An Axiomatic Approach to Block Rewards

Xi Chen
Columbia University
xichen@cs.columbia.edu

Christos Papadimitriou
Columbia University
christos@cs.columbia.edu

Tim Roughgarden
Columbia University
tr@cs.columbia.edu

September 25, 2019

Abstract

Proof-of-work blockchains reward each miner for one completed block by an amount that is, in expectation, proportional to the number of hashes the miner contributed to the mining of the block. Is this *proportional allocation rule* optimal? And in what sense? And what other rules are possible? In particular, what are the desirable properties that any "good" allocation rule should satisfy? To answer these questions, we embark on an axiomatic theory of incentives in proof-of-work blockchains at the time scale of a single block. We consider desirable properties of allocation rules including: symmetry; budget balance (weak or strong); sybil-proofness; and various grades of collusion-proofness. We show that Bitcoin's proportional allocation rule is the unique allocation rule satisfying a certain system of properties, but this does not hold for slightly weaker sets of properties, or when the miners are not risk-neutral. We also point out that a rich class of allocation rules can be approximately implemented in a proof-of-work blockchain.

1 Introduction

The Bitcoin protocol was a remarkable feat: eleven years after its sudden appearance [7], and without much adjustment and debugging, it has been used by millions of people and has launched the blockchain industry. Arguably, the most crucial and ingenious aspect of its design lies in the *incentives* the protocol provides to its miners to participate and follow it faithfully. We believe it is of great importance and interest to understand and scrutinize the incentives provided by blockchain protocols—and to do so through the point of view and the methodology of Economic Theory, the science of incentives.

Flaws in the incentives of a blockchain protocol can manifest themselves at multiple timescales. For longest-chain proof-of-work blockchains like Bitcoin, the most well-studied incentive-based attacks, such as selfish mining [4, 10, 5] and transaction sniping [3], concern miners reasoning strategically over multiple block creation epochs. For example, in selfish mining, a miner relinquishes revenue in the short term to achieve greater revenue (in expectation) in the long run via a type of forking attack.

This paper studies incentive issues and potential deviations from intended miner behavior at the most basic time scale, that of a single block creation epoch. We focus on the allocation of *block rewards*, which drives the incentive structure in Bitcoin and many other similar protocols. The dominant paradigm in proof-of-work blockchains is to fix a per-block reward, and for each block to allocate the entire reward to whichever miner first solves a difficult cryptopuzzle. Assuming that miners independently and randomly guess and check possible solutions to the cryptopuzzle,

the expected reward earned by a miner is proportional to their share of the total contributed computational power.

The proportional reward allocation scheme is a simple, natural, and compelling idea, and in all evidence it works quite well. *But is there any sense in which it is "optimal"?* To answer, one has to start by considering the whole spectrum of options; next one must articulate appropriate desiderata; and finally, characterize the full extent of possible solutions that satisfy these desiderata. This is in line with the axiomatic methodology, which has been traditionally employed in Economic Theory for the development of utility theory [13], of impossibility results in social choice [1], as well as of cooperative game theory [12], to name three salient examples. The advantage of the axiomatic approach is that through it one understands not only the domain of possibilities, but also the costs of transgressing the boundaries of this domain. This is our focus in this paper.

1.1 The Proportional Allocation Rule and Its Alternatives

We formalize the question above through the concept of *allocation rules*, functions that map profiles of contributed hashing power to profiles of expected block rewards. The input to such a function is an n-tuple $\mathbf{h} = (h_1, h_2, \ldots, h_n)$ of positive integers, where h_i is the hash rate contributed by miner i in a given block creation epoch, and n is the number of distinct miners (i.e., distinct public keys) that contribute a non-zero hash rate. When the epoch ends and a new block is authorized, one unit of reward becomes available, and the question is, *how should it be allocated to the miners?* The output of an allocation rule specifies an answer to this question, in the form of an n-tuple $\mathbf{p} = (p_1, p_2, \ldots, p_n)$, where p_i is the expected reward to miner i. As mentioned above, the allocation rule corresponding to the Bitcoin protocol is the *proportional rule*, with

$$p_i = \frac{h_i}{\sum_{j=1}^{n} h_j}.$$

Is the proportional rule "optimal"? Or, an even more basic question: What alternatives to the proportional allocation rule are possible? At first blush, it might seem as though a proof-of-work blockchain has no choice but to implement the proportional allocation rule, as presumably the probability distribution over which miner is first to produce a cryptopuzzle solution will be proportional to the contributed computational power. However, as we discuss in Section 2.4, a rich class of allocation rules can in principle be implemented (at least approximately) within a proof-of-work blockchain. The key idea is to wait for a large number of solutions to a medium-difficulty cryptopuzzle, rather than a single solution to a high-difficulty puzzle. Each solution acts as a single sample from the distribution proportional to miner hash rates, and with a large enough number of samples all miners' hash rates can be estimated with high accuracy. This, in turn, permits the (approximate) implementation of a wide range of allocation rules, even though the true miner hash rates are not a priori known to the protocol.

1.2 Properties of Allocation Rules

The definition of an allocation rule is generic enough; the question is, what kinds of properties should such allocation rules satisfy? We consider an array of possible properties, which a blockchain designer may (or may not) require of an allocation rule. The first several properties are motivated by economic viability and fairness rather than incentives per se.

- *Non-negativity.* Expected rewards (the p_i's) should be nonnegative. That is, the protocol cannot require payments from miners.
- *Budget-balance.* The protocol cannot be "in the red," meaning the sum of expected block rewards cannot exceed the unit of block reward available. *Strong* budget-balance insists that the entire unit of block reward is allocated, while *weak* budget-balance allows the protocol to withhold some of the block reward from miners.
- *Symmetry.* The allocation rule should not depend on the names of the miners (i.e., their public keys), only on their contributed hash rates.

Finally, there are two further properties aimed at disincentivizing certain behaviors by the miners that may be considered undesirable by the blockchain designer:

- *Sybil-proofness.* No miner can possibly benefit by creating many accounts and splitting its mining power among them.
- *Collusion-proofness.* Two or more miners cannot benefit by pooling their mining resources and somehow splitting the proceeds. This property has different variants depending on what types of payments between colluding miners are permitted; see Section 2.2.

The proportional allocation rule satisfies all five properties (presumably by design), including the strong version of budget-balance.

1.3 Our Results

We prove a number of characterization results that identify which allocation rules satisfy which sets of desired properties. We begin with risk-neutral miners, who care only about expected rewards (and no other details of the reward distribution). Our first result is:

1. The proportional allocation rule is the unique allocation rule that, with risk-neutral miners, satisfies non-negativity, strong budget-balance, symmetry, sybil-proofness, and a weak form of collusion-proofness (Theorem 1).

That Bitcoin's block reward scheme is a singularly good idea hardly comes as a surprise. Nevertheless, we believe there is value in formally articulating what makes it unique. Further, if one relaxes the requirements slightly, additional allocation rules become possible. For example:

(2) A family of allocation rules that we call the *generalized proportional allocation rules* constitute the only allocation rules that, with risk-neutral miners, satisfy non-negativity, weak budget-balance, symmetry, sybil-proofness, and a slightly stronger form of collusion-proofness (Theorem 2).

But is it reasonable to assume that miners are risk-neutral? The phenomenon of *mining pools* for Bitcoin and other cryptocurrencies (see e.g. [6]) is a behavior which, intuitively, aims to reduce the risk of each miner, and thus suggests that miners in the real world are *risk-averse.*[1] We prove an *impossibility result* that makes this intuition precise:

[1]There may also be other reasons to join a mining pool, for example to avoid the cost of maintaining a full node, but risk-aversion is undoubtedly a first-order factor. (Who is willing to wait an expected twenty years for their first reward?)

(3) If miners are risk averse (equivalently, their utility is the expectation of a strictly concave function of the reward), then there is *no* non-zero allocation rule that is symmetric, (weakly) budget-balanced, sybil-proof, and (weakly) collusion-proof (Theorem 3).

This result suggests that mining pools as a form of collusion are unavoidable. In contrast:

(4) If miners are *risk-seeking*, then Bitcoin's proportional allocation rule satisfies (strong versions) of all of the desired properties (Theorem 5).

(5) A deterministic implementation of the proportional rule—with the block reward split fractionally between miners—satisfies all of the desired properties even with risk-averse miners (Corollary 4).

The deterministic implementation in (5) can be viewed as a simulation of the functionality of a mining pool inside the blockchain protocol itself, analogous to the "Revelation Principle" from mechanism design theory.[2]

2 Model

2.1 Allocation Rules

Let $\mathbb{N}^* = \cup_{n \geq 1} \mathbb{N}^n$ denote the set of all finite tuples of positive integers. For a positive integer n, we write $[n]$ to denote $\{1, 2, \ldots, n\}$. An *allocation rule* is a function $\mathbf{x}$ over $\mathbb{N}^*$ that maps each tuple $\mathbf{h} = (h_1, \ldots, h_n) \in \mathbb{N}^n$ of length n to an n-tuple $\mathbf{p} = (p_1, \ldots, p_n)$ of nonnegative real numbers. Here h_i and p_i are the hash rate and expected reward of miner i.[3] We also write $x_i(\mathbf{h})$ for p_i and $\mathbf{x}(\mathbf{h})$ for $\mathbf{p}$. We refer to a tuple $\mathbf{h} \in \mathbb{N}^*$ as a *configuration* and $\mathbf{x}(\mathbf{h})$ as the corresponding *allocation*.

In this section and the next, we assume that miners care only about their expected rewards (with more being better), and in particular are risk-neutral. Section 4 addresses the case of non-risk-neutral miners and the interesting new issues that they raise.[4]

2.2 Axioms

With the language and notation of allocation rules, we can translate the properties in Section 1.2 into formal axioms.[5] See Section 2.3 for examples of allocation rules that satisfy different subsets of these properties.

[2] FruitChain [9] can be likewise interpreted as a lower-variance version of the proportional allocation rule, with "fruits" playing the role of medium-difficulty puzzle solutions. Bobtail [2] gives still another implementation of the proportional allocation rule using multiple puzzle solutions; their primary motivation was to reduce the variance in time between consecutive blocks, rather than that of miners' rewards per se.

[3] For example, h_i could be in units of hashes per second. For convenience, we assume this is an integer. All of our results for integral hash rates immediately imply the same results for arbitrary positive rational hash rates. (The proofs hold verbatim for hash rates that are integral multiples of $1/m$ for some $m \in \mathbb{N}$, and hence apply to all finite rational tuples of hash rates.)

[4] With risk-neutral miners, there is no need to specify details of the reward distribution beyond the expected reward for each miner. For example, $x_i(\mathbf{h})$ might represent a deterministic reward of $x_i(\mathbf{h})$ to miner i, or that miner i has a $x_i(\mathbf{h})$ probability of winning the entire block reward (and with the remaining probability receives no reward). We'll be more specific about the semantics of an allocation rule in due time, when we consider non-risk-neutral miners in Section 4.

[5] Non-negativity is already baked into our definition of an allocation rule.

A1. **Symmetry**: An allocation rule $\mathbf{x}$ is *symmetric* if $\mathbf{x}(\pi(\mathbf{h})) = \pi(\mathbf{x}(\mathbf{h}))$ for every configuration $\mathbf{h} \in \mathbb{N}^*$ and every permutation π.

That is, the expected reward of a miner does not depend on how the miners are ordered (or their public keys). An example of an asymmetric rule is a *dictator* rule, which always allocates the block reward to the miner with the (say) lexicographically smallest public key. We consider only symmetric rules in this paper.

A2a. **Strong budget-balance**: An allocation rule $\mathbf{x}$ is *strongly budget-balanced* if $\sum_i x_i(\mathbf{h}) = 1$ for every configuration $\mathbf{h}$.

A2b. **Weak budget-balance**: An allocation rule $\mathbf{x}$ is *weakly budget-balanced* if $\sum_i x_i(\mathbf{h}) \leq 1$ for every configuration $\mathbf{h}$.

A3. **Sybil-proofness**: An allocation rule $\mathbf{x}$ is *sybil-proof* if: For every configuration $\mathbf{h} \in \mathbb{N}^*$ and every configuration $\mathbf{h}'$ that can be derived from $\mathbf{h}$ by replacing a miner with hash rate h_i by a set S of miners with total hash rate at most h_i (i.e., with $\sum_{j \in S} h'_j \leq h_i$), the total expected reward to miners of S under $\mathbf{h}'$ is at most that of miner i in the original configuration:

$$\sum_{j \in S} x_j(\mathbf{h}') \leq x_i(\mathbf{h}).$$

We consider several natural definitions of collusion-proofness, depending on the type of reward sharing allowed inside the coalition and on whether a Pareto improvement for a coalition is required to be strict. We consider both arbitrary revenue-sharing agreements and proportional sharing. The latter corresponds to the reward schemes used in many Bitcoin mining pools (see e.g. [6]).

A4a. **Collusion-proofness (under arbitrary reward sharing)**: An allocation rule $\mathbf{x}$ is *collusion-proof (under arbitrary reward sharing)* if: For every configuration $\mathbf{h} \in \mathbb{N}^*$ and every configuration $\mathbf{h}'$ that can be derived from $\mathbf{h}$ by replacing a set T of miners with a new miner i^* (representing the coalition) with hash rate at most the total hash rate of miners in T (i.e., with $h'_{i^*} \leq \sum_{j \in T} h_j$), the total expected reward to miner i^* under $\mathbf{h}'$ is at most the total expected reward of miners of T in the original configuration:

$$x_{i^*}(\mathbf{h}') \leq \sum_{j \in T} x_j(\mathbf{h}).$$

A4b. **Strong collusion-proofness (under proportional sharing)**: An allocation rule $\mathbf{x}$ is *strongly collusion-proof (under proportional sharing)* if: For every configuration $\mathbf{h} \in \mathbb{N}^*$ and every configuration $\mathbf{h}'$ that can be derived from $\mathbf{h}$ by replacing a set T of miners with a new miner i^* with hash rate at most the total hash rate of miners in T, either: (i) no miner of T has strictly higher expected reward in $\mathbf{h}'$ (with proportional sharing) than in $\mathbf{h}$; or (ii) some miner of T has strictly lower expected reward in $\mathbf{h}'$ (with proportional sharing) than in $\mathbf{h}$. That is, if

$$x_{i^*}(\mathbf{h}') \cdot \frac{h_i}{\sum_{j \in T} h_j} > x_i(\mathbf{h})$$

for some miner $i \in T$, then

$$x_{i^*}(\mathbf{h}') \cdot \frac{h_\ell}{\sum_{j \in T} h_j} < x_\ell(\mathbf{h})$$

for some other miner $\ell \in T$.

A4c. **Weak collusion-proofness (under proportional sharing)**: An allocation rule $\mathbf{x}$ is *weakly collusion-proof (under proportional sharing)* if: For every configuration $\mathbf{h} \in \mathbb{N}^*$ and every configuration $\mathbf{h}'$ that can be derived from $\mathbf{h}$ by replacing a set T of miners with a new miner i^* with hash rate at most the total hash rate of miners in T, some miner of T has expected reward under $\mathbf{h}'$ (with proportional sharing) at most that in the original configuration. That is, for some miner $i \in T$,

$$x_{i^*}(\mathbf{h}') \cdot \frac{h_i}{\sum_{j \in T} h_j} \leq x_i(\mathbf{h}).$$

Every violation of (4c) also constitutes a violation of (4b), and similarly for (4b) and (4a). That is, (4a)–(4c) are ordered from strongest (i.e., most difficult to satisfy) to weakest. Impossibility and uniqueness results are most compelling for the weakest variants of budget-balance and collusion-proofness; possibility results are most impressive for the strongest variants. The proportional allocation rule satisfies the strongest versions of budget-balance and collusion-proofness (in addition to symmetry and sybil-proofness).

2.3 Examples

This section gives several examples of allocation rules which satisfy different subsets of the axioms in Section 2.2.

Example 1 (Proportional allocation rule)**.** *The proportional allocation rule is defined for each configuration* $\mathbf{h}$ *of length* n *by*

$$x_i(\mathbf{h}) = \frac{h_i}{\sum_{j \in [n]} h_j}.$$

This allocation rule is symmetric, strongly budget-balanced, sybil-proof, and collusion-proof under arbitrary reward sharing.

Example 2 (All-zero allocation rule)**.** *The all-zero allocation rule is defined for each configuration* $\mathbf{h}$ *of length* n *by*

$$x_i(\mathbf{h}) = 0$$

for $i = 1, 2, \ldots, n$*. The all-zero allocation rule is symmetric, weakly budget-balanced, sybil-proof, and collusion-proof under arbitrary reward sharing.*

Our next example comprises scaled versions of the proportional allocation rule, with the scaling constant dependent on the total hash rate.

Example 3 (Generalized proportional allocation rules)**.** *For every nondecreasing function* $c : \mathbb{N} \to [0, 1]$*, the corresponding generalized proportional allocation rule is defined for each configuration* $\mathbf{h}$ *of length* n *by*

$$x_i(\mathbf{h}) = c\left(\sum_{j \in [n]} h_j\right) \cdot \frac{h_i}{\sum_{j \in [n]} h_j}.$$

Examples 1 and 2 are the generalized proportional allocation rules corresponding to the functions $c(y) = 1$ and $c(y) = 0$, respectively.

Generalized proportional allocation rules are symmetric, weakly budget-balanced, sybil-proof, and collusion-proof under arbitrary reward sharing. To check sybil-proofness, let $\mathbf{h}$ be a configuration and $\mathbf{h}'$ derived from $\mathbf{h}$ by replacing some miner i with hash rate h_i by a set S of miners with total hash rate at most h_i (i.e., with $\sum_{j\in S} h'_j \leq h_i$). Then, the total expected reward under $\mathbf{h}'$ of the miners in S is:

$$\begin{aligned}
&c\left(\sum_{j\in[n]} h_j - h_i + \sum_{j\in S} h'_j\right) \cdot \frac{\sum_{j\in S} h'_j}{\sum_{j\in[n]} h_j - h_i + \sum_{j\in S} h'_j} \\
&\quad = c\left(\sum_{j\in[n]} h_j - h_i + \sum_{j\in S} h'_j\right)\left(1 - \frac{\sum_{j\in[n]} h_j - h_i}{\sum_{j\in[n]} h_j - h_i + \sum_{j\in S} h'_j}\right) \\
&\quad \leq c\left(\sum_{j\in[n]} h_j\right)\left(1 - \frac{\sum_{j\in[n]} h_j - h_i}{\sum_{j\in[n]} h_j}\right) \\
&\quad = c\left(\sum_{j\in[n]} h_j\right) \cdot \frac{h_i}{\sum_{j\in[n]} h_j},
\end{aligned}$$

with the inequality following from the fact that $\sum_{j\in S} h'_j \leq h_i$. Since the final expression is miner i's expected reward in the original configuration, this verifies sybil-proofness. Collusion-proofness can be checked using a similar argument.

Example 4 (Proportional-to-squares allocation rule)**.** *The proportional-to-squares allocation rule is defined for each configuration $\mathbf{h}$ of length n by*

$$x_i(\mathbf{h}) = \frac{h_i^2}{\sum_{j=1}^n h_j^2}.$$

This allocation rule is symmetric, strongly budget-balanced, and sybil-proof. It is not even weakly collusion-proof under proportional sharing.

Example 5 (Proportional-to-square-roots allocation rule)**.** *The proportional-to-square-roots allocation rule is defined for each configuration $\mathbf{h}$ of length n by*

$$x_i(\mathbf{h}) = \frac{\sqrt{h_i}}{\sum_{j=1}^n \sqrt{h_j}}.$$

This allocation rule is symmetric, strongly budget-balanced, and collusion-proof under arbitrary reward sharing. However, it is not sybil-proof.

2.4 Implementing Non-Proportional Rules

The proportional allocation rule is realizable, in that there is a proof-of-work blockchain protocol (namely, Bitcoin) that implements it. Are non-proportional allocation rules purely hypothetical?

This section demonstrates that, at least in principle, *every* weakly budget-balanced and continuous allocation rule can be approximately implemented within a proof-of-work blockchain.

Fix a power-of-2 M, a cryptographic hash function f, and a difficulty level b. A *full solution* is a preimage z such that $f(z)$ has at least b trailing zeroes, and a *partial solution* is a z such that $f(z)$ has at least $b - \log_2 M$ trailing zeros. The parameter b is chosen so that random guessing by miners produces a full solution in a prescribed amount of time (on average), such as 10 minutes. One expects partial solutions to be discovered at M times the rate of full solutions.

Fix a weakly budget-balanced allocation rule $\mathbf{x}$. A high-level description of one possible corresponding protocol is then:

1. Miners attempt to find partial and full solutions of the form $\langle pkey|\sigma|nonce\rangle$, where *pkey* is the miner's public key, σ is derived from the current blockchain state (e.g., the hash of the block of transactions at the end of the longest chain), and *nonce* is a number of free bits specified by the miner.

2. A miner who discovers a partial (non-full) solution can add it to the blockchain.

3. A miner who discovers a full solution can authorize a new block of transactions and add it to the blockchain (along with their full solution).

4. When a new full solution and corresponding block are published, block rewards are distributed according to the number of partial solutions contributed by each miner since the preceding full solution. Precisely, let $[n]$ denote the miners contributing at least one partial solution, g_i the number contributed by miner i, and $M' = \sum_{i \in [n]} g_i$ the total number of partial solutions reported in this epoch. Define an estimate $\hat{h}_i$ of miner i's hash rate by

$$\hat{h}_i = g_i \cdot \rho \cdot \frac{M}{M'},$$

   where ρ denotes the hash rate that would produce (on average) one partial solution in the prescribed amount of time. Define miner rewards by $\mathbf{x}(\hat{h}_1, \hat{h}_2, \ldots, \hat{h}_n)$.

Several comments are in order. First, this protocol effectively simulates the typical functionality of a mining pool *inside the protocol itself*. This is reminiscent of the *Revelation Principle* from mechanism design (see e.g. [8]), which is a simulation argument that shows how to eliminate non-truthful reporting of preferences by moving the deviations inside the mechanism itself.

Second, if we take $M = 1$, then partial and full solutions coincide, $M = M' = 1$, and the protocol essentially recovers Bitcoin's implementation of the proportional allocation rule.

Third, the larger we take M, the more accurate the estimated hash rates $\hat{\mathbf{h}}$. Thus any continuous allocation rule can be approximated as closely as desired by taking M sufficiently large.

Fourth, we ignore incentive issues involving miners' delaying the publication of full or partial solutions; Schrivjers et al. [11] discuss (in the context of mining pools) methods for mitigating such incentive issues.

Finally, we have deliberately avoided committing to the details of the implementation, such as the best way to record the partial solutions on-chain. Our point is simply that there appears to be no fundamental barrier to implementing a wide range of non-proportional allocation rules in a proof-of-work blockchain.

3 Characterizations with Risk-Neutral Miners

3.1 A Uniqueness Result for the Proportional Allocation Rule

We assume throughout this section that miners are risk-neutral and are concerned only with their expected rewards. Our first main result is a uniqueness result for the proportional allocation rule: It is the only rule that is symmetric, strongly budget-balanced, sybil-proof, and collusion-proof (even weakly collusion-proof with proportional sharing).

Theorem 1 (Characterization of Rules with A1, A2a, A3, A4c)**.** *The proportional allocation rule is the unique allocation rule that is symmetric, strongly budget-balanced, sybil-proof, and weakly collusion-proof (with proportional sharing).*

Proof. Let $\mathbf{x}$ be an allocation rule that is symmetric, strongly budget-balanced, sybil-proof, and weakly collusion-proof (with proportional sharing). We prove by induction that for every configuration $\mathbf{h} \in \mathbb{N}^*$, $\mathbf{x}$ satisfies $x_i(\mathbf{h}) = h_i / \sum_j h_j$. The induction is on the number t of entries in $\mathbf{h}$ that are larger than 1. The base case of $t = 0$ is trivial since $\mathbf{h}$ is then an all-1 tuple (of length n, say) and symmetry and strong budget-balance imply that $x_i(\mathbf{h}) = 1/n$ for every i.

For the inductive step, assume that the statement holds for all tuples with less than t entries larger than 1. Now consider a configuration $\mathbf{h} \in \mathbb{N}^n$ for some $n \geq t$ that sums to m and has t entries larger than 1. Because $\mathbf{x}$ is symmetric, we can assume without loss of generality that $h_1, h_2, \ldots, h_t > 1$ while $h_{t+1}, h_{t+2}, \ldots, h_n = 1$. Let $\mathbf{p} = (p_1, \ldots, p_n) = \mathbf{x}(\mathbf{h})$. We claim that $p_i = h_i/m$ for each $i \in [t]$. This claim, together with the assumptions that $\mathbf{x}$ is symmetric and strongly budget-balanced, implies that $x_i(\mathbf{h}) = 1/m$ for all $i > t$ and hence $\mathbf{x}$ indeed agrees with the proportional rule on $\mathbf{h}$.

Suppose $p_i > h_i/m$ for some $i \in [t]$. Then, consider the configuration $\mathbf{h}'$ obtained by splitting the miner i into h_i sybils, each with hash rate 1. By the inductive hypothesis and the fact that $\mathbf{h}'$ has $t - 1$ entries that are larger than 1, $\mathbf{x}$ is proportional on $\mathbf{h}'$ and so each of these miners with hash rate 1 receives $1/m$ which is strictly less than p_i/h_i. This means that, in $\mathbf{h}'$, these h_i players with hash rate 1 would all be better off by colluding and sharing their results proportionally (i.e., uniformly). This contradicts the assumption that $\mathbf{x}$ is weakly collusion-proof (with proportional sharing). We conclude that $p_i \leq h_i/m$ for all $i \in [t]$.

On the other hand, suppose $p_i < h_i/m$ for some $i \in [t]$, and consider the configuration $\mathbf{h}'$ obtained by splitting miner i into h_i sybils with hash rate 1 each. By the inductive hypothesis, each sybil receives expected reward $1/m$ under $\mathbf{x}$ in $\mathbf{h}'$. The total expected reward earned by the sybils therefore exceeds that of miner i in $\mathbf{h}$. This contradicts the assumption that $\mathbf{x}$ is sybil-proof, so we can conclude that $p_i \geq h_i/m$ (and hence $p_i = h_i/m$) for all $i \in [t]$. This completes the proof of the claim, the inductive step, and the theorem. □

3.2 Weak Budget-Balance and Generalized Proportional Rules

Example 3 shows that relaxing the strong budget-balance requirement to weak budget-balance enlarges the design space. One might suspect that, analogous to Theorem 1, generalized proportional allocation rules are the only ones that satisfy symmetry, weak budget-balance, sybil-proofness, and weak collusion-proofness with proportional sharing. The next example shows that this is not the case.

Example 6. *Consider the following allocation rule* $\mathbf{x}$*. For a configuration* $\mathbf{h}$ *in which no miner has more than half the overall hash rate (i.e.,* $h_i \leq \frac{1}{2}\sum_{j\in[n]} h_j$ *for every* i*),* $\mathbf{x}$ *agrees with the proportional allocation rule. For a configuration in which one miner* i *has more than half of the overall hash rate, miner* i *receives its fair share under the proportional rule (i.e.,* $x_i(\mathbf{h}) = h_i/\sum_{j\in[n]} h_j$*), while other miners receive 0 (*$x_j(\mathbf{h}) = 0$ *for* $j \neq i$*).*

The rule $\mathbf{x}$ *is symmetric, weakly budget-balanced, sybil-proof, and weakly collusion-proof under proportional sharing. It is not strongly collusion-proof under proportional sharing, however.*

Our second main result shows that if weak collusion-proofness is strengthened to strong collusion-proofness (with proportional sharing), then generalized proportional allocation rules are indeed the only ones that satisfy the axioms A1, A2b, A3, and A4b.

Theorem 2 (Characterization of Rules with A1, A2b, A3, A4b)**.** *Generalized proportional allocation rules are the only allocation rules that are symmetric, weakly budget-balanced, sybil-proof, and strongly collusion-proof under proportional sharing.*

Proof. Let $\mathbf{x}$ be an allocation rule that is symmetric, weakly budget-balanced, sybil-proof, and strongly collusion-proof under proportional sharing. We claim that, for each positive integer m, there is a nonnegative real number $c(m) \leq 1$ such that

$$x_i(\mathbf{h}) = c(m) \cdot \frac{h_i}{m}$$

for every configuration $\mathbf{h}$ with total hash rate $(\sum_j h_j)$ equal to m.

To prove the claim, fix a positive integer m. First, let $1^m \in \mathbb{N}^m$ denote the all-1 tuple of length m and define

$$c(m) = \sum_{i\in[m]} x_i(1^m).$$

Since $\mathbf{h}$ is symmetric and weakly budget-balanced, $c(m) \leq 1$ and $x_i(1^m) = c(m)/m$ for every $i \in [m]$. Second, let $\mathbf{h}^*$ denote the configuration with a single miner with hash rate m. Since $\mathbf{x}$ is sybil-proof and strongly collusion-proof (under proportional sharing), the expected reward assigned by $\mathbf{x}$ to the sole miner in $\mathbf{h}^*$ is $c(m)$.

We follow the approach in the proof of Theorem 1 and prove by induction that, for every configuration $\mathbf{h} \in \mathbb{N}^*$ with $\sum_i h_i = m$,

$$x_i(\mathbf{h}) = c(m) \cdot \frac{h_i}{m}$$

for every miner i. The induction is again on the number t of entries in $\mathbf{h}$ that are larger than 1. The base case of $t = 0$ is trivial by the choice of $c(m)$.

For the inductive step, we consider a configuration $\mathbf{h} \in \mathbb{N}^n$ for some $n \geq t$ with $\sum_{i\in[n]} h_i = m$ and exactly t entries larger than 1. Without loss of generality (since $\mathbf{x}$ is symmetric), assume that $h_1, \ldots, h_t > 1$ and $h_{t+1}, \ldots, h_n = 1$. Let $\mathbf{p} = (p_1, \ldots, p_n) = \mathbf{x}(\mathbf{h})$. With respect to the first t miners, the same argument as in the proof of Theorem 1 shows that $p_i = c(m) \cdot (h_i/m)$ for every $i \in [t]$. (This part of the argument requires only weak collusion-proofness under proportional rewards.) Example 6 shows that the rest of the proof (for miners $t+1, t+2, \ldots, n$) must differ from that of Theorem 1 and make use of strong collusion-proofness. The issue is that because $\mathbf{x}$ is only assumed to be weakly budget-balanced, it does not follow directly from $p_i = c(m) \cdot (h_i/m)$ for all $i \in [t]$ that $p_i = c(m)/m$ for all $i > t$.

So assume for contradiction that $p_i \neq c(m)/m$ for some $i > t$ (and hence, by symmetry, for all such i). If $p_i > c(m)/m$, then $\mathbf{x}$ fails sybil-proofness: the sole miner in configuration $\mathbf{h}^*$ can increase its expected reward by splitting into n sybils with hash rates as in $\mathbf{h}$. On the other hand, if $p_i < c(m)/m$, then $\mathbf{x}$ fails strong collusion-proofness under proportional sharing: if the n miners in $\mathbf{h}$ form the grand coalition, then with proportional sharing, every miner $i \in [t]$ receives the same expected reward $p_i = c(m) \cdot (h_i/m)$ as in $\mathbf{h}$ while the expected reward of every miner $i > t$ strictly increases (from p_i to $c(m)/m$). We conclude that $p_i = c(m)/m$ for every $i > t$ and hence $p_i = c(m) \cdot (h_i/m)$ for every miner i. This concludes the proof of the claim.

All that remains is to show that $c(m)$ must be a nondecreasing function of m. This follows from sybil-proofness: if $c(m+1) < c(m)$ for some m, then in the single-miner configuration with hash rate $m+1$, the miner would have an incentive to replace itself with a miner with hash rate m. □

4 Beyond Risk Neutrality: Possibility and Impossibility Results

We have been assuming so far that miners are *risk-neutral* and care only about the *expectation* of their reward, as opposed to other distributional characteristics (like variance). Going beyond this assumption reveals the interesting ways in which risk affects incentives in blockchain protocols.

4.1 Von Neumann-Morgenstern Utilities

The earliest, and most principled, treatment of risk in economics is through von Neumann and Morgenstern's *utility theory,* articulated more than seven decades ago [13]. One starts from each agent having a very general set of arbitrary preferences between *lotteries* (finite-support probabilistic distributions over different amounts of money), where the preferences of agents are assumed to satisfy four very natural and plausible axioms: completeness, transitivity, continuity, and independence. The remarkable result proved is that any such system of preferences of an agent is tantamount to the agent possessing *a utility function U,* a function mapping amounts of money to the reals, such that the agent prefers lottery L to lottery M if and only if $\mathbf{E}_L[U(p)] \geq \mathbf{E}_M[U(p)]$, where the expectation is over the random variable p (which is in units of money). That is, any agent with preferences satisfying the properties above can be modeled as an *expected utility-maximizer*.

For an agent with utility function U, we can interpret $U(p)$ as the amount of *utility* the agent receives from a reward of p. For ease of presentation, we assume throughout that $U(0) = 0$ and that U is twice-differentiable and strictly increasing. The risk sensitivity of an agent can then be gauged by the second derivative of U. Risk-neutrality corresponds to a linear utility function, with 0 second derivative; in this case, $\mathbf{E}[U(p)] = U(\mathbf{E}[p])$ for every distribution over rewards p. A *risk-averse* agent is an expected utility maximizer with a strictly concave utility function—a function with an everywhere negative second derivative. For a risk-averse agent, $\mathbf{E}[U(p)] \leq U(\mathbf{E}[p])$ for every distribution over p (by Jensen's inequality). For example, a risk-averse agent may not risk flipping a fair coin if the two outcomes are either losing 30% of their fortune, or doubling it. (Whereas a risk neutral — or *risk-seeking*, see Section 4.5 — agent would be happy to flip the coin.)

4.2 Collusion-Proofness Revisited

In this section and the next, we consider (randomized) allocation rules that allocate the entire reward to (at most) one miner. We interpret an output $x_i(\mathbf{h})$ of an allocation rule as the probability that miner i receives the entire reward in the configuration $\mathbf{h}$.

Of the four types of axioms in Section 2.2, symmetry and budget-balance are obviously independent of any model of miner utility. The sybil-proofness axiom requires only cosmetic changes:

A3. **Sybil-proofness (with general utility functions)**: An allocation rule $\mathbf{x}$ is *sybil-proof* if: For every allowable miner utility function U, every configuration $\mathbf{h} \in \mathbb{N}^*$, and every configuration $\mathbf{h}'$ that can be derived from $\mathbf{h}$ by replacing a miner with hash rate h_i by a set S of miners with total hash rate at most h_i (i.e., with $\sum_{j \in S} h'_j \leq h_i$), the total expected utility to i in $\mathbf{h}'$ is at most that in the original configuration:

$$\sum_{j \in S} x_j(\mathbf{h}') \cdot U(1) \leq x_i(\mathbf{h}) \cdot U(1).$$

Assuming that every allowable utility function U is strictly increasing (and thus, $U(1) > U(0) = 0$), this axiom is equivalent to the definition of sybil-proofness for risk-neutral miners given in Section 2.2.

We need to adjust the definition of collusion-proofness, since now the incentives of miners are affected by their utility functions. Fix a class $\mathcal{U}$ of allowable utility functions — for example, all linear functions, or all strictly concave functions. We next define analogs of axioms (4a) and (4c) from Section 2.2 (we skip (4b) because we do not need it in our statements). A *reward-sharing scheme* with miner set S specifies how the miners of S would split a block reward internally; formally, it is a collection of nonnegative random variables $\{\mathbf{r}_i\}_{i \in S}$ that satisfy $\sum_{i \in S} \mathbf{r}_i \leq 1$ with probability 1.

A4a. **Collusion-proofness against $\mathcal{U}$ (under arbitrary reward sharing)**: An allocation rule $\mathbf{x}$ is *collusion-proof against $\mathcal{U}$ (under arbitrary reward sharing)* if: For every configuration $\mathbf{h} \in \mathbb{N}^*$, every choice of utility function $U_i \in \mathcal{U}$ for each participating miner i, every configuration $\mathbf{h}'$ that can be derived from $\mathbf{h}$ by replacing a set T of miners with a new miner i^* with hash rate at most the total hash rate of miners in T, and every reward sharing scheme $\{\mathbf{r}_i\}_{i \in T}$, either: (i) no miner of T has higher expected utility in $\mathbf{h}'$ (with the given reward sharing scheme) than in $\mathbf{h}$; or (ii) some miner of T has strictly lower expected utility in $\mathbf{h}'$ (with the given reward sharing scheme) than in $\mathbf{h}$. That is, if

$$x_{i^*}(\mathbf{h}') \cdot \mathbf{E}[U_i(\mathbf{r}_i)] > U_i(1) \cdot x_i(\mathbf{h})$$

for some miner $i \in T$, then

$$x_{i^*}(\mathbf{h}') \cdot \mathbf{E}[U_\ell(\mathbf{r}_\ell)] < U_\ell(1) \cdot x_\ell(\mathbf{h})$$

for some other miner $\ell \in T$.

A4c. **Weak collusion-proofness against $\mathcal{U}$ (under proportional sharing)**: An allocation rule $\mathbf{x}$ is *weakly collusion-proof against $\mathcal{U}$ (under proportional sharing)* if: For every configuration $\mathbf{h} \in \mathbb{N}^*$, every choice of utility function $U_i \in \mathcal{U}$ for each participating miner i, every configuration $\mathbf{h}'$ that can be derived from $\mathbf{h}$ by replacing a set T of miners with a new miner i^* with hash rate at most the total hash rate of miners in T, some miner of T has expected utility under $\mathbf{h}'$ (with proportional sharing) at most that in the original configuration. That is, for some miner $i \in T$,

$$x_{i^*}(\mathbf{h}') \cdot U_i\left(\tfrac{h_i}{\sum_{j \in T} h_j}\right) \leq U_i(1) \cdot x_i(\mathbf{h}).$$

4.3 Risk Aversion and Impossibility

With risk-neutral miners, the proportional allocation rule implemented by the Bitcoin protocol satisfies all of the axioms in Section 2.2, including collusion-proofness (even with arbitrary reward sharing). In reality, however, Bitcoin is *not* collusion-proof, in that most miners join a mining pool that effectively acts like a large single miner.[6] There is anecdotal evidence that miners generally join mining pools to lower the variance of the rewards received, analogous to an insurance policy, and we view the ubiquity of mining pools as strong evidence that most miners are risk-averse. How does risk aversion affect the set of possible allocation rules satisfying our standard axioms?

The following *impossibility result* shows that the incentive to form mining pools is not an artifact of the specific allocation rule implemented in Bitcoin; rather, it is a fundamental difficulty with risk-averse miners.

Theorem 3 (Impossibility with Risk-Averse Miners)**.** *Assume that all miners are expected utility-maximizers with the same strictly concave utility function U satisfying $U(1) > U(0) = 0$. There is no non-zero allocation rule that is symmetric, weakly budget-balanced, sybil-proof, and weakly collusion-proof against $\mathcal{U} = \{U\}$ (with proportional sharing).*

Note that this impossibility result holds even with our weakest notions of budget-balance (A2b) and collusion-proofness (A4c). The assumption that all miners share the same utility function U only makes the impossibility result more compelling.

Proof of Theorem 3. Let $\mathbf{x}$ be a non-zero and symmetric allocation rule. Let $\mathbf{h}$ be a configuration such that $x_i(\mathbf{h}) > 0$ for some i, and assume for simplicity that h_i is even. Obtain $\mathbf{h}'$ from $\mathbf{h}$ by replacing miner i with two miners i_1 and i_2 with hash rate $h_i/2$ each. By symmetry, $x_{i_1}(\mathbf{h}') = x_{i_2}(\mathbf{h}')$; let q denote this common probability. If $\mathbf{x}$ is sybil-proof, then $2q \leq x_i(\mathbf{h})$; suppose this is indeed the case. Starting now from $\mathbf{h}'$, if the two miners i_1 and i_2 join forces and combine hash rates to produce the configuration $\mathbf{h}$ (sharing rewards proportionally), then their expected utilities change from

$$U(1) \cdot q$$

to

$$U\left(\tfrac{1}{2}\right) \cdot x_i(\mathbf{h}) \geq 2 \cdot U\left(\tfrac{1}{2}\right) \cdot q.$$

Our assumption on U implies that $2U(1/2) > U(1)$ and hence both miners are strictly better off in the coalition. We conclude that $\mathbf{x}$ is not weakly collusion-proof under proportional sharing. □

4.4 Possibility with Deterministic Rewards

The impossibility result in Theorem 3 holds for randomized allocation rules $\mathbf{x}$ that always allocate the entire block reward to a single miner according to the probability distribution specified by $\mathbf{x}(\mathbf{h})$. At the other extreme are *deterministic* allocation rules, for which each $x_i(\mathbf{h})$ represents a (fractional) block reward deterministically given to miner i. With risk-neutral miners, there is no difference between the two types of rules, and both the randomized and deterministic implementations of the proportional allocation rule satisfy all of our axioms.

[6] See `https://www.blockchain.com/en/pools` for the biggest Bitcoin mining pools. As of this writing, the largest pool (BTC.com) controls roughly 20% of the total hash rate.

With a deterministic allocation rule, the miner utility functions no longer matter—more (deterministic) reward is always better than less. (Remember that utility functions are strictly increasing.) In this case, miners effectively act as if they were risk-neutral, and so all results for risk-neutral miners carry over to deterministic allocation rules and miners with arbitrary utility functions. For example, the following corollary of Theorem 1 is immediate.

Corollary 4 (Possibility with Deterministic Rewards)**.** *For every family $\mathcal{U}$ of utility functions, the deterministic implementation of the proportional allocation rule is the unique deterministic allocation rule that is symmetric, strongly budget-balanced, sybil-proof, and weakly collusion-proof against $\mathcal{U}$ (with proportional sharing).*

Bitcoin's implementation of the proportional allocation rule is inherently randomized. But a deterministic version of the rule can be approximated as closely as desired using a protocol of the type described in Section 2.4 (with a sufficiently large value of the parameter M).

In the proposed implementation in Section 2.4, the choice of the miner who can authorize a new block of transactions and add it to the blockchain—the first miner to find a full solution—is still effectively chosen randomly (see step (3)). This highlights the fact that there are really two distinct problems to solve in each block creation epoch:

1. Electing a leader to authorize the next block of transactions.

2. Distributing block rewards.

In Bitcoin, the solutions to these problems are tightly coupled, in that the choices of the leader and of the recipient of the block reward are always the same. The solutions are decoupled in the deterministic implementation of the proportional allocation, with the leader elected randomly as in Bitcoin but with rewards distributed deterministically.

4.5 The Case of Risk-Seeking Miners

For completeness, this section studies *risk-seeking* miners, meaning expected utility maximizers with strictly convex utility functions. In a surprising contrast with the impossibility result for risk-averse miners (Theorem 3), here the (randomized implementation of the) proportional allocation rule satisfies all of the same axioms as in the case of risk-neutral miners.

Theorem 5 (The Proportional Rule with Risk-Seeking Miners)**.** *Let $\mathcal{U}$ be the set of all convex functions $U : [0,1] \to \mathbb{R}_{\geq 0}$ with $U(1) > U(0) = 0$. The proportional allocation rule is symmetric, strongly budget-balanced, sybil-proof, and collusion-proof against $\mathcal{U}$ under arbitrary reward sharing.*

Proof. The proportional allocation rule is obviously symmetric and strongly budget-balanced. It is sybil-proof for risk-neutral miners and hence also non-risk-neutral miners (see the discussion in Section 4.2).

To complete the proof, assume for the purposes of contradiction that the proportional allocation rule is not collusion-proof under arbitrary transfers against the set $\mathcal{U}$ of convex utility functions U that satisfy $U(1) > U(0) = 0$. Then, there is a configuration $\mathbf{h} \in \mathbb{N}^n$, utility functions $U_1, U_2, \ldots, U_n \in \mathcal{U}$ for the participating miners, a subset $T \subseteq [n]$ of the miners, and a reward-sharing scheme $\{\mathbf{r}_i\}_{i \in T}$ such that

$$\frac{\sum_{j \in T} h_j}{\sum_{j \in [n]} h_j} \cdot \mathbf{E}[U_i(\mathbf{r}_i)] \geq \frac{h_i}{\sum_{j \in [n]} h_j} \cdot U_i(1) \quad \text{for every } i \in T,$$

with the inequality strict for at least one miner $i \in T$.

Summing these inequalities over $i \in T$ and using that $U_i(1) > 0$ for every miner i, we then have

$$\frac{\sum_{j\in T} h_j}{\sum_{j\in[n]} h_j} \cdot \sum_{i\in T} \mathbf{E}\left[\frac{U_i(\mathbf{r}_i)}{U_i(1)}\right] > \sum_{i\in T} \frac{h_i}{\sum_{j\in[n]} h_j}$$

and, thus,

$$\mathbf{E}\left[\sum_{i\in T} \frac{U_i(\mathbf{r}_i)}{U_i(1)}\right] > 1.$$

However, since each U_i is convex with $U_i(1) > U_i(0) = 0$, we have $U_i(\mathbf{r}_i) \leq \mathbf{r}_i \cdot U_i(1)$ (with probability 1) and thus, since $\sum_{i\in T} \mathbf{r}_i \leq 1$ (again with probability 1),

$$\mathbf{E}\left[\sum_{i\in T} \frac{U_i(\mathbf{r}_i)}{U_i(1)}\right] \leq \mathbf{E}\left[\sum_{i\in T} \mathbf{r}_i\right] \leq 1.$$

This completes the contradiction and the proof. □

5 Acknowledgments

All three authors were supported in part by the Columbia-IBM Center for Blockchain and Data Transparency. The first author was also supported in part by NSF IIS-1838154 and NSF CCF-1703925. The second author was supported in part by NSF CCF-1763970. The third author was supported in part by NSF Award CCF-1813188 and ARO grant W911NF1910294.

References

[1] Arrow, K. J. A difficulty in the concept of social welfare. *Journal of Political Economy 58*, 4 (1950), 328–346.

[2] Bissias, G., and Levine, B. Bobtail: A proof-of-work target that minimizes blockchain mining variance. arXiv:1709.08750, 2017.

[3] Carlsten, M., Kalodner, H. A., Weinberg, S. M., and Narayanan, A. On the instability of Bitcoin without the block reward. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security* (2016), pp. 154–167.

[4] Eyal, I., and Gün Sirer, E. Majority is not enough: Bitcoin mining is vulnerable. In *Proceedings of the 18th International Conference on Financial Cryptography and Data Security* (2014), pp. 436–454.

[5] Kiayias, A., Koutsoupias, E., Kyropoulou, M., and Tselekounis, Y. Blockchain mining games. In *Proceedings of the ACM Conference on Economics and Computation* (2016), pp. 365–382.

[6] Lewenberg, Y., Bachrach, Y., Sompolinski, Y., Zohar, A., and Rosenschein, J. S. Bitcoin pools: a cooperative game theoretic analysis. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems* (2015), pp. 919–927.

[7] Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. Unpublished white paper, 2008.

[8] Nisan, N., Roughgarden, T., Tardos, E., and Vazirani, V. V. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.

[9] Pass, R., and Shi, E. Fruitchains: A fair blockchain. In *Proceedings of the ACM Symposium on Principles of Distributed Computing* (2017), pp. 315–324.

[10] Sapirshtein, A., Sompolinsky, Y., and A, Z. Optimal selfish mining strategies in bitcoin. In *Proceedings of the 20th International Conference on Financial Cryptography and Data Security* (2016), pp. 515–532.

[11] Schrijvers, O., Bonneau, J., Boneh, D., and Roughgarden, T. Incentive compatibility of Bitcoin mining pool reward functions. In *Proceedings of the 20th International Conference on Financial Cryptography and Data Security* (2016), pp. 477–498.

[12] Shapley, L. S. *Additive and Non-Additive Set Functions*. PhD thesis, Department of Mathematics, Princeton University, 1953.

[13] von Neumann, J., and Morgenstern, O. *Theory of Games and Economic Behavior*. Princeton University Press, 1947. Second edition.