Miner Collusion and the BitCoin Protocol

Abstract

Bitcoin users can offer fees to miners who record their transactions in the Blockchain. We document high variation of bitcoin fees, not only over time, but also within blocks. Further, the blockchain rarely runs at capacity, even though fees tend to be higher when blocks are fuller, so miners appear to be leaving "money on the table." We present a simple model of price discrimination to explain our results. We note that mining pools facilitate collusive equilibria, and estimate that they have extracted least 200 million USD a year in excess fees by making processing capacity scarce.

1 Introduction

On September 10, 2019 Nasdaq announced a new index, Defix, to track firms organized along the principle of decentralized finance. The index is consistent with the growing importance of financial entities that are designed to operate absent a trusted intermediary. While the production function of such entities is clearly novel, it is less obvious if their market conduct is also novel. In this paper, we examine whether the first successful decentralized financial product, Bitcoin, has evolved into a system that provides value transfer to consumers as it was promised to. Understanding how well the Bitcoin system works is important because it is the first widely-adopted fully decentralized finance system; it is also unregulated, and so provides us with insights into the sorts of economic structures that can arise endogenously.

The original white paper by Nakamoto presented a system that obviates the need for trusted intermediaries because "competitive miners" record and settle all BitCoin transactions. The code assigns BitCoin as an incentive for miners to do this. The first departure from the zero direct cost settlement, was that spontaneously, over time, consumers have added fees to their transactions to encourage speedy settlement. In this paper, we shed light on the determinants of Bitcoin transaction fees and provide suggestive evidence that miners act in a way consistent with fee maximization, as a monopolist intermediary might. We conclude that even a decentralized finance system can operate non-competitively.

We have three main results. First, we document that the BitCoin blockchain rarely operates at full capacity even in the presence of fees. Our sample comprises all Bitcoin transactions from the genesis block to November 4, 2018. There is no day over this period in which the BlockChain has run at full capacity. Fees began appearing in 2016, and amount to USD 844,537,503.30.

The existence of excess capacity, in the presence of fees, suggest miners appear to leave "money on the table." Our second result is that we illustrate that this behavior is observationally equivalent to collusive price-discrimination. Mining pools, in as far as they restrict the number of independent players in the system, facilitate such price discrimination. Third, consistent with a conservative interpretation of our model, we define excess fees as those paid above the 10% quantile of observed fees, which sum to USD 557,568,542.21, or approximately 200 million USD a year since the advent of mining pools.

Although the fees submitted by agents to the BlockChain appear akin to bids in a multi-unit first price private value auction, they arise from a different protocol. In a multiunit first price auction, higher bids are more likely to get the good (in this case, immediate settlement). Therefore, whether a particular bid results in the bidder winning the good depends on all the other bids because the seller commits to award the good to higher bids first. Effectively, the rationing rule is one of strict price priority. By contrast, under the BitCoin protocol, individual miners are not bound by any particular rationing rule, and may choose any set of transactions to work on. To complicate matters, given the decentralized system, miners may even observe different pools of potential transactions.

A fully dynamic model in which sellers have different information sets and do not commit, while buyers bid against an unknown number of future other buyers is beyond the scope of this paper. However, to understand how miners' ability to choose any set of buyers to mine

affects the consumer cost of using the system, we present a simple two type framework that compares bidding behavior in two specific cases. First, we characterize bids if there are capacity constraints, but miners always execute the highest bidders first. In this case, the bidders who are most desperate for immediate settlement only have to bid slightly higher than the lowest type. In other words, the high types retain some surplus. If miners can pick and choose when and which bids to process – price discriminate – miners can ignore any bids that are less than the high types' maximal valuation. Under this rationing rule, the miners extract the high types' full valuation. The optimal rationing rule obviously depends on various parameters such as how high the high types' valuation is and how many there are of them. This is the familiar price-quantity tradeoff.

The BitCoin protocol is designed so that miners are competitive, and it is well known that in repeated games it is more difficult to sustain collusive equilibria as the number of players increases. This suggests that mining pools provide an economic role besides diversifying risk for individual participants. By acting collectively, each mining pool effectively reduces the set of strategic players and so makes it easier to sustain price-discrimination. This observation is consistent with mining pools' habit of "signing" the blocks that they mine. If they do this, other, unsuccessful, mining pools have a credible way of checking whether the block was mined at full capacity (the pool deviated) or under-capacity. Empirically, we find a relationship between excess fees and mining capacity concentration.

Our paper fits into a small and growing literature on transaction fees in blockchain systems. David Easley, Maureen O'Hara & Soumya Basu (2019) explain the observed shift from no-fee to fee paying transactions and model the interactions of fee payments and waiting times. While the focus of their empirical analysis is on the time series of average transaction fees our paper documents a huge variation of Bitcoin transaction within blocks analyses the cross section of transaction fees. Gur Huberman, Jacob Leshno & Ciamac C Moallemi (2017) compare Bitcoin to a traditional payment system and derive closed form solutions for equilibrium fees.

This research on fees fits into a larger body of literature that focuses on the economics and incentives in blockchain ecosystems (among others Joseph Abadi & Markus Brunnermeier (2018), Lin William Cong & Zhiguo He (2019), Eric Budish (2018)) and the impact on financial markets (e.g. Katya Malinova & Andreas Park (2017) or Alexander Brauneis, Roland Mestel, Ryan Riordan & Erik Theissen (2018)). Lin William Cong, Zhiguo He & Jiasun Li (2019) analyze the incentives for miners to form pools to tradeoff risky mining against the amount pools charge to miners. Other research focuses on the pricing of crypto-currencies in the market including frictions causing pricing differences (e.g. Albert S. Hu, Christine A. Parlour & Uday Rajan (2018), Igor Makarov & Antoinette Schoar (2018), Kyoung Jin Choi, Alfred Lehar & Ryan Stauffer (2018)) .

2 The Bitcoin Protocol: Block Capacity, Usage and Fees

A Bitcoin transaction comprises inputs and outputs. Once a transaction is submitted, it is broadcast to the network. Then, each node determines if the transaction is valid (i.e., the Bitcoin have not already been spent elsewhere) and stores it, awaiting execution (also known

as mining). Each node keeps an inventory of valid transactions that have not yet been mined called the "mempool." It is important to note that the mempool is not a centralized entity but rather is specific to each node (which typically have different capacity RAM). Each time a node competes to mine a new block, it selects valid transactions from its own mempool to work on. Once a block is mined, the output is then broadcast to the network and all nodes remove the relevant transactions from their own mempools. Output in a mined block consists of a Bitcoin amount that is deposited to an address using a public key.

Miners create blocks by being the first to solve a computational puzzle and are compensated in two ways. First, they receive a mining or block reward, which was set at ₿50 initially and is cut in half every 210,000 blocks (roughly every four years). The block reward appears as the first transaction in each block and is also called the coinbase. Second, and more germane to our analysis, participants who want their transaction to be included in a block can offer fees to miners. Fees are offered implicitly as the difference between inputs and outputs. For example, a submitted transaction might call ₿2.2 as input but only assign ₿2.18 as output. Miners retain the difference and pay it to themselves as part of the coinbase if they successfully mine a block.

2.1 The Data and Stylized Facts

Our sample comprises all blocks from the Genesis block (January 3, 2009) to block number 548,684 (Nov 4, 2018) and includes 903,976,764 inputs and 961,308,921 outputs. For each block we observe the coinbase, the inputs and outputs (and hence fees paid to the miners), and the size of each transaction. While bytes is the usual metric to determine size, some nuances of the BitCoin protocol mean that it is more useful to describe the "weight" of transaction. We provide a detailed discussion of this below.

There is a technological limit on the number of transactions that can be processed in a block. In the original design, Satoshi Nakamoto introduced a 1MB limit to Bitcoin blocks in 2010. However, for technical reasons, block size was limited by the number of database locks required to process one (at most 10,000). This limit translated to around 500-750k in serialized bytes, and was forgotten until March 11, 2013, when an upgrade to V0.8.0 with a switch of databases caused an unplanned fork in the blockchain. After resolving the crisis, the community reached a consensus to remove this unknown limit and a hardfork was scheduled and cleanly activated on May 15, 2013. Subsequently, for the first time, 1MB became the effective maximum block size.[1]

Another upgrade, Segregated Witness (Segwit) was implemented on August 24, 2017. The Segwit is a way to store signatures and scripts associated with compliant transactions in a special area of a block (the witness section). The first block exceeding 1MB limit was block 481,947 mined on Aug 25, 2017 with a size of 1032 KB. Figure 1 shows the evolution of block size measured in bytes over time.

It is important to note that post Segwit size in bytes is neither an accurate measure of block capacity nor of transaction size. A fully compliant Segwit transaction takes about 25% the space

[1]Details of this system change are available at `https://en.bitcoin.it/wiki/Block_size_limit_controversy,https://blog.bitmex.com/bitcoins-consensus-forks/`

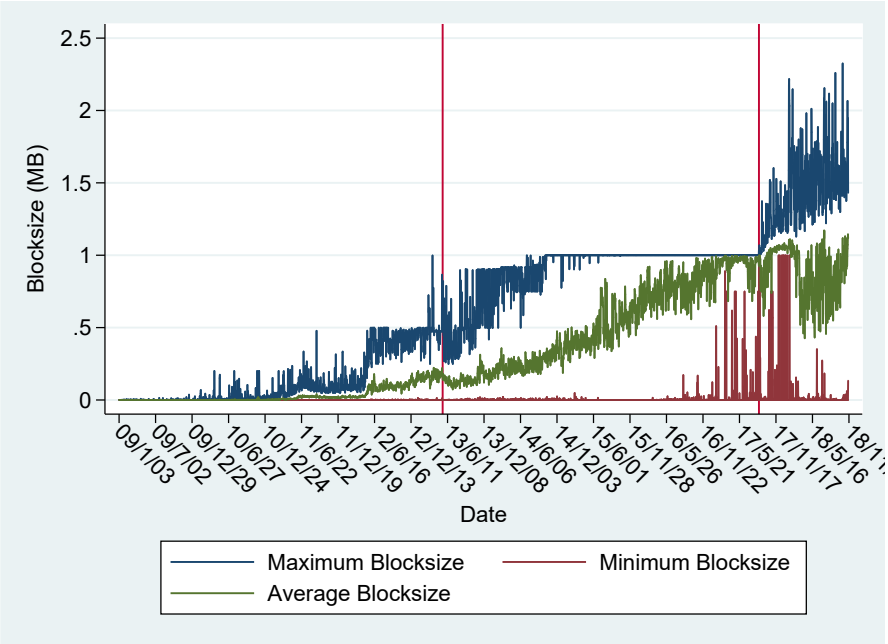

Figure 1. Minimum, average, and maximum daily block size. The red bars mark the safe implementation of the 1MB block size limit and the Segwit update, respectively. Days are defined over UTC.

(in bytes) of a traditional transaction because some components such as scripts are outsourced to the witness area and thus not part of the official block. However if coins are spent from an address locked up before the roll-out of Segwit, the full features of Segwit cannot be used and hence such transactions cannot take full advantage of the capacity increase. In short, they take up more space. The Segwit did not quadruple capacity: As pre-Segwit transactions are replaced with Segwit compliant transactions, the block capacity will gradually increase. To deal with this heterogeneity in transaction types, the concept of transaction weight was introduced with Segwit. Unless otherwise stated, in the rest of the paper, when we refer to post Segwit size or block capacity, we work with these weights which reflect true capacity utilization. In Appendix B we provide further institutional details on the measurement of block capacity post Segwit.

There is no difference in difficulty between mining a full and an empty block (empty spaces are also data). Further, for mining pools, there is an advantage to mining a non-empty block as these prevent free riding. This is because, to prevent any one pool member from claiming the whole reward, the pool coordinator does not disclose a full list of all transactions that should be included in a candidate block. Instead, pool participants get aggregate transaction information. Once a miner finds the right solution she has an incentive to report it to the pool as the the pool coordinator's information is required to claim the reward. Empty blocks do not allow the pool coordinator to withhold information and therefore increase the moral hazard problem. (We also note that empty blocks are sometimes mined with malicious intent to slow down transactions in Bitcoin, as in the fight between two competing BitcoinCash developer groups in November

2018, but these events are rare.[2])

Figure 2 illustrates the fraction of total blocks per day that are mined either completely full or empty as well as the used capacity per block (taking the possibilities of Segwit into account). Even though it seems as if the bitcoin blockchain ran at full capacity during the peak time of the bitcoin price it did not. Spare capacity exists because of empty blocks and blocks that are not filled to capacity.

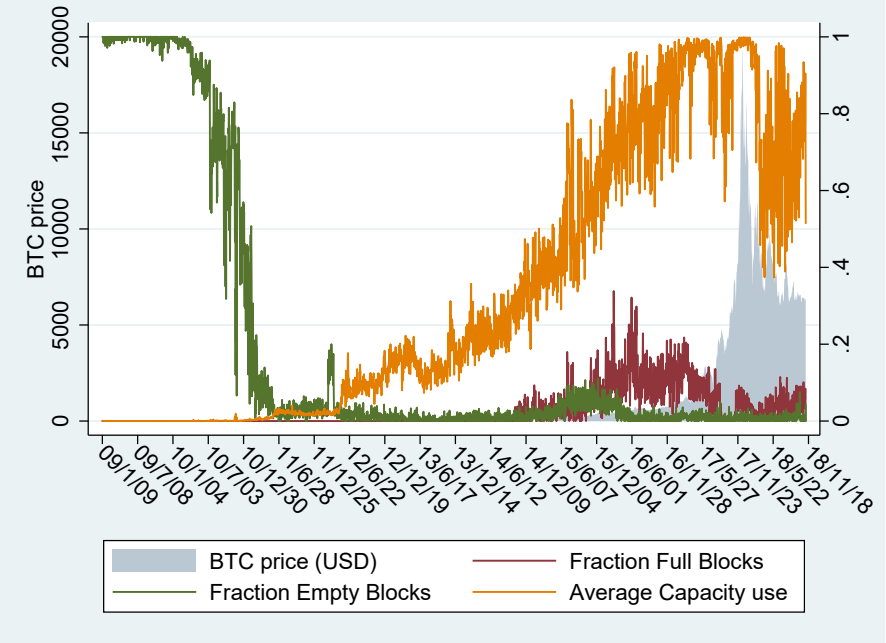

Figure 2. Number of blocks and fraction of empty and full blocks per day. Days are defined over UTC.

On Dec 17, 2017, for example, when Bitcoin was trading at a record price over USD 19,000, block 499704 and 499763 were mined empty by BTTC pool. On the same day, the same pool also mined 5 non-empty blocks making a technical problem unlikely. We can also rule out lack of transaction demand. Using mempool data as described in greater detail below we find that in the one hour interval around the time that block 499704 was mined more than 130,000 transactions were waiting to be mined. Over December 2017, 40 empty blocks were mined, out of which 39 were mined by 11 different identifiable mining pools. The largest pool in terms of share of mined blocks, AntPool, also mined the largest number of empty blocks.[3] We find that for all blocks mined in or after 2016 about 1.1% of all blocks mined by a given pool are empty. In unreported results we find a similar magnitude for all of the top five mining pools in our

[2]On November 15, 2008 the Bitcoin Cash blockchain forked in two parts as competing developer groups fought over the future development of the crypto-currency. Craig Wright, representing one group threatened to halt transactions in the competing fork by constantly adding empty blocks to the competitor's blockchain, therefore increasing processing times for transactions, thus making the other crypto-currency worthless.

[3]AntPool and BTTCPool mined 9 each, 6 are from BTC.top, 4 from 58coin, and BTC.Com, 2 from Slush pool, and one by 5 smaller pools. Ant pool and BTC top where also the largest mining pools in Dec 2017 with a share of 18.53% and 13.75% of all mined blocks, respectively.

sample. Since these results are consistent over time and across pools it is unlikely that they are due to random technical problems.

The second source of spare capacity are blocks that contain some transactions, but are also not filled to capacity. Excluding empty blocks documented above on the most congested day of our sample, August 22, 2017 (before Segwit), there was room for an additional 625 transactions. On December 17, when Bitcoin peaked, excluding the empty blocks, there was room for an additional 4,957 Segwit compliant transactions or 1,175 non-Segwit transactions. (We base our calculation on the median transaction size of a weight of 250 for Segwit and 1,000 for non-Segwit compliant transactions.) Together, the empty blocks and the empty capacity in filled blocks leave space for several thousand transactions each month. In the blockchain's busiest month so far, December 2017, on average room for 25,839 Segwit compliant transactions was left empty each day. For the broader sample since Jan 1 2014 on average there was empty space for 598,190 transactions per day.

To evaluate how efficiently miners use available capacity, we first quantify how much money miners could have made if they had filled up all blocks. To do so, we use a minute by minute mempool data set, which comprises summary statistics on the transactions that have been verified and are waiting to be included in a block.[4] This mempool is not empty: We find that on over 90% of the days in our sample all the excess capacity could have been filled with transactions from the mempool. Our measure of mempool size is conservative as we only keep observations immediately after a block was mined. Our analysis therefore represents that part of his mempool that was not picked up when miners confirm transactions in a new block.[5] In our sample the minimum size of the mempool was 112 KB, at any time there were at least 214 unconfirmed transactions in the mempool.

Combing the excess capacity per block with information in the mempool about the latent demand, allows us to calculate the direct cost of leaving transactions unmined in the mempool. We define 'money left on the table' as the additional fee revenue that could be obtained from filling up blocks with the transactions that offer the highest fees/byte and were in the mempool at the time that the block was mined. Figure 3 plots the total money left on the table by miners per day. The total amount of fees that miners chose not to pick up is not only determined by capacity utilization. Periods of relatively low excess capacity such as in December 2017 also coincide with miners leaving relative large amounts of money on the table because during this time transactions that waited in the mempool to be mined offered very high fees. Ignoring such transactions is therefore very costly for the miners. During periods with plenty of spare block capacity such as in late 2016 or around April 2018, many extra transactions could have been picked up by the miners but the price per space was low. We observe that miners leave money on the table throughout the whole sample period and not just in a specific period.

[4]The data are from Jochen Hoenicke's website, `https://jochen-hoenicke.de/queue/#0,all`, which contains minute by minute snapshots of the mempool. Transactions are grouped into fee buckets based on sat/byte.

[5]We go through the minute by minute snapshots of the mempool and look for instances where the size and the number of transactions in the pool drop and identify these drops as blocks being mined. We cannot reconcile blocks based on the timestamp as timestamps of blocks are inaccurate. Because mempool data is specific to each node as transactions are shared via peer-to-peer communication our data might have at a given time more or less transactions in the mempool than other nodes.

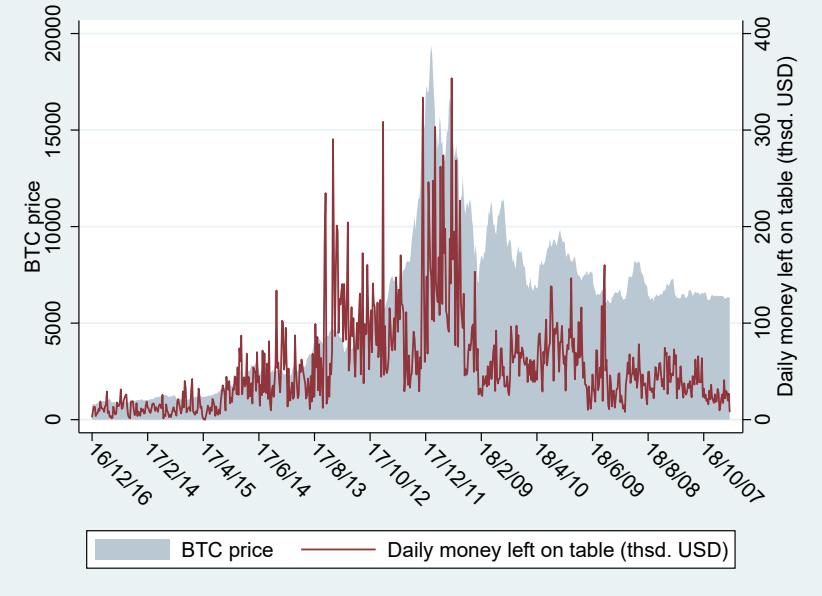

Figure 3. Money left on the table by miners and BTC price. We fill up empty capacity on the blockchain with unmined mempool transactions offering the highest fee/byte.

3 Optimally "leaving money on the table"

To motivate our framework, we present a simple model of on-chain settlement. The population comprises two types of bidders, those with high valuations for settlement v_h, and those with low, v_ℓ, where $v_h > v_\ell > 0$. These agents also differ in the cost incurred for every block that the transaction is delayed, where $c_h > c_\ell$, and we normalize c_ℓ to zero. We refer to the pair $\{c_i, v_i\}$ as the type of the agent i, where $i \in \{\ell, h\}$. For compactness we sometime denote that $c_h - c_\ell = \Delta_c$ and $v_h - v_\ell = \Delta_v$.

Each agent arriving at the market has the same outside option, $\bar{v} > 0$, which determines his willingness to participate. This is the expected utility any agent participating in the Bitcoin system obtains if he chooses his own optimal alternative.

We consider an infinite horizon, discrete time model. Here, one period corresponds to one mining block, so the culmination of each period is the addition of another block to the chain. The sequence of events within a period is as follows: first, λ_i agents of type $i = h, \ell$ arrive, and then submit their transaction (normalized to 1) and bid b_i to the network. Transactions and bids are then submitted to a common mempool and the transaction servicer then picks the bids to process.

Each block has a fixed capacity of κ. We assume that $\lambda_h < \kappa < \lambda_h + \lambda_\ell$, so that there is congestion in the blockchain and some types will have to be rationed. (We note that if there is no congestion, in the case in which the servicer exhausts capacity, the optimal bid is always 0.)

In reality, each miner keeps track their own mempool. We abstract from this and consider

a common mempool. Unprocessed bids remain in the mempool, as agents cannot revise or cancel their bids. However, as a practical matter, a mempool does not grow without bound because each mining pool flushes its mempool depending on its local conditions and capacity. For modeling purposes, we assume that the common mempool is flushed after ever period, but that agents whose transaction was not settled automatically resubmit their transaction at the same price. Thus, the arrival rate λ_i should be understood to include new transactions and those that were submitted earlier but have not been executed. The timeline for one period is presented in Figure 4 below.

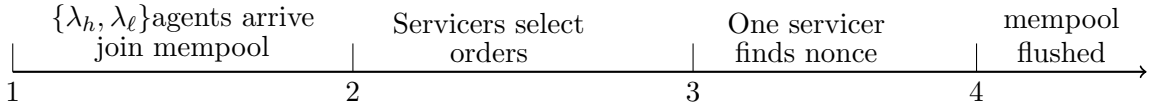

Figure 4. Sequence of Events within a block mining period

We characterize time invariant outcomes both because agents (and econometricians) do not observe the contents of the mempool, and the time between blocks – on average 10 minutes – probably does not admit time trends. We note, however, that in our empirical section, we use the random time between blocks as a proxy for congestion. In our base model, the arrival rates of different types are exogenous.

Let p_i be the time invariant probability that a transaction with bid b_i is executed in the current block. Then, an agent with cost c_i and valuation v_i, who submits a bid of b_i that executes after k periods, garners expected utility of $v_i - b_i - c_i(k-1)$. The costs of waiting are illustrated in Figure 5 below.

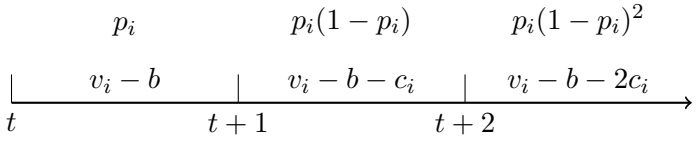

Figure 5. Probability of execution and payoff for an agent with type $\{c_i, v_i\}$. Here t refers to the block height.

Lemma 1 *The expected utility of an agent who submits a bid b_i and anticipates an execution probability p_i is*

$$V(b_i \mid c_i) \quad = \quad (v_i - b_i) - c_i \frac{1-p_i}{p_i}. \tag{1}$$

Agents take the execution probabilities as given. However, the execution probabilities depend on the actions of the miners, i.e., those who select transactions out of the mempool to process. We characterize two types of liquidity provision, first in which miners execute all orders by price priority and operate at full capacity, and second in which miners execute orders by price priority and ration strategically.

3.1 Different Rationing Rules

We know from the utility function of the consumer, and the participation constraint that the bid is

$$b_i^* \quad \leq \quad v_i - \bar{v} - c_i \frac{1 - p_i}{p_i}.$$

Further, because $v_h > v_\ell$ and $c_h > c_\ell = 0$, we know that the type h has an incentive to bid higher than the type ℓ, and so we consider the minimum bid at which the type h agents execute with probability 1 and the type ℓ are rationed. Let the superscript c denote bids, and execution probabilities if the servicer exhausts all available capacity.

Proposition 2 (Competitive Servicing) *If the servicers execute by price priority and exhaust all capacity then*

i. Type h bids no more than $b_h^c = \lim_{\epsilon \to 0} v_\ell - \bar{v} + \epsilon$, and executes with probability $p_h^c = 1$.

ii. Type ℓ bids at most $b_\ell^c = v_\ell - \bar{v}$.

If the servicer commits to exhaust all capacity, then the execution probability of the lower bidder type is determined by the arrival of traders and block capacity. The high types bids just enough to ensure that their orders will always execute. We note in passing that although we have continuous prices, in a model with arbitrarily small prices, the high types would bid the smallest price increment above the lowest types. In short, the high type retains rents and enjoys a surplus by participating in the system.

By contrast, if the servicer strategically chooses the execution probabilities, he can induce higher bids from agents who face higher costs. Specifically, the servicer only executes b_h^d with probability 1, and all other bids with probability $p_\ell^d < 1$.

Proposition 3 (Discriminatory Servicing) *If the servicers execute by price priority and ration, then*

i. Type h bids $b_h^d = v_h - \bar{v}$, and executes with probability $p_h^d = 1$.

ii. Type ℓ bids $b_\ell^d = v_\ell - \bar{v} - c_\ell\left(\frac{\Delta_v}{\Delta_c}\right)$ and executes with probability $p_\ell^d = \min\left[\frac{\Delta_c}{\Delta_v + \Delta_c}, \kappa - \lambda_h\right]$

iii. The profits of the servicer are $\pi^d = (v_h - \bar{v})\lambda_h + \frac{(v_\ell - \bar{v})\Delta_c - c_\ell \Delta_v}{\Delta_v + \Delta_c}\lambda_\ell$

.

These two different types of servicing strategies will generate different observables. First, it follows immediately from inspection of the two propositions that the maximum bid observed under discriminatory servicing is higher than that under full capacity servicing.

Corollary 1 *The maximum bid observed under discriminatory servicing is higher than the maximum bid under capacity servicing.*

We can therefore expect a wider dispersion in fees under discriminatory pricing than under competitive mining. Furthermore we can expect this price difference to increase when transactions become more valuable for the high type. Under discriminatory pricing we therefore expect large contemporaneous variations in fees and we expect that users who derive greater utility from getting their transaction executed to pay higher fees.

Second, under competitive servicing we should either see full blocks with positive fees when capacity is constrained, or we should see partially empty blocks and zero fees in case of excess capacity. Neither is consistent with the stylized facts presented above. What we see in the data, partially filled blocks with positive fees, is consistent with discriminatory pricing.

Finally our model is straightforward to extend to many types. No matter how many types we have under competitive pricing we should only have two outcomes: types above a certain reservation utility should be executed with probability one and all pay the same fee while all other types should be rationed if capacity constraints exist. Under discriminatory pricing a miner would offer a menu of fees and execution probabilities to price discriminate between users. Anecdotal evidence is consistent with the latter prediction. Online fee calculators like the one shown in Figure 6 provide users with a real time estimate on the fee they have to post to be confirmed with a 90% probability within 1,2,3,4,5, and 6 blocks, respectively.[6]

We will present more empirical evidence below but first we analyze how mining pools support collusive equilibria.

3.2 Sustaining Discriminatory Pricing with Competitive Miners

Suppose that discriminatory pricing is optimal. It is straight forward to show that even a decentralized financial system such as the BitCoin protocol can lead to discriminatory pricing equilibrium. Suppose that there are $i = 1, \dots N$ servicers of varying size. Let χ_i denote a servicer's hash rate as the proportion of the total Bitcoin system hash capacity. We assume that this is also the probability that a servicer wins the nonce.

Consider the payoff to servicer i. Mining garners a block reward in addition to fees. For simplicity, we normalize the block reward to zero as this is awarded mechanically. Under pricing regime $j = c, d$, a servicer obtains a per block profit at time t of π_t^j. Thus, under either regime $j = d, c$, the servicer anticipates from the next block onwards of

$$\Pi_i^j = \sum_{t=1}^{\infty} \delta^{t-1} \chi_i \pi_t^d \qquad j = d, c. \tag{2}$$

Thus, a servicer who is adhering to discriminatory pricing and wins the nonce receives a payoff

[6] See for example `https://www.buybitcoinworldwide.com/fee-calculator/` or `https://bitcoinfees.github.io/#30m`.

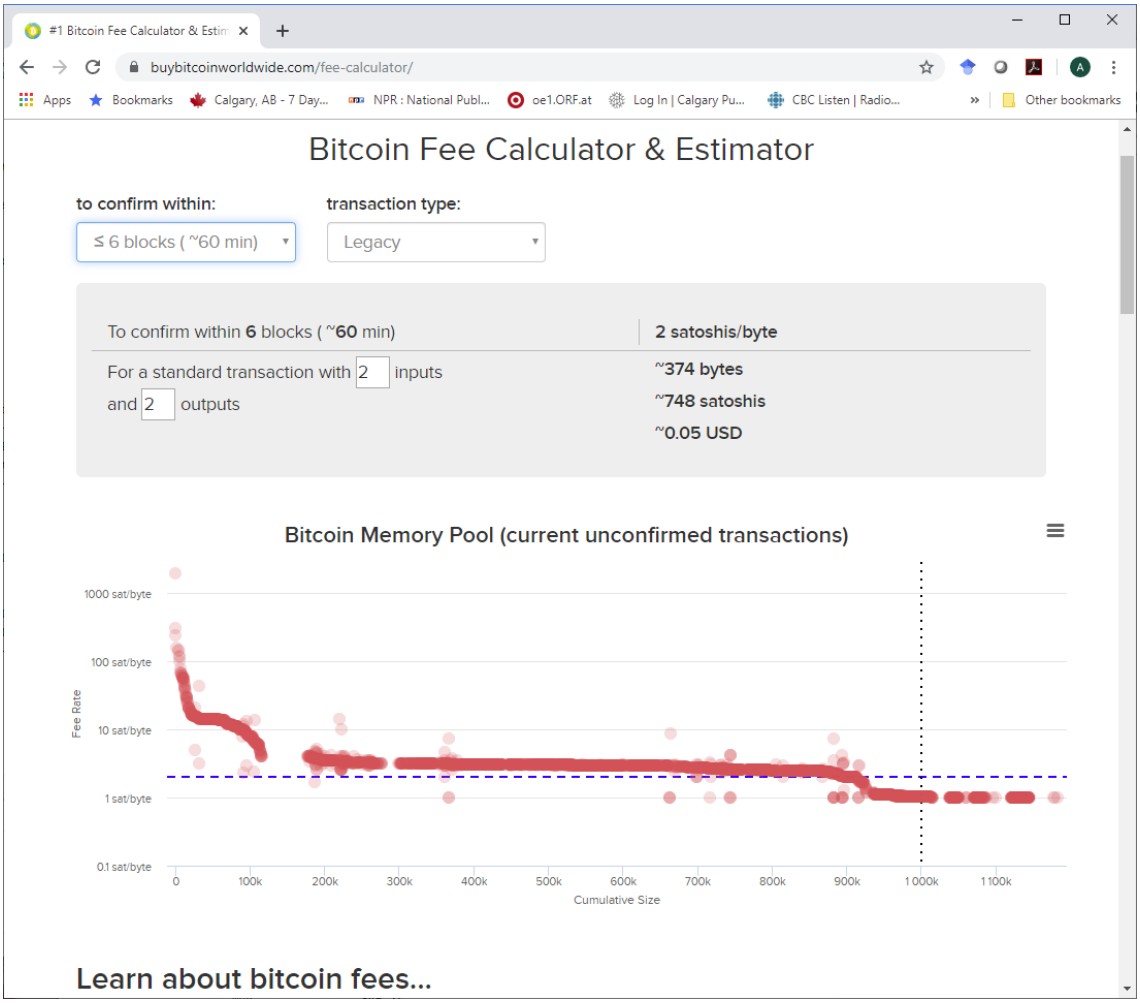

Figure 6. Screenshot from a BitCoin fee calculator that provides real time estimates of fees that users have to offer to get confirmed with 90% probability within 1,2,3,4,5, and 6 blocks, respectively.

of:

$$\Pi_i^{coop} \quad = \quad (v_h - \bar{v})\lambda_h + \Big((v_\ell - \bar{v}) - c_\ell \frac{1 - p_\ell^d}{p_\ell^d}\Big)\lambda_\ell p_\ell^d + \delta\Pi_i^d$$

Discriminatory pricing requires each miner to refrain from executing low bids to ensure that their expected execution probability is sufficiently low that the agents who value a rapid settlement are willing to bid aggressively up to their valuation. A miner who deviates would include too many transactions from the low bid types. If he finds the nonce, this would become public. Full capacity mining in perpetuity is a natural punishment strategy. The maximum profit to deviating is thus:

$$\Pi_i^{dev} \quad = \quad (v_h - \bar{v})\lambda_h + (\kappa - \lambda_h)\Big((v_\ell - \bar{v}) - c_\ell \frac{1 - p_\ell^d}{p_\ell^d}\Big) + \delta\Pi_i^c \tag{3}$$

It is immediate in this framework that if the payoff to discriminating is higher than the payoff to full capacity mining, then full capacity as a punishment strategy will sustain the discriminatory pricing outcome. The condition to sustain collusive outcome is

$$\Pi_i^{coop} \geq \Pi_i^{dev} \tag{4}$$

$$\delta(\Pi_i^d - \Pi_i^c) \geq \Big((v_\ell - \bar{v}) - c_\ell \frac{1 - p_\ell^d}{p_\ell^d}\Big)\Big(\kappa - \lambda_h - \lambda_\ell p_\ell^d\Big) \tag{5}$$

As with most collusive equilibria, the fewer the participants, the easier it is to sustain collusive equilibria. In this framework, fewer participants is equivalent to a higher probability that a particular miner or mining pool finds the nonce and is successful. This suggests that an effect of mining pools is to make collusive equilibria much easier to sustain. To see this note, the continuation payoffs in equation 5) are scaled by the probability that a miner wins the nonce. Thus, to sustain collusion, a minimum mining capacity as a function of the total capacity is required. Finally, we note that the miners' habit of signing the blocks they mine ensures that other pools can easily verify that a pool did not exhaust capacity.

4 Empirical Analysis

4.1 Fee Heterogeneity

Figure 7 plots the time series of the median as well as the 25 and 75 percentiles of transaction fees in BTC starting in March 2015. We observe a higher variation of fees over time with the peak in December 2017 when Bitcoin prices peaked. The graphs for fees in USD and KRW are similar.

One of the key predictions of discriminatory service is a high dispersion in fees. Figure 8 illustrates the heterogeneity of the Bitcoin fee distribution in December 2017, which was the peak of Bitcoin prices in our sample. The left panel shows the lower end of the fee distribution documenting that a substantial fraction of transactions occurred at low or modest fees. The right panel details the upper end of the distribution documenting that as Bitcoin fell from its peak price some people were willing to pay enormous fees to have their transactions processed. On Dec 24, 2017, for example, the 99 percentile fee was USD 433.23 while the largest fee paid was USD 14,174.64. We note that there are 80 transactions in our sample with fees greater than USD 10,000, of which 51 occur between Dec 20, 2017 and Dec 24, 2017. However over these same five days 1,674,141 transactions were processed out of which 752 had no fee and 16,191 transactions are mined with fees less than USD 5.

This observation is consistent with discriminatory pricing where the transactions with the very high fees were executed immediately and those transactions with very small fees were queuing in the mempool for a longer period in time. We cannot test this as the time that a transactions was first posted to the mempool is not recorded. However, if mining was competitive it is not

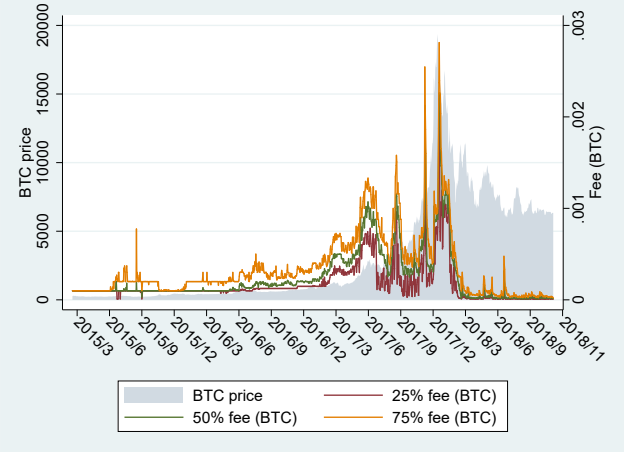

Figure 7. Bitcoin prices (gray area, left axis) and median, 25 and 75 percentile of daily fees (right axis) in BTC. Days are defined over UTC. Transactions exclude coinbase transactions

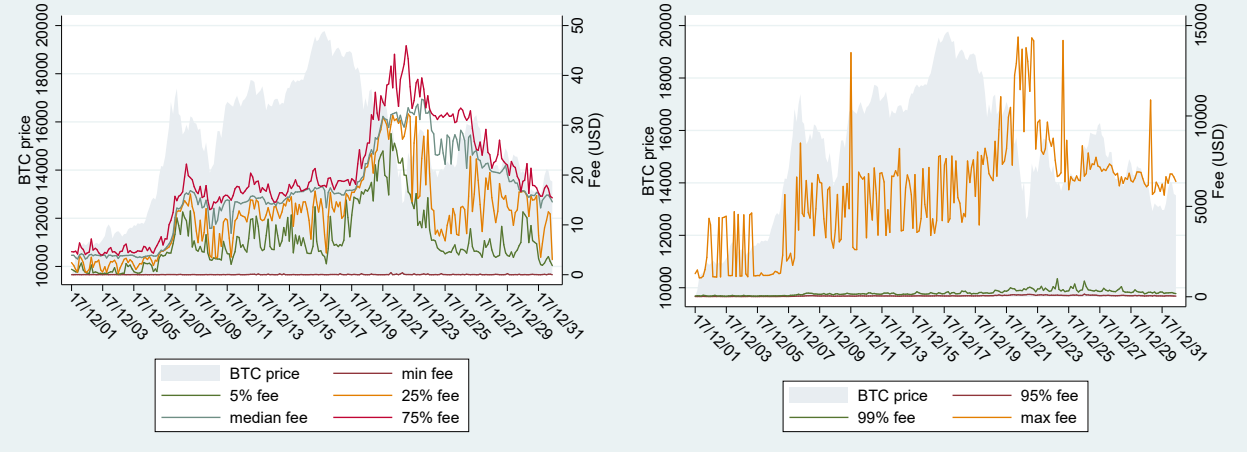

Figure 8. Fees in USD (right axis) and Bitcoin price in USD (left axis) in December 2017.

straightforward to find a rational reason why somebody would pay an excessive fee given that transactions at much lower fee levels were included in the most recent blocks.

4.2 Cross-sectional determinants

Our model predicts that under discriminatory pricing fees increase with the valuation of the high type, while under competitive pricing the valuation of the low type determines the fees. In this section we first analyze how transaction and blockchain specific variables drive fees and then provide evidence that is consistent with the idea that high value types pay higher fees.

Recall, that we posit two dimensions to liquidity, the cost of waiting per block and the valuation of having a transaction processed. We can identify various transaction attributes consistent with different demands for liquidity, for example, the speed with which the output from a transaction is used again, whether a transaction was simply used to insert data into the blockchain, the overall size of the transaction, and if the transaction can be attributed to financial trading.

We define minimum outtime as the time (in blocks) until the first output of this transaction is spent again. Recipient wallet owners that spend their funds very quickly have a more immediate need for funds and might thus put pressure on their debtors to send funds quickly. The Op-ret dummy is set to one for all data insertion transactions.[7] The Sum Inputs variable adds up all input values to a transaction. We choose inputs rather than outputs because some transactions, most of them data-insertion transactions, have almost all their inputs dedicated to fees and only have negligible output. Transaction size is the physical size of all inputs and outputs in bytes. The transaction size represents an opportunity cost for miners as block space is limited. Blocksize is the absolute size of the block in bytes. Larger blocks have less space for additional transactions and are an indication for transaction demand.

	Nobs	Mean	Std.Dev.	Median	Min	Max
Fee (Satoshi)	353,306,421	52,406	2,826,555	18,802	0	29,124,090,000
Fee (USD)	291,702,649	2.90	36.86	.20	0	137,186.10
Sumin (Satoshi)	353,306,421	1,370,000,000	40,500,000,000	19,800,000	0	55,000,000,000,000
Sumin(USD)	291,702,649	25,690.01	556,020.30	256.64	0	947,000,000.00
Blocksize (bytes)	353,306,421	834,072	325,063	998,126	267	2,324,736
Tx-Size (bytes)	355,725,559	537	2,210	250	62	999,657
dum-OPRet	353,306,421	.02	.15	.00	.00	1.00
Minouttime (blocks)	350,031,634	547.54	4,950.34	4.00	.00	474,195.00
Priceusd (USD)	291,702,649	3,604.66	4,162.71	1,188.28	204.84	19,828.76

Table 1. Summary statistics

Table 1 contains summary statistics. Some transactions contain unusually high fees which are either errors or payments miners make to themselves to consolidate their bitcoin balances together with the block award they earned under a new address. For example on April 26, 2016 someone paid BTC 291.241 to an output with BTC 0.0001, leaving fees of BTC 291.2409 (or USD 137,186.07 at the time) for the miner.[8] We found overall 80 transactions with fees over USD 10,000. These transactions have an average input of 140.89 BTC, and an average fee of 7.55 BTC or (USD 16,974.42 at the time). These transactions can be found between April 2016 and October 2018, but most of them are concentrated from Dec 21-24 in 2017. Bitcoin prices peaked on Dec 16 2017 at USD 19,200 and feel back to USD 15,191 on Dec 21. One potential explanation for these high fees is that investors wanted to move Bitcoin from their private wallet to an exchange quickly in a market panic. Transactions from the earlier days of Bitcoin may also show unusually high fees denominated in bitcoin. In December 2011 a transaction can be found with inputs of over 207 BTC and outputs of 36 BTC, leaving a huge fee for the miner. At that time bitcoin were worth very little, yet since there was no shortage of transaction space

[7] These non-financial transactions are used to store data from 2nd layer applications on the BitCoin blockchain.

[8] this is the highest fee transaction in BTC and USD terms in our sample, see transaction cc455ae816e6cdafdb58d54e35d4f46d860047458eacf1c7405dc634631c570d.

in the blockchain, there is no obvious rationale for paying such a high fee.

Many variables show some extreme observations. The average Bitcoin transaction was for 13.7 BTC (one bitcoin equals 100 million Satoshi) while the largest transaction was for 550,000 BTC on Nov 16, 2011 at a zero fee.[9] The largest transaction in dollar terms occurred on Dec 17th, 2017 at the peak of the bitcoin price when 48,500 BTC valued at over USD 946 million changed wallet for a fee of 80,908 Satoshi or USD 15.8.[10] In our sample we find 54,907 transactions with a value of more than USD 10 million. Out of those 20,956 were processed with a fee of less than USD 5 and the average fee of those transactions was USD 90.54.

Most transactions are small in size with a mean of 537 and a median of 250 bytes. The largest transaction in our sample consumes the entire block 364,292 with a size of 999,657 bytes. The transaction was mined on July 7th, 2015 and consolidates 5,569 inputs of 1,000 Satoshi each in a single output.[11] Minouttime measures the time (in number of blocks) until the first output of a transaction gets spent again. In some cases a transaction output gets spent in the same block that it is received, showing a minouttime of zero. The mining difficulty is set such that on average one block is mined every 10 minutes. We chose blocks and not calendar time as measure to make our findings robust to smaller variations in the time between blocks especially when the output is spent shortly after it is being received. For longer timespans the difference between calendar time and block time is irrelevant as blocks are mined on average every 10 minutes.

To purge our sample from extreme outliers we winsorize our fee data, the transaction size, the inputs, and the outtime at the 99.9% level. We find our results to be robust to different levels of winsorizing. Bitcoin fees also vary tremendously over time. To control for this time variation we include day-fixed effects in our regression analysis. To control for variation of fees across miners we cluster standard errors per block.

We first examine how transaction and blockchain specific explanatory variables influence transaction fees. Table 2 presents our findings for the fees in term of Satoshis (1 BTC is 100 million Satoshis) and Table 3 for fees in USD. In line with our intuition fees are higher when transactions space is scarce (larger blocksize in terms of weight) and when the transaction is larger in terms of bytes (Transaction Size), weight (Transaction Weight) and value (Sum Inputs). In the context of our model, this arises because more high fee transactions are in the mempool and therefore the block is larger.

Fees are also higher when the funds are spent sooner (lower Time until Spent) and especially if the output was spent in the next block. One possible explanation is that receivers who are in need for liquidity pressure senders to accelerate payments by offering higher fees. Another

[9] see transaction 29a3efd3ef04f9153d47a990bd7b048a4b2d213daaa5fb8ed670fb85f13bdbcf

[10] see transaction 261d69b25896034325d8ad3e0668f963346fd79baefb6a73b4eabd68c58c81ff

[11] According to some forum posts an unknown spammer created all these wallets to test the limits of the UTXO database. Somebody created many small outputs locked up under different addresses. To speed up the time it takes to verify the validity of a transaction each node holds in RAM memory a complete lust of all unspent BTC amounts per address. This list is called the Unspent Transaction Output (UTXO) database. The attack of the spammer forced each of the nodes to expand the UTXO list thereby increasing the memory requirement for each node. The Intentions of the spammer are unclear as all the transactions were protected by very weak private keys such as *'password'* or *'cat'*. Somebody eventually guessed the private keys and consolidated all the small inputs in block 364,292, freeing up space in the UTXO list.

	Whole Sample							w/o Peak
Block Size (Weight)	0.000765***						0.000591***	0.000611***
	(0.0000197)						(0.0000244)	(0.0000227)
Transaction Size		49.39***						
		(0.138)						
Transaction Weight			12.90***				12.92***	11.10***
			(0.0355)				(0.0356)	(0.0330)
Sum Inputs (Sat)				0.00000104***			0.000000561***	0.000000384***
				(8.65e-09)			(5.58e-09)	(4.42e-09)
OPRET					-10371.5***		-2107.3***	-2196.3***
					(232.5)		(231.6)	(223.9)
Spent next block						10049.4***	10417.5***	8580.7***
						(68.54)	(62.50)	(53.69)
Time until spent						-0.211***	-0.366***	-0.278***
						(0.00362)	(0.00420)	(0.00377)
R^2	0.0951	0.369	0.374	0.0973	0.0952	0.0957	0.376	0.345
Observations	353,306,421	353,281,736	353,306,421	353,306,421	353,306,421	350,031,634	350,031,634	320,966,799

Table 2: **Regression results of fees in Satoshis (1 BTC is 100 million Satoshis). Regressions include day fixed effects. Standard errors are clustered by block. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.**

possibility are arbritageurs who are keen to move Bitcoin from one exchange to another via a private wallet. Data insertion transactions (OPRET dummy) pay on average lower fees in BTC other users of the blockchain. While being statistically significant some results are economically insignificant. The value of the transaction, for example, only has a small impact on the fees, which is consistent with the fact that the miner's opportunity cost for including a transaction in a block of limited size is determined by the transaction size in weight and independent of the value. The coefficient on transaction weight is roughly one fourth that of transaction size in bytes because for all classic (i.e. before SegWit) transactions have by definition a weight equal to four times their size in bytes.

In Appendix A we provide more details on the determinants of transaction size. Transaction size has therefore an economically significant impact on fees. Adding one input (with an average size of 180 bytes) to a transaction (the average transaction has 2.5 inputs in our sample) increases the fee by 8,893 Satoshi or USD 0.41. Given that the median fee over the whole sample is USD 0.20 this effect is economically large. Results are mixed for the data-insertion (OP-Ret) dummy. In terms of BTC we see that data-insertion transactions post lower fees, yet when controlling for other variables we see that they pay a higher fee in USD. We attribute that to the increasing volume of data insertion transactions towards the end of our sample, where BTC prices were generally higher than at the beginning.

Outtime is measured in blocks, which are mined every 10 minutes on average. People spending their funds in the next block pay on average USD 0.577 more, and users spending a week ($\sim$1008 blocks) earlier pay on average 1.8 cent more in fees. Receivers that are keen to spend their coins sooner put a higher value on the execution. Consistent with discriminatory service those users pay higher fees as miners are able to price discriminate by delaying users that put a low priority on execution.

Our findings are not driven by the peak in Bitcoin prices around December 2017. The last column of each table shows the results for the subsample excluding all transactions between November 1, 2017 and January 31, 2018.

Next, we use the fact that different types of traders trade at different times. We argue that more sophisticated, high value, investors are active during the week and when the Bitcoin futures at CME are trading.[12] The results in Table 4 are consistent with discriminatory service. Transaction fees are lower on weekends by almost 20 cent, which is roughly equal to the median transaction fee for the whole sample. Looking at individual days of the week it seems that fees gradually increase during the work week with the highest observed fees on Fridays (which is also the settlement day for futures). Fees are also higher by about half a dollar, or twice the median fee, whenever the futures market is open (column (3)). To ensure that our findings are not driven by specific characteristics of CME futures trading hours we perform an event study using a 15 day window around December 18, 2017 in column (4). Fees in this short window were generally higher with a median fee of USD 15.6. We define *CME trading hours* as dummy set equal to one during the regular trading hours of Bitcoin futures at CME, *Post Dec 18th* is a

[12]BTC futures at CME are trading Sunday to Friday from 6pm to 5pm EDT with a daily one hour break between 5pm and 6pm EDT. Futures were first traded on December 18, 2017. Settlement is on the last Friday of the contract month. For this analysis we convert all block timestamps from UTC to Eastern Time considering summer daylight savings time when appropriate.

	Whole Sample							w/o Peak
Block Size (Weight)	2.48e-08***						2.24e-08***	2.18e-08***
	(1.07e-09)						(1.30e-09)	(1.08e-09)
Transaction Size		0.00233***						
		(0.00000989)						
Transaction Weight			0.000603***				0.000595***	0.000391***
			(0.00000256)				(0.00000252)	(0.00000184)
Sum Inputs (USD)				0.00000769***			0.00000615***	0.00000283***
				(6.65e-08)			(5.86e-08)	(2.14e-08)
OPRET					-0.368***		0.0463***	-0.0363***
					(0.0168)		(0.0152)	(0.00864)
Spent next block						0.636***	0.576***	0.298***
						(0.00646)	(0.00600)	(0.00247)
Time until spent						-0.0000146***	-0.0000178***	-0.00000728***
						(0.000000281)	(0.000000312)	(0.000000194)
R^2	0.274	0.400	0.400	0.286	0.274	0.275	0.408	0.265
Observations	291,702,649	291,678,054	291,702,649	291,702,649	291,702,649	288,555,299	288,555,299	259,490,464

Table 3: **Regression results of fees in USD. Regressions include day fixed effects. Standard errors are clustered by block. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.**

dummy euqal to one after December 18, 2017 and *CME Futures trading* is the product of *CME trading hours* and *Post Dec 18th*. We find that fees during CME trading hours are significantly higher once futures trading starts. This increase is robust to a general increase in fees after December 18, and with resepct to any other specific characteristics driving fees in these times.

	Whole Sample			Event
	(1)	(2)	(3)	(4)
Block Size (Weight)	1.27e-08***	1.76e-08***	2.02e-08***	0.00000786***
	(1.68e-09)	(1.59e-09)	(1.61e-09)	(0.00000266)
Transaction Weight	0.000595***	0.000595***	0.000595***	0.00264***
	(0.00000252)	(0.00000252)	(0.00000252)	(0.0000150)
Sum Inputs (USD)	0.00000613***	0.00000613***	0.00000613***	0.00000689***
	(5.91e-08)	(5.92e-08)	(5.92e-08)	(0.000000178)
OPRET	0.00511	0.00669	0.00242	1.142***
	(0.0181)	(0.0180)	(0.0180)	(0.276)
Spent next block	0.580***	0.582***	0.581***	2.564***
	(0.00679)	(0.00682)	(0.00681)	(0.118)
Time until spent	-0.0000182***	-0.0000182***	-0.0000183***	-0.000158***
	(0.000000323)	(0.000000323)	(0.000000323)	(0.00000601)
Monday	0.272***			
	(0.0205)			
Tuesday	0.380***			
	(0.0196)			
Wednesday	0.441***			
	(0.0188)			
Thursday	0.527***			
	(0.0196)			
Friday	0.643***			
	(0.0221)			
Saturday	0.485***			
	(0.0212)			
Weekend		-0.200***		
		(0.0120)		
CME Futures trading			0.492***	2.845***
			(0.0368)	(0.509)
CME trading hours				-1.356***
				(0.282)
Post Dec 18th				10.85***
				(0.430)
R^2	0.395	0.395	0.395	0.393
Observations	288,555,299	288,555,299	288,555,299	10,898,250

Table 4. Regression results of fees in USD - day of the week and opening hours of the futures market. Regressions include week fixed effects. Days are defined in UTC. Standard errors are clustered by block. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

Bitcoin is a pseudo-anonymous system, all wallet addresses can be identified and payments can be tracked moving from one wallet to another but the identity of the wallet owner is usually un-

known. In some cases, e.g. voluntary disclosure, court proceedings, the owner of some addresses becomes public. Gambling sites often use vanity addresses, such as '1dice....' or '1Lucky....' which are easily identified. Such vanity addresses have to be found by trial and error in a time consuming process and are therefore often reused.[13] Once an address is known, other addresses controlled by the same wallet can be inferred in a process commonly known as address clustering see e.g. Fergal Reid & Martin Harrigan (2013) or Sean Foley, Jonathan R Karlsen & Tālis J Putniņš (2018). The idea is that if multiple addresses are used as inputs in the same transaction these addresses most likely belong to the same person because the private key has to be used to sign the transaction.[14]

We use data from lists of known addresses such as wallet explorer and are able to identify the identity of the sender for 18,123,498 transactions, out of which 7,250,374 (or 2.05% of all transactions) were initiated by an exchange and 10,873,124 (or 3.07% of all transactions) were initiated by a gambling site. Similarly we are able to identify 32,771.960 payments to an exchange and 23,099,021 payments to a gambling site. Table 5 presents our findings for fees in USD. Flows to and from exchanges transact at higher than average fees. Since we control for day fixed effects our results cannot be driven by more exchange flows occurring on days when fees are generally higher. Our findings are also not driven by outliers since the data is windsorized. Transactions flowing out of exchanges pay USD 3.15 more, which is remarkable given that the median fee for the whole sample is USD 0.20. Fees into exchanges are paying USD 1.24 more than average. We argue that traders moving funds in and out of exchanges and gamblers put a high value on immediate execution. Consistent with discriminatory pricing, we observe that high types pay higher fees.

Another group of users with high valuations on execution are arbitrageurs who take advantage of cross exchange pricing differences. The arbitrageurs' preference for immediacy will be proportional to the price differential between exchanges as they need to process transactions on the blockchain to move Bitcoin from one exchange to the next. Under competitive pricing we should find that fees are independent of the value that the high types put on immediacy. Under discriminatory pricing we should find that miners can extract at least part of that surplus and therefore charge arbitrageurs a higher fee when the price differential is high.

To test this prediction empirically we take high frequency prices from Korea and the US and compute the Kimchi premium as the relative price difference of Bitcoin in the US and Korea at the block level. We interact a dummy for payments being made to and from wallets that can be identified as exchanges with the absolute value of the Kimchi premium and and find that fees made to exchanges increase in the Kimchi premium for these users. Since arbitrage profits are increasing in the absolute value of the price difference between BTC markets our evidence is consistent with discriminatory pricing and the idea that miners can extract more if the value of high types increases.

[13] Addresses are encoded in a Base58 alphabet (i.e. there are 58 possible 'letters' consisting of upper case, lower case letters and numbers with some combinations dropped that are often mixed up when printed on paper, e.g. capital i and lower case L) and start with 1. To get an address starting with '1Lucky' one has to try $58^5 \approx 656 million$ combinations. Vanity address companies offer computing resources to find custom bitcoin addresses.

[14] One notable exception are anonymizing services that for a fee combine transactions of several users into one large transaction so that it is not that clear who paid whom. See e.g. Malte Möser & Rainer Böhme (2017).

To Exchange	6.928***	3.973***		
	(125.78)	(104.91)		
To Gambling	0.392***	0.235***		
	(99.88)	(32.19)		
Block Size (Weight)		2.82e-08***		2.45e-08***
		(19.25)		(16.75)
Transaction Weight		0.000604***		0.000608***
		(237.63)		(236.52)
Sum Inputs (USD)		0.00000763***		0.00000774***
		(96.14)		(97.68)
OPRET		-0.224***		0.00812
		(-14.63)		(0.53)
Spent next block		0.616***		0.621***
		(89.28)		(91.25)
Time until spent		-0.0000171***		-0.0000164***
		(-53.47)		(-52.15)
From Exchange			2.198***	1.239***
			(156.89)	(124.17)
From Gambling			0.772***	0.524***
			(129.46)	(103.30)
R^2	0.268	0.408	0.264	0.406
Observations	294,586,994	291,439,644	294,586,994	291,439,644

Table 5. Regression results of fees in USD - day of the week. Regressions include week fixed effects. Days are defined in UTC. Standard errors are clustered by block. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

Our findings are in Table 6. Exchange is a dummy that identifies 35,163,580 payments to and from known exchange wallets, the Kimchi Premium is measured as relative price difference or the BTC price between Korbit in Korea and coinbase in the US, respecively. The interaction term of Exchange and Kimchi premium is statistically and economically significant. For an increase of the Kimchi premium by 10 percentage points users are willing to pay USD 3.22 more in fees. Our analysis most likely underestimate the importance that the demand for immediacy has for equilibrium fees for two reasons. First we can not identify all payments to exchanges. Observed high fees to exchanges might therefore coincide with similar high fee payments to unidentified exchanges making it harder to identify any effect in the data. Second, arbitrage between KRW and USD is only one of many potential trading strategies to exploit price differences, within one country, or between countries. We might therefore also observe high fees for payments to exchanges at times when two different markets have a large price difference which would not be captured in our regression. The results are robust to the introduction of Segwit and other control variables. Our finding cannot be driven by general variation in fees over time as we include day fixed effects.

In sum, fees are higher when the recipient spends the output faster, consistent with some liquidity need of the receiver. We find that transaction fees are higher on workdays and when CME bitcoin futures are trading consistent with the idea that miners can extract a portion of the higher value institutional investors put on execution. Using a sub-sample of identifiable wallets we find that payments to and from exchanges and gambling sites have substantially higher fees. We also find that the fees miners can extract from users making payments to exchanges increase in the

Kimchi Premium × exchange	40.20***	35.05***	32.46***
	(120.07)	(91.89)	(95.94)
Exchange		0.928***	-0.0962***
		(60.02)	(-7.10)
Kimchi Premium		-1.521**	-0.850
		(-2.23)	(-1.31)
Post Segwit			1.859***
			(4.66)
Block Size (Weight)			2.27e-08***
			(15.61)
Transaction Weight			0.000607***
			(237.15)
Sum Inputs (USD)			0.00000759***
			(97.90)
OPRET			0.0280*
			(1.84)
Spent next block			0.625***
			(91.43)
Time until spent			-0.0000159***
			(-51.11)
R^2	0.270	0.271	0.412
Observations	293,822,933	293,822,933	290,678,793

Table 6. Regression results of fees in USD including payments to exchanges and the size of the Kimchi premium. Regressions include day fixed effects. Days are defined in UTC. Standard errors are clustered by block. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

opportunities for cross country arbitrage as measured by the Kimchi premium.

4.3 Time Series evidence

In this section we analyze aggregated block level data to test some time series predictions of our model. Our model predicts that fee levels change in the value market participants put on execution but also on their opportunity cost and their costs of waiting. Since all of these parameters are unobservable we cannot use fee levels as explanatory variables.[15] Our model predicts that the dispersion of fees is higher under discriminatory service. We define feespread as the difference between the 90% and the 10% quantile of fees in a given block, standardized by the average fee. We choose this measure of fee dispersion over, say, a standard deviation, to reduce the influence of outliers

Discriminatory servicing is easier to maintain the higher mining concentration. We measure this by the market share of mining pools. To do this, we collect miners' signatures from each block's coinbase transaction. Unlike any other transaction on the bitcoin blockchain the bitcoin

[15] In the cross sectional regressions we control for time variation in values and costs by incorporating day fixed effects.

in the coinbase are newly created and therefore do not originate from another wallet. Miners use the space reserved for the input script to insert data into the blockchain. This space is used to send messages to the community like the miners' opinion on Bitcoin improvement proposals, it can contain other philosophical statements,[16] but most important for our purpose it usually contains a signature identifying the mining pool. We automatically search for commonly used signatures and then manually examine unidentified blocks for reoccurring signature patterns.[17]

We compute the daily Hirschman Herfindahl index (HHI) of mining concentration as the sum of the squared shares of each mining pool computed over the day where the block is mined. We also compute the mining pools' daily share of all mining activity. Table 7 presents our findings. We find that the feespread increases in both, the HHI and mining pool's aggregate share of mining activity. This finding is consistent with the idea that more mining by pools and more concentrated mining makes it easier to maintain the collusive discriminatory equilibrium in which the dispersion of fees increase within blocks.

HHI mining activity	3.274***	3.084***		
	(0.0380)	(0.0368)		
Fraction mined by pools			0.959***	0.996***
			(0.0130)	(0.0135)
Mined by pool	0.0329***	0.0298***	-0.0150***	-0.0124***
	(0.00456)	(0.00445)	(0.00482)	(0.00466)
Post Segwit		0.177***		0.171***
		(0.00302)		(0.00303)
Block weight		-3.85e-08***		-5.92e-08***
		(9.12e-10)		(9.54e-10)
Average tx weight		0.00000443***		0.00000475***
		(0.000000120)		(0.000000121)
Sum Inputs (USD)		-2.12e-08		-0.000000292***
		(4.81e-08)		(4.84e-08)
OPRET		1.074***		0.965***
		(0.0253)		(0.0254)
Time until spent		0.0000907***		0.0000758***
		(0.00000224)		(0.00000227)
R^2	0.0395	0.0996	0.0303	0.0927
Observations	198,111	198,105	198,111	198,105

Table 7. Regression results of fee spread per block defined as the difference of the 90% and 10% quantile over the average fee. HHI is the Hirschman Herfindahl index of mining pool activity and Agg Pool Share is the share of mining done by mining pools. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

The Bitcoin protocol calibrates the difficulty, i.e. the number of leading zeros that a block hash has to have to qualify as valid, in such a way that on average a new block is added every ten

[16] e.g. 'Welcome to the real world.' in block 328465, or 'smile to life and life will smile back at you' in block 328444, or 'the Lord of the harvest, that he send forth labourers into his harvest' in Block 143822.

[17] It is unclear why pools sign blocks. There is no clear gain in expected revenue from joining a larger pool. While larger pools are expected to mine a block more often, the reward has to be shared among a larger group. In the context of our model mining pools find it optimal to disclose their identity to show that they do not deviate from the collusive equilibrium.

minutes. Yet the times between blocks as they are recorded on the blockchain vary widely because mining a successful block is purely random and so sometimes blocks are found very quickly and sometimes it takes a long time. Another source of variation is that the time a block was mined is self reported by the miner and miners can have their local clocks not aligned with the world time. These clock mis-alignments can be substantial. Out of the 548,648 blocks in our sample we find 13,848 cases in which a block has an earlier time-stamp than its predecessor. This is technically impossible. Each block contains information from the previous block, which links the blocks together in a blockchain. The only rational explanations for the inconsistency in time-stamps is improper alignment of miners' clocks. To accommodate potential synchronization problems in miners' clocks the bitcoin protocol allows a block to have time-stamp up to two hours earlier than the previously mined block. We adjust for these mis-measurements heuristically by assuming that a block with an impossible timestamp has been mined half way between the two neighbouring blocks. Despite these problem cases for the vast majority of the sample the time-stamps seem to be properly recorded. Figure 9 illustrates the time between blocks. In the graph we focus on blocks after block 100,000 because dispersion in times between blocks was higher in the early days of bitcoin. Very few blocks have more than 50 minutes between them and these observations are omitted from the graph.

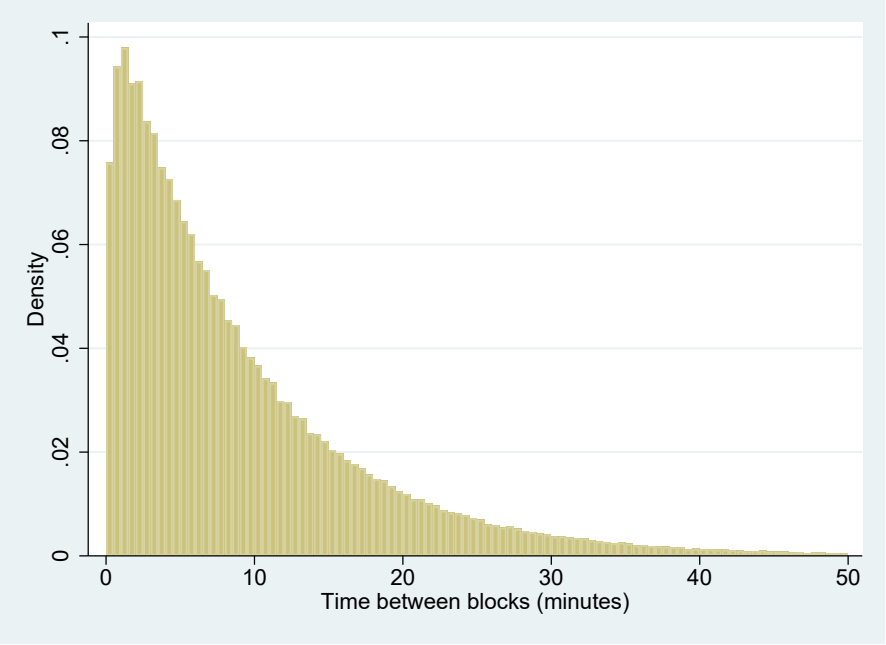

Figure 9. Histogram of time between blocks with blockheight larger than 100,000, capped at 50 minutes.

Price discrimination in our model works because miners can credibly delay low types. To keep the delay consistent miners can compensate for the variation in the arrival-time of blocks as documented in Figure 9 by adjusting block usage. Specifically we would expect that upon the random arrival of many blocks subsequent blocks would be filled up less because miners need to delay patient types. Similarly when few blocks arrived in the recent past we would expect

blocks to be fuller because more patient types have already waited their turn. Under collusion this effect should be more pronounced when mining is more concentrated. We test this intuition by regressing block-weight on dummies for terciles of the distribution of the number of blocks mined in the last hour. In column 2 we find that block weight increases when few blocks were mined in the last hour and that block weight decreases when many blocks were mined. When we interact the tercile dummies with the HHI of mining concentration we find a much stronger effect. Our findings are consistent with the idea that when miner collusion is strong capacity management my mining pools is more prevalent.

Low tercile recent blocks	83799.4	440940.1***	35386.9
	(61137.8)	(11970.6)	(68535.8)
HHI x low tercile recent blocks	3016703.1***		3429945.0***
	(525303.4)		(586159.6)
High tercile recent blocks	75236.4	-317545.1***	39521.3
	(106066.9)	(14592.9)	(114589.2)
HHI x high tercile recent blocks	-3295341.1***		-2894436.0***
	(875125.6)		(952862.4)
Sum Inputs (USD)	7.235***	7.224***	
	(0.834)	(0.838)	
OPRET	-1095141.8***	-1082040.9***	
	(251794.3)	(252135.6)	
Time until spent	293.9***	297.1***	
	(27.50)	(27.76)	
R^2	0.101	0.0974	0.0616
Observations	198,551	198,551	198,577

Table 8. Regression results of blockweight on the frequency of recent blocks. Standard errors are clustered by day. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

4.4 Economic impact of collusion

The total amount of fees paid in all transactions of our sample is USD 844,537,503.30. How much was extracted from miners via collusion is hard to estimate. According to our model the equilibrium fee under competitive mining should be below the value that the low type puts on execution. If there is no congestion, as it is for practically our whole sample, equilibrium fees should be zero without collusion. As conservative estimate of maximum fees under competitive mining we approximate the fees that the lowest type would be willing to bid with the 10%-quantile of the fee distribution per block. We then define excessive fees as the sum of all fees above the 10%-quantile. For the whole sample these excessive fees sum to USD 557,568,542.21.

To make the result robust to outliers we re-compute excessive fees and winsorize fees per block at the 99% quantile. With winsorizing, excessive fees for the whole sample add to USD 416,006,674.88. Overall we argue that excessive fees paid because of collusion are between half and two thirds of total fees paid. Figure 10 shows the daily ratio of excessive fees over total fees, bitcoin prices, and the mining concentration as measured by the HHI. Excess fees seem to

be positively correlated with mining concentration.

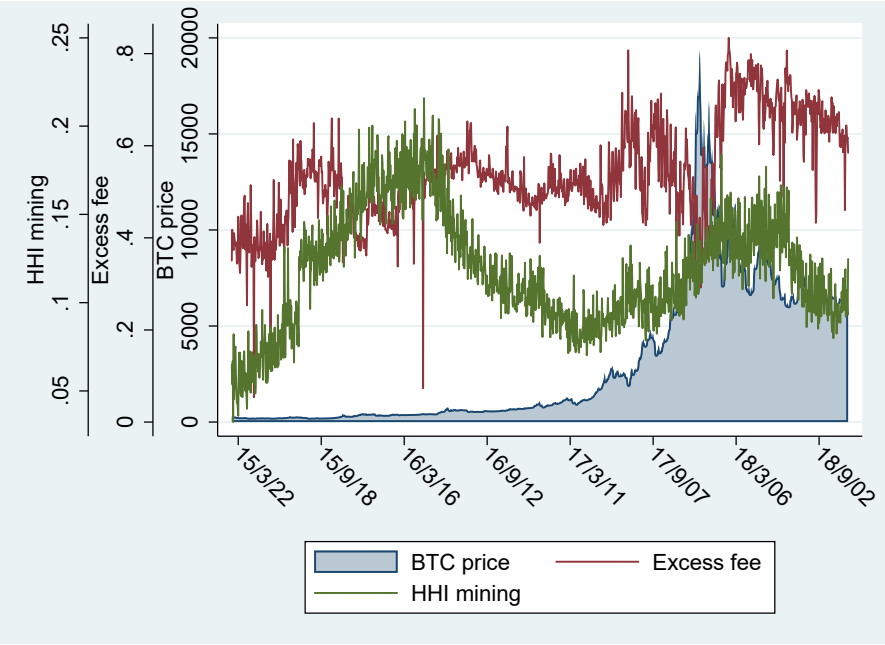

Figure 10. Daily excessive fees, mining concentration as measured by the HHI, and Bitcoin price. Days are defined over UTC.

5 Conclusion

We have documented stylized facts about the Bitcoin protocol. In particular, we observe that there appears to be excess capacity as a significant portion of blocks are empty or not at capacity. We note that this is consistent with collusive price discrimination. Indeed, the rise of fees coincided with the rise of mining pools. Given that the idea behind the bitcoin system was to provide a completely decentralized way of transferring value based on competitive mining, the possibility that there could be collusive equilibria raises questions about the viability of decentralized finance.

Our preliminary evidence suggests that one implementation of decentralized finance may operate in a way that is observational equivalent to traditional finance. Indeed, our analysis has highlighted the cost to the consumer of the higher fees they pay because of strategic unused capacity. However, we note that there is a positive side to these rents. Higher profits are one way to ensure that miners will view participating as a valuable exercise which ensures the continuity and stability of the BitCoin protocol. Similar to financial intermediaries, market power and the ability to extract rents provide an incentive to continue.

References

Abadi, Joseph, and Markus Brunnermeier. 2018. "Blockchain economics." National Bureau of Economic Research.

Brauneis, Alexander, Roland Mestel, Ryan Riordan, and Erik Theissen. 2018. "A high-frequency analysis of bitcoin liquidity."

Budish, Eric. 2018. "The economic limits of bitcoin and the blockchain." National Bureau of Economic Research.

Choi, Kyoung Jin, Alfred Lehar, and Ryan Stauffer. 2018. "Bitcoin Microstructure and the Kimchi premium."

Cong, Lin William, and Zhiguo He. 2019. "Blockchain disruption and smart contracts." *The Review of Financial Studies*, 32(5): 1754–1797.

Cong, Lin William, Zhiguo He, and Jiasun Li. 2019. "Decentralized mining in centralized pools." National Bureau of Economic Research.

Easley, David, Maureen O'Hara, and Soumya Basu. 2019. "From mining to markets: The evolution of bitcoin transaction fees." *Journal of Financial Economics*.

Foley, Sean, Jonathan R Karlsen, and Tālis J Putniņš. 2018. "Sex, drugs, and bitcoin: how much illegal activity is financed through cryptocurrencies?" *Review of Financial Studies, Forthcoming*.

Hu, Albert S., Christine A. Parlour, and Uday Rajan. 2018. "Cryptocurrencies: stylized facts on a new investible instrument." working paper.

Huberman, Gur, Jacob Leshno, and Ciamac C Moallemi. 2017. "Monopoly without a monopolist: An economic analysis of the bitcoin payment system."

Makarov, Igor, and Antoinette Schoar. 2018. "Trading and arbitrage in cryptocurrency markets." working paper.

Malinova, Katya, and Andreas Park. 2017. "Market design with blockchain technology." *Available at SSRN 2785626*.

Möser, Malte, and Rainer Böhme. 2017. "The price of anonymity: empirical evidence from a market for Bitcoin anonymization." *Journal of Cybersecurity*, 3(2): 127–135.

Reid, Fergal, and Martin Harrigan. 2013. "An analysis of anonymity in the bitcoin system." In *Security and privacy in social networks*. 197–223. Springer.

A Determinants of Btcoin Transaction Sizes

Transactions sizes are influenced by the number of inputs and outputs as well as by the type of the transaction. The Bitcoin eco-system allows several types of transactions from the simple pay to an address, to multi-signature transactions where, for example, two out of three people need to digitally sign a transaction for validation, to pay-to-script-hash transactions where the precise rules of redemption are not known at the time an output is locked but only the hash of the redemption script is recorded. Transaction size increases in the number of inputs and outputs with inputs contributing more since they have to hold the transaction signature and the public key. For basic transactions with i inputs and o outputs a common rule of thumb is that the transaction size in bytes is $180i + 34o \pm i$.[18]. To confirm this idea and to show that value does not drive transaction size we regress transaction size in bytes on the number of in, and outputs (numin and numout, respectively) as well as on the value of all inputs (sumin). The results in Table 9 document that the rough rule mentioned above is approximately true for our sample as well. We also find that that the value has no economically significant impact on the size of a transaction.

Numin	164.74***
	(0.06414)
Numout	37.781***
	(0.02734)
Sum Inputs	9.23e-12***
	(1.07e-12)
constant	19.617***
	(0.16118)
R^2	0.9394
N	350,008,404

Table 9. Regression of transaction size in bytes on the number of inputs, number of outputs, and the value of the inputs. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

B Block capacity post Segwit

A big part of the physical space that transactions take up in a block are the locking and unlocking scripts. Segregated Witness (Segwit) compliant transactions outsource these scripts into a separate data structure, the witness. The witness structure is organized as Merkel tree, a data structure where leaves hold data and each node is a hash of the underlying nodes. The root of the tree is linked to the Bitcoin block by including the root-hash in the coinbase transaction.

Segwit transactions are designed to be backward compatible. There are two basic types of Segwit transactions, Pay-to-Witness-Public-Key-Hash (P2WPKH) and Pay-to-Witness-Script-Hash (P2WPSH). In the former the locking script is marked as Segwit by including the Segwit

[18] see e.g. https://bitcoin.stackexchange.com/questions/1195/how-to-calculate-transaction-size-before-sending-legacy-non-segwit-p2pkh-p2sh

version number (currently 0) followed by a 20 byte hash of the public key. The signature and the full public key required for unlocking the Bitcoin are outsourced to the witness block. For P2WPSH transactions the locking script consists of the version number followed by a 32 byte hash of the unlocking script. These transactions are also often referred to as Bech32 transactions. A non-native and inefficient way of implementing Segwit transactions is to embed them in a classic Pay-to-Script-Hash (P2SH) transaction.

Outsourcing part of the transaction to the witness section reduces the amount of effective space a transaction takes up in the block. The measure of transaction size in bytes includes both the transaction and the witness data to make the measure comparable to pre-segwit transactions. For most purposes, e.g. to measure capacity use, the transaction size in bytes is not a useful measure because segwit-, partial segwit, and non-segwit transactions can be included in a block. To address this problem Bitcoin introduced a measure of transaction "weight." The weight of a transaction that does not take advantage of segwit is 4 times its size in bytes. The weight for a fully segwit compliant transaction is obtained by multiplying components that are part of the block (inputs, outputs, input- and output counts, version, and lock-time) by 4 and multiplying witness components by 1 and then adding up the weighted components. In our sample the weight is between 1.2 and 4 times the size in bytes.

The Segwit update did not have a big impact on variables of economic interest. The blue line in Figure 11 shows that the introduction of SegWit brought no immediate increase in capacity. The average weight per block stays between 3 and 4 MB, the latter being the maximum amount. The reason that Segwit brought no sharp increase unused transaction capacity is because of its slow adoption. The red line shows the fraction of transactions that use some Segwit features. Adoption is slow peaking at 15% after fifty days. With most transactions using the pre-Segwit format not much new capacity on the blockchain is being created. The green line illustrates the total fee revenue per block. While there is a peak around the introduction of Segwit the variation in fee revenue per block seems of similar or smaller magnitude than other variations in total fee revenue. It seems that there is no unusual variation in miners' fee revenue around the Segwit inroduction.

C Proofs

Proof of Lemma 1

The utility of an agent with cost c_i who bids b and executes in period k is $v - b(v, c) - c(k - 1)$. If the probability of execution is p_i, they obtain utility

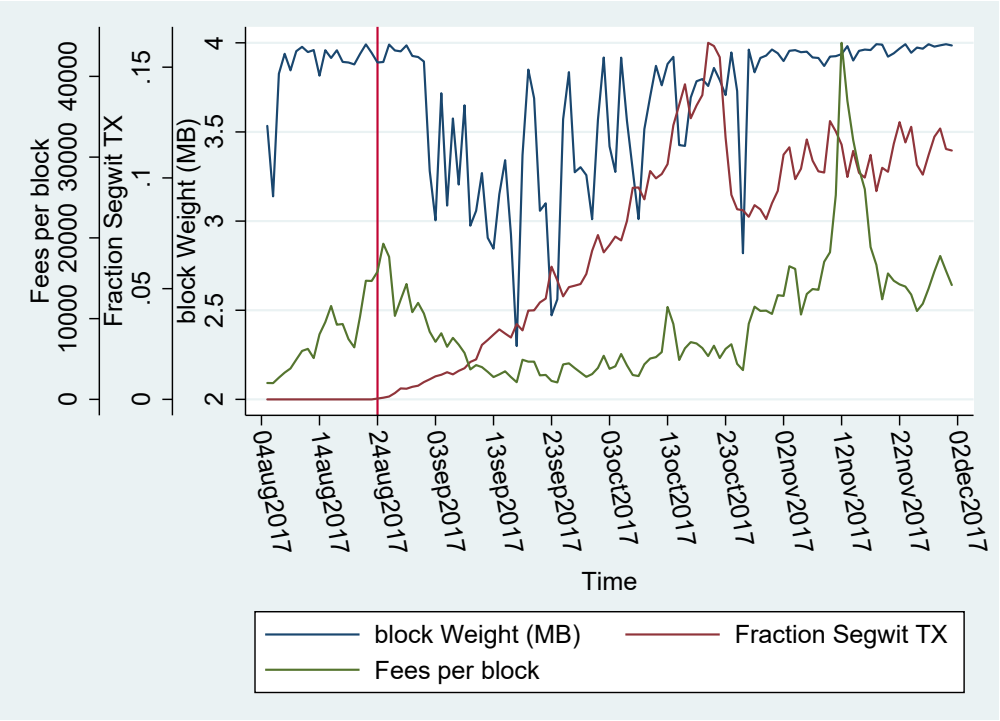

Figure 11. Average Weight per block, fraction of Segwit Transactions, and mining Revenue per block 20 days before to 100 days after the introduction of Segwit. Days are defined over UTC.

$$\begin{aligned} V_i(b) &= \sum_{k=1}^{\infty} p_i[1-p_i]^{k-1}(v-b-c_i(k-1)) \\ &= (v-b) - c_i p_i \sum_{j=0}^{\infty}[1-p_i]^j j \\ &= (v-b) - c_i \frac{1-p_i}{p_i}. \end{aligned} \tag{6}$$

■

Proof of Proposition 2

The high types incur a cost to delaying, and so they will always bid to ensure that their orders execute. We are looking for an equilibrium in which the high type executes with probability 1, and the low types randomize over a range. The utility of the low type is

$$V_\ell(b) = v_\ell - b - c_\ell \frac{1-p(b)}{p(b)}.$$

The highest bid a low type would make is $\bar{b_\ell^c} = v_\ell - \bar{v}$. Suppose that a fraction of at least $\frac{\kappa - \lambda_h}{\lambda_\ell}$ of the low types bid $\bar{b_\ell^c}$, then if an agent bids lower, her order will never execute. If she bids higher, she obtains negative utility. Thus, there is an equilibrium in which $[\frac{\kappa - \lambda_h}{\lambda_\ell}, 1]$ of the low types bid $\bar{b_\ell^c} = v_\ell - \bar{v}$.

Now, suppose that there is an equilibrium in which less than $\frac{\kappa - \lambda_h}{\lambda_\ell}$ bid $\bar{b_\ell^c}$. Agents have an incentive to reduce their bid to obtain positive surplus. Suppose all agents bid a maximum $b*$, which is strictly less than $\bar{b_\ell^c}$. An agent has a strict incentive to increase his bid and obtain execution for sure.

The high type will bid no more than $b_h^c = lim_{\epsilon \to 0}(v_\ell - \bar{v} + \epsilon)$ as at this price, he will obtain immediate execution.

Proof of Proposition 3

The high type bids $b_h^d = v_h - \bar{v}$ and is executed with probability 1.

The second highest type is executed with probability p_ℓ^d, which satisfies incentive compatibility, that ensures that type ℓ doesn't want to bid b_h^d.

$$\begin{aligned} v_\ell - b_h^d &\leq v_\ell - b_\ell^d - c_\ell \frac{1 - p_\ell^d}{p_\ell^d} \\ v_\ell - (v_h - \bar{v}) &\leq v_\ell - b_\ell^d - c_\ell \frac{1 - p_\ell^d}{p_\ell^d} \\ b_\ell^d &\leq (v_h - \bar{v}) - c_\ell \frac{1 - p_\ell^d}{p_\ell^d} \end{aligned}$$

Further, the highest type does not want to bid lower and experience waiting costs:

$$\bar{v} \geq v_h - b_\ell^d - c_h \frac{1 - p_\ell^d}{p_\ell^d} \tag{7}$$

$$b_\ell^d \geq v_h - \bar{v} - c_h \frac{1 - p_\ell^d}{p_\ell^d} \tag{8}$$

Subtracting these two equations from each other yields:

$$0 \leq (c_h - c_\ell)\Big(\frac{1 - p_\ell^d}{p_\ell^d}\Big)$$

Type	Bid	Execution Probability	Quantity
1	$v_h - \bar{v}$	1	λ_h
2	$v_\ell - \bar{v} - c_\ell\left(\frac{\Delta_v}{\Delta_c}\right)$	$\frac{\Delta_c}{\Delta_v+\Delta_c}$	$(\lambda_\ell)p_\ell^d$

Table 10. Bids and quantity executed if the servicer discriminates

As the RHS is strictly positive, these conditions are satisfied. To pin down b_ℓ^d, we note that

$$b_\ell^d = v_\ell - \bar{v} - c_\ell \frac{1-p_\ell^d}{p_\ell^d},$$

and use this in the incentive compatibility constraint Equation 7:

$$\begin{aligned}
\bar{v} &\geq v_h - b_\ell^d - c_h \frac{1-p_\ell^d}{p_\ell^d} \\
\bar{v} &\geq v_h - \left(v_\ell - \bar{v} - c_\ell \frac{1-p_\ell^d}{p_\ell^d}\right) - c_h \frac{1-p_\ell^d}{p_\ell^d} \\
0 &\geq \Delta_v - \Delta_c \frac{1-p_\ell^d}{p_\ell^d}
\end{aligned}$$

$$\begin{aligned}
\frac{1-p_\ell^d}{p_\ell^d} &\geq \frac{\Delta_v}{\Delta_c}, && (9) \\
\implies p_\ell^d &= \frac{\Delta_c}{\Delta_v+\Delta_c} && (10) \\
1-p_\ell^d &= \frac{\Delta_v}{\Delta_v+\Delta_c} && (11)
\end{aligned}$$

In this case, the flow revenue of the servicer is:

$$\begin{aligned}
\pi^d &= b_h^d \lambda_h + b_\ell^d p_\ell^d(\lambda_\ell) \\
&= (v_h - \bar{v})\lambda_h + \left(v_\ell - \bar{v} - c_\ell \frac{1-p_\ell}{p_\ell}\right) p_\ell^d(\lambda_\ell) \\
&= (v_h - \bar{v})\lambda_h + \left((v_\ell - \bar{v}) - c_\ell \frac{\Delta_v}{\Delta_c}\right)\left(\frac{\Delta_c}{\Delta_v+\Delta_c)}\right)(\lambda_\ell) \\
&= (v_h - \bar{v})\lambda_h + \frac{(v_\ell - \bar{v})(\Delta_c) - c_\ell(\Delta_v)}{\Delta_v+\Delta_c}(\lambda_\ell)
\end{aligned}$$

■