Miner Collusion and the Bitcoin Protocol

Alfred Lehar*
Haskayne School of Business
University of Calgary

Christine A. Parlour†
Haas School of Business
UC Berkeley

April 27, 2021
Comments Welcome

Abstract

Bitcoin users can offer fees to the miners who record transactions on 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 there appears to be excess demand. We argue that this is inconsistent with competitive mining, but is consistent with strategic capacity management. If agents believe that only high fee transactions are executed in a timely fashion then strategic capacity management can be used to increase fee revenue. We note that mining pools facilitate collusion, and estimate that they have extracted least 200 million USD a year in excess fees by making processing artificially capacity scarce.

Keywords: Bitcoin, Transaction Costs, Blockchain, Decentralized Finance

*Corresponding author, email: alfred.lehar@haskayne.ucalgary.ca, Tel: (403) 220 4567.

†email:parlour@berkeley.edu. We thank Christina Atanasova, Lin William Cong, Jaisun Li, Ye Li, Fahad Saleh, Asani Sarkar, Shu Yan, Peter Zimmerman, and seminar participants at CBER, SF BlockChain Week, IWFSAS, Philadelphia Fed FinTech Conference, Toulouse School of Economics, WFA 2020, EFA 2020, the China Fintech Research Conference, 2021, INFORMS 2020, the University of Graz, P2P Financial Systems 2020, Toronto Fintech Conference 2020, 3rd UWA Blockchain, Cryptocurrency and FinTech conference 2020, 4th SAFE Market Microstructure Conference 2020. Lehar is grateful to the Canadian Securities Institute Research Foundation for financial support.

1 Introduction

Bitcoin is the first successful proof of concept of a decentralized financial product. Understanding how the Bitcoin system works is thus important. In this paper, we investigate whether Bitcoin has evolved into a system that provides value transfer to consumers competitively, or if there is a concentration of market power. Bitcoin 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 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 first document that the Bitcoin blockchain rarely operates at full capacity. 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, even though there appears to be excess demand for transaction processing.

Fees began appearing in 2016, and over the course of sample close to 850 million USD have been paid by users of the system. We document the dispersion of fees both across time and within blocks. The existence of excess capacity, in the presence of fees, suggest miners appear to leave "money on the table." However, we argue that this is consistent with strategic capacity management designed to increase fee revenue. Mining pools (aggregations of miners) also began appearing in 2016. In as much as they restrict the number of independent players in the system, they facilitate strategic capacity management.

The aggregate effect of strategic capacity management is large. Consistent with a conservative interpretation, we define excess fees as those paid above the 10% quantile of the empirical daily fee distribution. These sum close to 560 million USD 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. Specifically, there is no commitment by miners to an allocation rule. 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.

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. Our specific aim is to understand if the Bitcoin system behaves as a competitive system or if it

yields outcomes consistent with strategic capacity management. We first observe that if miners process the highest bids first, we should not observe empty blocks or blocks processed at less than capacity if there are orders waiting. We conclude that processing does not follow highest bid first.

If there is strategic capacity management based on fees, then arriving agents effectively face a menu that relates fees to waiting times. Given the random arrival of agents, such a menu implies that some transactions will be processed even if transactions with higher fees attached to them remain in waiting. This is consistent with our data. If the purpose of strategic capacity management is to induce higher bids, then bids should be more dispersed. Intuitively, a menu that induces higher revenue will induce agents to pay fees higher than their unconstrained choice. This is consistent with our data.

If there are a large number of miners, the probability of "winning the nonce" and processing a block is small, individual miners have an incentive to fill up a block with all positive fee transactions, which would not induce strategic capacity management. 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 enhance revenue. 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.

In our sample of over 350 million observations, we document that users who are more likely to have higher valuations for consummated transactions are more likely to pay a higher fee, which is consistent with rent extraction. Specifically, transactions that are more likely to originate from institutional sources (proxied through day of the week), transactions that are more likely to be arbitrage trades (proxied by the Kimchi premium), transactions that involve gambling sites and exchanges and transactions associated with rapid redeployment pay higher fees. All of these effects are stronger when mining power is more concentrated, which suggests that the way in which transactions are processed affects bidding behavior.

To investigate the effect of mining pools, we find that fee dispersion is higher when mining capacity is more concentrated and more mining is done by pools. Consistent with strategic capacity management, blocks tend to be fuller after periods of low mining output and more empty after periods of high mining output. This effect is more pronounced the higher mining concentration. Our empirical analysis does not admit causal identification, but the preponderance of evidence is observationally equivalent to revenue enhancement on the part of miners.

Our paper fits into a small and growing literature on transaction fees in blockchain systems. Easley, O'Hara, and 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. Huberman, Leshno, and Moallemi (2017) compare Bitcoin to a traditional payment system and derive closed form solutions for equilibrium fees. In their framework, the chain runs at full capacity.

This research on fees fits into a larger body of literature that focuses on the economics and incentives in blockchain ecosystems (among others Abadi and Brunnermeier (2018), Cong and He (2019), Budish (2018)) and the impact on financial markets (e.g. Malinova and Park (2017) or Brauneis, Mestel, Riordan, and Theissen (2018)). Cong, He, and 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. Hu, Parlour, and Rajan (2018), Makarov and Schoar (2018), Choi, Lehar, and Stauffer (2018)).

Strategic manipulation of capacity to extract rents has been analyzed in the industrial organization literature. For example, Gilbert and Klemperer (2000) show that in markets for which consumers have to make a fixed investment to enter a market, rationing may induce more entry and thus be profitable, while Denicolò and Garella (1999) show that rationing may allow a durable good monopolist to maintain high prices. Similarly, in the operations research literature, rationing has been shown to be optimal to induce consumers to accelerate purchases (Liu and van Ryzin (2008)) and to convince consumers to ascribe a higher value to the good (see for example Debo, Parlour, and Rajan (2011) and Debo, Rajan, and Veeraraghavan (2020)).

An interesting contemporaneous theory paper, Malik, Aseri, Singh, and Srinivasan (2019) considers consumers who decide between processing payments with Bitcoin or going to the banking system. They provide conditions under which increasing Bitcoin capacity leads large miners to collude tacitly to undo such increases by only partially filling blocks; this crowds out low value payments who prefer to use the banking system. They further show that providing incentives for miners to operate at full capacity increases system security risk which could reduce value. We too consider how miners strategically manipulate capacity but our focus is on whether the blockchain system is competitive. Our predictions and findings on the within block dispersion of fees are inconsistent with their framework.

It is important to our discrimination framework that servicers can choose which transactions to include. The lack of commitment in the protocol is the starting point for Basu, Easley, O'Hara, and Sirer (2019). The authors observe that over time the prices for the same service have fluctuated, and argue that there is no dominant strategy equilibrium. They propose a mechanism that is manipulation proof as the number of users and miners increases.

Finally, although we are the first paper to document strategic capacity management in the Bitcoin system, there is a computer science literature on miner extractible value. Briefly, this literature documents that miners on the Ethereum blockchain systematically front run arbitrage trades. This further supports the idea that miners are profit maximizing and seek revenue from all aspects of the mining process. The seminal paper in this literature is Daian, Goldfeder, Kell, Li, Zhao, Bentov, Breidenbach, and Juels (2019).

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." While the mempool is not a centralized entity but rather is specific to each node (which typically have different capacity RAM), empirical results from computer science indicate that mempools are effectively the same.[1] Each time a miner competes to mine a new block, it selects valid transactions from its own mempool to work on. Once a block is mined, it is then broadcast to the network and all nodes remove the relevant transactions from their own mempools.

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.

Figure 1 plots the time series of the median as well as the 25 and 75 percentiles of transaction fees in ₿ starting in March 2015. We observe a higher variation of fees over time with the peak in December 2017 when Bitcoin prices peaked. The graph for fees in USD is similar.

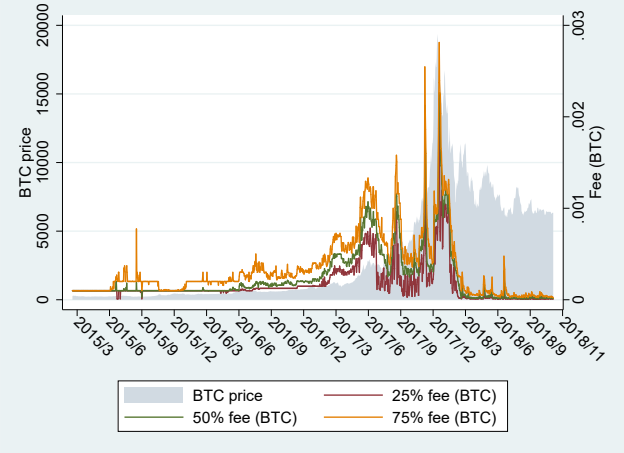

Figure 1. Bitcoin prices (gray area, left axis) and median, 25 and 75 percentile of daily fees (right axis) in Bitcoin. Days are defined over UTC. Transactions exclude coinbase transactions

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. In total we

[1] See for example, Dae-Yong, Meryam, and Hongtaek (2020).

have 353,306,421 transactions. 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. For technical reasons, however, effective block size was smaller until May 15, 2013.[2] An upgrade, Segregated Witness (Segwit), was implemented on August 24, 2017. Briefly, Segwit is a way to store signatures and scripts associated with compliant transactions in a special area of a block (the witness section).[3] We emphasize that adoption of this technology was voluntary and Bitcoin users slowly converted to the new system.

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 (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. Segwit did not quadruple capacity: As pre-Segwit transactions are replaced with Segwit compliant transactions, block capacity gradually increased. 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 size or block capacity, we work with these weights which reflect true capacity utilization. In Appendix A we provide further institutional details on the measurement of block capacity post Segwit.

The blockchain allows us to observe precisely how much capacity was used. We observe information about capacity demand imprecisely. However, we have obtained two sets of mempool data to measure transaction demand. First, we have partially aggregated mempool data that provides information on the number and size of transactions grouped by fee buckets. Second, we set up two nodes and collected a shorter sample of detailed mempool data that tracks each transaction. This second data set allows us to trace each transaction from the time it entered the mempool to the time that it was mined. Details of both data sets appear in Appendix B.

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. In our complete sample the median transaction size has a weight of 904. We count a block as full if there is less than free capacity of 2,000 weight units. Thus, our measure of a full block is conservative. Mining a full and an empty block are equally difficult (empty spaces are also data).[4] From Figure 2, in the early days of

[2]Block size was limited by the number of database locks required to process a block (at most 10,000). This limit translated to around 500-750 thousand 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. Details of this system change are available at `https://en.bitcoin.it/wiki/Block_size_limit_controversy,https://blog.bitmex.com/bitcoins-consensus-forks/`

[3]The first block exceeding 1MB limit was block 481,947 mined on Aug 25, 2017 with a size of 1032 KB.

[4]Miners hash over data that include the root of the Merkle tree that contains all transaction information. The

Bitcoin, most blocks were mined empty. This is most likely due to a lack of transaction demand.

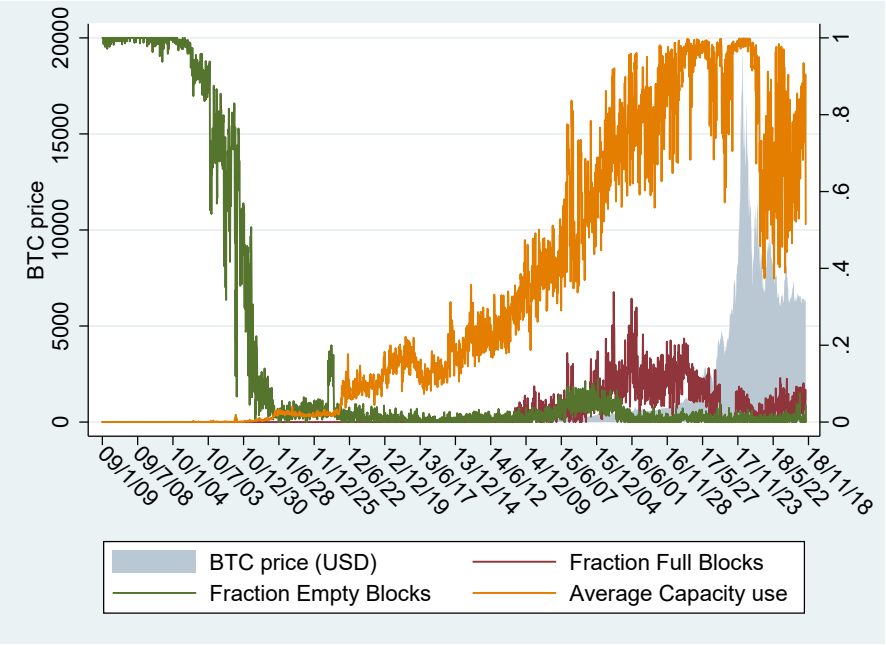

Figure 2. Number of blocks and fraction of empty and full blocks per day. Days are defined over UTC.

Our sample also includes the 2017 runup in Bitcoin prices. Even in this period, the blockchain did not run at full capacity. 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 one of our mempool data sets 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.[5] 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 sample.

Our results are consistent over time and across pools and so it is unlikely that they are due to random technical problems or due to network latency. The is because of special high speed connections between miners. When a new block gets discovered it has to be transmitted to other nodes so that the other nodes know the block's hash which they have to include in their own block and know which transactions have been included in the block so that they do not include

size of this root is independent of the number of transactions in the block.

[5]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.

the same transactions in their own block. Over time innovations such as the Fibre network (Fast Internet Bitcoin Relay Engine) or compact blocks as outlined in Bitcoin Improvement Proposal (BIP) 152 have drastically reduced block transmission times. Most miners participate in Fibre, which is a special network protocol started in 2016 to deliver Bitcoin blocks around the world with delays as close to the physical limits of signal transmission as possible. At the time of writing the paper, the median transmission time of blocks is on the order of 5 milliseconds.[6] Anecdotal evidence suggests block validation times for fast hardware to be 45 milliseconds.[7] Next, miners have to compile a candidacy block from transactions in the mempool on which they want to mine. Transactions that are in the mempool have to be verified before being included in a candidacy block. Verification includes for example the check of the signature and processing of the script and is computationally expensive but is completed as transactions enter the mempool and thus before a candidacy block is composed. At the time when a new candidacy block is compiled only relatively trivial consistency checks have to be performed which take usually less than 1 millisecond.[8] Overall the process from block discovery to compiling a candidacy block for mining transpires in less than a second.

In our data we observe filled blocks mined within a very short period and empty blocks mined with delays. Starting in 2013 we find that the average time after which an empty block was mined to be 1.96 minutes compared to 9.29 minutes for a full block. However we find 25,552 non-empty blocks mined within less than minute, and 1,566 non-empty blocks mined within less than 5 seconds. Similarly we find 192 empty blocks mined more than 10 minutes after their predecessor. Later in Subsection 3.4 we provide evidence that instead of occurring randomly, empty blocks are correlated with recent capacity usage. This is consistent with strategic capacity management rather than technical mishaps.

Another reason why technical problems are unlikely is the example of the Ethereum network. Ethereum works similarly to Bitcoin, with proof of work, mining pools, a mempool, and transactions that need to be confirmed, yet it is designed to add a new block to the chain every ten seconds.[9]

Empty blocks could be consistent with an empty mempool. This was certainly the case in the very early days of bitcoin (the first non-coinbase transaction occurs in block 170). However, using our mempool data which starts in 2016 we find that most empty blocks are mined when there are orders waiting in the mempool. Empty blocks are evenly distributed over time in our sample and do not cluster around times when transaction demand might be low. We therefore conclude that we do not observe empty blocks because of zero transaction demand.

Finally, we note that adding transactions to a candidate block is not irreversible. A miner can take all transactions in a mempool and start mining. When a higher fee transaction arrives before the correct nonce is found it can replace a low fee transaction without affecting the probability of finding a valid nonce. Miners do not have to keep block space set aside should

[6] See data from `http://bitcoinfibre.org/stats.html`

[7] See e.g., `https://bitcoin.stackexchange.com/questions/50349/how-long-does-block-validation-take`.

[8] see e.g. `https://bitcoin.stackexchange.com/questions/84045/block-verification-time`.

[9] We download data on over a million Ethereum blocks ranging from block 10,000,000 to 11,722,614. The average time between blocks is 13.32 seconds, the median is 9 seconds. 25% of blocks are mined within four seconds.

higher fee transactions arrive during the mining process. When confronted over mining empty blocks Jihan Wu from Antpool tweeted in 2016 'sorry, we will continue mining empty blocks. This is the freedom given by the Bitcoin protocol.'

In addition to empty blocks, the second source of unused capacity are blocks that contain some transactions, but are also not completely filled. 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 25,838 Segwit compliant transactions was left empty each day (including space in empty blocks). For the broader sample since Jan 1 2014 on average there was empty space for 909,137 transactions per day.

In spite of this unused capacity, unconsummated orders, often with fees attached, were waiting to be added to the blockchain. To quantify how much money miners would have made if they had filled up all blocks to capacity, we use minute by minute mempool data, which comprises partially aggregated transactions that have been verified and are waiting to be included in a block. (A detailed description of the data set appears in Appendix B). In these data, transactions are grouped by fee buckets based on sat/byte. Two things are worth noting. First, the mempool we observe is not empty: For 78% of the blocks in our sample all the excess capacity could have been completely filled with transactions from the mempool and further 20% could be partially filled. Only for 0.005% of blocks in our sample the mempool was exhausted.[10] Second, specialized mining pools have access to have better hardware and more peer connections and therefore have better and more up-to-date information on potential transactions. We therefore conclude that we have conservative snapshots of their mempools.

Combining 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. Miners also forgo profits by not picking the transactions that pay the highest fee per blockspace which is discussed in Section 3.5. For each block, we match the excess capacity in the block with the excess demand for transactions in the mempool at the time the block was mined. Figure 3 plots the total money left on the table by miners per day. Foregone revenue or money left on the table is observed consistently throughout our sample.

[10] We are able to match mempool snapshots to 101,045 mined blocks. Out of these 78,820 could be completely filled, and 20,398 could be partially filled. 1,822 blocks were completely full, and for 5 blocks the mempool snapshot was empty. Note that invalid transactions are not broadcast and immediately discarded from the mempool.

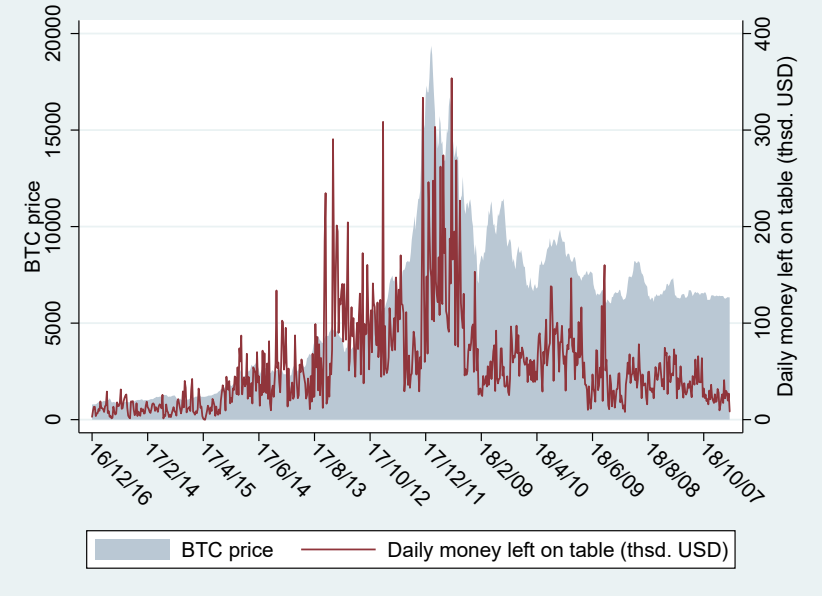

Figure 3. Money left on the table by miners and Bitcoin price. We fill up empty capacity on the blockchain with unmined mempool transactions offering the highest fee/byte.

3 Optimally "leaving money on the table"

The BitCoin protocol does not specify the sequence in which miners process orders waiting in the mempool. Although a protocol that selects orders with highest fees first is a natural one to consider, it is also possible that orders are selected in a different way. We seek to understand how the lack of a specific sequencing protocol affects consumers, and so our empirical strategy uses observed fees in each block to infer agents' beliefs about the protocol, and thus draw inferences about how it affects their behavior. To do this, we specify a flexible framework that admits joint observations on bids and used block capacity that would be observed under two relevant protocols: highest fee first (HFH) and strategic capacity management (SCM).

Timing Convention: Our data comprise all mined blocks, and so we describe time in "block time," which is analagous to "transaction time." Our timing convention is illustrated in Figure 4 below. The block mined at time $t-1$ is drawn from the time $t-1$ mempool.

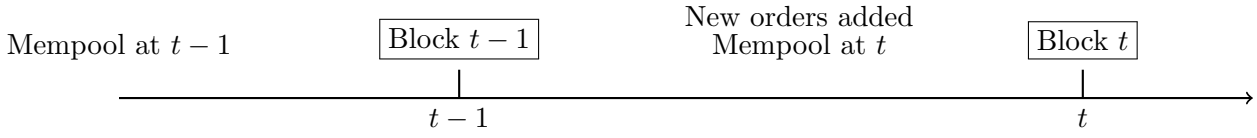

Figure 4. Sequence of Events from Block to Block

Composition of the mempool: Entries in the mempool comprise a transaction size and a fee associated with the transaction. For simplicity, we consider all transactions to be of the same size, which we normalize to one. The mempool after block t is mined is defined by $M_{t+1} \equiv \{m_{t+1}^f\}_{f=0}^F$ where m_{t+1}^f are the submitted but unmined orders offering fee f, and F is the maximal bid. (Across all the blocks in our sample it is Ƀ291.24). We note that fees are effectively discrete, as the minimum price increment we observe is 1 satoshi.

The $t+1$ mempool comprises the orders in the time t mempool, less the orders that were mined in block t and augmented by the flow of orders that come in after the tth block was mined. From the point of view of an agent who submits an order, the future flows are unknown.

We have aggregate data on one mempool and how it changes after each block is mined. Our data are from a researcher and thus likely a subset of the orders maintained by mining pools. For each block, let $\bar{\kappa_t}$ denote the maximum capacity of block t, which is determined by the technology. Let κ_t denotes the observed used capacity in block t, so $0 \leq \kappa_t \leq \bar{\kappa_t}$ for all t. Consistent with our data, we suppose that capacity is scarce, so $\bar{\kappa_t} < \sum_{f=0}^F m_t^f$, for all t.

Preferences: Agent types are denoted by θ. An agent of type θ who broadcasts his order at time t is defined by an intrinsic valuation $v_\theta > 0$ and a maximum number of blocks over which he is willing to wait, Δ_θ before his order is mined. For simplicity, we assume that his payoff is zero if he does not execute within this window. We also assume that intrinsic values and maximum waiting times are ordered so if $\theta > \theta'$, then $v_\theta > v_{\theta'}$ and $\Delta_\theta < \Delta_{\theta'}$. In short, agents with higher valuations want to be processed more quickly. Notice, that because of periodic block processing, the natural notion of time is discrete, thus are types are discrete.

Waiting time and block usage: Virtually all transactions submitted to the blockchain are eventually executed. However, orders can be delayed which is akin to rationing. Pivotal to the fees submitted by any agent are their beliefs on how their fees translate into waiting times.

When an agent broadcasts a transaction, he has some conditioning information, which we denote x_t. This could include a signal about the current state of the mempool, and beliefs about the servicers' protocol. For any fee f that agent of type θ submits, he ascribes a probability $p_\theta(f \mid x_t)$ that his order will execute within his desired number of blocks.

Let $q_{t+i}(f \mid M_{t+i})$ denote the probability that a transaction submitted at time t with an associated fee, f is processed in the ith block after the order is broadcast when the mempool at that time is M_{t+i}. The probability $q_{t+i}(f \mid M_{t+i})$ is determined by both the composition of the mempool before block $t+i$, and the protocol that the servicer uses to choose which orders to mine. Then, an agent who wishes to have his order mined within Δ_θ blocks determines it as

$$p_{\Delta_\theta}(f \mid x_t) \quad = \quad E[q_{t+1} + \sum_{j=1}^{\theta} \Pi_{i=1}^{j}(1 - q_{t+i}) q_{t+j} \mid x_t].$$

Thus, given his conditioning information, x_t, and his beliefs on how orders are executed, the agent faces a menu that relates fees to expected waiting times, $\{(f^i, \Delta(f^i))\}_{i=1}^N$. Consider the

choice of agent θ faced with this menu. He will choose the bundle $(f^i, \Delta(f^i))$ if for any other bundle $(f^j, \Delta(f^j))$,

$$p_{\Delta_\theta}(f^i \mid x_t)v_\theta - f_i \quad \geq \quad p_{\Delta_\theta}(f^j \mid x_t)v_\theta - f^j. \tag{1}$$

Highest Fee First

Suppose the protocol is to process the highest fee first until block capacity is reached. This protocol is akin to competitive mining. In this case, 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.

Prediction 1 *Suppose that the servicers always process highest fees first, then*

(i) The mempool will never contain transactions after a partially empty block is mined.

(ii) A mined block will not include transactions with a fee lower than existing transactions in the mempool.

The structure of the highest fee first protocol is akin to an all pay auction (Krishna(2009)). Higher fees will be processed first, and fees for the block at t will be generated by the set of order statistics $f_t^1, \ldots f_t^{\bar{\kappa}}$, and transactions remaining in the mempool will have lower fees.

Strategic Capacity Management

Consistent with websites that dynamically report the relationship between fees and block waiting times, we consider bidding by agents who believe that the menu that relates bids to expected waiting times, $\{(f^i, \Delta(f^i))\}_{i=1}^N$ is not necessarily highest fee first.

Prediction 2 *Suppose that the servicer uses strategic capacity management, then*

(i) The mempool may contain transactions after a partially empty or fully empty block is mined.

(ii) A mined block may include transactions with a fee lower than existing transactions in the mempool.

Anecdotal evidence is consistent with the latter prediction. Online fee calculators like the one shown in Figure 5 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.[11]

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

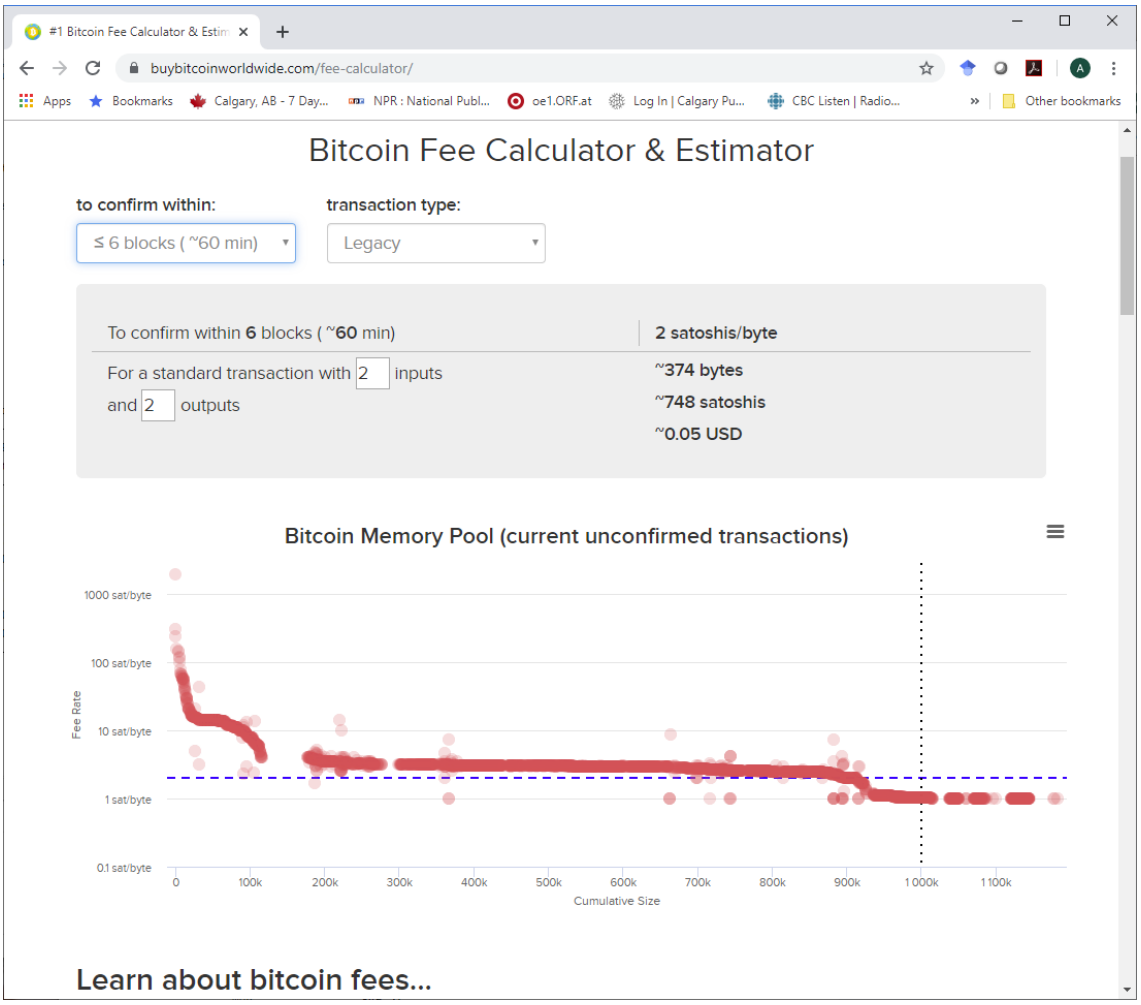

Figure 5. Screenshot from a Bitcoin fee calculator that provides real time estimates of fees that users have to offer to get confirmed with 90% probability within 1,2,3,4,5, and 6 blocks, respectively.

By itself, strategic capacity management does not imply any particular ordering between offered fees and the expected waiting times. We note, however, that there are certain implications that come from fee revenue maximization. Intuitively, customers facing SCM will choose the fees to maximize their expected utility. By contrast, a servicer trying to maximize revenue will only offer fast processing times to customers that offer fees that are sufficiently high. In short, the servicer will choose a menu that induces agents with high valuations to bid higher than they would if they chose fees as if under a highest fees first protocol.

Prediction 3 *Suppose that Strategic Capacity Management is undertaken to maximize fee revenue, then observed fees may be more dispersed than under the highest fee first protocol.*

The stylized facts we have presented on capacity utilization are consistent with strategic capacity management. The further evidence we present is consistent with strategic capacity management

being used to increase revenue. We next document fee dispersion per block and the fact that types with a higher willingness to pay bid more.

3.1 Fee Dispersion

Figure 6 illustrates Bitcoin fee heterogeneity in December 2017, which was the peak of Bitcoin prices in our sample. The left panel illustrates that fees up to the median were less than USD 35. By contrast, from the right panel the maximum fee was USD 14,174.64, and many other transactions paying several thousand dollars in fees. Overall, 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.

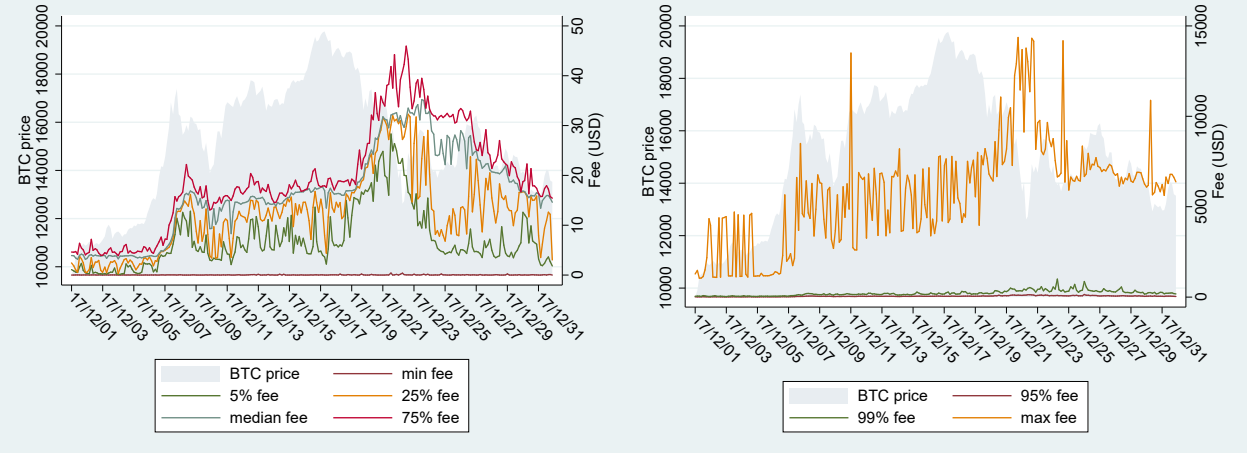

Figure 6. Fees in USD (right axis) and Bitcoin price in USD (left axis) in December 2017.

A high fee dispersion is consistent with the use of capacity management to induce induce different fees from different types. By constrast, if agents believe that mining is competitive, it is difficult to envisage a rational agent paying such excessive fees given that transactions with lower fees were recently included in blocks.

3.2 Who pays more? Evidence from Blockchain characteristics

Central to our argument is the fact that different participants have different values for completed transactions. We 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. The following variables differ across users' types and are plausibly related to their demand for immediate execution. First, the speed with which the output from a transaction is used again. We define rest time as the minimum time (in blocks) until the first output of a transaction is spent again. Recipient wallet owners that spend their funds very quickly have demonstrated an

immediate need for funds. Similarly, we use a dummy variable for transactions for which an output is spent within one block. Second, some transactions are used to insert data into the blockchain; such transactions have no obvious need for speed. The Data dummy is set to one for all such data insertion transactions.[12]

We also include variables that are related to both the capacity of the block and the size of the transactions. Transaction size is the physical size of all inputs and outputs in bytes or weight units. The transaction size represents an opportunity cost for miners as block space is limited. Blocksize is the absolute size of the block in weight units.

Finally, we include variables as controls. 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.

	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	0.20	0	137,186
Sum Inputs (thsd. Sat.)	353,306,421	1,367,292.33	40,523,154.35	19,800	0	55,000,000,000
Sum Inputs (USD)	291,702,649	25,69	556,02	256.64	0	946,541,056
Blocksize (bytes)	353,306,421	834,072	325,063	998,126	267	2,324,736
Block weight	353,306,421	3,209,784	1,212,422	3,991,416	624	3,999,120
Transaction Size	353,281,736	538	2,211	250	62	999,657
Transaction Weight	353,306,421	2,084	8,544	904	248	3,998,628
Data	353,306,421	.022	.147	0	0	1
Spent next block	353,306,421	.140	.347	0	0	1
Resttime (blocks)	350,031,634	547.54	4,950.34	4	0	474,195
₿ Price (USD)	291,702,649	3,604.66	4,162.71	1,188.28	204.84	19,829.76

Table 1. Summary statistics *Fee (Satoshi)* is the transaction fee in Satoshi. One bitcoin is 100 million Satoshi. *Fee (USD)* is the transaction fee in USD, *Sum Inputs (thsd. Sat.)* is the sum of input values for the transaction measured in thousands of Satoshi, *Sum Inputs (USD)* is the sum of input values for the transaction measured in USD, *Blocksize (bytes)* is the size of the block measured in bytes, *Block weight* is the size of the block measured in weight units, *Transaction Size* is the size of the transaction measured in bytes, *Transaction Weight* is the size of the transaction measured in weight units, *Data* is a dummy set to one of a transaction inserts non-transactional data (identified by the OP_RET instruction in the script), *Spent next block* is a dummy set to one an output of a transaction was re-spent within one block, *Resttime* is the average time (measured in blocks) until transaction outputs are re-spent, *₿ Price (USD)* is the Bitcoin price in USD.

Many variables in Table 1 present extreme observations which we carefully control for in our subsequent analysis. For example, some transactions with unusually high fees are payments miners make to themselves as they consolidate Bitcoin balances with the coinbase under a new address. For example on April 26, 2016 someone paid ₿291.241 to an output with ₿0.0001, leaving fees of ₿291.2409 (or USD 137,186.07 at the time) for the miner.[13] We identified 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 or (USD 16,974.42 at the time). Transactions from the

[12]These so-called "Op-Ret" transactions are used to store data from 2nd layer applications on the Bitcoin blockchain.

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

earliest 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 and outputs of ₿36, leaving a huge fee for the miner. At that time Bitcoin were worth very little, yet since there was no shortage of transaction space in the blockchain, there is no obvious rationale for paying such a high fee.

The average Bitcoin transaction was for ₿13.7 (one Bitcoin equals 100 million Satoshi) while the largest transaction was for ₿550,000 on Nov 16, 2011 at a zero fee.[14] The largest transaction in dollar terms occurred on Dec 17th, 2017 at the peak of the Bitcoin price when ₿48,500 valued at over USD 946 million changed wallet for a fee of 80,908 Satoshi or USD 15.8.[15] In our sample there are 54,907 transactions with a value of more than USD 10 million. Of those, 20,956 were processed with a fee of less than USD 5, while the average fee was USD 90.54.

Most transactions are small as the mean is 538 bytes or 2084 weight units while the median is 250 bytes or 904 weight units. However, the largest transaction in our sample consumes the entire block 364,292 and has a size of 999,657 bytes. This transaction was mined on July 7th, 2015 and consolidates 5,569 inputs of 1,000 Satoshi each into a single output.[16]

For our regressions, we winsorize the fee data, the transaction size, the inputs, and the restime at the 99.9% level. (The coinbase of the Genesis block has never been spent, implying a resttime equal to the sample length.) Our results are 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.

In Tables 2 and 3, we regress fees in Satoshis and USD respectively on transaction characteristics. Fees are higher when transaction space is scarce (larger blocksize weight) and when the transaction is larger in bytes (Transaction Size), weight (Transaction Weight) and value (Sum Inputs). The coefficient on transaction weight is roughly one fourth that of transaction size in bytes because pre SegWit transactions have, by definition, a weight equal to four times their size in bytes. Transaction size has an economically significant impact on fees. Specifically, 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. Fees are also higher when the funds are spent sooner (lower rest time) and especially if the output was spent in the next block. The dollar value of the transaction (SumInputs), for example, only has a small and economically insignificant effect on 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. Results are mixed for the data-insertion (OP-Ret) dummy. In

[14] see transaction 29a3efd3ef04f9153d47a990bd7b048a4b2d213daaa5fb8ed670fb85f13bdbcf

[15] see transaction 261d69b25896034325d8ad3e0668f963346fd79baefb6a73b4eabd68c58c81ff

[16] According to some forum posts an unknown spammer created all these wallets to test the limits of the UTXO database. To speed up the time it takes to verify the validity of a transaction each node holds in RAM a complete lust of all unspent BTC amounts per address. This list is called the Unspent Transaction Output (UTXO) database. Somebody created many small outputs locked up under different addresses. The attack 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 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)
Data					-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)
Resttime						-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).** *Block weight* is the size of the block measured in weight units, *Transaction Size* is the size of the transaction measured in bytes, *Transaction Weight* is the size of the transaction measured in weight units, *Sum Inputs (USD)* is the sum of input values for the transaction measured in USD, *Data* is a dummy set to one of a transaction inserts non-transactional data (identified by the OP_RET instruction in the script), *Spent next block* is a dummy set to one an output of a transaction was re-spent within one block, *Resttime* is the average time (measured in blocks) until transaction outputs are re-spent. 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.

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.

We stress that 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.

3.3 Who pays more? Evidence from trader specific characteristics

Different trader types plausibly have different values for transactions and different costs of waiting. More sophisticated investors such as institutional investors or hedge funds are more likely to be active during the week and when the Bitcoin futures at CME are trading.[17] To investigate this, we augment the regression presented in Tables 2 and 3 to include time dummies. The results are presented in Table 4. Average transaction fees on weekends are 20 cents lower than week days (Column (2)) which, while small, is economically meaningful as it is the median transaction fee for the whole sample. Also, fees gradually increase during the work week so that the highest observed fees are on Fridays, which is also the settlement day for futures (column (1)). Fees are also higher by about half a dollar, or twice the median fee, whenever the futures market is open (column (4)).

It is possible that CME trading hours are picking up higher transaction demand that is unrelated to futures and hence institutional trading. To ensure that our findings are not driven by specific characteristics of CME futures trading hours we perform a robustness check using a 15 day window around December 18, 2017 when futures were first traded. These results appear in column (5). The dummy *CME trading hours* is one during the regular trading hours of Bitcoin futures at CME, *Post Dec 18th* is a dummy equal 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. We defer our discussion of Column 3, and the HHI variable, to Section 3.4 below.

Bitcoin is a pseudo-anonymous system, while wallet addresses are public and payments can be observed moving from one wallet to another, the identity of the wallet owner is usually unknown. In some cases, e.g. voluntary disclosure, or court proceedings, the owner of some addresses becomes public. In addition, gambling sites often use vanity addresses, such as '1dice....' or

[17] BTC futures at CME trade 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 with the appropriate adjustment for summer daylight savings time.

	Whole Sample							w/o Peak
Block 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)
Data					-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)
Resttime						-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.** *Block weight* is the size of the block measured in weight units, *Transaction Size* is the size of the transaction measured in bytes, *Transaction Weight* is the size of the transaction measured in weight units, *Sum Inputs (USD)* is the sum of input values for the transaction measured in USD, *Data* is a dummy set to one of a transaction inserts non-transactional data (identified by the OP_RET instruction in the script), *Spent next block* is a dummy set to one an output of a transaction was re-spent within one block, *Resttime* is the average time (measured in blocks) until transaction outputs are re-spent. 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.

	Whole Sample				Event
	(1)	(2)	(3)	(4)	(5)
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***	-0.0381		
		(0.0120)	(0.0264)		
Weekend × HHI			-1.366***		
			(0.217)		
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)
Block Size (Weight)	1.27e-08***	1.76e-08***	1.70e-08***	2.02e-08***	0.00000786***
	(1.68e-09)	(1.59e-09)	(1.61e-09)	(1.61e-09)	(0.00000266)
Transaction Weight	0.000595***	0.000595***	0.000595***	0.000595***	0.00264***
	(0.00000252)	(0.00000252)	(0.00000252)	(0.00000252)	(0.0000150)
Sum Inputs (USD)	0.00000613***	0.00000613***	0.00000613***	0.00000613***	0.00000689***
	(5.91e-08)	(5.92e-08)	(5.92e-08)	(5.92e-08)	(0.000000178)
Data	0.00511	0.00669	0.00683	0.00242	1.142***
	(0.0181)	(0.0180)	(0.0180)	(0.0180)	(0.276)
Spent next block	0.580***	0.582***	0.582***	0.581***	2.564***
	(0.00679)	(0.00682)	(0.00682)	(0.00681)	(0.118)
Resttime	-0.0000182***	-0.0000182***	-0.0000182***	-0.0000183***	-0.000158***
	(0.000000323)	(0.000000323)	(0.000000323)	(0.000000323)	(0.00000601)
R^2	0.395	0.395	0.395	0.395	0.393
Observations	288,555,299	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. The regression includes day of the week dummies, *Weekend* is a dummy set to one if the day is either Saturday or Sunday, *HHI* is the Herfindahl–Hirschman index of daily mining shares, *CME Futures trading* is a dummy that is set to one during the trading hours of Bitcoin futures at the CME after Dec 18, 2017, the day when Bitcoin futures started trading, *CME trading hours* is a dummy that is set to one during the hours when Bitcoin futures trade at the CME for the whole sample period, i.e. also before Dec 18, 2017, *CME trading hours* is a dummy that is set to one after Dec 18, 2017, *Block weight* is the size of the block measured in weight units, *Transaction Weight* is the size of the transaction measured in weight units, *Sum Inputs (USD)* is the sum of input values for the transaction measured in USD, *Data* is a dummy set to one of a transaction inserts non-transactional data (identified by the OP_RET instruction in the script), *Spent next block* is a dummy set to one an output of a transaction was re-spent within one block, and *Resttime* is the average time (measured in blocks) until transaction outputs are re-spent. 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.

'1Lucky....' which are easily identified and more importantly are reused.[18] 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. Reid and Harrigan (2013) or Foley, Karlsen, and 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.[19]

We use data from lists of known addresses and are able to identify the sender for 18,123,498 transactions, out of which 7,250,374 (2.05% of all transactions) were initiated by an exchange and 10,873,124 (3.07% of all transactions) that were initiated by a gambling site.[20] 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. Notably, flows to and from exchanges transact at substantially 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 as the data is winsorized. Transactions flowing into exchanges pay on average USD 3.97 more than the average fee paid on the same day. This is economically large, given that the median fee for the whole sample is USD 0.20. Flows from exchanges pay USD 2.20 more than same-day average. Gamblers also pay significantly higher fees. Traders moving funds in and out of exchanges and gamblers put a high value on immediate execution. Consistent with revenue maximization, such transactions pay higher fees.

Arbitrageurs who take advantage of cross exchange pricing differences also value quick execution. Arbitrageurs' preference for immediacy is higher, the higher 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 strategic capacity management 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 we match minute time-stamped prices from Korea and the US with block creation times. The Kimchi premium is the relative price difference of Bitcoin in the US and Korea.[21] 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 find that fees on transactions to and from exchanges increase in the Kimchi premium. 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 from high value types.

Our findings are in Table 6. Exchange is a dummy that identifies 35,163,580 payments to and from known exchange wallets. The interaction term of Exchange and Kimchi premium

[18] 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.

[19] 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. Möser and Böhme (2017).

[20] The data are primarily from `walletexplorer.com` and are available from the authors upon request.

[21] The Kimchi premium is calculated as the absolute value of the Bitcoin price at Korbit in Korea converted to USD minus the Bitcoin price on Coinbase in the US as a percentage of the US price.

To Exchange	6.928***	3.973***		
	(125.78)	(104.91)		
To Gambling	0.392***	0.235***		
	(99.88)	(32.19)		
From Exchange			2.198***	1.239***
			(156.89)	(124.17)
From Gambling			0.772***	0.524***
			(129.46)	(103.30)
Block 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)
Data		-0.224***		0.00812
		(-14.63)		(0.53)
Spent next block		0.616***		0.621***
		(89.28)		(91.25)
Resttime		-0.0000171***		-0.0000164***
		(-53.47)		(-52.15)
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 and known wallet addresses. *To exchange* and *From exchange* are dummy variables set to one if the transaction makes a payment to or receives a payment from a wallet identified as belonging to an exchange, respectively. *To Gambling* and *From Gambling* are dummy variables set to one if the transaction makes a payment to or receives a payment from a wallet identified as belonging to a gambling site, respectively. *Block weight* is the size of the block measured in weight units, *Transaction Weight* is the size of the transaction measured in weight units, *Sum Inputs (USD)* is the sum of input values for the transaction measured in USD, *Data* is a dummy set to one of a transaction inserts non-transactional data (identified by the OP_RET instruction in the script), *Spent next block* is a dummy set to one an output of a transaction was re-spent within one block, and *Resttime* is the average time (measured in blocks) until transaction outputs are re-spent. 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.

is statistically and economically significant. For an increase in the Kimchi premium of 10 percentage points, users are willing to pay USD 3.22 more in fees. We note that our analysis is most likely an underestimate of how the demand for immediacy affects fees for two reasons. First, we cannot 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. Once again, we defer our discussion of HHI, our concentration measure, to Section 3.4 below.

Kimchi Premium	40.20***	35.05***	32.46***	-24.01***	-30.79***	-32.48***
×exchange	(120.07)	(91.89)	(95.94)	(-17.33)	(-24.38)	(-29.80)
Kimchi Premium				534.5***	546.2***	538.8***
×exchange×HHI				(43.58)	(48.04)	(54.09)
Exchange		0.928***	-0.0962***		0.967***	-0.0564***
		(60.02)	(-7.10)		(65.55)	(-4.57)
Kimchi Premium		-1.521**	-0.850		-1.117	-0.450
		(-2.23)	(-1.31)		(-1.64)	(-0.70)
Post Segwit			1.859***			1.861***
			(4.66)			(4.67)
Block Weight			2.27e-08***			2.30e-08***
			(15.61)			(15.77)
Transaction Weight			0.000607***			0.000607***
			(237.15)			(237.18)
Sum Inputs (USD)			0.00000759***			0.00000757***
			(97.90)			(97.90)
Data			0.0280*			0.00714
			(1.84)			(0.47)
Spent next block			0.625***			0.617***
			(91.43)			(90.47)
Resttime			-0.0000159***			-0.0000160***
			(-51.11)			(-51.61)
R^2	0.270	0.271	0.412	0.271	0.271	0.413
Observations	293,822,933	293,822,933	290,678,793	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. *Kimchi Premium* is the absolute value of the Bitcoin price in Korea converted to USD minus the Bitcoin price in the US as a percentage of the US price, *Exchange* is a dummy equal to one if the transaction involves a wallet identified as belonging to an exchange, *HHI* is the Herfindahl–Hirschman index of daily mining shares, *Post Segwit* is a dummy equal to one after the introduction of Segwit, *Block weight* is the size of the block measured in weight units, *Transaction Weight* is the size of the transaction measured in weight units, *Sum Inputs (USD)* is the sum of input values for the transaction measured in USD, *Data* is a dummy set to one of a transaction inserts non-transactional data (identified by the OP_RET instruction in the script), *Spent next block* is a dummy set to one an output of a transaction was re-spent within one block, *Resttime* is the average time (measured in blocks) until transaction outputs are re-spent. 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.

3.4 Strategic Capacity Management, Transaction fees over time and Mining Concentration

Our framework has implications for within block fee dispersion: dispersion is higher if strategic capacity management is undertaken to increase fee revenue. In this section, 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. We then investigate how this variable is affected by mining concentration.

Suppose that strategic capacity management increases revenue. Consistent with collusion in

repeated games, servicers will maintain strategic capacity management if the expected stream of future profits exceeds one shot deviation plus the future punishment payoffs. A miner who deviates would include too many transactions i.e., fill up a block. If he finds the nonce, this would become public. Full capacity mining in perpetuity is a natural punishment strategy.[22]

To investigate whether fees are only driven by transaction demand, or also influenced by miner behavior, we measure mining concentration by the share of daily blocks mined by specific mining pools. To do this, we collect miners' signatures from each block's coinbase transaction. Unlike any other transactions, the Bitcoin 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 on topics as diverse as miners' opinion on Bitcoin improvement proposals, or philosophy,[23] but most important for our purpose is that 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.[24] 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.

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 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.

To measure mining concentration we use HHI as defined above and also compute the daily fraction of blocks that were mined by mining pools. Table 7 presents our findings. The feespread (i.e., dispersion) 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 collusive strategic capacity management.

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 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.[25] Figure 7 illustrates the time between blocks. In

[22] Another punishment strategy is to purposefully orphan the blocks produced by deviating miners. Orphan blocks are rare in the data. An examination of the approximately 57 orphan blocks in our data exhibit a relationship between capacity usage and orphaning, these results are presented in Subsection 3.6.

[23] 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.

[24] 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. Mining pools may find it optimal to disclose their identity to show that they do not deviate from the collusive equilibrium.

[25] 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

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)
Data		1.074***		0.965***
		(0.0253)		(0.0254)
Resttime		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 mining activity* is the Herfindahl–Hirschman index of daily mining shares, *Fraction mined by pools* is the daily fraction of blocks mined by identifiable pools, *Mined by pool* is a dummy set to one if a block was mined by an identifiable pool, *Post Segwit* is a dummy equal to one after the introduction of Segwit, *Block weight* is the size of the block measured in weight units, *Average tx weight* is the average weight of transactions in the block, *Sum Inputs (USD)* is the average input transaction value measured in USD per block, *Data* is the fraction of data insertion transactions (identified by the OP_RET instruction in the script) in the block, and *Resttime* is the per block average of the time (measured in blocks) until transaction outputs are re-spent. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

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.

Strategic capacity management can increase revenue because miners can credibly delay low fee transactions. To keep delay consistent miners can compensate for the variation in the arrival-time of blocks as documented in Figure 7 by adjusting block usage. Specifically, we would expect that if a few blocks arrive close together, subsequent blocks would be more empty as miners delay patient types. Similarly after a long interval between blocks, subsequent blocks would be fuller to accommodate the patient types. Under collusion this effect should be more pronounced when mining is more concentrated. We test this intuition by regressing block-weight on dummies for

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.

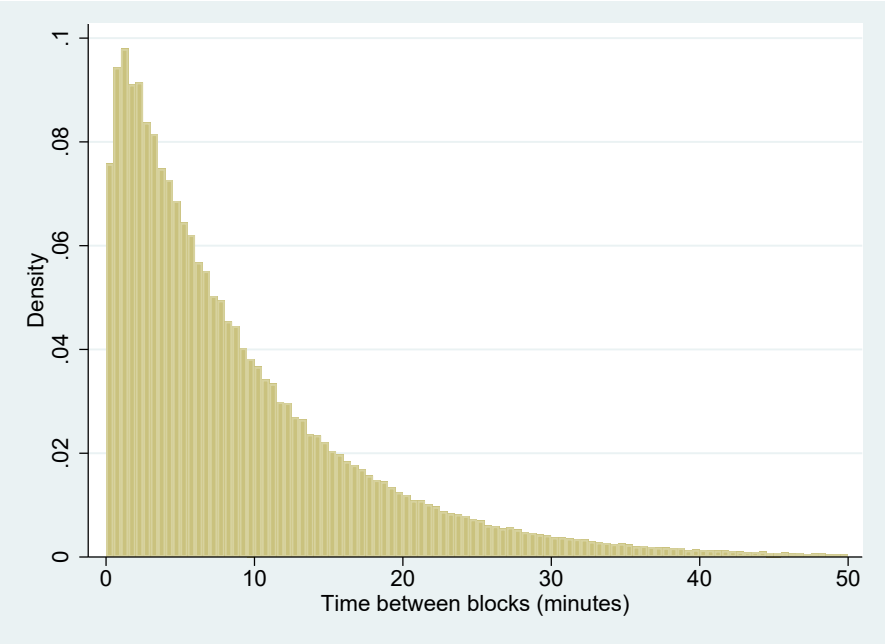

Figure 7. Histogram of time between blocks with blockheight larger than 100,000, capped at 50 minutes.

terciles of the distribution of the number of blocks mined in the last hour. In column 2 of Table 8 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 larger effect. Our findings are consistent with the idea that when miners collude, capacity management by mining pools is more prevalent.

Another way to manage capacity is by including empty blocks. In Table 9 we document that the probability of an empty block being mined increases in the number of blocks found in the last hour and in the used capacity over the last hour.[26] This evidence is consistent with miners delaying transactions and thus inserting empty blocks when by chance too many blocks were found.

Finally, we refer back to our analyses of the previous section that related trader types to the fees that they paid. First, from the results in Table 4 column (3) we can see that the drop in fees over the weekend increases in mining concentration. If fees were driven by strategic bidding alone we would expect to see mining concentration being irrelevant. Our empirical findings are consistent with miners being able to extract more when high types get greater utility out of having their transaction processed at times when collusion is easier to sustain. The last three columns of Table 6 document that our findings are stronger when mining is more concentrated and, therefore easier to sustain strategic capacity management.

[26] Our findings are similar with two hour windows.

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)	
Data	-1095141.8***	-1082040.9***	
	(251794.3)	(252135.6)	
Resttime	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. *Low tercile recent blocks* and *High tercile recent blocks* are dummy variables equal to one if the number of blocks mined in the previous hour is in the lowest/highest tercile of its distribution, HHI is the is the Herfindahl–Hirschman index of daily mining shares, *Sum Inputs (USD)* is the average input transaction value per block measured in USD, *Data* is the fraction of data insertion transactions per block (identified by the OP_RET instruction in the script), and *Resttime* is the block-average of the time (measured in blocks) until transaction outputs are re-spent. Standard errors are clustered by day. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

Number blocks 1h	0.0501***		0.0619***		0.0611***
	(0.00185)		(0.00200)		(0.00534)
Used capacity 1h		0.180***	0.673***		0.513***
		(0.0390)	(0.0431)		(0.0992)
Size mempool (MB)				3.53e-10	7.30e-10
				(8.53e-10)	(8.68e-10)
pseudo R^2	0.1448	0.1372	0.1474	0.0187	0.0332
Observations	447,626	447,626	447,626	103,974	103,974

Table 9. Probit regression explaining the probability of mining an empty block. *Number blocks 1h* is the number of blocks mined in the previous hour, *Used capacity 1h* is the fraction of the capacity used in the blockchain in the previous hour, *Size mempool (MB)* is the size of the mempool in MB. All regressions include week fixed effects. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

3.5 Detailed Mempool data

To investigate how individual transactions are delayed we also collected detailed transaction level mempool data on 671,025 transactions out of which 660,870 eventually weremined. A detailed description of our data can be found in Appendix B. Figure 8 illustrates how long transactions had to wait in the mempool for processing conditional on their place in the fee distribution. Specifically we look at the subset of transactions that eventually got mined and for the purpose of the graph drop all transactions that waited more than 10 blocks. For each transaction we compute the fee distribution of all unmined transactions at the time that this transaction entered

the mempool. The left panel shows wait times in blocks for transactions that were in the top 5% of the fee distribution at the time they entered the mempool. Similarly the middle panel and right panel show wait times for transactions in the 5% around the median and at the bottom of the fee distribution, respectively. Consistent with strategic capacity management we find that transactions with lower fees have to wait longer to be processed than high fee transactions.

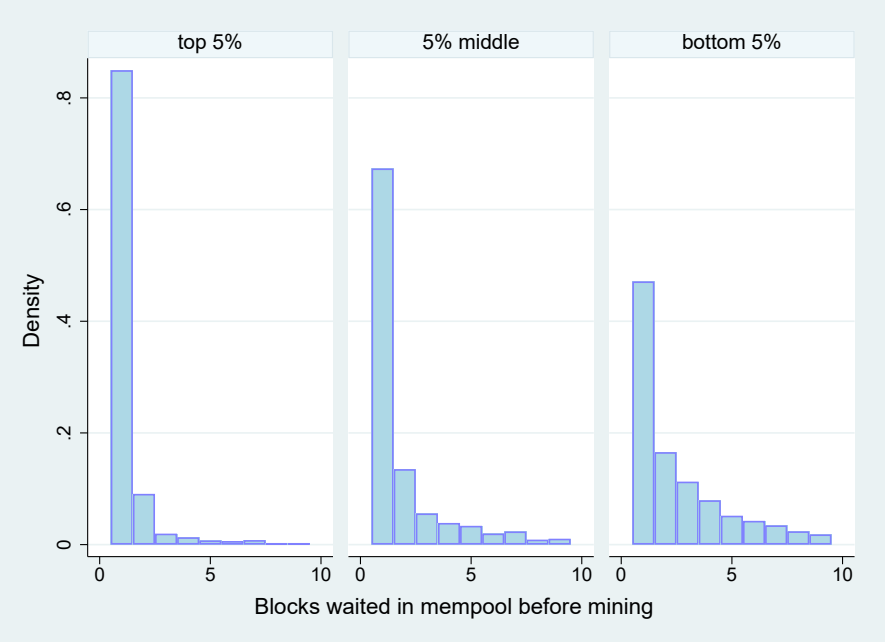

Figure 8. Histogram how long transactions that eventually got mined had to wait in the mempool for transactions that were in the top 5% of the fee distribution at they time they entered the mempool (left panel), in the 5% around the median (middle panel), and in the bottom 5% (right panel), respectively.

A priority violation occurs if a transaction paying a higher fee per weight unit is not mined but a lower fee transaction is. Table 10 shows the results of a probit regression explaining the probability that a transaction suffers a priority violation. We find that higher fee transactions are significantly less likely to be delayed. Further, the blockweight and the number of transactions in the pool do not determine the probability of priority violations. We also find no significant difference across mining pools with the exception of BTC.TOP, which has a smaller probability of priority violations in our sample. We draw two conclusions from this analysis. First, the existence of priority violations is inconsistent with a highest price first processing rule. Second violations are not random, but are more likely among low fee transactions consistent with strategic delay.

3.6 Orphaned blocks

Collusion requires deviators from the collusive equilibrium to be punished. While implementing competitive servicing is a possible punishment, in our setting miners could also punish a specific

Fee/Wght	-0.0132^{***}	-0.0142^{***}	-0.0150^{***}
	(-3.59)	(-3.92)	(-4.00)
Blockweight		0.0000974	-0.0000197
		(0.31)	(-0.80)
Number TX in pool		0.00000354	0.00000344
		(0.83)	(0.81)
TX weight		-0.00000785^{***}	-0.00000793^{***}
		(-4.87)	(-4.90)
AntPool		0	
		(.)	
BTC.COM		-0.122	
		(-0.42)	
BTC.TOP		-0.719^{**}	
		(-2.57)	
Bitcoin.com		-0.0929	
		(-0.49)	
F2Pool		-0.596	
		(-0.35)	
Huobi		0.299	
		(0.65)	
SlushPool		-0.139	
		(-0.49)	
Unknown		0.236	
		(1.26)	
ViaBTC		-0.106	
		(-0.46)	
poolin		0.0190	
		(0.09)	
pseudo R^2	0.0089	0.0271	0.0129
Observations	1,226,983	1,226,983	1,226,983

Table 10. Probit regression explaining the probability of a priority valuation. A priority violation is defined as the inclusion of a lower fee transaction in a block when a higher fee transaction was left waiting in the mempool. *Fee/Wght* is fee paid by a transaction over its size measured in weight units. *Blockweight* is the weight of the block, and *Number TX in pool* is the number of transaction in the mempool. The regression includes dummies for large mining pools. Standard errors are clustered per block. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

deviator by strategically orphaning one of its blocks.[27] This is a potential punishment because in Bitcoin, consensus on the correct ledger is achieved via the length of the blockchain. A fork is created when two miners simultaneously add two valid blocks to an existing blockchain. Subsequently miners can add blocks to either fork but only the branch that becomes longer is recognized as the true consensus blockchain and the shorter branch gets "orphaned". All transactions recorded in orphaned blocks are not recognized in the consensus and are thus treated as if they never happened. Rather than an inadvertent fork, miners could deliberately cause a fork by ignoring the block of a (deviating) miner and focus their efforts on another branch. With enough computing power (as in the case of a coalition of large mining pools) they can build a longer blockchain and so strategically orphan a block. The miner of the orphaned block would not only lose the fee revenue from that block but also the often more valuable

[27]We thank Bruno Biais for this suggestion.

coinbase, the reward for finding the block.

To test this possible disciplining channel, we manually collect data on orphaned blocks from various sites on the internet.[28] Since orphan blocks are rare we end up with a small sample of 57 orphan blocks from January 2016 to August 2019.

We then compare the mining behavior of the pool, whose block got orphaned - the victim, to that of other mining pools in a window of 8000 blocks (approximately 4 days) around the orphaned block. Victims are generally large pools, the median victim ranks third in mining share at the time of the orphaned block. In the first column of Table 11 we examine mining behavior before the orphan block. We regress blockweight of all blocks mined by large mining pools (at least 5% mining share) on a dummy for the victim. Event fixed effects control for inter-temporal variation in blockweights. Our findings are consistent with the idea that miners that deviate from rationing by mining larger blocks are more likely to be victim of an orphan attack. Given the median transaction weight of 904 units, victims include about 164 transactions more per block than other large mining pools. The second column also includes data after the orphan attack. While we cannot show that that the victim reduces blocksize after the orphan block we do find that other large mining pools increase blocksize as well, consistent with the idea of a transition away from the collusive equilibrium.

Victim	148573.9***	125017.8**
	(48304.7)	(50702.0)
Post		51361.5**
		(22269.2)
Victim × Post		5322.5
		(69978.6)
R^2	0.696	0.608
Observations	354	723

Table 11. Regression explaining blockweight around orphan block events. Victim is a dummy equal to one for the mining pool whose block was orphaned. Post is a dummy equal to one after the orphan block. Dummies are included for each orphan block event. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

3.7 Economic impact of collusion

To estimate the maximum fees that would obtain under competitive mining we approximate the fees that the lowest type would be willing to bid with the 10%-quantile of the realized fee distribution per block. We note that we do not observe the transactions that were not incorporated in a block. We then define excessive fees as the sum of all fees above the 10%-quantile.

[28] For example `https://bitcoinchain.com/block_explorer/orphaned` Orphaned blocks are not consistently stored in the local database of a bitcoin node. Orphaned blocks are transmitted and thus stored in the local database as long as there is uncertainty which branch of the blockchain will succeed. Nodes do not transmit blocks that are known to be orphan. Thus longer running nodes have more orphan blocks in their local storage (see `https://bitcoin.stackexchange.com/questions/93455/why-do-two-different-fully-synced-bitcoin-core-nodes-differ-in-the-blockchain-si`).

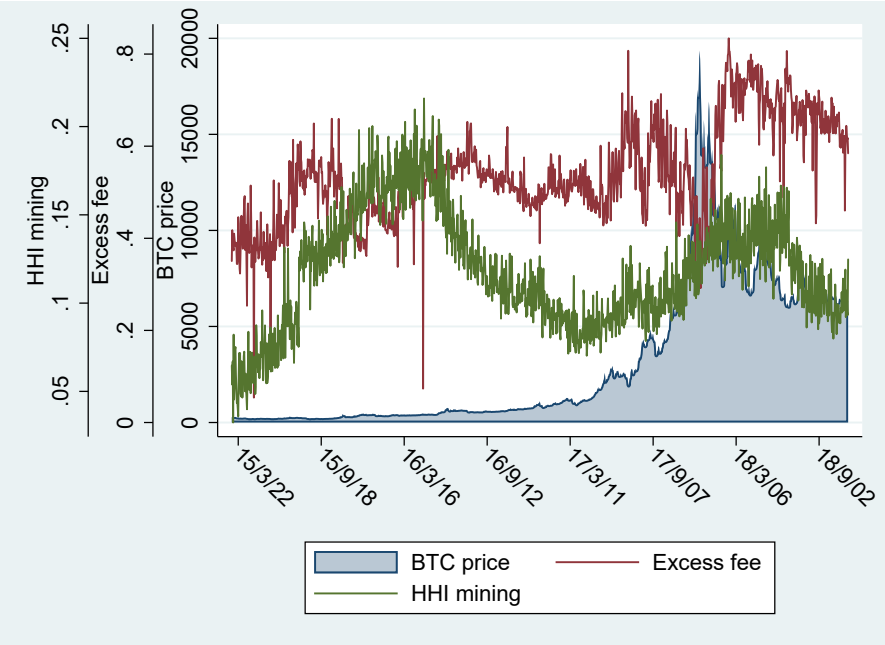

Figure 9. Daily excessive fees, mining concentration as measured by the HHI, and Bitcoin price. Days are defined over UTC.

HHI mining	0.549***		0.414***	0.444***		
	(0.00728)		(0.00595)	(0.00615)		
Fraction pools		0.380***			0.328***	0.319***
		(0.00222)			(0.00206)	(0.00214)
Mined by pool			0.0293***		-0.0000892	
			(0.000719)		(0.000716)	
Post Segwit			0.134***	0.122***	0.131***	0.121***
			(0.000483)	(0.000532)	(0.000462)	(0.000511)
Block weight			2.44e-09***	3.85e-10***	-4.24e-09***	-5.09e-09***
			(1.47e-10)	(1.49e-10)	(1.46e-10)	(1.48e-10)
Tx weight			0.000000145***	0.000000150***	0.000000190***	0.000000190***
			(1.42e-08)	(1.40e-08)	(1.36e-08)	(1.35e-08)
Sum Inputs (USD)			-0.000000487***	-0.000000505***	-0.000000579***	-0.000000581***
			(7.76e-09)	(7.68e-09)	(7.42e-09)	(7.40e-09)
Data			0.215***	0.205***	0.186***	0.181***
			(0.00365)	(0.00361)	(0.00348)	(0.00347)
Resttime			0.00000101***	8.69e-08	-0.00000495***	-0.00000496***
			(0.000000355)	(0.000000351)	(0.000000341)	(0.000000339)
Pool fixed effects	No	No	No	Yes	No	Yes
R^2	0.0278	0.128	0.384	0.403	0.440	0.449
Observations	198,577	198,577	198,551	198,551	198,551	198,551

Table 12: **Regression explaining the fraction of excessive fees over total fees in USD.** *HHI mining* is the Herfindahl–Hirschman index of daily mining shares, *Fraction pools* is the daily fraction of blocks mined by identifiable pools, *Post Segwit* is a dummy equal to one after the introduction of Segwit, *Block weight* is the average daily size of blocks measured in weight units, *Tx weight* is the daily average weight of transactions, *Sum Inputs (USD)* is the daily average input transaction value measured in USD, *Data* is the fraction of daily data insertion transactions (identified by the OP_RET instruction in the script), *Resttime* is the daily average of the time (measured in blocks) until transaction outputs are re-spent. One, two, and three stars indicate significance at the 10%, 5%, and 1% level, respectively.

Figure 9 shows daily values of the ratio of excessive fees over total fees, Bitcoin prices, and the mining concentration as measured by the HHI. In Table 12 we present the results from a regression of the daily excess fees as a fraction of total fees on the HHI of mining concentration and on the fraction of blocks mined by pools. Excessive fees increase in both measures of mining concentration, consistent with capacity management. The results are statistically and economically significant. A ten percentage point increase in the fraction of block mined by pools coincides with a 3 percentage point rise in fees. An increase in the HHI of 0.05, which corresponds to a transition from 5 to 4 equally sized miners, causes excessive fees to increase by approximately 2.15 percentage points.

The sum of fees paid in all transactions is USD 844,537,503.30. 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. After winsorizing, excessive fees for entire whole sample amount to USD 416,006,674.88. Overall excessive fees paid due to strategic capacity management are between half and two thirds of total fees paid.

4 Conclusion

We have documented stylized facts about the Bitcoin protocol. In particular, we observe that there appears to be excess capacity. A significant portion of blocks are empty or not at capacity. We note that this is consistent with revenue enhancing strategic capacity managment. 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.

The preliminary evidence we present 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, J., and M. Brunnermeier, 2018, "Blockchain economics," working paper, National Bureau of Economic Research.

Basu, S., D. Easley, M. O'Hara, and E. Sirer, 2019, "Towards a Functional Fee Market for Cryptocurrencies," *Cornell Working Paper*.

Brauneis, A., R. Mestel, R. Riordan, and E. Theissen, 2018, "A high-frequency analysis of bitcoin liquidity," .

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

Choi, K. J., A. Lehar, and R. Stauffer, 2018, "Bitcoin Microstructure and the Kimchi premium," .

Cong, L. W., and Z. He, 2019, "Blockchain disruption and smart contracts," *The Review of Financial Studies*, 32(5), 1754–1797.

Cong, L. W., Z. He, and J. Li, 2019, "Decentralized mining in centralized pools," working paper, National Bureau of Economic Research.

Dae-Yong, K., E. Meryam, and J. Hongtaek, 2020, "Examining Bitcoin mempools Resemblance Using Jaccard Similarity Index," in *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 287–290. IEEE.

Daian, P., S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels, 2019, "Flash boys 2.0: Frontrunning, transaction reordering, and consensus instability in decentralized exchanges," *arXiv preprint arXiv:1904.05234*.

Debo, L. G., C. A. Parlour, and U. Rajan, 2011, "Signallng Quality via Queues," *Management Science*, 58(5), 44–55.

Debo, L. G., U. Rajan, and S. Veeraraghavan, 2020, "Signaling Quality via Long Lines and Uninformative Prices," *Manufacturing and Service Operations Management*, 22(3), 513–537.

Denicolò, V., and P. G. Garella, 1999, "Rationing in a Durable Goods Monopoly," *Rand Journal of Economics*, 30(1), 44–55.

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

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

Gilbert, R. J., and P. Klemperer, 2000, "An Equilibrium Theory of Rationing," *Rand Journal of Economics*, 31(1), 1–21.

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

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

Liu, Q., and G. J. van Ryzin, 2008, "Strategic Capacity Rationing to Induce Early Purchases," *Management Science*, 54(6), 1115–1131.

Makarov, I., and A. Schoar, 2018, "Trading and arbitrage in cryptocurrency markets," working paper.

Malik, N., M. Aseri, P. V. Singh, and K. Srinivasan, 2019, "Why Bitcoin will fail to scale?," *Tepper Working Paper*.

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

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

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

A 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 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 10 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.

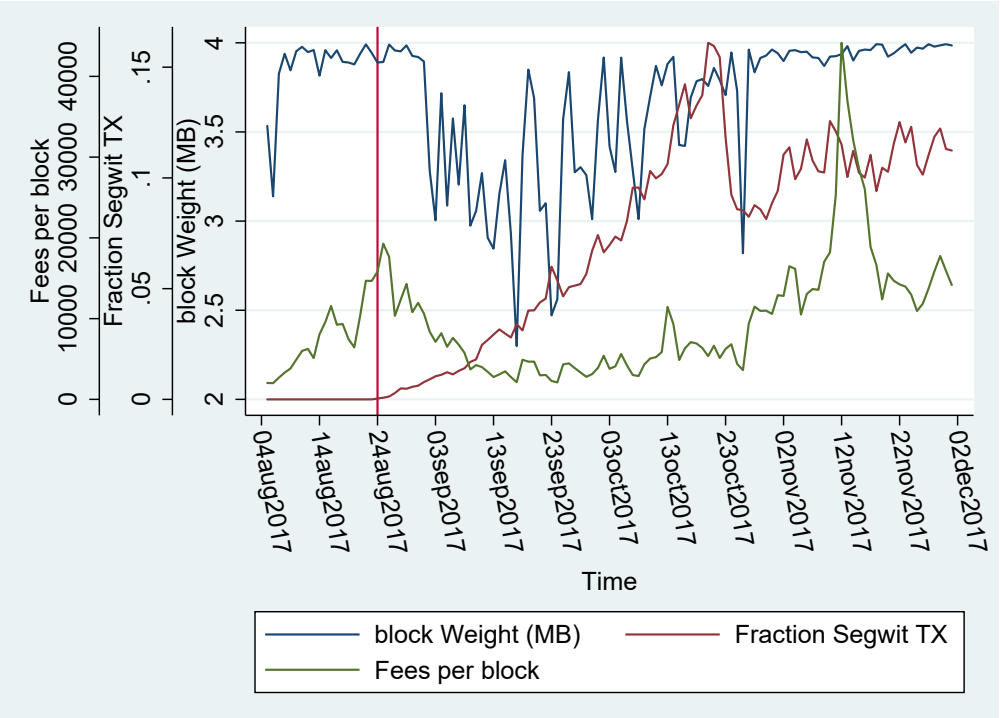

Figure 10. Average Weight per block, fraction of Segwit Transactions, and mining Revenue per block 20 days before to 100 days after the introduction of Segwit. Days are defined over UTC.

B Mempool data

We collect two sets of mempool data to examine transaction demand for Bitcoin. The partially aggregated dataset is used for the monay-left-on-the-table calculation in Section 2.1, the detailed mempool data is used in Section 3.5.

B.1 Partially aggregated data

We collect minute by minute snapshot data of the mempool from Jochen Hoenicke's website, `https://jochen-hoenicke.de/queue/#0,all`. The data ranges from Dec 16, 2016 to the end of our sample period. For each snapshot, transactions are grouped into 45 fee buckets based on sat/byte and contain for each bucket the number of transactions in that bucket at that time, the sum of fees offered by all transaction in that bucket, and the size of all transactions in the bucket. The sample contains over 1.14 million snapshots with a total of 51 million time/bucket observations. There are some gaps in the data, most likely because outages of the server collecting the data. Out of 1,140,218 snapshots we observe 852 snapshots that are more than 70 seconds apart, with the longest gap being 35 hours.

We match mined blocks to mempool data based on the timestamp that the block was mined and

by looking for sharp drops in the size and the number of transactions in the pool. We identify these drops as blocks being mined. We cannot reconcile blocks based on the timestamp alone, as timestamps of blocks are sometimes inaccurate. We therefore have a record of the mempool immediately after a block was mined. For our estimate of money left on the table we start filling any empty blockspace with transactions from the highest fee/byte bucket, until we exhaust this bucket and so forth until the block is full.

Because mempool data is specific to each node, any individual miner may face a different mempool. However, we note that transactions which enter the mempool are shared via peer-to-peer communication. We expect that miners have better hardware, faster connections, and are connected to more peers than our data source. Therefore we provide a conservative estimate of the money miners appear to leave on the table.

B.2 Detailed data

We set up our own Bitcoin node and collect the precise composition of the mempool on a transaction level for a small sample of 277 blocks from block 629,408 to 629,684. We observe a total of 671,025 transactions out of which 660,870 eventually get mined. For each transaction we observe precisely when it entered the mempool, its weight, the fee, any dependencies on other unmined transactions, and if and when it was eventually mined. We also collect information on the weight, time, and transaction count of the mined blocks.

C Proofs

Proof of Prediction 1

(i.) Immediate: If $\bar{\kappa}_t > \kappa - t \leq \epsilon$, then $M_t = 0$, else another transaction could have been processed.
(ii.) Let $f_t^{\bar{\kappa}}$ denote the $\bar{\kappa}$th highest fee in the t mempool. If an order with some fee $\widetilde{y} < y^{\bar{\kappa}}$ appears in the tth mined block then the highest fee orders were not executed at time t, which is a contradiction.

■

Proof of Prediction 2

We proceed with an example. Consider a menu that ascribes a waiting time of at least $\Delta^* > 2$ to fees of f^* and below, while orders with fees $f > f^*$ transact in two blocks. The arrival rate of orders into the mempool is random.

(i) If $\bar{\kappa}$ orders with fees less than f^* arrive every block so that the contents of the mempool could fill a block, then to maintain the expected waiting time, at least one empty block will be mined, and the mempool will contain unmined orders.

(ii) If one high fee order arrives at time t, while the mempool is filled with lower fee orders that have been waiting for $\Delta^* - 1$ blocks, the high fee order will remain in the mempool and lower fee orders will be mined.

■

Proof of Prediction 3

Under the highest fee first protocol, customers choose a fee which maximizes their expected utility given their information as in Equation 1. Consider $\hat{f}$, the highest fee chosen by a customer who obtains a surplus over the next highest fee of $\bar{u} > 0$. A servicer could increase revenue by only offering the waiting time $\Delta(\hat{f})$ if the customer offers a fee $\hat{f} + \epsilon$, with $\epsilon < \bar{u}$. ■