Stabilizing Congestion in Decentralized Record-Keepers*

Assimakis Kattis† Fabian Trottner‡

October 2022

Abstract

We provide an economic analysis of a market for record-keeping instantiated using succinct proof systems. Our model allows for adaptive adjustments in predicate (block) size according to pre-specified update rules. We show that when block rewards are absent, there exists a predicate size update rule, based on observed changes in difficulty, that induces stable equilibrium levels of transaction fees and congestion at varying levels of user demand. Our theoretical results highlight the importance of economic incentives for the design of cost-efficient, scalable systems around a Non-Interactive Zero-Knowledge proof as the consensus puzzle.

*We would like to thank Joseph Bonneau, Tarun Chitra, and Georgios Konstantopoulos for helpful feedback and discussions.

†akis@kattis.io

‡Department of Economics, University of California, San Diego, ftrottner@ucsd.edu

1 Introduction

Blockchain technology Nakamoto (2008) offers a decentralized alternative to traditional record-keeping systems. In this work, we study the economic properties of record-keepers instantiated using succinct proof systems. Miners are responsible for producing proofs that a set of records (or transactions) provided to them by end-users is valid, and to add these to an append-only ledger while maintaining consensus. Such an approach generalizes the Bitcoin model, in that end-users can request a record of any truth claim (such as smart contract verification). Since the underlying proof-of-work (PoW) consensus mechanism ensuring correctness poses substantive costs (growing linearly with system size) Abadi & Brunnermeier (2018); Budish (2018); Thum (2018), it is desirable for it to output useful computation.

We work within a specific consensus framework, in which the proof of block validity and the consensus algorithm are combined into one process, while retaining the Nakamoto consensus Nakamoto (2008) security properties. In this model, a (suitably designed) Non-Interactive Zero-Knowledge (NIZK) proof is used as the PoW puzzle Kattis & Bonneau (2020) (instead of an arbitrary hash function). If the NIZK is performing state verification, this connection provides a link between miners' computational work in generating a proof (and thus a block) and the number of transactions that can be verified in a given block (also referred to as *predicate size*). Our analysis shows how to exploit this link so as to design decentralized record-keepers that maintain cost-efficiency at varying levels of demand. Our contributions are most closely related to previous work that extracts usefulness out of the PoW procedure Ball *et al.* (2017); King (2013), and recent initiatives that permit adaptive block sizes in cryptocurrency implementations (such as Ethereum Buterin (2014)).

1.1 Operating at Maximal Block Size S_{max}

A fundamental question concerning the viability of distributed payments is throughput. If the maximal amount of transactions that we can add is S_{max}, it is natural to consider a protocol which produces a proof for S_{max} transactions in each block. This system would always be operating at maximal throughput, generating a proof for S_{max} transactions regardless of how full the block is. If S_{max} is large enough to act as a suitable ceiling[1], then the PoNW computation required would always be the same (subject to the difficulty parameter) and the costs of generating blocks are going to be

[1]Recent developments in NIZK-based applications such as zk-rollups Buterin (2018 (accessed June 12, 2020) give S_{max} values that correspond to $\sim$ 500+ transactions per second, which is substantial enough for commercial scaling purposes.

maximal for all miners. Such a system is inefficient from an economic perspective, as miners will not receive any cost reduction in proof generation times from including fewer transactions in the block. Indeed, the properties of PoW mean that this puzzle would take the same amount of time to solve regardless of its inputs (which in this case is the transaction set, which can be full or empty). This implies a lower bound for the number of transactions required for miners to continue operating at a profit.

Further, a system always operating at maximal throughput is also 'brittle' to large deviations in demand. Previous work Carlsten *et al.* (2016) has shown that setting block rewards to zero creates an inherently unstable system if demand is sufficiently low so that all transactions can be included in a block. By relying solely on transaction fee revenue, miners are susceptible to large losses in periods when there is little demand. Although a drop in difficulty could rectify miners' losses, this creates a security vulnerability for the system, making it susceptible to attack. Thus, adaptable throughput capacity can help ensure that prices compensate miners sufficiently during periods of low demand. Indeed, by shrinking the predicate size S with falling transaction fees, the system would bolster mining revenues by restricting the supply of proofs. This puts upwards pressure on difficulty and mitigates the system's susceptibility to the instability result of Carlsten *et al.* (2016).

1.1.1 From Adaptability to Stability

A crucial design choice in distributed payment systems is that of setting transaction fees so that they are stable over changes in demand ('demand shocks'). How Bitcoin sets fees with a limited amount of block space was analyzed in Huberman *et al.* (2019), in which closed form solutions for how transaction fees vary with system congestion are derived.[2] If we parametrize proof size by the number of transactions S that a given proof verifies (i.e. the number of transactions processed in the given block), then transaction fees, system congestion, and the computational costs to miners depend on the size of predicates. Intuitively, this dependence enables suitable update rules for predicate sizes to adapt throughput to changes in demand in a cost-efficient manner. We will provide a formal framework to verify this intuition, alongside concrete suitable update rules.

[2] Basu *et al.* (2019) propose an equilibrium model with exogenously given transaction fees and block size assumed to be restricted to one transaction. Our work focuses on understanding how transaction fees evolve endogenously as block size changes.

1.2 Our Contributions

We provide an economic model of a market for record-keeping instantiated using succinct proof systems, whose predicate size adjusts according to pre-specified update rules. Predicate size corresponds to block size (thus characterizing throughput), and miners generate the corresponding NIZK as a PoW puzzle. We show that the proposed protocol implies a stable equilibrium over time in the difficulty and predicate size parameters, where a shock to demand leads to proportional increases in throughput without any long-term changes in transaction fees and system congestion. This is due to two complementary features - PoNW consensus and an adaptive block size update rule - which together ensure that adjustments in predicate size and thus block capacity incentivize miners to incur higher/lower computational cost in exchange for higher/lower transaction fee revenue. This adaptability allows the system to achieve stability.

More specifically, we provide the following contributions:

1. An empirically relevant equilibrium framework permitting an analysis of the determinants of transaction fees and (entry) behavior of mining pools in a decentralized market for record-keeping featuring adjustments of system throughput according to arbitrary predicate size update rules.

2. A formal analysis of the dynamic equilibrium behavior induced by various update rules. Specifically, we derive an update rule that induces constant equilibrium transaction fees under arbitrary levels of demand (i.e., equilibrium elasticity of fees with respect to demand is zero due to offsetting throughput increases).

3. An incentive-compatible adaptive PoW system that also performs state verification in succinct blockchain architectures. This protocol admits stable levels of transaction fees that are robust to bounded demand shocks, where the bound can be made arbitrarily large.

We model miners as profit-maximizing firms that compete for the right to verify an incoming block of transactions by engaging costly . as to characterize the competitive equilibrium, and analyze the equilibrium response of transaction fees and computational costs posed by the system to changes in the demand for throughput. We characterize the competitive equilibrium and use our model to formally derive the adjustment of throughput to changes in difficulty under which changes in demand lead to efficient system cost changes. We prove that under the derived optimal rule, and when block rewards are zero, changes in demand lead to the balanced growth of throughput and mining revenues, while transaction fees and wait times remain stable.

Our analysis yields two conditions under which stable user fees and wait times across changing levels of demand can be sustained. First, block rewards need to equal zero. Only if block rewards are equal to zero, system difficulty is a sufficient statistic for changes in the demand for throughput. Although necessary at the early stages of network growth, positive block rewards act as a subsidy in this context and decrease market efficiency. Second, predicate size updates need to be gradual enough so that sudden large changes in demand do not destabilize the system. Up to some maximally tolerated change in demand, an update rule which takes into account changes in difficulty to alter predicate size proportionately to the shift in demand would admit stability in its average transaction fees and wait times. Note that this admits a trade-off between the magnitude of the change in demand that can be tolerated and the time it takes for the system to converge to equilibrium.

2 Preliminaries

Many blockchain protocols ensure security by requiring miners to publicly verify energy consumption (known as proof-of-work or PoW Back *et al.* (2002); Dwork & Naor (1992)) in order to gain the right to write transactions on the blockchain. We model this security constraint as the requirement that as demand λ increases, a sufficient statistic (which in Bitcoin is difficulty d) also causes a proportional increase in miner costs.

Since the total number of miners is an endogenous variable that varies over time, the protocol needs a way to ensure that a PoW solution is provided at a fixed frequency (on average) over *all* participants in the network. This is achieved by requiring the PoW puzzle to satisfy a 'difficulty' requirement, which would mean that the expected time taken to generate a solution can be calibrated according to a difficulty parameter that can be changed over time. Through the use of time-stamps, the system 'self-corrects' the difficulty of the PoW puzzle to ensure constant block frequency. We model block frequency as a random variable μ (the block arrival rate) with a constant expectation (the 'block time') mirroring the above technological constraints.

The underlying system consists of a set of users and miners, who each are responsible for requesting and providing proofs from and to the system respectively. Everyone has access to the underlying state of the system, which consists of a sequence of valid transactions that have been processed into the system (the blockchain) along with their respective proofs of correctness: proofs that each block added to the chain is valid.

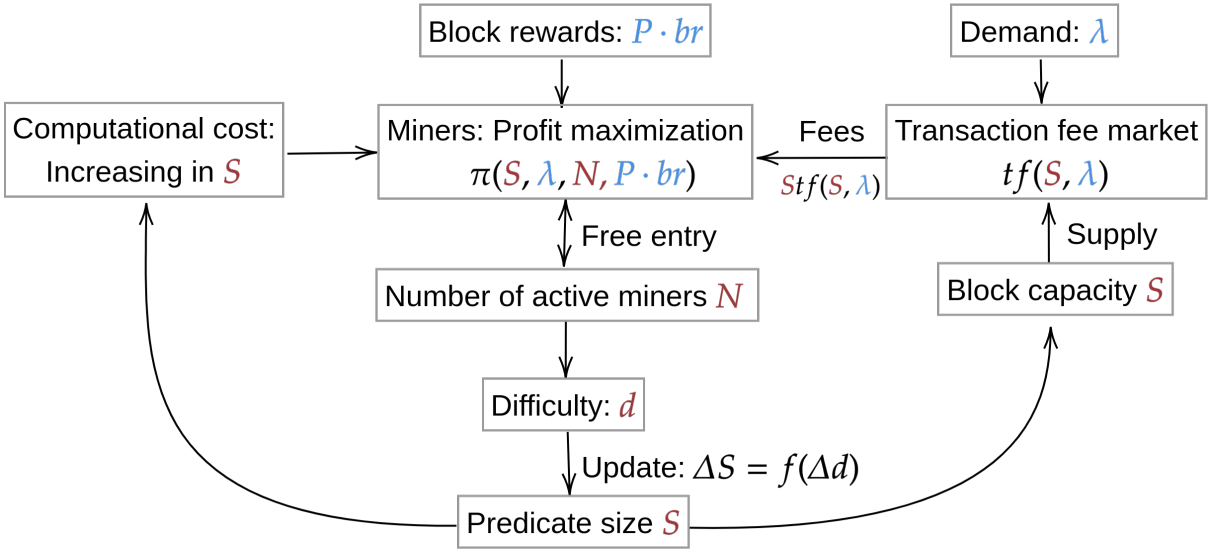
Users submit transactions (or 'records') to miners along with an associated transaction fee. The miners in turn are responsible for adding these transactions to some block

which is then appended to the blockchain. We denote by S the total number of transactions verified in the given block. In order for a miner to win the right to add the next block (and hence collect the corresponding transaction fees and block reward), they need to win the PoW game, which is a puzzle yielding a valid block in time proportional to the underlying difficulty parameter. In this model, only miners are required to keep the whole blockchain (or some kind of 'state' data structure) in order to efficiently generate new blocks and proofs. End-users should be able to efficiently check provided proofs in order to find the correct chain with minimal bandwidth requirements.

The process of generating succinct proofs can be quite resource intensive. One way to deal with this issue is by stipulating that the PoW procedure computes a NIZK as a by-product of each successful iteration. If the NIZK can be randomly resampled according to different nonces without affecting the underlying knowledge soundness claims, it can be used as an effective PoW puzzle. In such a model, the protocol can restrict miners to produce a NIZK attesting to the validity of S transactions at any time step and accept it if a hash of the NIZK satisfies some difficulty condition. Using an update rule for S, such a system can be made to pick the number of transactions verified in each block based on endogenous system variables by stipulating how many transactions the NIZK needs to validate. We investigate how this extra degree of freedom can be leveraged to produce incentive-compatible equilibria in the number of transactions processed by the system per block.

Efficiency in this system is achieved by artificially constraining the block size to $S \leq S_{max}$, where S_{max} a universal upper bound specified by a maximally accepted block size. This restriction to predicate size has the effect of decreasing proof generation times for miners. This is desirable in the context where S_{max} is very large and blocks are (mostly) empty, as the majority of the computational work done to generate a NIZK of size S_{max} will not be needed. Thus, enforcing a lower predicate size would remove the requirements for generating a very large NIZK even when the amount of transactions is substantially lower. Moreover, the restriction of block size to S is a throughput bottleneck if the transaction demand rises sufficiently. We therefore need to investigate how to incentivize the underlying protocol to change the value of S based on changes in transaction demand. This can only be achieved in a stable fashion by taking into account the incentives inherent to changing the size of the block. Since miner profits are based on transaction fees, they are directly proportional to predicate size. We use this observation as the 'driving force' behind throughput adaptability.

Figure 1 A Decentralized Market for Throughput



Notes: This figure illustrates the key forces in our model. Exogenous variables are highlighted in blue, endogenous equilibrium outcomes are highlighted in red.

3 Economic Model

The chart displayed in figure 1 represents the key forces in our model. We model a decentralized market for throughput where transactions are verified in batches/blocks that arrive at a random rate μ in each unit of time. Miners compete for the right to verify an incoming set of transactions in exchange for transaction fees and block rewards. They choose the profit-maximizing number of workers, among which to distribute proofs to simultaneously verify block correctness and generate a PoW solution. In equilibrium, all miners include the transactions offering the highest fees, up to block capacity S.[3] The probability that a miner gains the right to write the next block is proportional to the number of computations that she performs. Free entry requires that all miners make zero profits in expectation, and pins down the equilibrium number of miners N.

Predicate size S corresponds to the number of transactions that can be verified in each new block. Users arrive at the pool of pending transactions at rate λ and offer fees for transaction verification. Transaction fees arise from a VCG auction, where users weigh the cost of higher fees against a reduction in their expected wait time. In equilibrium, average transaction fees tf depend on both block capacity (and thus predicate size S), and on the level of demand λ.

Unlike standard markets, the level of supply - system throughput/block capacity S -

[3] In practice, miners may choose to include different transactions in each block. As we assume that miners observe the same pending transactions at any given point in time and miners have incentives to include transactions with higher fees, miners all aim to verify the same block of transactions.

cannot be directly chosen by miners, but is instead set by the protocol. The inability of miners to choose both their computational cost *and* throughput poses a market failure that leads to inefficient adjustments of system cost to changes in demand. To overcome this issue, we model a system that implements periodic updates to predicate size in accordance with an ex-ante specified update rule. Within this framework, we ask how such an updating rule can be designed to ensure that throughput scales efficiently with changes in demand.

3.0.1 Hash Rate, Predicate Size, and Pools

Miners compete for the chance to mine the next arriving block and collect the associated mining rewards. To do so, miners distribute proof computations corresponding to predicate size S to pools, and provide PoW in terms of proofs per second H to the system. A miner employing n workers to solve proofs corresponding to a predicate size S provides H proofs per second according to,

$$\log H(n; S) = \log U + \sigma \log n - \log S, \tag{1}$$

where U is a technological constant. The parameter σ governs the returns to pool size n: It captures the elasticity of the hash rate H with respect to pool size n, and can be thought of as a parallelization coefficient. Consistent with existing performance benchmarks, we assume that $\sigma < 1$.[4] A value of $\sigma = 1$ would imply perfect conservation of work in a parallel system of n workers, and as such, no higher value of σ is empirically achievable.

As evident in equation (1), an increase in predicate size S reduces a pool's hash rate for any given pool size. Thus, an increase in S poses a cost to the system as pools have to distribute proofs across more workers to maintain the same hash rate.

3.0.2 The Problem of Miners

Miners incur a cost $c_m \cdot n$ per time unit to distribute proofs across n workers.[5] Further, miners incur fixed costs f_m every period they maintain a mining pool. Both variable and fixed costs are denominated in USD.

Miners compete for the chance to mine the next arriving block. In equilibrium, all miners include transactions offering the highest rewards up to block capacity S. Total

[4] State-of-the-art distributive systems of SNARK proofs achieve $\sigma \approx 0.88$ Wu *et al.* (2018).

[5] In practice, miners distribute and assemble proofs while workers solve small computational problems. The cost c_m summarizes both margins.

rewards from mining are given by $Rev + P \cdot br$, where Rev denotes fee revenue and br block rewards. Transaction fees are proposed by users, and, thus, denominated in USD. We specify Rev further below after having solved the problem of users. The system exogenously regulates the size of the block reward , implying that the value of br to miners depends on the exchange rate of the currency to USD, which we denote by P.

As in PoW, the probability that a given pool is selected to mine the next block depends on its hash rate relative to the aggregate hash rate of the system.[6] Denoting individual mining pools by i, the aggregate hash rate of the system is given by $\mathcal{H} = \sum_{i \in \{\text{Active Pools}\}} H_i$.

Taking predicate size S and the aggregate hash rate $\mathcal{H}$ as given, a miner i chooses pool size n_i to maximize expected profits π given by:

$$\pi\left(n_i; S, \mathcal{H}\right) = \frac{H\left(n_i, S\right)}{\mathcal{H}} \left(Rev + P \cdot br\right) - c_m \cdot n_i - f_m, \tag{2}$$

where $H\left(n, S\right)$ is defined in equation (1).

In a Nash equilibrium, the optimal pool size n_i^* solves,

$$n_i^* = \arg\max_n \pi\left(n; S, \mathcal{H}\right). \tag{3}$$

The solution to this problem is given by:[7]

$$n_i^* = \left(\frac{\sigma\left(Rev + P \cdot br\right)}{c_m (S/U) \mathcal{H}}\right)^{1/(1-\sigma)}. \tag{4}$$

eq:optimal$_p$ool$_s$izeshowsthattheoptimalpoolsizen$_i^*$ is increasing in mining rewards, and decreasing in the variable costs of proof distribution c_m, predicate size S, and market competition captured by the aggregate hash rate $\mathcal{H}$.

3.0.3 Entry and the Equilibrium Number of Miners

Anyone willing to incur the overhead cost f_m is free to enter the system as a miner. Miners enter until expected profits

[6] Chen *et al.* (2019) and Leshno & Strack (2019) show that proportional selection is the only allocation rule that satisfies a set of desirable properties, e.g. anonymity, collusion-resistance, and sybil-resistance.

[7] When setting $\frac{\partial \pi}{\partial n_i} = 0$, we assume that miners are small in the sense that they don't internalize the effect their computations have on aggregate hash rate $\mathcal{H}$. Our qualitative results do not crucially depend on this assumption.

eq:profits equal zero. Thus, a free entry condition pins down the number of active mining pools N.

In a symmetric, equilibrium, all active miners employ the same number of workers, $n_i^* \equiv n^*$. Then, the aggregate hash rate is given by $\mathcal{H} = NH(n^*)$. Using equation (4), the optimal pool size n^* is given by,

$$n^* = \frac{\sigma}{c_m} \cdot \frac{Rev + P \cdot br}{N}. \tag{5}$$

In equilibrium, the probability that a miner is chosen to mine the next block is proportional to the number of active pools, $\frac{H_i}{\mathcal{H}} = \frac{H}{NH} = 1/N$. Substituting equation (5) into profits given by equation (2) and imposing that expected profits equal zero upon entry, the equilibrium number of miners equals,

$$N = \frac{(1-\sigma)}{f_m} \cdot (Rev + P \cdot br). \tag{6}$$

The equilibrium number of miners N is increasing in mining revenues and decreasing in the fixed costs f_m.

Miners include transactions that offer the highest transaction fees up to the capacity limit. Intuitively, mining costs are increasing in predicate size S, so miners have no incentive to verify less than S transactions. Further, miners can neither force others into a coalition, nor have an incentive to form coalitions with the intent to temper with transaction fees[8] due to the presence of fixed costs.

3.1 Demand for Transactions and Determination of Fees

We now describe the problem of users, and characterize the equilibrium level of transaction fees. The demand side of the model closely follows Huberman *et al.* (2019). Users wish to verify single transactions and are heterogeneous in the costs that verification delay poses for them. New users enter the pool of pending transactions at a constant rate per time unit, and offer transaction fees for service. Users' willingness to pay arises out of the opportunity cost of delay.

3.1.1 Users

Users arrive at Poisson rate λ at each point in time. λ parametrizes the level of demand by describing the rate at which the pool of pending transactions grows over time.

[8] For example, a large coalition may choose not to include transaction fees below a certain threshold.

An arriving user is identified by a wait cost c, which is drawn from a continuous cumulative density function $F(c)$ with domain $C \equiv [0, \bar{c}] \subseteq \mathbb{R}_0^+$ and probability density function $f(c)$. The expected benefit of using the system to a user endowed with wait cost c and offering transaction fee tf is given by:

$$U(tf; W, c) = v - tf - cW(tf; G),$$

where $v > 0$ is the value of a verified transaction to users[9], and $W(tf, G)$ is the expected wait time for a user posting fee tf. The wait-time W depends on the endogenous distribution of transaction fees $G(tf)$ posted by all users of the system.[10]

3.1.2 The Dependence of Fees on Wait Time

Each active user posts a fee tf so as to maximize utility, weighing the costs of higher fees against the benefit of lower expected wait times. Standard arguments imply that equilibrium transaction fees are a continuous, monotonous function of user wait-cost $tf(c)$ satisfying $tf(0) = 0$ and $tf'(c) > 0$. Monotonicity implies that $G(tf(c)) = F(c)$. Equilibrium transaction fees $tf(c)$ solve the first order condition $W'(tf|G) = -\frac{1}{c}$ which is equivalent to the following differential equation:

$$\tilde{W}'(c|F)\, cf(c) = tf'(c),$$

where $\tilde{W}$ directly maps wait cost into wait times and satisfies $-W'(tf|G) = W'(c|F)$.[11] Integrating and imposing as a boundary condition that users with no costs of delay post a fee equal to zero, transaction fees can be fully expressed in terms of equilibrium wait times:

$$tf(c) = \int_0^c \tilde{c} f(\tilde{c}) \tilde{W}'(\tilde{c}|F)\, d\tilde{c}. \tag{7}$$

Inspecting equation (7) reveals that equilibrium fees have the externality correcting property inherent to the Vickrey-Groves-Clark (VCG) auction mechanism.[12] For our purposes, the fact that offered transaction fees are socially optimal implies that none of the potential inefficiencies present in the decentralized equilibrium arise from sub-

[9] In practice, users have an outside option and only participate if their utility from doing so exceeds this outside option. Throughout the analysis, we assume that v is sufficiently high so that users are willing to participate.

[10] The assumption is that users know the distribution of wait times F and are assumed to anticipate others' optimal behavior correctly. However, users do not observe the current pool of pending transactions when submitting fee offers.

[11] Note that $G(tf(c)) = F(c)$ implies $G'(b(c))b'(c) = f(c)$. Thus $W'(tf|G) = W'(tf|G)f(c)/b'(c)$.

[12] An externality arises as each user delays the verification of the transactions of other users. Despite the fact that users do not internalize the cost that they pose on others, the VCG auction mechanism incentivizes "truthful bidding," implying that the users with higher valuations submit higher bids.

optimal user behavior.

3.1.3 Wait Time

Blocks can process up to S transactions per time period, and arrive randomly at Poisson rate μ. The equilibrium wait time depends on system congestion, which is given by $\rho \equiv \lambda/(S\mu)$ and captures the average number of transactions that can be processed per time unit. $\rho < 1$ implies that while users may experience delays, the system will eventually process all transactions.

To characterize the equilibrium wait time, we build on results derived in Huberman *et al.* (2019), who show that for sufficiently high levels of block capacity S, wait times depend solely on system congestion ρ.[13] We restate this result in the following theorem.

Theorem 1 1. *For any $\rho \equiv \lambda/(\mu S) \in (0,1)$ the equilibrium wait time for a user with delay cost c, $W(c; S, \mu, \lambda)$ is given by:*

$$W(c;\rho) = \frac{\mu^{-1}}{1 - (1 + \alpha(\rho\bar{F}(c)))\, e^{-\alpha(\rho\bar{F}(c))}}, \tag{8}$$

where $\bar{F}(c) \equiv 1 - F(c)$ and $\alpha(x)$ is the real root of the algebraic equation $e^{-\alpha} + x\alpha - 1 = 0$.

2. *Equilibrium wait time is increasing and convex in congestion ρ and decreasing in user wait cost c.*

Users with higher costs of delay offer higher fees, and wait less for transaction verification. Higher levels of system congestion ρ, in turn, increase wait times - and, therefore, transaction fees - for all users. The key implication of Theorem 1 for our analysis is that since wait times - and, therefore, transaction fees - are fully characterized by system congestion ρ, equilibrium fees are stable at varying levels of demand λ only if the system adjusts S so as to maintain a stable level congestion ρ.

3.1.4 Transaction Fees

Combining Theorem 1 and equation (7), we can derive the average equilibrium fee revenue *Rev* per unit of time as function of block capacity S and system congestion ρ:

[13] Huberman *et al.* (2019) show wait times are appropriately approximated for $S = 20$. Block capacity in Bitcoin is at around 2000. As outlined earlier, we expect our proposed protocol implementation to be capable of handling a volume of transactions per block that outperforms Bitcoin by magnitudes. We, therefore, feel comfortable working with the characterization of wait times in equation (8) for the remainder of the analysis.

$$Rev\left(S,\rho\right) = S \cdot \int_0^{\bar{c}} t f(c) dF(c) \tag{9}$$

$$= S \cdot \rho \int_0^{\bar{c}} \left(\bar{F}(c) - c f(c)\right) W\left(c;\rho\right) dc \equiv S \cdot \psi\left(\rho\right).$$

$\psi(\rho)$ in equation (9) is the average level of fees per unit of time that users pay for service.[14] *Rev* summarizes the optimal bidding behavior of users, and, thereby, how changes in either block capacity S or demand λ affect the demand-side of the market.

The focus of our analysis is to understand how the system may induce an increase in throughput in response to an increase in demand so as to incentivize miners to scale proof production at constant per-transaction fee revenues. For this purpose, the responsiveness of average fees $\psi\left(\rho\right)$ with respect to changes in demand λ - or generally changes in congestion ρ - is key. To this end, we define the elasticity of $\psi(\rho)$ in equation (9) with respect to ρ as $\varepsilon\left(\rho\right) \equiv \frac{\partial \log \psi(\rho)}{\partial \log \rho} \equiv \frac{\rho}{\psi(\rho)} \frac{\partial \psi(\rho)}{\partial \rho}$. $\varepsilon\left(\rho\right)$ captures the percentage increase in equilibrium fees in response to a percentage increase in congestion ρ. The following theorem characterizes key properties of $\varepsilon\left(\rho\right)$ and in particular shows that it admits a uniform upper bound, independently of the underlying distribution of wait cost $F(c)$.

Theorem 2 *The elasticity of equilibrium average transaction fees $\psi\left(\rho\right)$ with respect to system congestion ρ, $\varepsilon\left(\rho\right) \equiv \frac{\partial \log \psi(\rho)}{\partial \log \rho}$, is bounded below by 1, increasing and convex in ρ. Further, there exists $\overline{\varepsilon}\left(\rho\right)$ such that $\varepsilon\left(\rho\right) \leq \overline{\varepsilon}\left(\rho\right)$ for all $\rho \in (0,1)$ and any distribution of user wait cost $F(c)$.*

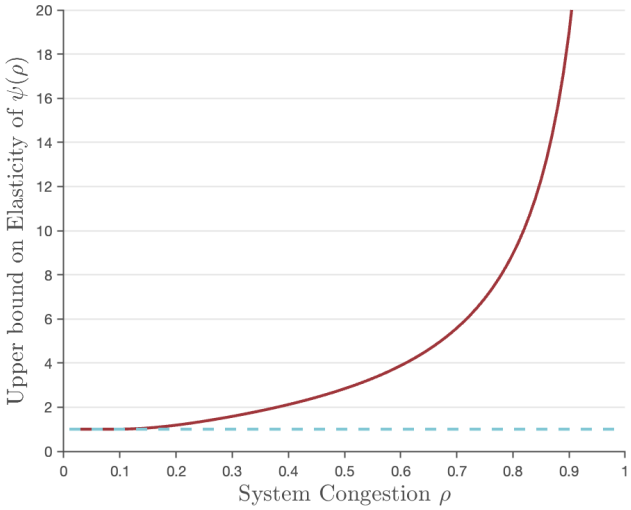
Figure 2 plots the elasticity of average transaction fees as a function of system congestion ρ.

3.2 Equilibrium at Fixed Predicate Size S

We characterize the mass of entrants and level of system difficulty in partial equilibrium, holding fixed the predicate size S.

[14] As we characterize steady states in a fee market where there is a constant flow of transactions in and out of a system that eventually verifies all transactions and where all users participate, the integral is taken over the entire domain of c.

Figure 2 Upper bound on the elasticity of equilibrium transaction fees with respect to system congestion



Notes: This figure plots the uniform bound on the elasticity of equilibrium average fess given in equation (9) with respect to system congestion $\rho = \lambda/\mu S$ as a function of ρ on the x-axis.

3.2.1 Mass of active mining pools

A partial equilibrium for a given predicate size S and demand level λ is defined by the condition that miners make zero profits in expectation upon entry.[15] Using equation (6) and the characterization of total mining fee revenues, the equilibrium number of miners N at predicate size S, demand λ and nominal block rewards $P \cdot br$ is given by:

$$N(S, \lambda, P \cdot br) = \frac{1-\sigma}{f_m} \cdot (S\psi(\rho) + P \cdot br). \tag{10}$$

Predicate size S, demand λ and nominal block rewards $P \cdot br$ summarize the profit incentives of miners. Adjustments in predicate size S, therefore, change the profit incentives and entry behavior of miners. Cost-efficiency in a decentralized equilibrium crucially depends on whether adjustments of predicate size S can be optimally calibrated to induce incentive-compatible equilibrium behavior of miners that is *as if* miners internalized the externality that they pose on others through Nakamoto consensus. In order to build up an intuition for our later results, it is useful to briefly pause to discuss how adjustments in predicate size S shape profit incentives of miners through the condition in equation (10).

[15] While this equilibrium is technically a steady-state - owing to the continuous entrance and exit of users - we effectively treat it as a static equilibrium condition.

An increase in demand λ initially increases mining fee rewards through an increase in equilibrium user fees $\psi(\rho)$. This increases entry, the equilibrium number of miners, and the computational cost of the system without adding benefits for system capacity or users. To restore cost-efficiency, a subsequent increase in predicate size S has to direct additional cost incurred by the system toward increases in throughput. Our results imply that competition by miners ensures that this is achieved by additional entry and lower success probabilities for each active pool.

3.2.2 Difficulty

As opposed to for example Bitcoin, difficulty in our model depends only on the total number of proofs computed per time unit.[16] As pools are symmetric, solely distribute one proof computation among pool participants and operate at hash rate H, system difficulty is given by:

$$d = \log\left(\mu \times H \times N\right). \tag{11}$$

By using the equilibrium expression for the number of miners and the solution to the miner's problem, we can express difficulty as:[17]

$$d = \log\left(\beta \times \frac{S\psi(\rho) + P \cdot br}{S}\right), \tag{12}$$

where $\beta \equiv \mu U \left(\frac{\sigma}{c_m}\right)^{\sigma} \left(\frac{1-\sigma}{f_m}\right)^{1-\sigma}$ is a constant.

The inability of miners to autonomously adjust cost *and* throughput upwards as demand rises causes rising levels of system congestion, resulting in high wait times and fees for users, and costly increases in mining activity at no increase in throughput as demand rises. In light of this, our protocol design's core innovation is to provide an "updating rule" for predicate size S and, therefore, block capacity that correctly infers changes in demand. While demand λ is not directly observable, equation (12) highlights that difficulty is a sufficient statistic for changes in demand when block rewards are zero. In this case, changes in equilibrium difficulty are proportional to changes in average transaction fees, which in turn depend solely on the congestion parameter ρ.

[16] This is because the solution of a single proof commits to the nonce before distributing computation to the pool. This means that, unlike the distribution of double-SHA computations in Bitcoin (that effectively break up nonce generation among pool participants), here the nonce is set by the operator, and all computation forwarded to workers is nonce-specific.

[17] Note that the solution to the miners' problem implies that equilibrium hash rates are given by $U\left(\frac{\sigma f_m}{(1-\sigma)c_m}\right)^{\sigma}/S$.

3.3 Dynamic Throughput Adjustment and Equilibrium Predicate Size S

We endow the system with the ability to periodically adapt throughput S according to a pre-specified updating rule.

Definition 1 *Let* $\mathbf{X}$ *be a vector of statistics. A law of motion for throughput* S *is a function* f *such that:*

$$S_{t+1} = f\left(S_t, S_{t-1}, ...; \mathbf{X}_t, \mathbf{X}_{t-1}, ...\right). \tag{13}$$

Given an updating rule for predicate size S, we can define a dynamic equilibrium and the notion of a steady-state.

Definition 2 *Given a starting predicate size* S_0*, level of demand* λ *and nominal block rewards* $P \cdot br$*, an equilibrium is a sequence of predicate sizes* $\{S_t\}_{t=1,2,...}$ *such that at each* t*, (i) transaction fees posted by users satisfy equation* (9)*, (ii) the equilibrium number of miners* N_t *satisfies equation* (6) *and (iii) the law of motion for* S *is given by equation* (13)*. A steady state is reached when changes in predicate size* S *converge to 0.*

So far, we have analyzed how the optimal behavior of miners and users determines fees and the energy usage of the system. In the following, we utilize our framework to analyze how transaction fees and system congestion respond to demand shocks under different pre-specified laws of motion for throughput.[18]

4 Stabilizing Congestion

Unlike standard markets, the level of supply is not directly chosen by miners, but, instead, is set by the protocol through the size of the predicate S. This results in cost inefficiencies. Intuitively, absent changes in S, an increase in demand λ raises system congestion $\rho = \lambda/(\mu S)$. Ultimately, this increases the computational costs of the system without lowering the costs to users. In this section, we show that suitable updates of throughput help overcome this limitation. Specifically, as equilibrium difficulty encodes changes in demand, we show how to design an updating rule for the predicate size S in accordance with changes in difficulty so as to induce cost-efficient adjustments of throughput in response to changes in demand.

[18] Definition 2 defines an equilibrium taking both block rewards Pbr and demand λ as given. The underlying assumption is therefore that adjustments in predicate size S occur sufficiently frequently - or equivalently that the length of periods dt is sufficiently small - for the system to reach steady states between changes in demand or block rewards.

To develop intuition for our result, note that the net benefit produced by the system can be characterized in terms of the value of transaction verification across all consumers, wait times as well as the the costs of energy used by pools:

$$v\lambda - \text{Average Wait-time}\,(\lambda, S) - \text{Total Energy Cost}\,(S)\,.$$

Free entry implies that the total energy cost of the system must equal mining rewards. Abstracting from block rewards, total mining revenues increase at least proportionally to system congestion, as do average wait times. To ensure that the net benefits of the system do not decrease as demand λ increases, changes in demand λ should be offset by proportional changes in S. It is easy to see that this would imply that the benefit of the system would remain invariant to how many people use it. In conclusion, an ideal adjustment of predicate size S to changes in difficulty keeps wait times and congestion $\rho = \lambda/(\mu S)$ constant, allowing for proportional growth in demand and supply.

4.1 Floating Difficulty Level

Consider a policy rule that upon observing a change in difficulty between perdiods $t-1$ and t, updates predicate size S in $t+1$ as follows:

$$\log\left(S_{t+1}/S_t\right) = \gamma\left(d_t - d_{t-1}\right). \tag{14}$$

We focus on analyzing transitions between equilibrium steady states in response to changes in demand when block rewards are zero. We analyze constraints on the updating parameter γ stemming from two objectives. First, γ should ensure convergence of the system to a new steady state after a demand shock. Second, a shock to demand should yield minimal change s in congestion.

Assume the system is in steady state with predicate size S_0 and congestion $\rho^* = \lambda_0/\mu S_0$. We consider a change in demand $d\log\lambda \equiv \log(\lambda_1) - \log(\lambda_0)$ from λ_0 to λ_1. The object of interest is the elasticity of steady state system congestion with respect to this demand shock, $\frac{d\log\rho^*}{d\log\lambda}$, which measures by how much steady state system congestion changes in response to a change in demand. In the following proposition, we summarize the properties of this object.

Proposition 1 *Consider an initial steady state $\{S_0, \lambda_0\}$ and assume that block rewards are equal to zero. Under a floating difficulty regime, upon a change in demand $d\log\lambda \equiv \log(\lambda_1) - \log(\lambda_0)$,*

1. *the system converges to a new steady state if $\gamma < \frac{1}{\overline{\varepsilon}(\overline{\rho})}$, where $\overline{\varepsilon}\,(\rho)$ denotes the upper*

bound on $\varepsilon(\rho)$ *derived in Theorem 3.2. and* $\overline{\rho} \equiv \max\{\lambda_0/\mu S_0, \lambda_1/\mu S_0\}$,

2. *if* $\gamma \in \left(0, \frac{1}{\overline{\varepsilon}(\overline{\rho})}\right)$, *then steady state changes in system congestion are strictly less than proportional to changes in demand:* $\frac{d\log\rho^*}{d\log\lambda} \in \left(\frac{1}{2}, 1\right)$.

The first part of Proposition 1 shows that dynamic stability imposes a feasibility restriction on the rate at which predicate size (and thus throughput) can be increased in response to some demand shock. For $\gamma > 1/\overline{\varepsilon}(\overline{\rho})$, changes in predicate size would be large enough at every step to ensure that the system diverges from equilibrium. Thus, the value of γ needs to be low enough so that this is prevented. Note that this imposes a new trade-off in efficiency: a lower γ ensures that higher demand shock levels can be 'absorbed' without instability, but also means that the system will take longer to arrive at equilibrium (as predicate updates are less sensitive to changes in difficulty).

The second part of the above result demonstrates that such a record keeping system with $\gamma > 0$ does strictly better than one where S is fixed. This is due to the upper bound on the elasticity of steady state system congestion, which shows that congestion will increase at a fraction of the non-adaptive case. Although they will increase comparatively less, it is immediate that transaction fees remain unbounded: an arbitrary sequence of positive demand shocks will always increase difficulty arbitrarily high, pushing up equilibrium fees as well. Further, to ensure dynamic stability at possibly high levels of congestion requires sufficiently low γ, which impedes the speed of convergence. In fact, the only choice for γ that ensures dynamic stability under a floating difficulty regime as $\lambda \to \infty$ is $\gamma = 0$.

In the traditional Bitcoin protocol, an unbounded difficulty parameter is required due to the security benefits that it provides. This is by making the cost of disrupting the network scale with d. Note, however, that an increase in difficulty here is not necessarily required for system security: the costs of participation are increasing with respect to both difficulty *and* predicate size. This means that 'difficulty' in the Bitcoin sense is actually an increasing function $g(S, d)$ here, as both of these parameters are monotonically increasing with respect to costs per unit time. This motivates the question of whether *constant* equilibrium transaction fees can be achieved without compromising system security.

4.2 Pegged Difficulty Level

Consider a policy rule that updates predicate size S as follows:

$$\log \frac{S_{t+1}}{S_t} = \gamma (d_t - d^*), \tag{15}$$

where d^* is a *pre-specified* level of difficulty.

Again, consider a demand shock $d \log \lambda$. The following proposition draws out conditions on the update parameter γ under which the system maintains a stable steady state level of congestion ρ^* under zero block rewards.

Proposition 2 *Assume that block rewards are equal to zero, and difficulty is pegged to d^*. Denote $\rho^* \equiv \psi^{-1}(\exp(d^*/\beta))$ with β given in equation (12). Under a pegged difficulty regime, upon a change in demand $d \log \lambda \equiv \log(\lambda_1) - \log(\lambda_0)$ yielding an initial change in congestion from ρ^* to ρ, congestion converges back to ρ^* for any $\gamma \in \left(0, \frac{2}{\bar{\varepsilon}(\overline{\rho})}\right)$, where $\overline{\rho} \equiv \max\{\rho^*, \rho\}$.*

Equation (15) allows for the *a priori* selection of some acceptable congestion level ρ^* by fixing (or 'hardcoding') the corresponding d^* in the update rule. In this case, the system will increase predicate size S accordingly until system difficulty *drops* back to d^*. Therefore, the long-term equilibrium effect on congestion is zero. Such a system has the desirable property that transaction fees at equilibrium will always revert to constant: the effects of any demand shock will be fully compensated by changes in predicate size, which will revert transaction fees to their previous equilibrium level.

Technically, the crucial change that permits such desirable behavior lies in the 'decoupling' between the sufficient statistics for system security and demand in the above update rule. Indeed, the costs of participation rise monotonically with S, while costs of transactions are only functions of d. Thus, by ensuring that the system will always update S as a reaction to changes in demand so as to fully compensate for any changes in d, we are able to keep equilibrium transaction fees constant *regardless* of the demand shock as long as γ is low enough.

4.3 Positive Block Rewards

We conclude our analysis by briefly discussing how positive block rewards, $br > 0$, affect our key results. In the presence of positive nominal block rewards, miners' profit incentives are not fully determined by mining fees and therefore entry and changes in difficulty only are an imperfect measure of demand. Consequently, changes in demand cannot be fully absorbed through proportional scaling of throughput. Further, changes in nominal block rewards - either due to an increase in P or a (scheduled) decrease in block rewards br - will cause changes in throughput that are (potentially) detached from fundamental user demand. While nominal block rewards serve as a subsidy to sustain mining incentives at low levels of initial demand, here they pose a cost in terms of economic efficiency.

5 Conclusion

We provided a protocol specification for a decentralized record-keeper that addresses economic cost efficiencies from correctness verification, PoW security guarantees and throughput adaptability. Drawing on recent developments aimed at optimizing the scaling of distributed payment systems, we have shown how to link throughput to an observable system statistic that summarizing changes in user demand. Our work shows to autonomously regulate incentives of system participants so as to ensure cost-efficient scaling to user demand. We believe that an implementation of this approach is feasible given recent advances in the computer science literature concerned with the scaling of decentralized record-keepers.

References

Abadi, Joseph and Markus Brunnermeier, 'Blockchain Economics.' NBER Working Papers 25407, National Bureau of Economic Research, Inc, *2018*.

Back, Adam *et al.*, 'Hashcash-a denial of service counter-measure.' *2002*.

Ball, Marshall, Alon Rosen, Manuel Sabin and Prashant Nalini Vasudevan, 'Proofs of Useful Work..' *IACR Cryptology ePrint Archive*, **2017**, p. 203, *2017*.

Basu, Soumya, David Easley, Maureen O'Hara and Emin Sirer, 'Towards A Functional Fee Market For Cryptocurrencies.'Technical report, *2019*.

Budish, Eric, 'The Economic Limits of Bitcoin and the Blockchain.' working paper, *2018*.

Buterin, Vitalik, 'Ethereum: A next-generation smart contract and decentralized application platform.' *2014*, Accessed: 2016-08-22.

_ , *On-chain scaling to potentially 500 tx/sec throughmass tx validation., 2018 (accessed June 12, 2020)*.

Carlsten, Miles, Harry Kalodner, S Matthew Weinberg and Arvind Narayanan, 'On the instability of bitcoin without the block reward.' In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 154–167, *2016*.

Chen, Xi, Christos Papadimitriou and Tim Roughgarden, 'An Axiomatic Approach to Block Rewards.' In *Proceedings of the 1st ACM Conference on Advances in*

Financial Technologies, AFT '19, p. 124–131, New York, NY, USA: Association for Computing Machinery, *2019*, DOI: http://dx.doi.org/10.1145/3318041.3355470.

Dwork, Cynthia and Moni Naor, 'Pricing via processing or combatting junk mail.' In *Annual International Cryptology Conference*, pp. 139–147, Springer, *1992*.

Huberman, Gur, Jacob Leshno and Ciamac C. Moallemi, 'An economic analysis of the Bitcoin Payment System.' Columbia Business School Research Papers 17-92, Columbia Business School, *2019*.

Kattis, Assimakis and Joseph Bonneau, 'Proof of Necessary Work: Succinct State Verification with Fairness Guarantees.' Cryptology ePrint Archive, Report 2020/190, *2020*, https://eprint.iacr.org/2020/190.

King, Sunny, 'Primecoin: Cryptocurrency with prime number proof-of-work.' *July 7th*, **1** (6), *2013*.

Leshno, Jacob and Philipp Strack, 'Bitcoin: An Impossibility Theorem for Proof-of-Work based Protocols.' Cowles Foundation Discussion Papers 2204R, Cowles Foundation for Research in Economics, Yale University, *2019*.

Nakamoto, Satoshi, 'Bitcoin: A peer-to-peer electronic cash system.' *2008*, http://bitcoin.org/bitcoin.pdf.

Thum, Marcel, 'The Economic Cost of Bitcoin Mining.' *CESifo Forum*, **19** (1), pp. 43–45, *2018*.

Wu, Howard, Wenting Zheng, Alessandro Chiesa, Raluca Ada Popa and Ion Stoica, '{DIZK}: A Distributed Zero Knowledge Proof System.' In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pp. 675–692, *2018*.

A Proofs

Theorem 1. See Lemma 14 in Huberman *et al.* (2019). □

Theorem 2. Let $\psi(\rho) = \rho \int_0^{\bar{c}} (\bar{F}(c) - cf(c)) W(c,\rho)\, dc$. First, the derivative of the wait time is equal to:

$$\frac{\partial W(\rho)}{\partial \rho} = e^{-\alpha(\rho)} \alpha(\rho)^3 W(\rho)^3.$$

Therefore, the elasticity is given by:

$$\frac{\rho}{W(\rho)}\frac{\partial W(\rho)}{\partial \rho} = \rho e^{-\alpha(\rho)}\alpha(\rho)^3 W(\rho)^2. \tag{A.1}$$

Evidently, this elasticity is increasing in ρ. Thus, the elasticity of average transaction fees is given by:

$$\frac{d\log\psi(\rho)}{d\log\rho} = 1 + \rho\int_0^{\bar{c}} \omega_c \bar{F}(c) e^{-\alpha(\rho\bar{F}(c))}\alpha(\rho\bar{F}(c))^3 W(\rho\bar{F}(c))^2 dc,$$

where $\omega_c \equiv \frac{\bar{F}(c)-cf(c)W(c,\rho)}{\int_0^{\bar{c}}(\bar{F}(c)-cf(c))W(c,\rho)dc}$. Given that $\alpha(\rho)\to\infty$ as $\rho\to 0$, a uniform bound on this elasticity is given by:

$$\frac{d\log\psi(\rho)}{d\log\rho} \leq 1 + \rho e^{-\alpha(\rho)}\alpha(\rho)^3 W(\rho)^2 \equiv \overline{\varepsilon}(\rho).$$

□

Proposition 1. Using the recursive structure of the system, the elasticity of steady state congestion with respect to a demand shock is:

$$\frac{d\log\rho^*}{d\log\lambda} = 1 + \sum_{t=1}^{\infty}(-\gamma)^t\left(\prod_{k=0}^{t}\varepsilon(\rho_t)\right).$$

Using Theorem 2, the proposition follows from standard properties of geometric sums. □

Proposition 2. Under a pegged difficulty regime, the change in steady state congestion in response to a demand shock and an initial level of congestion ρ^* can be written as $\varepsilon(\rho^*)\prod_{t=1}^{\infty}(1-\gamma\varepsilon(\rho_t))$. The result then follows from Theorem 3.2. permitting to bound all terms in the product above by 1. □