The Three Woes of Bitcoin's Fee Market (and How BlockDAGs Can Fix Them) – Part I: Background

Parallel Thoughts (Shai Deshe) • 10 Mar 2025

> This post was written as a cursory version of a future part in my open book, understanding blockDAGs and GHOSTDAG

The fee market of a cryptocurrency is *extremely important* for long-term security. Once block rewards become too minuscule to fund mining (or have diverged away from existence altogether) fees remain the most dominant subsidy for network security. A crucial aspect of designing a proof-of-work cryptocurrency is understanding the fee market it induces and the dynamics admitted thereof.

In this series:

1. In the current part, we provide a qualitative discussion while quickly introducing some game theoretic jargon
2. In Part II we analyze the fee market induced by a standard blockchain such as Bitcoin, and use game theoretic insight to track these dynamics back to the property that blockchain consensus is *single-leader*: whoever creates the next blocks necessarily wins the entire pot regardless of what transactions were included in blocks created by their peers.
3. In Part III we generalize our analysis to the multi-leader consensus induced by blockDAG networks, and notice how it smooths out many of the pathologies of a blockchain fee market.
4. Finally, in Part IV, we compare the dynamics of blockchains vs. blockDAGs and draw conclussions.

We use Bitcoin as an archetypical example, as it is the only proof-of-work chain utilized enough so that all these phenomena manifest, but these dynamics lay dormant in any one-leader-per-round type consensus, even outside proof-of-work.

I will concentrate on those unsavory properties I like to call *the three woes of Bitcoin's* fee market:

- Race-to-the-bottom: the plummeting of fees when the network is not congested
- Aberration: fees poorly representing acquired quality of service
- Starvation: the perpetual exclusion of low-fee transactions

There is a reason I decided to list the woes in this particular order. The first phenomenon is of an *underutilized* market, where demand exceeds supply to the extent that everyone gets the same service regardless of fees. The last phenomenon is of an *overutilized* network, where excessive demand creates strict entry barriers that exclude all but the wealthiest users. The middle one? Well, it's a consequence of *both*. Whatever direction your market is biased towards, it will dominate the cost of quality of service, removing the desires of actual users from the equation.

In single-leader chains, we will find that the switch happens abruptly. As long as the network is not congested, it is underutilized. As long as the network is congested, it is overutilized. Consequentially, the regime in which supply meets demand, which is arguably the healthiest in terms of monetizing desired service through fees, is compressed into a singular point.

Having accrued a more precise understanding of the observation above, we will describe a simple yet powerful model for the distribution of fees among miners in a *multiple-leader* environment. The model is very simple: if several leaders include the same transaction, the fee goes to one of them selected uniformly at random. We will describe the analysis of the resulting fee market and argue that it widens the desired gap. In particular, we will find that if there are k leaders per round, the network only needs to be at $1/k$ capacity to avoid a race-to-the-bottom. We will also find that the point of starvation (the threshold below which miners will not profit from including your transaction) decays exponentially with the number of blocks per round, showing that many rounds also effectively solve starvation.

The remaining woe is aberration, and it is much more difficult to argue about, as the meaning of "desired quality of service" depends on the *utility* of each user, and isn't uniform across the network. However, we will argue that the most dominant pathologies of a single-leader fee market (namely, the discontinuous bidding war dynamics) are resolved.

> If you find this content educational, interesting, or useful, please consider supporting my work.

Related Works

My observations are a direct continuation of the analysis carried by Yoad Lewenberg, Yonatan Sompolinsky, and Aviv Zohar in 2015. Their paper is a general discussion of inclusive protocols (that is, protocols that include data from parallel blocks), but the part that interests me is Chapter 4, discussing the dynamics imposed by the fee market.

The MEV resistance and oracle implementation schemes currently under work by Sompolinsky et al. are closely related yet not quite the same. The essential difference is that Yonatan's work considers imposing bidding protocols on the multitude of leaders, whereas I consider the natural dynamics imposed by an inclusive fee market, and how they are different from Bitcoin's.

Lewenberg et al. had a particular goal: to prove that parallelism is *not detrimental*. It is definitely a valid concern that parallel blocks might contain too many redundancies to claim that a blockDAG actually increases throughput, making the entire thing somewhat of a wasted effort. Lewenberg et al. find that this is not the case, and that even though some redundancies appear, the fraction of unique transactions remains above a constant. In other word, they find that *the throughput increases linearly with the block rate*, alleviating many concerns.

However, my focus is different, I don't want to convince just you that parallelism is not a disadvantage, but that it is actually a huge *advantage*, and in particular makes it much more feasible to maintain a vibrant fee market.

A **very important** takeaway is that such advantages **are the motivation for high BPS**. There is a common misconception that high BPS is motivated by throughput and confirmation times, but that's only true to some extent. Throughput can be extended much more easily by e.g. increasing block sizes (this might be dangerous for Bitcoin, but for inclusive protocols like GHOSTDAG this has no bearing on the security as long as they are parameterized correctly). Smaller block delays do increase confirmation times, but only in the regime where the network delay is significantly smaller than the block delay. Sure, if the network delay is three seconds, then producing a block per second will definitely provide **much better** confirmation times than producing a block once per ten minutes. But it will remain the same if we further

reduce the block time to one-hundredth of a second. In this regime, the network delay dominates the confirmation times, and all the blocks in the world will not change that.

Game Theory Primer

Games, Utilities, and Strategies

If we want to have a productive discussion, we are going to need some vocabulary. None of the terms I am about to present are exceptionally deep, but they might be a bit abstract for readers with no mathematical background. To provide some concreteness, I will use the familiar game of Poker as an ongoing example. However, one should remember that Poker, Chess, and other connotations of the word “game” are typically far too complicated to be effectively analyzed in these terms.

A *game* contains several pieces of information. First, the *state* of the game: all information about what is going on in this particular instance of the game. The state of a Poker game will specify whose turn it is, the current blinds, banks, players’ hands, the remainder of the deck, and so on. An important part of the state is the *division of information*: what information was revealed to what player so far. Each player has a different set of facts known to them. In the poker table, *any* player knows *only* their own hand, *all* players know the result of the last river, but *no* player knows the burn card.

Then there are the rules. The rules specify, at each point, the choices available to the player. In Poker, if it is not your turn you are not allowed to do anything, and if it is your turn you might be able to call, raise, pass, fold, and so on, depending on the state of the game.

A *strategy* is any way to map the information you have to a choice from the available options. Crucially, a strategy *need not be deterministic*. A player can introduce *randomness* into how they play, e.g. by flipping a coin to determine close calls. A random strategy is often called *mixed*, as it mixes together several of the allowed actions.

Next, there is the *result function*. This function tells us the next state of the game, based on the current state and the decisions made by all players. Poker is a turn-based game, so the result is applied to each player’s decision in turn. However, there are round-based games where all players make a decision

simultaneously, and then all decisions together determine the next state. A good example is *closed bidding*. During the round, each player bids some amount of value, and the winner of the bid is determined based on all values combined.

To analyze a game, we need to somehow model the player's ambitions. We do this by defining a *utility*. A utility is a function that quantifies a state into a numerical value, and saying that a player "has this utility" means that they try to *maximize this value*. For example, the utility of a Poker player is typically to maximize their own hand (though this is not always true! In high-level professional poker it is not uncommon for players to employ a strategy where they deliberately lose a little money for potentially a more meaningful edge later).

Finally, given a utility, we can define a *rational* player as someone who tries to maximize that utility. It should be noted that "rationality" is not a judgment call, just a term for the utility we consider the typical one. There could always be *externalities* - incentives that are not encoded into the utility, and a good mechanism design will provide resiliency against them. For example, Bitcoin mining provides resiliency by only expecting half of the miners to be rational. We will not consider the resiliency of fee markets in the current post, but it is a subject that has been researched, by myself and others, and that I might address in a different post. The bottom line is that fee markets naturally offer resiliency since fees couple disturbing the market with direct costs. In fact, my recurring criticism of the Nano network boils down to a lack of resiliency. In a vacuum, their feeless "market" has a nice equilibrium. But since there are no fees to provide robustness, the slightest of externalities destabilizes this equilibrium and potentially creates new, much worse stable equilibria.

Equilibria

Games like poker and chess are far too complicated for the kind of analysis we are after. Computing or even understanding things like equilibria becomes intractable very fast for even moderately complex games. Luckily, game theory often deals with much simpler games, the most famous one is probably the *prisoner's dilemma*.

Imagine a detective interrogating two suspects separately, to each suspect he says: - If **you both** snitch on each other, **you both** get **two years** - If **only you** snitch, **you** get to **walk** but your friend gets **ten years** - If **none of you** snitch, both of you get **one year**

We can neatly pack all of this information into a *payoff table*. In this table, each row corresponds to a (pure) strategy of one of the players, and each column to a strategy of the other player. In each cell we write a pair of numbers (x, y) that describe the utility of the first and second player in case they chose these strategies. For example, the payoff table of the prisoner dilemma looks like this:

$p_1 \setminus p_2$	Snitch	Don't
Snitch	$(-2, -2)$	$(-10, 0)$
Don't	$(0, -10)$	$(-1, -1)$

You will note that the numbers in the table are negative. Why is that? Well, duh! Because more years in prison = less good!

Say that you are one of the suspects, what would you do?

Note that if you choose to be loyal to your friend, he can improve his utility by switching strategy from not snitching (which would land him five years) to snitching (which will set him free). This might make you averse to loyalty and incentivize you to snitch. With some more thought, you realize that **your accomplice is facing the same quandary!** They might want to stay loyal, but they know that they are taking a risk, and the only way this risk pays off is for you to take the same risk. The only strategy where this kind of loyalty loop does not occur is if *both snitch*. In this case, no one has anything to gain by changing strategy. Sure, there is a strategy which is better for both, namely both staying loyal, but this strategy is *unstable* since any one of them can *improve their situation at the expense of their accomplice*.

We see that the mutual snitching strategy has a special quality to it. In game theory parlance, this quality is called being in *a (Nash) equilibrium*.

It is important to note that a Nash equilibrium is not necessarily a utility-maximizing strategy! That the players want to increase their utility as much as possible does not mean that they will be willing to take any risk. Assuming players will prefer the equilibrium strategy captures this tendency well, and this assumption is backed by many empirical data.

Expected Values

When we finally discuss mining, we will find that in BlockDAGs, the profit of a miner is not deterministic but probabilistic. In other words, they can't tell exactly how much fee they are going to earn before seeing all the other blocks.

That means that their profit is not a deterministic value, but is rather what we call a *random variable*: something that we don't know exactly, but we know how it *distributes*.

Hence, when we consider the "profit" of a miner who follows a particular strategy, we are not actually considering a *concrete number* but a *distribution*. The way we bridge this is by replacing this distribution with its *average* value, which is what we call its *expectation*. (For more details, you can see the appendix of my book).

I want to take a moment to explain roughly what this means, and why it is a reasonable way to quantify the profits of a miner.

Say we play a game where I flip a fair coin and if the result is heads, you give me one kaspa (otherwise, nothing happens). Intuitively you can probably see what I mean when I say that "my expected value is one half of a kaspa": in half of the cases I gain one kaspa, and in half of them I get zero kaspa, so in the "average" case I get the average between zero and one, which is one half.

This intuition of "averaging the output over many repetitions" is not how expectation is usually *defined*, but it is equivalent under many conditions, including all probability in this post (such equivalences between the standard definition of the expectation, and the average of infinitely many samples, are called laws of large numbers).

For illustration, let us discuss how fast the coin-flipping game above *converges*. That is, how far do we expect the average to be from one-half if we have repeated the experiment many times?

(There is a subtle nuance here that is probably not of any concern to most readers, but I'll point it out for the math-savvier crowd: the appropriate form of *convergence* here is a bit different than the convergence of a sequence of numbers you might know from calc 101. In particular, if we flip a coin one million times, there is still *some* chance that all flips are heads. Yes, it is a very small chance, but it is not zero and will never be zero. There are several ways to overcome this leading to different *modes of convergence* that are *not* equivalent. Implicitly, the discussion below implicitly a weak mode of convergence called convergence in probability, corresponding to the first known law of large numbers, called the *weak* law of large numbers).

When we think about expectation as averaging the outcome of many repetitions, we have to remember that each repetition uses *fresh randomness*.

So say that we repeat the game above many times. However, since coin flipping is a complex physical procedure, I don't have nearly enough dexterity to flip it the same every time. Not even close. The result of any repetition is *independent* of previous results.

Say that in the first repetition, I got heads and won. I also won the second time, hooray! However, in the third time, I flipped tails and won nothing. Across all three attempts, I made two kaspa, making my *average* gain $2/3$ kaspa. If I repeat the process many times, the fraction of flips that turned out heads will get increasingly close to one-half, whereby my average profit converges to my expected profit of one-half of a kaspa.

Why should the fraction of fair coins converge to one-half, and how fast does this happen? Using a theorem called Chernoff's bound, one can prove the following: if we flip a fair coin n times, the probability that we got the same result more than $n/2 + \sqrt{6n \cdot \ln n}$ is less than $2/n^4$.

Don't get intimidated by the expression above. It might be a bit ugly, but all we really have to do is substitute numbers into n and see what happens: If we flip a fair coin one thousand times, the probability we get more than 700 heads is less than one in 500 billion. If we throw it a million times, the probability to get more than $501,000$ heads is less than $0.00000000000000000000002$.

This instrumental description of expectation also reveals why it is the appropriate metric to describe mining: because miners are in it for the *long run*. They aren't mining for just one or two rounds but, at the very least for thousands of them (Bitcoin has 1440 rounds per week, the round length in Kaspa is a bit harder to pin down as it fluctuates with network conditions, but it is in the hundreds of thousands per *day*). Consequentially, the *expected* revenue of their mining strategy provides a *very good approximation* of their *actual* fee revenue.

Remark: Interestingly, the quality of the approximation decreases with the number of **rounds**, but is completely agnostic to *how long* each round is in terms of real-time (or alternatively, in terms of the number of offering turns in each round). Hence, in high BPS networks like Kaspa, a shorter period of time is required to strongly approach the average. For example, in Bitcoin you'll have to wait about a week to obtain the same quality of approximation Kaspa will provide within two minutes. This (and another important property called variance) actually has strong implications on entry barriers for miners. A small miner is arguably less resilient to the tides and troughs of high-variance mining and might prefer a network where uncertainty averages away much faster.

Part II >>

> If you find this content educational, interesting, or useful, please consider supporting my work.