Achieving State Machine Replication without Honesty Assumptions

Conor McMenamin[1], Vanesa Daza[2], and Matteo Pontecorvi[3]

[1,2]Department of Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona, Spain
[2]CYBERCAT - Center for Cybersecurity Research of Catalonia
[3]NOKIA Bell Labs, Nozay, France

December 18, 2020

Abstract

State machine replication protocols have reached a crucial juncture in their widespread deployment. Tokenised state machine replication protocols, which utilise an internal token for rewarding player participation, have brought about major advances in the areas of finance, internet of things, supply chain, legal systems, and data storage, to name but a few. However, the viability of these protocols as replacements for their centralised alternatives requires guarantees of player actions at all times which at present do not exist. Current standards for player characterisation in tokenised state machine replication protocols allow for honest players who will always follow the protocol, regardless of possible token increases for deviating. Given the ever-increasing market capitalisation of these tokenised protocols, honesty is becoming more expensive and more unrealistic. As such, this out-dated player characterisation must be removed to provide true guarantees of safety and liveness in a major stride towards universal trust in state machine replication protocols and a new scale of adoption. As all current state machine replication protocols are built on these legacy standards, it is imperative that a new player model is identified and utilised to reflect the true nature of players in tokenised protocols, now and into the future.

To this effect, we propose the ByRa player model for state machine replication protocols. In the ByRa model, players either attempt to maximise their tokenised rewards, or behave adversarially. This merges the fields of game theory and distributed systems, an intersection in which tokenised state machine replication protocols exist, but on which little formalisation has been carried out. In the ByRa model, we identify the properties of strong incentive compatibility in expectation and fairness that all protocols must satisfy in order to achieve state machine replication. We then provide FAIRSICAL, a protocol which provably satisfies these properties, and by doing so, achieves state machine replication in the ByRa model.

email: *conor.mcmenamin@upf.edu* — *vanesa.daza@upf.edu* — *matteo.pontecorvi@nokia.com*

This Technical Report is part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 814284

Contents

1 Introduction **3**
1.1 Our Contribution 4
1.2 Organisation of the paper 4

2 Related Work **4**

3 Preliminaries **6**

4 A Game Theoretic Framework for SMR **7**

5 Achieving SMR in the ByRa Model **10**

6 The FAIRSICAL Protocol **12**
6.1 Threat Model 12
6.2 Protocol Description 12

7 Proving FAIRSICAL achieves ByRa SMR **16**
7.1 Safety and Liveness when Correct Players Control a Majority of Stake 16
7.2 Proving FAIRSICAL is Strong Incentive Compatible in Expectation 19
7.3 Proving FAIRSICAL is Fair 21

8 Conclusion **21**

9 Acknowledgements **22**

Bibliography **22**

1 Introduction

Tokenisation has emerged as one of the most successful tools for instantiating state machine replication (SMR) protocols. Adding tokenised rewards to SMR protocols explicitly quantifies the gains and losses of players within protocols based on the actions they take. If a majority of players seek to maximise their tokens in an SMR protocol, and players maximise their tokens by following the protocol, then guarantees of SMR can be made.

Unfortunately, current SMR protocols fundamentally assume the existence of some subset of players who ignore token changes and honestly follow the protocol. If a player can deviate from a protocol to increase their tokens with no perceived effect on safety and liveness, it must be assumed that every such individual will choose to do this. In any large-scale usage scenario, most, if not all players will not consider their deviations as affecting SMR. Therefore, it is essential that we assume non-Byzantine players will seek to maximise their tokens in tokenised protocols. As a direct consequence, SMR guarantees can no longer depend on honest-by-default users in tokenised protocols. We explicitly outline the ByRa model as an updated player characterisation framework to reflect this weakness in current standards. By moving to the ByRa model, which we formally define in Definition 4.1, the caveat of honest player dependencies in current SMR protocols is removed. Furthermore, we demonstrate that it is possible to achieve SMR in the ByRa model by providing the FAIRSICAL protocol.

To progress towards global adoption, a tokenised SMR protocol must first ensure that all players will maximise their tokens by following the protocol. Implementing an SMR protocol that maximises a player's tokens by following the protocol is known as incentivisation, and is a fundamental requirement of any SMR protocol. Much of the work on incentivisation in SMR protocols stems from the seminal work on selfish mining in Nakamoto-consensus [20]. In [20], it is demonstrated that certain players are incentivised to deviate from the prescribed protocol. This eventually leads to a scenario where SMR properties are violated, as discussed in [20]. It is only upon the performing of actions as required by the protocol by some majority that it is possible to guarantee the SMR properties of safety and liveness. This has remained the case in the age of tokenisation.

While the concept of incentivisation is elegant, its implementation is fraught with complications. Tokenisation of SMR protocols came to popular practice as a direct consequence of Bitcoin and Nakamoto consensus [27]. However, due to the decoupling of on-chain tokens and the resource necessary to participate in the protocol, hashing power, adverse side-effects such as the emergence of oligopolies were witnessed [4, 8] and its widespread adoption limited. Making on-chain tokens and consensus resources interchangeable was a logical progression for SMR, and varying flavours of such an implementation, under the broad heading of Proof-of-Stake, have appeared [14, 16, 17, 18, 22, 24].

Despite these, and many other works, there has been no thorough treatment and analysis of tokenised SMR protocols from a game theoretic standpoint involving rational players, who want to maximise their net tokenised gains (referred to as utility increases in game theoretic literature), and an adversary, who can corrupt the owners of some amount of the tokenised consensus resource and behave arbitrarily. These corrupted players are known as Byzantine. This characterisation model of players as either Byzantine or Rational, which we refer to as the ByRa model, was first considered in distributed systems literature in [26], but never successfully with respect to SMR protocols, although attempts have been made [5, 24]. The closest semblance to this model which has seen wide-scale adoption with respect to SMRs is the BAR (Byzantine, Altruistic and Rational) model [3]. The BAR model crucially includes some portion of altruistic players who disregard tokenised utility, and always follow the protocol. Examples of authors echoing our desire to move away from altruistic dependencies are numerous, but this from Fairledger [24] puts it concisely: "We have to take into account that every entity may behave rationally, and deviate from the protocol if doing so increases its benefit". Non-adversarial, honest-by-default characters do not exist in competitive games, and cannot be depended on in tokenised SMR protocols due to their gamified nature. Although many works state the need to move away from altruistic dependencies, none have proven the critical nature of this dependency or provided protocols which achieve SMR in the ByRa model. In this paper, we fulfil both of these essential tasks.

Without the safety net of altruistic players, any successful instantiation of an SMR protocol in the ByRa model must guarantee that rational players will always follow the protocol. To ensure this, rational players must expect to strictly maximise their utility by following the protocol, a property we define as *strong incentive compatible in expectation* (SINCE).

Moreover, we must also guarantee that within such an incentive compatible protocol, the adversary cannot increase their share of tokens to a point where they control enough tokens to prevent SMR. Despite the existence of strong incentive compatibility in expectation, it may be possible for an adversary to receive more than their share of the tokens that get distributed, increasing their share of control. Therefore, we must additionally ensure that an adversary cannot increase the share of tokens they control in the presence of all other players following the protocol. The property that an adversary cannot increase their share of tokens in the presence of correctly participating non-adversarial players is *fairness*.

1.1 Our Contribution

We define the ByRa player characterisation model, the properties of SINCE and fairness, and in Definition 4.5, the basic requirements a prospective SMR protocol must meet in order to guarantee safety and liveness in the ByRa model. If these requirements are met for a protocol in the ByRa model, the protocol *achieves ByRa SMR*. Informally, to achieve ByRa SMR we require that players controlling a majority of tokens follow the protocol at all times. We then prove that the properties of SINCE and fairness are necessary and together sufficient to achieve ByRa SMR in the main theorem of the paper.

Theorem 5.8. For an SMR protocol Π, Π achieves ByRa SMR if and only if Π is strong incentive compatible in expectation and fair.

In addition to this new game theoretical framework, we provide FAIRSICAL as a concrete instantiation of an SMR protocol that provably achieves SINCE and fairness in the ByRa model. Using Theorem 5.8, we then prove FAIRSICAL achieves SMR in the ByRa model.

Corollary 7.7. FAIRSICAL achieves ByRa SMR.

1.2 Organisation of the paper

In Section 2 we review related work and present an overview of attempts to implement, and works in favour of, the ByRa model for SMR protocols. In Section 3 we provide a background on the SMR and game theory concepts needed to define the ByRa model. Section 4 introduces a new game theoretic framework for analysing SMR protocols. This new framework defines the ByRa model, and outlines what we require from SMR protocols in the ByRa model, introducing the properties of SINCE and fairness. In Section 5 we prove that SINCE and fairness are necessary for a protocol to achieve ByRa SMR. We then prove that together, SINCE and fairness are sufficient properties for a protocol to achieve ByRa SMR. In Section 6 we outline the FAIRSICAL protocol as an example, for the first time in literature, of a SINCE and fair ByRa SMR protocol. In Section 7.1 we prove that FAIRSICAL satisfies the necessary and sufficient properties of safety and liveness for SMR when players controlling a majority of the consensus votes follow the protocol in every round. We prove that the FAIRSICAL protocol is SINCE in Section 7.2, and fair in Section 7.3 which, using Theorem 5.8, implies FAIRSICAL achieves ByRa SMR. We conclude in Section 8.

2 Related Work

There is a growing appreciation that incentivisation is not only important, but necessary, to ensure the successful instantiation of an SMR protocol. Many papers have argued for the incentivisation of players in SMR protocols [2, 5, 9, 10, 15, 17, 22, 23, 25, 26, 30, 31, 32] while many other papers demonstrate the critical need for incentive compatibility in tokenised SMR protocols [4, 7, 8, 11, 12, 13, 20, 21, 27, 28].

In-keeping with the distributed nature of SMR protocols we must also account for a metaphysical adversary who can control some portion of the SMR participants (Byzantine) with unknown utility. The characterisations of Byzantine and rational, coupled with that of altruistic players who always follow the protocol, segues into the BAR player characterisation model as introduced in [3]. However, as discussed in Section 1, tokenised SMR protocols cannot depend on altruistic players to ensure the critical properties of safety and liveness. As a result, we do not consider altruistic actors in this paper.

We amend the player characterisations to only include those of Byzantine and rational players in what we call the ByRa model. A very similar player model is discussed in [26]. We extend their basic binary action space for players to allow for complex action profiles in line with those of SMR protocols. We introduce the necessity for strict maximisation of expected utility if we are to ensure rational players always follow a protocol. This is opposed to [26], where it is claimed that equality of utility will suffice to ensure a rational player will choose one strategy over another. This is logically insufficient. Related to this concept of insufficient proof mechanisms, a common pitfall of legacy incentive compatible proofs is to prove that following a protocol is a Nash Equilibrium in the presence of honest players [17, 21, 22, 30]. In the ByRa model this assumption is not possible, and therefore those proofs are not complete. We also allow the adversary to behave arbitrarily, as opposed to [26] where the adversary only tries to minimise the utility of rational players. Although there are buzzwords associated with this paper such as *Price of Malice* and *Price of Anarchy*, no name is attributed to the player model. We refer to our version of this player model as the ByRa model. The only example of this player model in SMR literature without an explicit utilisation of an altruistic entity is in [5].

Critically, the authors of [5] depend on a dominating cost for rational players for increasing the probability of reaching consensus on an invalid block which is unrelated to change in tokens, something which is also utilised in [32]. Such a critical cost of accepting an invalid block is not related to the utility of players trying to maximise their quantity of tokens. By moving to the ByRa model and removing this perceived cost for rational players, utility assumptions become sufficiently weak so as to capture the true nature of uncooperative rational players in the large-scale tokenised protocols of today, and beyond. In [5], it is also implicitly assumed rewards are paid to all players who reach consensus on a block. This is non-trivial in the ByRa model as rewards need to be recorded by a proposer at some point in the protocol, and rational proposers may be incentivised to omit players, as is the case in previous works from subsets of the same authors [6, 7]. We address this omission in the FAIRSICAL protocol, providing an explicit solution in the ByRa model.

A purely economic approach to SMR protocols is taken in [32], which focuses on Proof-of-Stake SMR protocols. Their player model only considers rational players, but they provide some novel approaches to incentivisation in SMR protocols, and agree that in the tokenised setting, players must be incentivised to follow the protocol to guarantee SMR. This is extended in [31] where it is demonstrated that if rewards are distributed in (expected) proportion to the amount of stake a player owns, and all players follow the protocol, bounds can be placed on player stake distribution over time. Again, they restrict their player model to only contain rational players. As their rewards are only paid fairly in expectancy, they cannot guarantee the fairness that we define in Definition 4.9.

One of the legacy works in relation to fairness and incentive compatibility of SMR protocols is Fruitchains [30]. The Fruitchains player model consists of an altruistic majority of players and a cooperative rational minority, despite stating: "Assuming honest participation, however, is a strong assumption". Fruitchains crucially relies on an underlying blockchain satisfying an SMR protocol in order to guarantee fairness of rewards. They fail to consider the incentives of all parts of the system, relying on an altruistic majority in order to guarantee the underlying blockchain satisfies the required SMR properties. They then add a small section where claims of incentive compatibility for non-cooperative rational players are made. The authors claim a protocol is incentive compatible if fairness of rewards has already been guaranteed. As fairness in their system is only guaranteed if a majority of players follow the protocol, there is no logical result which proves that rational players will always follow the protocol, required for incentive compatibility. This is insufficient to guarantee SMR in the ByRa model. This fatal dependence

on an underlying correct-by-default SMR protocol/ trusted third-party is also demonstrated in [15, 23, 24], where claims of incentive compatibility and fairness do not hold in the ByRa model.

3 Preliminaries

This section covers the concepts and definitions required to reason about SMR protocols from a game theoretic perspective. First we define SMR and a general notion of a blockchain which provides some intuition for our SMR definitions, and primes the reader for our description of the FAIRSICAL protocol in Section 6. We then provide the game theory framework necessary to formally reason about SMR protocols involving rational and adversarial players, and how SMR can be achieved in the presence of these types of players.

In this paper, we are interested in a distributed set of n players $\{P^1, ..., P^n\}$ interacting with one and other inside a protocol which will produce some output that all players correctly participating in the protocol can agree on. This output will be a *replicated state machine*. First, we define a state machine.

Definition 3.1. A *state machine* consists of set of variables, and sequence of commands/ updates on those variables, producing some output.

The concept of a state machine alone does not capture the notion that potentially many players can reconstruct a common view of the same state of a machine, and requires extension.

Definition 3.2. For a set of players $\{P^1, ..., P^n\}$ and a state machine, *state machine replication* (SMR) is a process that allows each player to execute a common sequence of commands acting on the machine's state in the same order, thus maintaining a common view of the machine's state.

Progressing towards our goal of analysing SMR protocols, we must first define what we require from an SMR protocol. We take inspiration for our definition from [1], where their system model is clearly and concisely explained, and is very similar to ours.

Notation 3.3. With respect to protocols and recommended protocol actions, a *correct* player is a player who always follows the recommended protocol actions.

Definition 3.4. An *SMR protocol* Π deciding on a potentially infinite sequence of state machine updates satisfies the following properties:

- *Safety*: For any two correct players P^i, P^j in Π, $i \neq j$, if P^i decides on an SMR update V^i at position k, and P^j decides on an SMR update V^j at position k in the sequence, then $V^i = V^j$.
- *Liveness*: For any position k in the sequence, every correct player eventually decides on an SMR update for position k.

To achieve SMR, we utilise the concept of a blockchain. This is done in a sufficiently generic manner as to allow for direct comparison with most blockchain instantiations.

Definition 3.5. A *block* B is a data structure used to communicate changes to the state machine view of each player. Blocks consist of a pointer(s) to previous block(s), and a set of instructions with which to update the state. State machine updates in a block are applied to the state described by the block(s) to which they point. The *genesis block* B_1 describes the starting state of the system and is a priori agreed upon by all players. The global state at any point in the system is then described by applying the state machine updates according to some ordering rule starting from the genesis block. A *blockchain* $C = [B_1, ..., B_H]$ is the ordered data structure created by traversing the block pointers from the genesis block to all blocks to be applied to the global state according to the ordering rule. H denotes the *height* of the blockchain.

In our system, an SMR protocol Π consists of n players owning shares of a finite resource, which we will refer to as *stake*, and denoted $Stake_1$ at initialisation. Π proceeds in fixed-time periods, which we refer to as *rounds*, beginning in round 1. For any height $H > 1$ of the blockchain, players participate in Π to decide on a block for that height. Reaching consensus on a block will involve one or more successful consensus rounds. After a block has been decided for height $H \geq 1$, the total stake in the system is denoted $Stake_H$ with player shares of $Stake_H$ denoted $s_H^1, ..., s_H^n$. Without loss of generality, we assume $\sum_{i=1}^{n} s_H^i = 1$, and for all $i \in \{1, ..., n\}$, $H \geq 1$, $s_H^i < \frac{1}{2}$.

Now we introduce some basic game theory to allow us to properly reason about SMR protocols in our system as games, taking inspiration for our definitions from [29]. The games we are concerned with, SMR protocols, are played by players with strict incomplete information, meaning some subset of players will not know the action choices of other players for certain rounds when they are required to choose their own actions. As such, we need to be able to describe what a player knows (and implicitly what they do not), which we call their private information. Furthermore, we must be able to describe what motivates players in games. This motivation is provided by a utility function, which attributes a numerical score to each action a player can take. In games, players choose the action which maximises their utility function.

Definition 3.6. A *game*, denoted $\mathbb{G}$, progressing in rounds with strict incomplete information for a set of n players $\{P^1, ..., P^n\}$ can be described by the following:

- For every P^i, a set of *actions* X^i. We denote by X^{-i} the set of actions that each player excluding P^i can take. For $x^{-i} \in X^{-i}$, x^{-i} is described by a vector of actions of length $n-1$, with each vector position mapping to a unique player.
- For every player P^i and round r, a set of *private informations* T_r^i. A value $t_r^i \in T_r^i$ is a private information value that P^i can have at round r. We denote the private informations held by all players excluding P^i at round r by t_r^{-i}.
- For every player P^i, current round $r \geq 1$, and some round $r' \geq r$, the *utility function* for P^i with respect to round r' is defined as :

$$u_{r'}^i : T_r^i \times \underbrace{X^i \times ... \times X^i}_{r'+1-r} \times \underbrace{X^{-i} \times ... \times X^{-i}}_{r'+1-r} \to \mathbb{R} \tag{1}$$

  where $u_{r'}^i(t_r^i, x_r^i, ..., x_{r'}^i, x_r^{-i}, ..., x_{r'}^{-i})$ is the utility achieved by P^i in round r' with private information t_r^i, if player P^i takes the actions $x_r^i, ..., x_{r'}^i$ in rounds $r, ..., r'$ respectively, and the actions of all other players are described by $x_r^{-i}, ..., x_{r'}^{-i}$ in rounds $r, ..., r'$ respectively.

Although utility functions evaluate actions given the actions of all other players, the actions of the other players may not be known in advance. Therefore, players will need to be able to choose their actions solely based on their private informations. The actions a player takes given some private information is computed through a strategy, which is defined in Definition 3.7.

Definition 3.7. A *strategy* of a player P^i is a function $str^i : T_r^i \to X^i$, $r \geq 1$, which defines the actions to be taken by P^i given some private information value. A strategy str^i is *mixed* if for a player P^i with m_i possible strategies $Str^i = \{str_1^i, ..., str_{m_i}^i\}$, they select a strategy to follow from Str^i according to some probability distribution. For every player P^i, str^{-i} describes the mixed strategies taken by all players excluding P^i.

Notation 3.8. For an SMR protocol Π, the *recommended strategy*, denoted str^Π, is the strategy that Π requires players to follow in order to successfully achieve SMR.

4 A Game Theoretic Framework for SMR

As reasoned in the earlier sections, we only consider adversarial or rationally motivated participants in SMR protocols. This reflects the fact that SMR protocols can be described as games

with strict incomplete information as defined in Definition 3.6. This is a crucial progression from existing standards in distributed systems literature where some number of players are honest-by-default. Due to the distributed nature of SMR protocols, as a baseline we must account for some portion of adversarial players who can behave arbitrarily with unknown utility functions. With SMR protocols considered as games, the remaining non-adversarial players must follow some known utility function, and attempt to choose the actions which maximise it. To ensure the honest behaviour of rational players in this setting, following the protocol strategy must maximise the utility of rational players. We define these player characterisations here formally as the ByRa model.

Definition 4.1. The *ByRa model* consists of *Byzantine* and *Rational* players. A player is:

- *Byzantine* if they deviate arbitrarily from the recommended strategy within a game with unknown utility function. Byzantine players are chosen and controlled by an adversary A.
- *Rational* if they choose uniformly at random from all mixed strategies which maximise their known utility function.

How the adversarial choices are made, and to what extent the adversary can control Byzantine players must be outlined in the threat model. After deciding on a block at height $H \geq 1$, we denote the adversarial of stake by s_H^A. This is the share of stake Byzantine players control at height H.

Remark 4.2. Our definition of rational players represents the weakest assumption possible about rational players. We omit tie-breaking assumptions that bias a rational player to certain strategies over others with equal utility. For example, if we have a fair coin tossing game that costs 1 token to play and correct guesses gain 2 tokens, a rational player in our system will choose heads with probability 0.5. If we have a protocol that requires rational players to always choose heads, it is necessary to make the payoff for heads strictly greater than that of tails.

In the ByRa model, it is necessary to have a notion of an adversarial control of stake threshold, below which SMR can be achieved if all non-adversarial players follow the protocol.

Notation 4.3. For an SMR protocol Π, we denote by b the maximal share of stake such that for players controlling greater than $1 - b$ of the stake following the SMR protocol, safety and liveness are achieved. The exact value of b will depend on the network distribution assumptions, in line with the results of [19], which must be contained in the threat model.

For some $\kappa \in \mathbb{N}$, our goal is to guarantee that SMR can be achieved (that is, both safety and liveness are satisfied) in the ByRa model with probability greater than $1 - e^{-\kappa}$ over any polynomial in κ rounds.

We first need to introduce an equivalence relation for mixed strategies over finite rounds. When we state the protocol strategy which needs to be followed to achieve SMR, although there is an infinite number of strategy encodings, we only require players to follow strategies which result in actions as outlined by the protocol. We are indifferent to how this is achieved. If a strategy is encoded differently to the recommended protocol strategy, but results in actions as prescribed by the protocol with probability greater than $1 - e^{-\kappa}$ over any polynomial in κ rounds, we see this as equivalent to the recommended protocol strategy.

Definition 4.4. For a player P^i at initialisation, and round $r' \geq 1$, two mixed strategies str_1^i and str_2^i are *equivalent with respect to round* r' if for all rounds r, $1 \leq r \leq r'$, and private informations $t_r^i \in T_r^i$, it is the case that $str_1^i(t_r^i) = str_2^i(t_r^i)$. We use $str_1^i \equiv_{r'} str_2^i$ to denote this equivalence relation. If $str_1^i \equiv_{r'} str_2^i$ for all rounds r' polynomial in κ, str_1^i and str_2^i are *equivalent*, with this denoted by $str_1^i \equiv str_2^i$.

With this equivalence relation, we can now define what it means for a protocol to achieve SMR in the ByRa model.

Definition 4.5. For an SMR protocol Π, let the probability that players controlling more than $1-b$ of the total stake follow a mixed strategy $str \equiv_r str^{\Pi}$ up to and including round r for any $s_1^A < b$ be denoted p_r^{Π} . If for all rounds r' polynomial in κ it holds that $p_{r'}^{\Pi}$ is greater than or equal to $1 - e^{-\kappa}$, then Π *achieves ByRa SMR*. Otherwise, Π *fails in the ByRa model*.

To consider rational players in any game, it is necessary to explicitly define what their utility functions are. Inkeeping with the tokenised assumptions of our model, we let rational player utility be measured in stake. By their nature, tokenised SMR protocols require it to be expensive to deviate from the protocol actions, encouraging honest behaviour through stake rewards, and/or stake punishments for dishonest behaviour. Given the unprecedented levels of SMR protocol usage as a result of tokenisation, we see stake as the driving utility measure for the players who participate in these protocols.

As total stake is only meaningful with respect to a particular time-point, and SMR protocols are played indefinitely, rational players will seek to maximise their total stake at all possible rounds sufficiently far into the future. Therefore, when discussing incentivisation and player utility, it is necessary to refer to stake/share/total stake with respect to rounds. As we are using the round variable as a counter, and some consensus rounds may be unsuccessful, it cannot be independently used to determine the height, and vice versa. Rather than add notation to relate the two, we treat them separately, and make it clear from context which is being used. When referring to stake/share/total stake with respect to particular rounds, we use subscripts involving r, whereas when discussing these variables with respect to the height of the blockchain, we use subscripts involving H.

For a rational player P^i with private information t_r^i and round $r' \geq r$, we have:

$$u_{r'}^i(t_r^i, x_r^i, ..., x_{r'}^i, x_r^{-i}, ..., x_{r'}^{-i}) = s_{r'}^i \cdot Stake_{r'}. \tag{2}$$

However, in a game with strict incomplete information as is the case in an SMR protocol, a rational player P^i with private information t_r^i will not know their own future private information values (required to choose their actions), the private informations of the other players, or str^{-i}, before choosing str^i. Therefore, P^i must choose the mixed strategy which maximises P^i's expected stake at round r', denoted $E(s_{r'}^i \cdot Stake_{r'})$, according to the probability distribution that P^i attributes to possible values for these unknowns. This distribution will be contained in t_r^i.

Thus, knowing t_r^i is sufficient to calculate P^i's expected utility of a particular strategy at round r', which we express mathematically by $E(s_{r'}^i \cdot Stake_{r'} | t_r^i, str^i)$. We state this formally in Definition 4.6.

Definition 4.6. For a rational player P^i, with private information t_r^i, mixed strategy str^i, and a particular round $r' \geq r$, the *expected utility of* str^i *for* P^i *at round* r' is denoted $\overline{u}_{r'}^i(t_r^i, str^i)$ and is described by $\overline{u}_{r'}^i(t_r^i, str^i) = E(s_{r'}^i \cdot Stake_{r'} | t_r^i, str^i)$.

As such, for a rational P^i in an SMR protocol Π with private information t_r^i, P^i will choose the mixed strategy str^i which maximises $\overline{u}_{r'}^i(t_r^i, str^i)$. To establish the existence, or not, of such a mixed strategy, we introduce an inequality in Definition 4.7 which allows us to pairwise rank mixed strategies by expected utility.

Definition 4.7. For a rational player P^i and two mixed strategies str_1^i, str_2^i, if there exists $r'' \geq r$, $r'' =$ polynomial in κ , such that for all $r' > r''$, $\overline{u}_{r'}^i(t_r^i, str_1^i) > \overline{u}_{r'}^i(t_r^i, str_2^i)$ then we say str_1^i *strictly dominates* str_2^i *in expectation*. If str_1^i strictly dominates str_2^i in expectation, we denote this relationship by $str_1^i >_u str_2^i$.

Using the strict dominance in expectancy relationship, we can formally define what we require from an SMR protocol in order for rational players to follow the recommended protocol strategy. This requirement is strong incentive compatibility in expectation, and is defined in Definition 4.8.

Definition 4.8. An SMR protocol Π is *Strong INcentive Compatible in Expectation* (SINCE) if for any rational player P^i with set of mixed strategies Str^i, for all mixed strategies $str^i \in Str^i$ such that $str^i \not\equiv str^{\Pi}$, it is the case that $str^{\Pi} >_u str^i$.

For a protocol to be SINCE in the ByRa model ensures that all rational players will follow the recommended protocol strategy. However, SINCE is not on its own sufficient to ensure the safety and liveness of an SMR protocol in ByRa model. It is still possible for an adversary to gain more than their fair share of rewards, and as such, increase their total share above the critical threshold of b. Towards achieving SMR in the ByRa model, it must be ensured that the adversarial share remains strictly bounded by the threshold b required to achieve SMR if all non-adversarial players follow the protocol. We explicitly define what we mean by fairness in the ByRa model in Definition 4.9.

Definition 4.9. An SMR protocol Π is *fair* in the ByRa model if given an adversary A, for any round $r \geq 1$ in Π, $P(s_r^A \leq s_1^A) \geq 1 - e^{-\kappa}$.

With SINCE and fairness, we have two intuitive properties which turn out to be crucial in achieving ByRa SMR. In Section 5, we show that it is impossible to guarantee the actions of players controlling more than $1 - b$ of the stake if these properties do not hold. Explicitly, we prove that the properties of SINCE and fairness are necessary, and together sufficient, to achieve ByRa SMR.

5 Achieving SMR in the ByRa Model

Towards our final goal of proving that the properties of SINCE and fairness are necessary, and together sufficient, to achieve ByRa SMR, the first step is to prove in Lemma 5.6 that SINCE is necessary. To allow us to prove this result, we introduce notation which allows us to consider, for a potential SMR protocol, the strategies from which rational players choose.

Definition 5.1. For a rational player P^i with a set of mixed strategies Str^i, let $Str^i_{\text{NSD}} \subseteq Str^i$ be such that for all $str^i \in Str^i_{\text{NSD}}$, there does not exist a $str^i_u \in Str^i$, such that $str^i_u >_u str^i$.

That is, if a mixed strategy $str \in Str^i$ is in the set Str^i_{NSD}, there is No strategy for P^i which Strictly Dominates str in expectancy. We provide the following Lemmas towards establishing that rational players will choose strategies exclusively from Str^i_{NSD}.

Lemma 5.2. For an SMR protocol Π, a rational player P^i, any strategy $str^i_1 \in Str^i$, and $|Str^i| \geq 2$, either $str^i_1 \in Str^i_{\text{NSD}}$ or there is some $str^i_2 \in Str^i_{\text{NSD}}$ such that $str^i_2 >_u str^i_1$.

Proof. We will do this by induction over the cardinalities of Str^i. First we check $|Str^i| = 2$. If str^i_1 is in Str^i_{NSD}, we are finished. Assume otherwise. That is, $str^i_2 >_u str^i_1$, which implies $str^i_1 \not>_u str^i_2$, and as such, $str^i_2 \in Str^i_{\text{NSD}}$ as required.

Assume the inductive hypothesis for $|Str^i| = k$.

Now, given this assumption, we must prove our hypothesis holds for $|Str^i| = k+1$. Consider a strategy $str^i_3 \in Str^i$. We need to prove either $str^i_3 \in Str^i_{\text{NSD}}$, or there exists $str \in Str^i_{\text{NSD}}$ with $str >_u str^i_3$. If str^i_3 is not strictly dominated by any strategy $str \in Str^i$, then $str^i_3 \in Str^i_{\text{NSD}}$.

Assume instead there exists some strategy $str^i_1 \in Str^i$, $str^i_1 >_u str^i_3$. Consider $Z^i = Str^i \backslash \{str^i_3\}$. By the inductive assumption, either $str^i_1 \in Z^i_{\text{NSD}}$, or there exists $str^i_2 \in Z^i_{\text{NSD}}$ such that $str^i_2 >_u str^i_1$. If $str^i_1 \in Z^i_{\text{NSD}}$, then $str^i_1 \in Str^i_{\text{NSD}}$, which implies there exists $str \in Str^i_{\text{NSD}}$ such that $str >_u str^i_3$. Otherwise, if $str^i_1 \notin Z^i_{\text{NSD}}$, there exists $str^i_2 \in Z^i_{\text{NSD}}$, with $str^i_2 >_u str^i_1$. As $str^i_2 >_u str^i_1$, and $str^i_1 >_u str^i_3$, this implies $str^i_2 >_u str^i_3$. As $str^i_2 \in Z^i_{\text{NSD}}$, and $str^i_2 >_u str^i_3$, this implies $str^i_2 \in Str^i_{\text{NSD}}$. Therefore, there exists $str \in Str^i_{\text{NSD}}$ such that $str >_u str^i_3$.

⊠

As rational players choose uniformly at random from all mixed strategies which maximise utility, from Lemma 5.2 for a rational player P^i these mixed strategies will be contained in Str^i_{NSD}. Moreover, Definition 4.1 states P^i chooses from these mixed strategies in Str^i_{NSD} with uniform probability. Therefore, to ensure rational players follow str^Π with probability at least $1 - e^{-\kappa}$, we must identify the conditions where for any rational player P^i, $Str^i_{\text{NSD}} = \{str^\Pi\}$. We state this explicitly in Observation 5.3.

Observation 5.3. A rational player P^i follows str^Π with probability at least $1 - e^{-\kappa}$, if and only if $Str^i_{\text{NSD}} = \{str^\Pi\}$.

The precise conditions where $Str^i_{\text{NSD}} = \{str^\Pi\}$ for a rational player P^i are identified in Lemma 5.4.

Lemma 5.4. For an SMR protocol Π and a rational player P^i, $Str^i_{\text{NSD}} = \{str^\Pi\}$ if and only if Π is strong incentive compatible in expectation.

Proof. If an SMR protocol Π is SINCE, then for any rational player P^i, str^Π strictly dominates all other strategies in expectation. This implies $Str^i_{\text{NSD}} = \{str^\Pi\}$.

Now we need to show if $Str^i_{\text{NSD}} = \{str^\Pi\}$, then Π is SINCE. From Lemma 5.2, we know for any strategy str^i_1, either $str^i_1 \in Str^i_{\text{NSD}}$ or there is some $str^i_2 \in Str^i_{\text{NSD}}$ such that $str^i_2 >_u str^i_1$. As the only strategy in Str^i_{NSD} is str^Π, this implies for any strategy $str^i_1 \not\equiv str^\Pi$, $str^\Pi >_u str^i_1$. This implies Π is SINCE, as required. ⊠

Corollary 5.5. For an SMR protocol Π and a rational player P^i, $P(P^i \text{ chooses } str^\Pi) \geq 1 - e^{-\kappa}$ if and only if Π is strong incentive compatible in expectation.

Proof. Follows from Observation 5.3 and Lemma 5.4. ⊠

This allows us to prove SINCE is a necessary property to achieve ByRa SMR.

Lemma 5.6. For an SMR protocol Π, if Π is not strong incentive compatible in expectation, then Π fails in the ByRa model.

Proof. Consider such a protocol Π. As a consequence of not SINCE, for a rational player P^i, this means $P(P^i \text{ chooses } str^\Pi) < 1 - e^{-\kappa}$, applying Corollary 5.5. From Definition 4.5 we are required to consider s^A_1 maximal. Given this rational P^i and a maximal adversary, there is now players controlling greater than or equal to b of the total stake who will not choose a strategy equivalent to str^Π with probability greater than $e^{-\kappa}$. Using the notation of Definition 4.5, this means $p^\Pi_{r'} < 1 - e^{-\kappa}$ for some $r' \geq 1$, which implies Π fails in the ByRa model. ⊠

Using similar arguments, we are able to prove fairness is also necessary for a protocol to achieve ByRa SMR.

Lemma 5.7. For an SMR protocol Π, if Π is not fair then Π fails in the ByRa model.

Proof. If Π is not fair, there exists $r' \geq 1$ such that $P(s^A_{r'} > s^A_1) > e^{-\kappa}$. From Definition 4.5, we are required to consider the case where s^A_1 is maximal. In this case, the probability that the adversary controls greater than or equal to b of the stake at round r' is greater than $e^{-\kappa}$ given $P(s^A_{r'} > s^A_1) > e^{-\kappa}$. Given the uniform strategy selection probability of Byzantine players across all possible strategies, this implies that $p^\Pi_{r'} < 1 - e^{-\kappa}$. Therefore, Π fails in the ByRa model. ⊠

Collecting the results of this section, with some additional proof-work, we are equipped to prove the main theorem of the paper, Theorem 5.8.

Theorem 5.8. For an SMR protocol Π, Π achieves ByRa SMR if and only if Π is strong incentive compatible in expectation and fair.

Proof. For an SMR protocol Π, we will first prove that if Π achieves ByRa SMR then Π is SINCE and fair. Using the contrapositive of Lemma 5.6, we have that if Π achieves ByRa SMR (does not fail in the ByRa model), then Π is SINCE. Similarly, using the contrapositive of Lemma 5.7, we have that if Π achieves ByRa SMR, then Π is fair.

We now need to prove if Π is SINCE and fair then Π achieves ByRa SMR. By SINCE and Corollary 5.5, this implies all rational players will always choose str^Π. Furthermore, as Π is fair, from Definition 4.9, we know rational players will maintain greater than $1 - b$ of the stake in every round with probability greater than $1 - e^{-\kappa}$. Therefore, we have players controlling greater than $1 - b$ of the stake who will follow str^Π with probability greater than $1 - e^{-\kappa}$, which is precisely the definition of Π achieving ByRa SMR from Definition 4.5. ⊠

This crucial theorem completes the first part of the paper, identifying the properties of SINCE and fairness as both necessary, and together sufficient, for a protocol to achieve ByRa SMR, independently of network assumptions and adversarial capabilities. We now proceed to outline the FAIRSICAL protocol, demonstrating that it is possible to satisfy SINCE and fairness in the ByRa model.

6 The FAIRSICAL Protocol

Having demonstrated the vital nature of SINCE and fairness for SMR protocols in the ByRa model, we now describe FAIRSICAL, a FAIR, Strong Incentive Compatible in expectation ALgorithm for achieving ByRa SMR. FAIRSICAL demonstrates that is possible to achieve SINCE and fairness in SMR protocols. Furthermore, the detailed algorithmic description in our generic blockchain framework serves as a template for any SMR protocol to progress towards SINCE and fairness.

6.1 Threat Model

We assume the existence of a functionality Broadcast for delivering all messages, including those sent by the adversary. Messages sent through Broadcast are authenticated using an unforgeable signature scheme so that only P^i can convince any other player P^j that a message signed by P^i was sent by P^i. Messages sent through Broadcast during some round are guaranteed to be delivered to all players at the beginning of the following round.

In our model, we consider an adversary A with the following properties:

1. The adversary A can read all messages sent during a particular round before deciding on the Byzantine player actions for that round.
2. The adversary A can control and coordinate all Byzantine players in any way, with unknown utility function.
3. At initialisation we have $\frac{1}{2} - \delta < s_1^A < \frac{1}{2} = b$, for $\delta > 0$, in line with the synchronous network distribution [19] enforced by the Broadcast functionality.
4. After deciding on a block for height $H \geq 1$, the adversary A chooses any f players, say $P^1, ..., P^f$, to become Byzantine before beginning consensus on a block for height $H + 1$ with $1 \leq f < n - 1$, such that $\sum_{i=1}^{f} s_H^i = s_H^A$.
5. Given the adversary chooses players $P^1, ..., P^f$ as Byzantine for consensus on a block at height H with shares $[s_{H-1}^1, ..., s_{H-1}^f]$, the adversarial share for the proceeding consensus height is calculated as $s_H^A = \sum_{i=1}^{f} s_H^i$.

Remark 6.1. As Byzantine players have unknown utility functions, rational players a priori assume Byzantine players will choose from all possible strategies with uniform probability.

6.2 Protocol Description

As mentioned in Section 3, the protocol proceeds in rounds. In the FAIRSICAL protocol, there are 4 rounds which need to be successfully completed in order to decide on a block to be added to the blockchain. These are Propose, Validate, Commit, and Decide, formally described at the end of this section, but which we describe informally here for reader intuition:

- Propose: Players propose a block to be added to the blockchain.
- Validate: Players select a valid block to be added to the blockchain from the set of proposed blocks based on a protocol-defined selection rule.
- Commit: Players agree on (commit to) a block if it was validated by players controlling more than b of the stake.
- Decide: If players controlling more than b of the stake committed to a block, then it is added to the blockchain.

Blocks will only get added to the blockchain after a successful Decide round. Each round consists of 3 phases for each player P^i:

- Phase 1. Receive: All messages sent via Broadcast during previous rounds are received and available to read by P^i.

- Phase 2. Compute: Players perform polynomial in κ binary operations, with the prescribed protocol requiring polynomial in κ binary operations in each round.
- Phase 3. Send: Messages are sent.

In FAIRSICAL, following a successful Decide round, rewards in the form of newly minted stake are distributed to players who followed the protocol, while the stake of players decided to have deviated are deleted. Stake changes are computed during the Decide round, so will be received by the network at the beginning of the proceeding Propose round.

Each player P^i owns a public key, secret key pair ($pubKey^i$, $privKey^i$). At initialisation, there is $Stake_1$ stake in the system, with P^i owning $s_1^i \cdot Stake_1$ total stake. To participate in consensus for a block at height $H > 1$, a player P^i must own stake as decided in the block at height $H - 1$. As FAIRSICAL is intended as a template for SMR between players with pre-existing communication channels and some known distribution of stake between them, we omit any further setup implementation from this paper.

For $H \geq 1$, blocks are described by the tuple $(h(C[H-1]), V, \xi)$, where $h : \{0,1\}^* \to \{0,1\}^\kappa$ is a cryptographically secure hash function, $C = [B_1, ..., B_H]$ is a vector representation of a blockchain with indices corresponding to block heights in the blockchain, V contains state machine updates, and $\xi \in \{0,1\}^\kappa$ is a shared randomness value for the current block.

Each non-Byzantine player maintains a set of variables $\{H, Stake, R, S, C, \xi\}$. At initialisation, these variables are set to $\{H, Stake, R, S, C, \xi\} \leftarrow \{1, Stake_1, c, [s_1^1, ..., s_1^n], [B_1], \{1\}^\kappa\}$, for some $Stake_1, c > 0$ amounts of stake, with the genesis block $B_1 = (0, (Stake_1, S), \xi)$, .

For each height $H > 1$, we use $J_{\text{temp}} \subset \{1, ...n\}$ to track the indices of players that are observed to behave incorrectly during consensus for a block at height H. The total reward available for height $H > 1$ is $R = c \cdot d_{H-1}$ (algorithm: 5, line: 14), with $d_H = d_{H-1}(1 - \sum_{j \in J_{\text{temp}}} s_{H-1}^j)$, $d_1 = 1$.

Note that any player not in J_{temp} implicitly follows the protocol. However, we impose the requirement that for all players not included in the J_{temp} of P^i, there must be accompanying proofs that these players correctly followed the protocol with respect to P^i's state. The precise nature of these proofs are beyond the scope of this paper.

For P^i behaving correctly up to and including height H, they will receive $c \cdot d_{H-1} \cdot s_{H-1}^i$ stake (algorithm: 5, line: 20) for deciding on the block at height H. As we discount R after every successful block decision by the incorrect share of stake, $(\sum_{j \in J_{\text{temp}}} s_{H-1}^j)$ (algorithm: 5, line: 14), this has the same effect as recursively calculating the discount d_H, and applying it to c as described. During the Validate, Commit and Decide rounds for some height H in the protocol, if deviation by a player P^j is identified, j is added to J_{temp}, which happens at line 4 in each of the corresponding algorithms. If a majority of players identify deviation from P^j, P^j's stake is destroyed (algorithm: 5, line: 18). Note, $sig()$ is an unforgeable signature scheme, with $sig(privKey^i, ...)$ encrypting using the secret key of P^i, and can only be decrypted using the public key of P^i, while $sig(pubKey^i, ...)$ encrypts using the public key of P^i, and can only be decrypted using the private key of P^i. We also assume at all times it is possible to generate a valid SMR update. The function $validSMRUpdate()$ takes as input an SMR update V and blockchain C as described previously, returning TRUE if V is valid with respect to the state described by applying the state updates contained in C.

We now describe in detail the recommended protocol strategy of FAIRSICAL for a player P^i attempting to decide on a block at height $H + 1$, with $H \geq 1$.

Propose: P^i chooses a valid state machine update V with respect to the state described by the current chain C (algorithm: 2, line: 1) to generate a block $B_{\text{temp}} = (h(C[H]), V, \xi)$ to be proposed for height $H + 1$, then generates a randomness value ξ_{temp} (algorithm: 2, line: 2), and submits the vector $[B_{\text{temp}}, sig(privKey^i, \xi), sig(pubKey^i, \xi_{\text{temp}})]$ to Broadcast. The second value in this vector is P^i's private key encryption of the randomness ξ decided upon for height $H + 1$. This will be used to determine the unique block to be added to C in the Validate round. As it is encrypted using the private key, anyone knowing the public key can verify that it is indeed P^i who encrypted the value. The third entry is P^i's commitment to a randomness share ξ_{temp} using public key encryption. ξ_{temp} will be used to generate the final randomness value for height $H + 2$ during the Commit round.

Algorithm 1 FAIRSICAL

$Stake = Stake_1$ # total stake
$S \leftarrow [s_1^1, ..., s_1^n]$ # vector of player shares
$pubKeyArray \leftarrow getPublicKeys()$ # vector of public keys, accessible by P^i
$pubKey \leftarrow pubKeyArray[i]$ # P^i's public key
$privKey \leftarrow getPrivateKey(i)$ # retrieve P^i's private key
$\xi \leftarrow \{1\}^\kappa$ # initialise κ-bit randomness
$H \leftarrow 1$ # blockchain height
$R \leftarrow c$ # total reward for block decision
$C \leftarrow [(0, (Stake, S), \xi)]$ # vector representation of blockchain
while $H \geq 1$ **do** # Run Protocol
  $J_{\text{temp}} \leftarrow \emptyset$ # temporary set of deviating players for height H
  $B_{\text{temp}} \leftarrow ()$ # prospective block for height H
  $\xi_{\text{shares}} \leftarrow [\{0\}^\kappa$ **for** $i \in \{1, ..., n\}]$ # initialise array for randomness shares
  $\xi_{\text{temp}} \leftarrow \{0\}^\kappa$ # temporary variable for height $H+1$ randomness
  Propose(H, C, ξ, ξ_{temp}, $privKey$)
  Validate(S, B_{temp}, J_{temp}, ξ, ξ_{temp}, ξ_{shares})
  Commit(S, B_{temp} J_{temp}, ξ_{temp}, ξ_{shares}, $pubKeyArray$)
  Decide(H, $Stake$, R, S, C, B_{temp} J_{temp}, ξ, ξ_{temp})
end while

Algorithm 2 Propose(H, C, ξ, ξ_{temp}, $privKey$)

$V \leftarrow v$ **where** $validSMRUpdate(v, C) ==$ TRUE
$\xi_{\text{temp}} \leftarrow genRandomBitString(length = \kappa)$ # P^i's randomness share for ξ_{temp}
$B_{\text{temp}} \leftarrow (h(C[H]), V, \xi)$ # P^i's block proposal for H
Broadcast.$send([B_{\text{temp}}, sig(privKey, \xi), sig(pubKey, \xi_{\text{temp}})])$

Algorithm 3 Validate(S, C, B_{temp}, J_{temp}, ξ, ξ_{temp}, ξ_{shares})

$M \leftarrow$ Broadcast.$receive()$ # Read all Propose messages
for $m \in M$ **do**
  if isValid([Validate,m, S, ξ, $pubKeyArray$, C]) == FALSE **then**
    $J_{\text{temp}}.append(m.sender())$ # Identify deviators
    $M.remove(m)$ # remove invalid messages from consideration in block selection
  else
    $\xi_{\text{shares}}[m.sender()] \leftarrow m[3]$ # record randomness share commitments
  end if
end for
$d \leftarrow 1 - \sum_{j \in J_{\text{temp}}} S[j]$ # Calculate remaining share of correct stake
if $d \leq \frac{1}{2}$ **then** # Check if no majority in consensus on valid values
  continue # Ends current iteration of while loop in algorithm 2
end if
$selectedBlockHash = minimum([m[2]$ **for** $m \in M])$ # Criterion for block selection
for $m \in M$ **do** # Extract block to be validated based on $selectedBlockHash$ criterion
  if $m[2] == selectedBlockHash$ **then** # Checks if message
  # corresponds to $selectedBlockHash$
    $B_{\text{temp}} \leftarrow m[1]$ # Update B_{temp} to selected block
  end if
end for
Broadcast.$send([B_{\text{temp}}, J_{\text{temp}}, \xi_{\text{temp}}])$

Validate: P^i retrieves all messages from players sent through Broadcast in the Propose round for height $H+1$ from Broadcast. For all invalid Propose messages, P^i adds the index of the sender to J_{temp} (algorithm: 3, line: 4). If the sum of shares owned by players included in J_{temp} is greater than or equal to $\frac{1}{2}$, consensus cannot be reached for this set of Propose messages. As such, P^i returns to the Propose round for height $H+1$ (algorithm: 3, line: 12). Otherwise, P^i selects the block to be decided upon for height $H+1$ (algorithm: 3, line: 15) based on a selection rule (algorithm: 3, line: 14) that will select a unique block. In FAIRSICAL, we select the block that corresponds to the the smallest encrypted randomness value from the Propose round (the second entry in the tuple sent in the Propose round) received from a player P^k, $k \in \{1, ..., n\} \backslash J_{\text{temp}}$. Given P^k's block is selected, P^i sets B_{temp} equal to P^k's proposal for B_{temp} (algorithm: 3, line: 18). P^i submits the vector $[B_{\text{temp}}, J_{\text{temp}}, \xi_{\text{temp}}]$ to Broadcast. By doing so, P^i reveals their randomness share ξ_{temp}, which will be used to generate the shared randomness for height $H+2$.

Algorithm 4 Commit(S, B_{temp} J_{temp}, ξ_{temp}, ξ_{shares}, *pubKeyArray*)

```
1: M ← Broadcast.receive()
2: J ← J_temp.copy()                    # make a distinct record of deviators observed so far,
3:                                      # allows J_temp to remain updatable during isValid() checks
4: for m ∈ M do
5:   if isValid([Commit, m, S, pubKeyArray, ξ_shares, J ]) == FALSE then
6:     J_temp.append(m.sender())
7:   else
8:     ξ_shares[m.sender()] ← m[3]                    # record valid randomness shares
9:   end if
10: end for
11: d ← 1 − Σ_{j∈J_temp} S[j]                         # Calculate remaining share of correct stake
12: if d ≤ 1/2 then
13:   continue                          # Ends current iteration of while loop in algorithm 2
14: end if
15: ξ_temp ← {0}^κ
16: for x ∈ ξ_shares do                 # Calculate ξ_temp as XOR sum of valid randomness shares
17:   ξ_temp ← ξ_temp ⊕ x
18: end for
19: Broadcast.send([B_temp, J_temp, ξ_temp])
```

Commit: P^i retrieves all messages from players sent through Broadcast in the Validate round for height $H+1$ from Broadcast. For all invalid Validate messages with respect to the isValid() function, P^i adds the index of the sender to J_{temp}. If a message satisfies isValid() $== TRUE$, P^i adds the corresponding randomness share to the set ξ_{shares} (algorithm: 4, line: 8) for consideration when calculating the proposed global randomness for height $H+2$. If the sum of shares owned by players included in J_{temp} is greater than or equal to $\frac{1}{2}$, P^i returns to the Propose round for height $H+1$ as no consensus can be reached for the current set of Validate messages. Otherwise, P^i computes the prospective randomness value, setting ξ_{temp} equal to the XOR sum of all valid randomness shares received during this Commit round (algorithm: 4, lines: 16-17). P^i submits the vector $[B_{\text{temp}}, J_{\text{temp}}, \xi_{\text{temp}}]$ to Broadcast.

Decide: P^i retrieves all messages from players sent through Broadcast in the Commit round for height $H+1$ from Broadcast. For all invalid Commit messages, P^i adds the index of the sender to J_{temp}. If the sum of shares owned by players included in J_{temp} is greater than or equal to $\frac{1}{2}$, P^i returns to the Propose round for height $H+1$. Otherwise, given the remaining correct share of stake d, P^i sets the total stake in the system $Stake \leftarrow d \cdot Stake$ (algorithm: 5, line: 13), and reward $R \leftarrow d \cdot R$ (algorithm: 5, line: 14). This discounts both the total stake in the system and the total reward to be paid out for the current block to reflect the removal of the malicious stake in line with our recursive formulae for d_H. P^i updates the

Algorithm 5 Decide(H, *Stake*, R, S, C, B_{temp} J_{temp}, ξ, ξ_{temp})

```
1:  M ← Broadcast.receive()
2:  J ← J_temp.copy()                    # make a distinct record of deviators observed so far,
3:                                       # allows J_temp to remain updatable during isValid() checks
4:  for m ∈ M do
5:    if isValid([Decide, m, S, ξ_temp, J]) == FALSE then
6:      J_temp.append(m.sender())
7:    end if
8:  end for
9:  d ← 1 − Σ_{j∈J_temp} S[j]            # Calculate remaining share of correct stake
10: if d ≤ 1/2 then
11:   continue                           # Ends current iteration of while loop in algorithm 2
12: end if
13: Stake ← d · Stake                    # Delete deviator stake
14: R ← d · R                            # Discount total rewards by deviator stake
15: Stake ← Stake + R                    # Add rewards to the system
16: for i ∈ {1, ..., n} do
17:   if i ∈ J_temp then
18:     S[i] ← 0
19:   else if i ∉ J_temp then
20:     S[i] ← S[i]/d                    # Adjust honest shares to reflect deletion of deviator shares
21:   end if
22: end for
23: ξ ← ξ_temp                           # Update global randomness now that block decided
24: C.append(B_temp)                     # Add block to blockchain, applying the state machine update
25:                                      # in B_temp to P^i's view of the SMR
26: H ← H + 1
```

total stake $Stake \leftarrow Stake + R$ (algorithm: 5, line: 15) to include the reward for this round. Then, P^i adjusts the stakes in the system to reflect the removal of deviating players from the protocol (algorithm: 5, line: 16). First, for all $j \in J_{\text{temp}}$, P^i sets $S[j] \leftarrow 0$, deleting the stake of deviating players (algorithm: 5, line: 18). Secondly, for all $i \in \{1, ..., n\} \backslash J_{\text{temp}}$, P^i sets $S[i] \leftarrow \frac{S[i]}{d}$ to normalise the remaining non-deviating stake in the system to 1 (algorithm: 5, line: 20). P^i sets $\xi \leftarrow \xi_{\text{temp}}$ to reflect the consensus on ξ_{temp} as the randomness for height $H + 2$ (algorithm: 5, line: 23), and appends B_{temp} to C (algorithm: 5, line: 24). Once B_{temp} is added to C, the state machine update contained in B_{temp} is applied to the state described by $C[H]$. P^i increments H, and proceeds to the Propose round for the next block.

7 Proving FAIRSICAL achieves ByRa SMR

To ensure FAIRSICAL achieves ByRa SMR, we first prove that the FAIRSICAL protocol achieves SMR in the legacy honest players controlling a majority of stake setting (Section 7.1). Given this result, we then prove that FAIRSICAL satisfies the properties of SINCE (Section 7.2) and fairness (Section 7.3) in the ByRa model, and as such, using Theorem 5.8, that FAIRSICAL achieves ByRa SMR.

7.1 Safety and Liveness when Correct Players Control a Majority of Stake

Before it is possible to prove a protocol can achieve SMR in the ByRa model, we must prove that it achieves SMR in the presence of correct players controlling a majority of the stake. This is the minimum requirement a protocol must satisfy to be considered an SMR protocol. In this section, we prove that FAIRSICAL satisfies safety and liveness when correct players control a majority of stake in each round. To prove safety, we first identify an important implicit

Algorithm 6 isValid(*input*)

```
1: step ← input[1]
2: m ← input[2]                                              # message vector
3: S ← input[3]                                              # shares vector
4: j = m.sender()                                            # sender's player index
5: if S[j] == 0 then                                         # check if sender has non-zero share
6:    return FALSE
7: end if
8: if step == Validate then                                  # m = [B_temp, sig(privKey, ξ), sig(pubKey, ξ_temp)]
9:    ξ ← input[4]                                           # global randomness for current height
10:   pubKeyArray ← input[5]
11:   C ← input[6]
12:   if validSMRUpdate(m[1], C) == FALSE then              # check SMR update validity
13:      return FALSE
14:   end if
15:   if sig(pubKeyArray[j], m[2]) == ξ then                # check sender's encryption of global
16:                                                         # randomness is correct
17:      return TRUE
18:   else
19:      return FALSE
20:   end if
21: else if step == Commit then                             # m = [B_temp, J_temp, ξ_temp]
22:   J_temp ← m[2]                                         # sender's suggestion for J_temp
23:   ξ_temp ← m[3]                                         # sender's randomness share
24:   pubKeyArray ← input[4]
25:   ξ_shares ← input[5]                                   # vector of randomness share encryptions
26:   J ← input[6]                                          # P^i's version of J_temp
27:   if sig(pubKeyArray[j], ξ_temp) ≠ ξ_shares[j] then     # check share matches commitment
28:      return FALSE
29:   end if
30:   if J ≠ J_temp then                                    # check P^i's deviator set matches sender's
31:      return FALSE
32:   else
33:      return TRUE
34:   end if
35: else if step == Decide then                             # m = [B_temp, J_temp, ξ_temp]
36:   J_temp ← m[2]                                         # sender's suggestion for J_temp
37:   ξ_temp ← m[3]                                         # sender's proposed new height randomness
38:   ξ ← input[4]                                          # P^i's proposed new height randomness
39:   J ← input[5]                                          # P^i's version of J_temp
40:   if ξ ≠ ξ_temp then                                    # check P^i's randomness matches sender's
41:      return FALSE
42:   end if
43:   if J ≠ J_temp then                                    # check P^i's deviator set matches sender's
44:      return FALSE
45:   else
46:      return TRUE
47:   end if
48: end if
```

property of deviator sets and their complements within the FAIRSICAL protocol. If a player P^i never appears in the deviator set of some other correct player P^j, this implicitly means P^i always provably agrees with P^j. Specifically, P^i always provably agrees with P^j's SMR updates. This is stated formally in Lemma 7.1.

Lemma 7.1. For any two correct players P^i and P^j, $i \neq j$, if P^i is never included in P^j's deviator set, and P^j decides on an SMR update V^j for some round r, P^i also decides on V^j for round r.

Proof. As P^j is correct, and decides on an SMR update V^j in round r, that must mean P^j also decides on a deviator set corresponding to round r. We know P^i is never added to P^j's deviator set, meaning P^j must observe P^i as following the protocol in all rounds up to, and including, round r. This means P^i submits a valid commit message including V^j in round $r-1$. As P^j decides on V^j, P^i is correct, and P^i receives the same messages as P^j due to the Broadcast functionality in the Threat Model (Section 6.1), P^i also decides on V^j in round r. ⊠

We are now equipped to prove the safety of FAIRSICAL given correct players control a majority of stake.

Lemma 7.2. Given correct players control a majority of stake, FAIRSICAL satisfies safety.

Proof. For safety, we require for any two correct players P^i and P^j, $i \neq j$, if P^j decides on an SMR update V^j at height H, and P^i decides on an SMR update V^i at height H, then $V^i = V^j$.

From Lemma 7.1, this is equivalent to proving P^i will never appear in P^j's deviator set at any point in FAIRSICAL, and vice versa. Without loss of generality, we prove that P^j never adds P^i to their deviator set, with the proof that P^i never adds P^j analogous. To do this, we will assume otherwise and reach a contradiction. Therefore, let us assume at some round in the protocol, P^i is added to P^j's deviator set.

Assume P^i is added to P^j's deviator set during a Validate round (algorithm: 3, line: 4). Therefore, either P^i did not submit a valid SMR update (algorithm: 6, line: 12), or P^i sent an incorrect encryption of the height randomness (algorithm: 6, line: 15). As P^i is correct, neither of these occur. Therefore, P^i is not added to P^j's deviator set during a Validate round.

Assume P^i is added to P^j's deviator set during a Commit round (algorithm: 4, line: 6). Therefore, either P^i sent a different deviator set to that of P^j (algorithm: 6, line: 30), or P^i sent an incorrect encryption of their randomness share (algorithm: 6, line: 27). We have already seen that no correct player will add a player to a deviator set who sends a message satisfying the isValid() conditions of a Propose round. Let P^k be a player sending a Propose message not satisfying the isValid() conditions of a correct player, $i \neq k \neq j$. Due to the Broadcast functionality, all players observe P^k's deviation. By the correctness of P^i and P^j, both will add P^k to their deviator sets. Therefore, P^i and P^j must have the same deviator set at the end of the preceding Validate round. Furthermore, as P^i is correct, P^i will send the correct randomness share encryption. This contradicts the assumption. Therefore, P^i is not added to P^j's deviator set during a Commit round.

Assume P^i is added to P^j's deviator set during a Decide round (algorithm: 5, line: 6). Analogously to a Commit round, for a Decide round, P^i and P^j will have the same deviator set at the end of the preceding Commit round. Therefore, P^i must have submitted an incorrect randomness share (algorithm: 6, line: 40). As P^i is correct, this is not the case, meaning P^i will not appear in P^j's deviator set during a Decide round.

Therefore, P^i is never added to P^j's deviator set, as required. ⊠

Next, we prove that FAIRSICAL satisfies liveness given correct players control a majority of stake.

Lemma 7.3. Given correct players control a majority of stake, FAIRSICAL satisfies liveness.

Proof. To prove liveness, we require for any height H in the blockchain, every correct player eventually decides on an SMR update for height H.

To do this, we will prove that for any correct player P^i in the presence of correct players controlling a majority of stake decides on an SMR update in exactly 4 rounds, meaning an SMR update will be decided for height H in exactly $4(H-1)$ rounds.

As P^i is correct, they will complete the Propose round and progress to the Validate round. P^i receives valid Propose messages from at least the correct players controlling a majority of stake. This means the share of correctly participating stake for the Propose round is strictly greater than $\frac{1}{2}$, which implies P^i will progress to the Commit round. Again, we know that at least the correct players controlling a majority of stake will have submitted valid Validate messages. This means the share of correctly participating stake in the Validate round is strictly greater than $\frac{1}{2}$, sufficient for P^i to progress to the Decide round. Consequently, due to the correct players controlling a majority of stake, there will be valid Commit votes from players controlling more than $\frac{1}{2}$ of the total stake. This results in P^i successfully completing the Decide round, and as such, deciding on the SMR update contained in the block they append to their blockchain.

⊠

Now that we have proved FAIRSICAL achieves SMR when correct players control a majority of stake, we can advance to proving it achieves ByRa SMR.

7.2 Proving FAIRSICAL is Strong Incentive Compatible in Expectation

To prove the FAIRSICAL protocol is SINCE, we show that following the protocol is the strategy which strictly maximises rewards. Recall, rewards for height H are paid to players not in J_{temp}, for J_{temp} decided upon by the protocol.

We first show that the reward for a player P^k who decides on a block at height H is equal to $c \cdot s_1^k$, for all $H > 1$, where c is the total reward to be shared by all players for a block decision at initialisation, and s_1^k is P^k's share of stake at initialisation. This means the per-block reward received by any rewarded player is constant.

Lemma 7.4. In FAIRSICAL, for any player P^k who decides on a block at height $H > 1$, they receive a reward of $R_H^k = c \cdot s_1^k$ for height H.

Proof. From Section 6, we know $R_H = c \cdot d_{H-1}$, $d_H = d_{H-1}(1 - \sum_{j \in J_{\text{temp}}} s_{H-1}^j)$ for J_{temp} the decided on deviator set for height H, $\forall\, H > 1$, $d_1 = 1$, with $R_H^k = R_H s_{H-1}^k$ (algorithm: 5, line: 14). We want $R_H^k = c \cdot s_1^k$, $\forall\, H > 1$. We will prove this by induction. In the following, we will let $Y_H = (1 - \sum_{j \in J_{\text{temp}}} s_{H-1}^j)$.

First we check $H = 2$. As $d_1 = 1$, this gives $R_2 = c$ (algorithm: 1, line: 8). Therefore, $R_2^k = R_2 s_1^k = c \cdot s_1^k$.

Assume the inductive hypothesis for $H = h$. That is, $R_h^k = c \cdot s_1^k$. This means $R_h = c\frac{s_1^k}{s_{h-1}^k}$, which implies $d_{h-1} = \frac{s_1^k}{s_{h-1}^k}$ (as $R_H = c \cdot d_{H-1}$). Looking ahead to $H = h+1$, we want an expression for d_h in terms of $\frac{s_1^k}{s_h^k}$. We know $d_h = d_{h-1} Y_h$. Observe that $Y_h(N_{h-1} + R_h) = N_h$ (algorithm: 5, lines: 13-15), and also that for a player P^k who decides on a block at height h, that $s_h^k = \frac{s_{h-1}^k (N_{h-1}+R_h)}{N_h}$. We can rewrite the former as $\frac{N_{h-1}+R_h}{N_h} = \frac{1}{Y_h}$. This implies $s_h^k = s_{h-1}^k \frac{N_{h-1}+R_h}{N_h} = s_{h-1}^k \frac{1}{Y_h}$, which gives $Y_h = \frac{s_{h-1}^k}{s_h^k}$. Therefore, as $d_{h-1} = \frac{s_1^k}{s_{h-1}^k}$ and $Y_h = \frac{s_{h-1}^k}{s_h^k}$, we have $d_h = d_{h-1} Y_h = \frac{s_1^k}{s_{h-1}^k} \frac{s_{h-1}^k}{s_h^k} = \frac{s_1^k}{s_h^k}$.

Now, given this assumption, we must prove our hypothesis holds for $H = h+1$. We know $R_{h+1}^k = R_{h+1} s_h^k$, and $R_{h+1} = c \cdot d_h$. From the previous point, we have $d_h = \frac{s_1^k}{s_h^k}$, which implies $R_{h+1} = c\frac{s_1^k}{s_h^k}$. Therefore $R_{h+1}^k = c\frac{s_1^k}{s_h^k} s_h^k = c \cdot s_1^k$ as required.

⊠

Now we show that for a rational player P^i, P^i is strongly incentivised in expectation to follow the protocol.

Theorem 7.5. FAIRSICAL is strong incentive compatible in expectation in the ByRa model.

Proof. For FAIRSICAL to be SINCE, we require that for a rational P^i, their expected utility is maximised by following the protocol. For a player to be rewarded at height H, they must not be included in the decided deviator set J_{temp} for that height (algorithm: 5, line: 19), which means no player will ever submit a J_{temp} set containing themselves. To incorrectly abort a block at a Decide round foregoes the chance to earn a reward for a decided block by entering the Propose round of an incorrect height, which no rational player will do. Therefore, we only need to examine strategies for rounds where messages are sent.

From Lemma 7.4, if a player P^i decides on a block, they will always receive $c \cdot s_1^i$, so all strategies which result in deciding on a block have the same positive reward for players who reach a decision on the block.

As stake change only occurs after successful Decide rounds (algorithm: 5, lines: 13-19), the maximum possible stake gain at round r' for P^i given the current round is r is $\lceil \frac{r'+1-r}{4} \rceil c \cdot s_1^i$. To maximise the expected stake at some round r', this is equivalent to successfully completing as many Decide rounds as possible in expectation, then finishing as close as possible to a Decide round, while also maximising the probability of remaining in the protocol at round r'. We will show these conditions are uniquely achieved by following the protocol.

We can represent any deviator set J_{temp} as a binary string of length n, with 1 in position j representing the statement 'P^j has deviated'. For P^i, an invalid message from P^j will be identified by the isValid() algorithm (algorithm 6). An invalid message from P^j therefore is equivalent to at least one of the following:

- Violating the *validSMRUpdate*() predicate (algorithm: 6, line: 12).
- Sending an incorrect randomness derivative (algorithm: 6, lines: 15, 27, 40).
- Submitting an incorrect J_{temp} (algorithm: 6, lines: 30, 43).

Based on the unforgeability of messages in our protocol, it is impossible for P^i to claim correct behaviour of a deviating player P^j if no such behaviour was observed with respect to P^i's state (noted in Section 6.2). Therefore, for all players submitting a binary string representation of the J_{temp} set, the indices of players who violate the *validSMRUpdate*() predicate or send an incorrect randomness derivative are set to 1. Due to the Broadcast functionality, these identified deviators will be consistent for all players. We already know a rational player P^i always sets position i in their own deviator string to 0, which means they will never violate the *validSMRUpdate*() predicate or send an incorrect randomness derivative. Therefore, to prove SINCE, we need to show that submitting the correct J_{temp} in every round simultaneously maximises the expected positive payoff of a player while minimising the probability of being removed from the protocol.

We first prove that submitting the correct J_{temp} in every possible round has the unique highest probability of being agreed upon, which is equivalent to maximising the expected number of successful (Decide) rounds. Every message sent by a player P^i must, for every other P^j, $j \in \{1, ..., n\} \backslash \{i\}$, either prove P^j honestly followed the protocol up until that point in the protocol, or claim they deviated. We know for any $j \neq i$, P^j will set position j of their own string to 0. Furthermore, in the worst case scenario for agreement, P^j will set all other positions of non-deviating players, in P^j's view, to 0 with probability 0.5. As such, for m players following the protocol up to a particular round according to P^i, P^i rational, the correct deviator set will match that of any other player P^j with probability greater than or equal to 0.5^{m-1}.

Consider a player P^j, $j \neq i$, such that P^j's share is $\delta > 0$. For an incorrect deviator set from P^i accusing P^j of deviating, the probability of choosing the same deviator string as another player, sampling in proportion to player shares, reduces by at least $0.5^{m-1}\delta$. This is because for an incorrectly assigned 1 in position j of P^i's deviator string, the respective deviator strings of P^i and P^j will be different. This is opposed to a probability of matching of at least 0.5^{m-1}

if position j was 0. The probability of having a matching deviator string with any other player P^k, $j \neq k \neq i$, does not change in expectancy in this worst-case assignment of 0s and 1s. This means the most probable submitted deviator set is the correct one, which implies following the protocol strictly maximises the number of Decide rounds that can be reached.

As following the protocol strictly maximises the expected positive payoff, to prove that following the protocol maximises expected utility, we only have to show that no other strategy increases the probability of remaining in the protocol at round r'. Any other strategy involves choosing some incorrect J_{temp} to submit during some round $r'' \leq r'$.

P^i is only removed from the protocol (algorithm: 5, line: 18) if the players remaining in the protocol at round r' submit a J_{temp} set in a previous round r'' not matching that of P^i. Therefore, to maximise their probability of remaining in the protocol, P^i must choose the most probable J_{temp}. We have already seen that this is achieved by submitting the correct J_{temp}.

As such, a rational player submits the correct J_{temp} to maximise their expected utility. This corresponds to SINCE.

⊠

7.3 Proving FAIRSICAL is Fair

As we have shown FAIRSICAL is strong incentive compatible in expectation, we now need to show that there will always exist a majority of rational players. To ensure such a majority always exists, it is required that the adversary is unable to increase their share in any round of the protocol. For this, we require fairness.

Theorem 7.6. FAIRSICAL is fair in the ByRa model.

Proof. From Theorem 7.5, we have that in the presence of rational players controlling a majority of stake, this majority will always follow the protocol. If the adversary follows the protocol given this majority, they will always be rewarded, and thus for a starting share s_1^A, receive $c \cdot s_1^A$ out of a total reward of c at each height $H > 1$ (from Lemma 7.4), and thus $s_H^A = s_1^A$.

If the adversary deviates from the protocol with one or more of the Byzantine players given rational players control a majority of stake, it will be identified by the rational players and thus these Byzantine players will be added to the corresponding J_{temp} set, destroying the offending players' stake, resulting in an overall loss in adversarial share. Therefore, as the adversarial share is decreasing, with equality if and only if the adversary follows the protocol, the protocol is fair. ⊠

Corollary 7.7. FAIRSICAL achieves ByRa SMR.

Proof. The result follows by applying Theorems 7.5 and 7.6 to Theorem 5.8. ⊠

8 Conclusion

We provide a game theoretic framework for analysing SMR protocols. Although many previous attempts have been made, we are, to the best of our knowledge, the first to formally treat SMR protocols as games involving only rational and adversarial players. We detail the ByRa model for player characterisation in SMR protocols, an update to the legacy BAR model, removing the dependency on altruistic players in an era of unprecedented market capitalisation of tokenised SMR protocols. We demonstrate that the properties of strong incentive compatibility in expectation and fairness as described in this paper, are both necessary, and together sufficient to achieve SMR in the ByRa model. We then provide the FAIRSICAL protocol as an example of a protocol that achieves ByRa SMR, which is of independent interest both as a strong incentive compatible in expectation and fair protocol in the ByRa model, but also as an easy-to-understand standard for addressing the shortcomings of current protocol guarantees in the ByRa model. The proof techniques we use provide accessible methodologies with which SMR protocols can be analysed in this new game theoretic framework.

Although we provide FAIRSICAL as an example of a protocol which achieves ByRa SMR under weak adversarial assumptions, weaker communication assumptions will be required to

successfully instantiate protocols that achieve ByRa SMR in certain real-world settings. Our development and thorough detailing of FAIRSICAL stands as a template for this important future work.

9 Acknowledgements

Thanks to Bruno Mazorra for his helpful feedback and discussions throughout the development of this paper.

References

[1] Ittai Abraham, Srinivas Devadas, Danny Dolev, Kartik Nayak, and Ling Ren. Efficient synchronous byzantine consensus. https://eprint.iacr.org/2017/307, 2017. Retrieved: 07/12/2020. 6

[2] Ittai Abraham, Dahlia Malkhi, Kartik Nayak, Ling Ren, and Alexander Spiegelman. Solida: A blockchain protocol based on reconfigurable byzantine consensus. https://arxiv.org/abs/1811.08572, 2016. Retrieved: 05/12/2020. 4

[3] Amitanand S. Aiyer, Lorenzo Alvisi, Allen Clement, Mike Dahlin, Jean-Philippe Martin, and Carl Porth. Bar fault tolerance for cooperative services. *SIGOPS Oper. Syst. Rev.*, 39(5):45–58, October 2005. 3, 5

[4] Humoud Alsabah and Agostino Capponi. Pitfalls of bitcoin's proof-of-work: R&d arms race and mining centralization. https://ssrn.com/abstract=3273982, 2020. Retrieved: 07/12/2020. 3, 4

[5] Yackolley Amoussou-Guenou, Bruno Biais, Maria Potop-Butucaru, and Sara Tucci-Piergiovanni. Rational behavior in committee-based blockchains. https://eprint.iacr.org/2020/710, 2020. Retrieved: 06/12/2020. 3, 4, 5

[6] Yackolley Amoussou-Guenou, Antonella Del Pozzo, Maria Potop-Butucaru, and Sara Tucci-Piergiovanni. Correctness and fairness of tendermint-core blockchains. https://arxiv.org/pdf/1805.08429, 2018. Retrieved: 06/12/2020. 5

[7] Yackolley Amoussou-Guenou, Antonella Del Pozzo, Maria Potop-Butucaru, and Sara Tucci-Piergiovanni. On fairness in committee-based blockchains. https://arxiv.org/pdf/1910.09786, 2019. Retrieved: 06/12/2020. 4, 5

[8] Nick Arnosti and S. Matthew Weinberg. Bitcoin: A natural oligopoly. https://arxiv.org/abs/1811.08572, 2018. Retrieved: 08/12/2020. 3, 4

[9] Sarah Azouvi and Alexander Hicks. Sok: Tools for game theoretic models of security for cryptocurrencies. https://arxiv.org/abs/1905.08595, 2020. Retrieved: 25/11/2020. 4

[10] Shehar Bano, Alberto Sonnino, Mustafa Al-Bassam, Sarah Azouvi, Patrick McCorry, Sarah Meiklejohn, and George Danezis. Sok: Consensus in the age of blockchains. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, AFT '19, pages 183–198, New York, NY, USA, 2019. Association for Computing Machinery. 4

[11] Bruno Biais, Christophe Bisière, Matthieu Bouvard, and Catherine Casamatta. The blockchain folk theorem. IDEI Working Papers 873, Institut d'Économie Industrielle (IDEI), Toulouse, 2017. Retrieved: 05/12/2020. 4

[12] Georgios Birmpas, Elias Koutsoupias, Philip Lazos, and Francisco J. Marmolejo-Cossío. Fairness and efficiency in dag-based cryptocurrencies. https://arxiv.org/pdf/1910.02059, 2019. Retrieved: 20/11/2020. 4

[13] Eric Budish. The economic limits of bitcoin and the blockchain. Working Paper 24717, National Bureau of Economic Research, June 2018. 4

[14] Vitalik Buterin and Virgil Griffith. Casper the friendly finality gadget. https://arxiv.org/pdf/1710.09437, 2017. Retrieved: 15/11/2020. 3

[15] Vitalik Buterin, Daniel Reijsbergen, Stefanos Leonardos, and Georgios Piliouras. Incentives in ethereum's hybrid casper protocol. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 236–244, Seoul, South Korea, 2019. IEEE. 4, 6

[16] Jing Chen and Silvio Micali. Algorand*. https://arxiv.org/pdf/1607.01341, 2017. Retrieved: 04/12/2020. 3

[17] Phil Daian, Rafael Pass, and Elaine Shi. Snow white: Robustly reconfigurable consensus and applications to provably secure proof of stake. In *Financial Cryptography and Data Security*, pages 23–41, Cham, 2019. Springer International Publishing. 3, 4, 5

[18] Bernardo David, Peter Gaži, Aggelos Kiayias, and Alexander Russell. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018*, pages 66–98, Cham, 2018. Springer International Publishing. 3

[19] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Consensus in the presence of partial synchrony. *J. ACM*, 35(2):288–323, April 1988. 8, 12

[20] Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. *Commun. ACM*, 61(7):95–102, June 2018. 3, 4

[21] Giulia Fanti, Leonid Kogan, Sewoong Oh, Kathleen Ruan, Pramod Viswanath, and Gerui Wang. Compounding of wealth in proof-of-stake cryptocurrencies. In Ian Goldberg and Tyler Moore, editors, *Financial Cryptography and Data Security*, pages 42–61, Cham, 2019. Springer International Publishing. 4, 5

[22] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 357–388, Cham, 2017. Springer International Publishing. 3, 4, 5

[23] Abhiram Kothapalli, Andrew Miller, and Nikita Borisov. Smartcast: An incentive compatible consensus protocol using smart contracts. In Andrew Miller, Michael Brenner, Kurt Rohloff, Joseph Bonneau, Vanessa Teague, Andrea Bracciali, Massimiliano Sala, Federico Pintore, Markus Jakobsson, and Peter Y.A. Ryan, editors, *Financial Cryptography and Data Security - FC 2017 International Workshops, Revised Selected Papers*, pages 536–552, Sliema, Malta, 2017. Springer-Verlag Berlin Heidelberg. 4, 6

[24] Kfir Lev-Ari, Alexander Spiegelman, Idit Keidar, and Dahlia Malkhi. Fairledger: A fair blockchain protocol for financial institutions. https://arxiv.org/pdf/1805.08429, 2019. Retrieved: 07/12/2020. 3, 6

[25] Ziyao Liu, Nguyen Cong Luong, Wenbo Wang, Dusit Niyato, Ping Wang, Ying-Chang Liang, and Dong In Kim. A survey on blockchain: A game theoretical perspective. *IEEE Access*, 7:47615–47643, 2019. 4

[26] Thomas Moscibroda, Stefan Schmid, and Roger Wattenhofer. When selfish meets evil: Byzantine players in a virus inoculation game. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing*, PODC '06, pages 35–44, New York, NY, USA, 2006. Association for Computing Machinery. 3, 4, 5

[27] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin, 2008. Retrieved: 04/12/2020. 3, 4

[28] Kevin Alarcón Negy, Peter R. Rizun, and Emin Gün Sirer. Selfish mining re-examined. In Joseph Bonneau and Nadia Heninger, editors, *Financial Cryptography and Data Security*, pages 61–78, Cham, 2020. Springer International Publishing. 4

[29] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, Cambridge, 2007. 7

[30] Rafael Pass and Elaine Shi. Fruitchains: A fair blockchain. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, PODC '17, pages 315–324, New York, NY, USA, 2017. Association for Computing Machinery. 4, 5

[31] Ioanid Rosu and Fahad Saleh. Evolution of shares in a proof-of-stake cryptocurrency. http://dx.doi.org/10.2139/ssrn.3377136, 2020. Retrieved: 04/12/2020. 4, 5

[32] Fahad Saleh. Blockchain without waste: Proof-of-stake. http://dx.doi.org/10.2139/ssrn.3183935, 2020. Retrieved: 04/12/2020. 4, 5