theta}}$, and (2) n received a broadcast $(n', z-1, \hat{\boldsymbol{\theta}})$ from all n' such that $\hat{\boldsymbol{\theta}} = 1$.

The communication protocol: We fix a maximum phase number $Z^* > 1$ that we specify later. Player n is instructed to communicate according to the following protocol.

1. Send a broadcast $(n, z_n, \hat{\boldsymbol{\theta}}^n)$ to all other n'.

2. If a broadcast $(n', z_n, \hat{\boldsymbol{\theta}}')$ is received from any n' with $\hat{\boldsymbol{\theta}}' \neq \hat{\boldsymbol{\theta}}^n$, update $\hat{\boldsymbol{\theta}}^n$ to a vector of zeros.

3. If a broadcast $(n', z_n, \hat{\boldsymbol{\theta}}^n)$ is received from all other n' such that $\hat{\theta}^n_{n'} = 1$, then

   - If $z_n < Z^*$, update phase to $z_n + 1$ and return to Step 1.
   - If $z_n = Z^*$, send broadcast $(n, Z^*, \hat{\boldsymbol{\theta}}^n)$ to all other n' and terminate communication.

Clearly, if all players enter the revision algorithm with the same value $\hat{\boldsymbol{\theta}}$ and behave according to the protocol, then they will all exit with the same value as well. We will be interested in the properties of the revision algorithm when some players behave dishonestly, since this algorithm will be used to impose off-equilibrium punishments.

Proposition 2. *Let S be a set of players who behave according to the majority rule algorithm followed by the revision algorithm. Then the probability that any two players $n, n' \in S$ terminate the revision algorithm with different values $\hat{\boldsymbol{\theta}} \neq \hat{\boldsymbol{\theta}}'$ (different from $\mathbf{0}^N$) is at most $1 - \left(\frac{Z^*}{1+Z^*}\right)^{|S|-1}$.*

Proof. Let n be the first player in S who terminates communication (with non-zero value $\hat{\boldsymbol{\theta}}$), and suppose that some other honest player n' terminates communication with a non-zero value $\hat{\boldsymbol{\theta}}' \neq \hat{\boldsymbol{\theta}}$. For this to occur, n must receive more than $\frac{N}{2}$ costly broadcasts of $\hat{\boldsymbol{\theta}}$ and n' must receive more than $\frac{N}{2}$ costly broadcasts of $\hat{\boldsymbol{\theta}}'$, so there must be n'' who sent a costly broadcast of $\hat{\boldsymbol{\theta}}$ to n and $\hat{\boldsymbol{\theta}}'$ to n'.

By the protocol definition, since n and n' terminate with different values, it must be that n delivers at least Z^* messages to n'' before a single message sent by n is delivered to n'. Let t_0 be the first time at which n receives a costly broadcast of $\hat{\boldsymbol{\theta}}$. Define d_0 to be the delivery time of the message sent by n to n' at t_0, and define $d_1, \ldots, d_{Z^*}$ to be the delivery times of the Z^* messages sent by n to n'' after t_0. We have assumed that message delivery times are I.I.D. according to a distribution G, so by symmetry, the probability that this occurs is

$$\Pr\left(d_0 > \sum_{z=1}^{Z^*} d_z\right) = \Pr\left(d_z > \sum_{z' \neq z} d_{z'}\right).$$

There are $1 + Z^*$ such probabilities, so this probability is at most $\frac{1}{1+Z^*}$.

Consider the probability

$$P^* = \Pr(\exists\ \tilde{n} \in S : \tilde{n} \text{ terminates with a non-zero value different from } n).$$

We have shown that a necessary condition for this event to occur is for n to deliver Z^* messages to n'' before delivering a single message to $\tilde{n}$. Denote the probability that this happens by $P_{n,\tilde{n}}$. Using the independence of message delivery times, P^* then satisfies

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

□

B.4.3 The final phase

Now we describe the communication protocol in the final phase, once players have learned the allocation they will implement. A player n who concludes the revision algorithm with a non-zero value $\hat{\boldsymbol{\theta}}^n$ has learned a subset of agents S_n who are non-faulty, i.e., all n' such that $\hat{\theta}^n_{n'} = 1$. The allocation learned by this player is then $x^n_{\iota^n}$, where $\iota^n = (\mathcal{Y}^F, \mathcal{N}\backslash S_n)$ is the instance in which players S_n are non-faulty. On the other hand, a player n who terminates the revision algorithm with value $\hat{\theta}^n = \mathbf{0}^N$ learns allocation $\underline{x} = \{\underline{\mathbf{v}}\}$, where $\underline{\mathbf{v}}$ is a vector that will be specified later.

The message space, rules of the communication game, and the communication protocol are precisely as in the quorum algorithm of Section B.1. The quorum for an allocation x_ι is set to the subset of non-faulty players in instance $\iota = (\mathcal{Y}^F_\iota, \mathcal{F}_\iota)$; that is, $Q_x = \mathcal{N}\backslash\mathcal{F}_\iota$.

B.4.4 Proof of the result

We now define the full communication game $\mathcal{G}$ and record-keeping protocol $\boldsymbol{\sigma}$ used in our third case. We prove that the pair $(\mathcal{G}, \boldsymbol{\sigma})$ achieves fault-tolerance and allocative efficiency.

The communication game: The message set available to agents is the union of the message sets defined for the majority rule algorithm (Section B.4.1), the revision algorithm (Section B.4.2), and the quorum algorithm (Section B.1). The rules determining which messages players can send and which actions they can take are also as defined in those sections.

We must still define the cost $\kappa(n, \hat{\boldsymbol{\theta}})$ of sending a costly broadcast $(n, 0, \hat{\boldsymbol{\theta}})$ in the revision algorithm, since that quantity was left unspecified. For simplicity, letting $x_\iota = \{y_1, \ldots, y_K, \mathbf{v}\}$ be the allocation corresponding to $\hat{\boldsymbol{\theta}}$, we set

$$\kappa(n, \hat{\boldsymbol{\theta}}) = \sum_{n \in S(y_k)} u_n(y_k) + v_n.$$

The record-keeping protocol: The protocol $\boldsymbol{\sigma}$ specifies that players should follow the majority rule algorithm (Section B.4.1), the revision algorithm (Section B.4.2), and the quorum algorithm (Section B.1) in sequence.

We must still specify the number of phases Z^* in the revision algorithm. Define

$$U_{\max} = \max_{n \in \mathcal{N}} \sum_{y \in \mathcal{Y}} u_n(y) \mathbf{1}\{u_n(y) \geq 0\} + V.$$

This is the maximum possible utility n could receive in any allocation. Then let

$$\kappa_{\min} = \min_{n, \hat{\boldsymbol{\theta}}} \kappa(n, \hat{\boldsymbol{\theta}}) \text{ s.t. } \kappa(n, \hat{\boldsymbol{\theta}}) > 0.$$

This is the minimum broadcast cost in the communication game. Finally, set Z^* so that

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

Incentive-compatibility: We now show that there does not exist a profitable deviation from the record-keeping protocol for any coalition of players. First, observe that if players follow the record-keeping protocol honestly and a majority are non-faulty, then under any allocation, all players receive a total payoff of at most zero, regardless of which transactions are feasible and which players are faulty. Furthermore, when players follow the protocol, they receive a payoff of exactly zero. When no player sends multiple distinct costly broadcasts, players never have an incentive to delay the implementation of an allocation along the equilibrium path, since all allocations implemented by this protocol are assumed to be individually rational. No coalition has an incentive to pursue any strategy that would lead to the implementation of a different allocation (without sending multiple costly broadcasts).

It remains to determine whether a coalition can profitably deviate by, at some point, sending multiple distinct costly broadcasts. The probability that *any* transactions are implemented after the second costly broadcast is sent is at most $1 - \left(\frac{Z^*}{1+Z^*}\right)^N \leq \frac{\kappa_{\min}}{U_{\max}}$ (by Proposition 2). The utility realized by any player from such transactions is at most $U_{\max}$, and the cost of engaging in this strategy is at least $\kappa_{\min}$ (for any player who receives positive utility from engaging in that strategy). Therefore, such deviations cannot be profitable for any coalition.

We have shown that the prescribed record-keeping protocol $\boldsymbol{\sigma}$ is an equilibrium of the communication game $\mathcal{G}$ in every instance with a majority of non-faulty players. Moreover, by construction, an efficient allocation is implemented with positive probability in each instance (which follows from the second part of Proposition 1). Therefore, $(\mathcal{G}, \boldsymbol{\sigma})$ achieves fault-tolerance and allocative efficiency, as desired.