TADGE DRYJA: OK, so today I will talk about mostly fees, which is an interesting and somewhat contentious issue in Bitcoin and all these other systems. So schedule for today, problem set 3 description, we'll talk about fees, these unknowable acronyms-- Child Pays for Parent, Replaced By Fee-- and then talk a little bit about long-term incentives in Bitcoin, and how that relates to fees.

OK, so the problem set 3 is to grab UTXOs. Hopefully it's fun. You're going to be making transactions on the real Bitcoin test network. So one of the parts of it that hopefully is not too hard is downloading and install Bitcoin D. So I'll put a link. The current, most up-to-date version is 0.16.0.

I believe the next version will just be called 17, because a lot of the developers are like, this is annoying, it's always starting with a zero, it's like 0.1, 0.2, 0.3. And they're like, well, at some point, should we get to version 1? And they're like, well, everyone's going to think version 1 means it's better. Let's just call the next version 17 instead of 0.17.

So download that, and then sync up to testnet. If you just run it by default, it will sync up to the main network, which will download something like 170 gigabytes. You don't want that. And the homework assignment cannot be completed on the main network. There's no free money-- I haven't put any free money on the main network.

However, on the test network, it'll work. And it's about 11 gigabytes of download. So I think most people, on your laptop, probably have 11 gigs free. It's doable. And it's kind of fun to see how it works. So if you don't have access to 11 gigs, you can probably get an Athena node to do it. It's not too big. You can run a tiny little node with not much RAM, not much space, and it'll work. You can also sync up on a Raspberry Pi. It just takes a little longer. So I don't think it'll be too hard to do.

You can probably get around-- sort of the goal of the assignment is to check out the actual Bitcoin D and Bitcoin CLI. You can use a block explorer if you want. I'm not going to say that's cheating, but it's also not as fun. So the block explorers websites will help you find a lot of things, because they have different databases and different indexes so you can sort of search through things. It's more fun if you write it yourself. But if you try this,

OK, that works too. Yeah, so hopefully you get familiar with the Bitcoin CLI. If you want to do your own scripting with Bitcoin, you can use anything you want. So Bitcoin CLI is a command-line utility that will let you interact with the Bitcoin node.

So for example-- is this readable? Kind of, right? So if I want to get peer info. I want to see-- this is a computer down on the first floor, running Bitcoin Mainnet, I want to see, OK, who's connected to me? And then it'll tell me. It returns JSON, and it tells me all about the people I'm connected to. If I want to say, get wallet info, it'll tell me I don't have any money-- things like that.

And then there's, like, Help. It's weird and it's not super well documented. It's fairly well documented in this stuff. There's also tons of undocumented hidden RPC calls that are deprecated, or new, or programmers put in and they don't want people to use them. They only want to use them for themselves. So it's kind of a mess, but it's fun to see exactly how this software works. So hopefully you can try that out. And I will add more UTXOs to grab in the next few days. So that might be fun.

You could, if you wanted to be a jerk, just steal all the money, and then no one else could do it. But please don't do that. I mean, this is not as adversarial a setting as actual Bitcoin. If someone does it, I'll be like, OK, fine, and I'll just like reseed the place. So hopefully we don't have too many trolls trying to prevent everyone from having fun with this assignment.

That said, there is at-- aspect some of them are-- the one I put in the assignment last night, there's five outputs that you can grab. One of the sort of hints-- there's only five outputs. So the first five people to grab it get the coins. So there is a little bit of timing stuff there. Any questions? Cool? Yeah, so if you have questions about it, office hours tomorrow, also IRC, stuff like that.

OK, so today's main topic-- transaction fees. So we described earlier that the transaction fee is just the difference between the sum of the input amounts and output amounts. What's interesting is it's implicit. So if you actually look at a transaction, you can't tell what the fees are. You have to look at its ancestors to tell what the fees are.

So for example, if you want to go through here and-- oh, how am I going to do this? Jeez. If I want to say, OK, what's the most current block? getblockchaininfo. OK, so the most current block is this. And then I can say, OK, getblock-- can I just say getblock? Yeah, I want to get that block.

OK, there's all the transactions in this block. I can do getblock verbose true, and it'll give me all the transaction data. But instead, I can just pick one of these transactions and say getrawtransaction, get that thing.

Oh, here's another weird thing. So it'll just give you hex, which is not very useful, unless you somehow can read hexadecimal. And then in the actual thing, it says, like, examples. Wait, true? Did they change it? You used to have to do 1 at the end. Yeah. But true works too? It used to not work if you said true, and you had to put the number 1 instead of true.

There is a lot of weird bugs in Bitcoin that have been there for years, and I guess they fixed that one. The most egregious is probably the multisig bug, where when you're spending from a multisig output, you have to push a zero-byte onto the stack first, or else it doesn't work. It was just a bug Satoshi had. And now it's consensus critical. And people joke about it, like, oh, we also have to put a zero-byte in whatever new RPC we create, because there's just, like, bugs.

Anyway, so for example, this transaction that I just brought up, this is how it'll look. Here's the TXID, it's 370 bytes, it's got this lock time, which-- ooh, I should mention that. I'm guessing this transaction was created by Bitcoin Core Wallet, not a different wallet. That's a hint. It's got one input-- no, two inputs. OK, so inputs are specified-- this TXID input, this TXID input-- two inputs and one output of value, 0004500. We can't tell what the fee is from this. No, sorry, two outputs. Here are the two outputs.

So we can tell these two outputs, however, we don't know what the fees are. We know the total output amounts, but the input amounts are not encoded in the transaction itself. In order to figure that out, we'd have to go to this transaction and see what vout 0's output amount is, which we can do. And it's 0052-- you know, so we can add it up that way, but it's not written directly, which is a little weird.

And the idea of the fee is, whoever mines that block that includes that transaction gets the fee. So the sum of all the inputs and outputs within a block do add up, modulo the new coins that are created. But long-term, when there's no coins being created, when they've all been created, you can say, OK, we can look at a block, add up all the inputs spent in all the transactions in the block, add up all the outputs created in all the transactions in the block, and those should equal. Otherwise, it's invalid.

Well, they don't have to equal. You're always allowed to destroy coins. So if the miner says, hey, there's all these fees that can be paid to me, and I just don't grab them, you can do that. So it's a less-than-or-equals operator that-- it's the check. And people have destroyed coins, usually inadvertently. There was one in January where there was a block that just had a zero output for the miner reward, so all whatever, 13, 14 coins were just destroyed. Yeah, it seemed like a bug. So that's how fees work.

What's interesting is, you don't know who you're paying when you create the fee. You sign sort of, OK, here's my transaction, I sign it off, whoever confirms this transaction gets $5 or whatever. And you throw it out there.

So it's kind of interesting. There could be legal-- I've heard there's legal questions, like, well, what if one of the miners is in Iran, now you're paying someone in Iran, are you in trouble for that? You don't know. You can't tell who going to mine it beforehand.

And generally, we talk about fee rates, or sort of a specific fee-per-byte. And it's expressed in Satoshis per byte, and one Satoshi being the minimum possible bitcoin amount, which is 1, 2, 3, 4, 1, 2, 3 zeros, so seven years. There's eight digits in bitcoin. And you usually keep that. So you pad it out. So even in something where it's 0.0134, they put the extra four zeros. In the actual code, this is just a 64-bit signed integer, and there is no decimal point.

So it's sort of a UI thing to put a decimal point in so that people can think of, oh, that's two bitcoins, or that's half a bitcoin. But really, the code only deals with the Satoshis, and the decimal point is purely UI layer, which can be confusing sometimes, when you're dealing with JSON, and when you're dealing with different programs, tossing numbers around. You can get a factor of 100 million off, sometimes. It's a big enough factor that, usually, it's pretty obvious.

So from a miner's point of view, they want to prioritize, based on the TX size and the fee rate because space is limited. So one of the other rules that's probably one of the most controversial and argued-about rules in Bitcoin was a 1-megabyte block size limit. So the idea is, OK, you can put all these transactions in the block, but the total block, when serialized, including the headers and everything, needs to be 1 million bytes or less, which is now not quite true because there's ways around that. But there's still a hard limit of 4 megabytes today.

What's also interesting is it's totally unrelated to the amount of coins being transferred. So in that sense, the fee is flat. Whether you're sending 100 Bitcoins, or one Bitcoin, or a millionth of a Bitcoin. The miner doesn't really care. It's just based on space. You're paying for your data usage of the network, and your actual sort of monetary value transferring is unrelated to that.

To some extent, later on, you might think, well, miners might try to charge people sending more money more because they're more able to pay. But so far, we haven't seen anything like that. It's mostly just the miners looking at, I've got a megabyte of size to parcel out to people, I'm going to try to pack that in is at as high a rate as I can. And whether someone's paying a little bit or a lot-- whether someone is moving a little bit or a lot, the miners don't care. Any questions about these things? Yeah.

AUDIENCE: I have a question about actually the miners getting the fees. So how does that-- because that's not part of the Coinbase transaction.

TADGE DRYJA: It is, it is. So if we look-- how am I going to do this?

AUDIENCE: You could look into the Coinbase transaction in the block and find out what the fees would be.

TADGE DRYJA: Well, if you assume that the block is valid. So OK, the first transaction is always the Coinbase transaction. That's little. So I'm going to do getrawtransaction, get this one. And yeah, it says, Coinbase. So Coinbase's arbitrary data, you've got a sequence number, there is no witness here-- or is-- OK, anyway. Yeah, there is witness.

OK, so this block is-- well, two blocks have come out since I first looked at it, because there's now two confirmations. This is SegWit-related, which I think I'm going to talk about Monday. So that's sort of this weird opportune thing. This is the actual Coinbase payout that the miner is getting, and it's 12.79. So he got almost 0.4 extra coins in fees, which is what, like $4,000, pretty good. It was super high a few months ago. So it all gets aggregated. 12.5 comes out of nowhere. 12.5 is the new coins being generated. And then the 0.398 is the sum of all the fees in that block.

AUDIENCE: I guess you don't know [INAUDIBLE]

TADGE DRYJA: Well, you have to look at all the transactions. So if you look at a transaction, it doesn't say in the transaction itself. You have to look at a transaction, look up its inputs, see the sum of all the inputs, and then subtract and say, OK, I can calculate the fee for this transaction, calculate the fee for all these different transactions, add it all up, it should equal-- and it does exactly equal 0.398 da da da da. So for the computer, it's pretty quick. But yeah, it is a little bit weird accounting to try to add it all up yourself. Any other questions about fee stuff, about this? Yes.

AUDIENCE: Do miners choose what blocks to mine based on their [INAUDIBLE]?

TADGE DRYJA: Yes, but I'm not sure I'd put it that way. The miners construct the block to mine. So in the case of-- here, getblock head dash 30, I don't want to show all of the-- but this is, OK, the miner mined this block. Here's the size, weight, height, version, vertical route. All these transactions that get included are produced by the miner. So the miner sort of aggregates the transactions themselves, and then decides to mine. So I think I talk about that in the next-- wait, do I say that? Yeah, OK. I'll talk about that in the next slide.

So basically the fee rate is set by the transaction signer. When you're creating a transaction-- OK, I want to sign one Bitcoin, I will spend 500 Satoshis, my transaction is about 250 bytes, OK, that's a two-Satoshis-per-byte fee rate. You choose your fee rate, sign-- the way you increment and decrement your fee rate is to say, OK, I've got some change output, I'm going to increase or decrease the change output amount. If someone's saying, hey, pay me one coin, I can't really change their amount and say, like, oh, sorry man, there was a high fee, you didn't get a bitcoin.

And that's sort of a cultural-- that's not a software thing. If someone says, pay me a coin, you need to pay the coin. It's generally the sender who's covering the fee. Which is different than in credit cards, right? In credit cards, it says $10, I swipe my card, I get charged $10, and the merchant is the one absorbing the fee. And I don't even know what the fee is. There's actually contractual law that I'm not allowed to hear what the fee is, the merchant can't tell me, things like that. But in general, in Bitcoin, this fee is set by the sender.

The transactions, however, are chosen by the miners. So the idea is, you create your transaction, set your fee, throw it out there, everyone sort of broadcasts it to everyone else. And the miner then looks at all these fees, looks at all these transactions, and chooses the best ones. It's essentially an auction process where you're bidding for us for space in a future block that has not yet been created. So the simplest miner-side algorithm is, take the mempool, sort the mempool transactions by their fee rate, choose the top megabyte, put them all in order, compute a merkle route from all those transaction IDs, and then mine it. So the-- yeah, I explained the merkle route stuff, right?

So you're going to pick a couple thousand-- 5,000, maybe-- transactions, sort them by fee rate, put them all in a row, hash it a bunch of times, and then just that 80-byte header you keep mining. However, this keeps changing. This mempool changes every second. There's new transactions coming up multiple times a second. And so you're going to re-compute that every second, probably, and say, oh, this new transaction came out, oh, the fee's really low, ignore it, that doesn't meet my threshold. Versus, new transaction comes out, oh, it's got a really high fee rate, OK, put it in near the top, and push out one or two transactions at the bottom. And now I'm mining a block that will have slightly higher fees.

So this is sort of a real-time process, where this is constantly changing. The mining is even faster, so you're going to mine trillions and trillions of attempts for each specific merkle route. But the merkle routes are also going to change multiple times a second. The nonce is going to change a gazillion times a second. Yes.

AUDIENCE: How do you know that you're not mining a stale transaction?

TADGE DRYJA: So you know it's in the mempool, you know it hasn't been-- well, you don't know, you're pretty sure that it has not been confirmed in a block already. But it may be the case that someone just came out with a block a few seconds ago, and you haven't seen it, and then you are mining a stale transaction.

The thing is, whether the transaction is stale or not, even if you mine a block and it's got a completely different set of transactions than the block you were unaware of, you still lose out. Because you're pointing-- so OK, this comes out, but you're not aware of it yet, and you mine this. Even if they have a completely disjoint set of transactions, it doesn't matter because they're still pointing to the same parent block. So you still, now, have two blocks of the same height. You might win. If this happens, hey, maybe this one gets built upon.

Generally, miners pick the first seen block to build off of. But you could change that. And you can't really verify that that's the case. So yeah, it is possible that you're mining transactions that have already been included in the block that you're not aware. Yes.

AUDIENCE: Because blocks are changing all the time based on the transactions you want to put in it, are you kind of mining different blocks at the same time? So the first one, you try-- have started mining, do you keep mining that one or no?

TADGE DRYJA: You could. So in practice, probably yes, but on a very small timescale. So what you'll have is we'll get block template, you'll have-- it's like network latency. So you'll have, let's say, a bunch of different warehouses that all have thousands and thousands of these mining chips. And they will connect in, and they'll pull every second or so, and get the current transaction list to mine. And that'll get updated every second.

And so there may be sort of lags between different warehouses, between different machines, that are mining slightly different sets of transactions. But generally, they're going to be synchronized. Over the same time scale of a few seconds, they're all going to be synchronized.

They're going to be updated pretty quickly. Because every time a new transaction comes in, that's potentially more fees that you can get. So you want to push that out and update it. Let's say a transaction comes in that's got a nice $5.00 fee, and you can push out some $4.00 fee transactions, that's an extra buck. And if you don't get it in time, and mine a block anyway, hey, you still mined a buck, but that $5 fee transaction now goes to the next person mining the block. So you want to minimize your latency so that you get things as quickly as possible. But there will be some skew. Yes.

AUDIENCE: How does mining stale transactions affect the block order and block transaction fees?

TADGE DRYJA: Sorry, mining what transactions?

AUDIENCE: Stale transactions.

TADGE DRYJA: If you have-- wait, wait. OK, wait. Stale transactions? A transaction that's already been in a block?

AUDIENCE: Yeah.

TADGE DRYJA: It just invalidates the whole thing. If you have if you have a transaction here that was already confirmed here, this block is invalid, and no one will-- they'll just drop it. So stale transaction isn't really a term. It's a stale block. This is a stale block because, well, someone made it, but it got orphaned out, and no one's using it.

AUDIENCE: So the answer is, there's no reward for that, right?

TADGE DRYJA: Worse. You lose all your-- you lose everything. So if there's transaction A in here, and you try to include transaction A in here, the block itself is invalid, even if it has valid proof of work. So you lose everything.

The thing is, in practice, it's not a problem. Because you're pointing to this block, so you must have already seen it. So it's really easy to just say, well, don't include any of the transactions in here.

OK, wait, sometimes it's a little complicated. Sometimes you see this block, and then you immediately want to start mining the block on top of it. But it's a fraction of a second, and you haven't actually validated it. And you don't even know what's in it. Maybe you only have the 80-byte header.

So one optimization which is sort of bad in a way but optimal for the miners is build a block that's empty on top of it. You haven't even seen what transactions are in this block. You don't even know if it's valid. But it's probably valid, because most blocks are.

And so what you do is you build on top of it, you point to it, but since you don't know what transactions are in it, the safest thing is just to include no transactions in your so there's no overlap. And you only do this for a fraction of a second until you've figured out the Transaction List here and subtracted all the-- removed all those from your set. But it saves you maybe 1/10 of a second, 1/20, whatever it is. Which, over time, statistically, you could be mining in that 1/10 of a second. You don't you want your chips to not be doing anything productive for those fractions of a second.

So that does happen. That is not software that's in the normal Bitcoin D, but miners have written their own software to do that. And so you will occasionally see completely empty blocks with just a Coinbase transaction. And they're almost always very soon after the previous block, where you see one that comes out at 10:30 and 27 seconds, 10:30 and 29 seconds, and it's empty.

So that does happen, and that can work. Although what was it, two years ago that happened, where a miner mined a blocked that was invalid. Another miner mined a block here, very quickly, that was empty. And then a bunch of miners started building off of it because they're like, well, this one looks valid, there's no transactions in it. And they mined like six or seven blocks in a row, and the whole thing was just invalid. Because they figured, the stuff I'm building off of must be OK. So they lost some money there. This makes sense, right?

Sort the mempool by fee rate, choose the top megabyte, compute a merkle route, mine. Why is it not this simple? And I mean on an algorithmic way. If you're just like, yeah, sort, choose fee rate, sort, quick-sort or whatever, that's super quick, and then mine. Why is it actually, complexity-wise, not-- what is it, n log n for quick-sort? It's definitely not n log n for the optimal. Although, wait, quick-sort is worst-case scenario n-squared [INAUDIBLE] right? No, it's way worse than n-squared.

Anyway, any guesses onto why the algorithmic complexity is actually quite a bit worse?

AUDIENCE: Is it because the example keeps on changing?

TADGE DRYJA: No, I'm saying, even at any-- yeah, that changes it over time. But at any one state, let's say, here's a mempool, find the optimal transaction set to put into your block.

AUDIENCE: Because Bitcoin is slow to search, so you have to go into each transaction and find the fee?

TADGE DRYJA: Yeah, but that's still n log n, right? Yeah, just because the look-up is slower, that doesn't change the O.

AUDIENCE: The problem is [INAUDIBLE] because you have to find [INAUDIBLE].

TADGE DRYJA: Yeah, so there's dependencies. So the thing is, there may be a transaction-- you can't just take all these transactions independently and sort them. There may be a transaction with a very high fee rate which depends on a transaction with a very low fee rate. And so you know it's spending an output that hasn't yet been created. So now it becomes-- it's NP-hard.

In practice, the heuristics are fine. But it is pretty ugly in that you're like, well, this transaction has a very low fee, I'm not going to include it. This transaction that depends on it has a high fee rate. Now I have to take the sort of mean of the two fee rates, and keep computing those things. So it can get quite complex.

In practice, it's usually OK. You have some heuristics in the code that sort of don't deal with that. Also, I think you have max-- there's no hard rule, but the algorithms that mine usually have a max depth of like three or four transactions, where they say, look, if there's a chain of like four dependent transactions, I'm not going to keep going down that path. Because you could potentially have enormous chains, where you have 100 transactions, each depending on another. And then the one at the very end has a very high fee rate, and you're going to have to go through all this. So they limit their search to like four or something like that.

Oh, sorry. Yeah, so the TX dependencies make it kind of hard. Cheap transaction which allows an expensive transaction to be confirmed, we call this Child Pays for Parent, in that you've got this-- in a graph, you've a child transaction, and it's essentially paying for the parent transaction by having a higher fee.

There is no Parent Pays for Child, in that if you've got a transaction with a high fee rate, and then a descendant transaction with a low fee rate, you just take the parent and you ignore the child and someone else in a later block can grab that child transaction. However, the dependency graph is, no, if you want in on this child, the parent has to be confirmed first. It actually has to be confirmed earlier in the list of TXIDs in the block.

So it's not-- so like in this, when you look at a block, the transactions are sequenced. And this transaction can spend that transaction. But you need to be able to go in through linearly. Update your UTXO set at every row here, and it should be consistent.

AUDIENCE: I have a question.

TADGE DRYJA: Yes.

AUDIENCE: You mentioned LVIV, those are [INAUDIBLE] terms. You mentioned a threshold earlier. How is that threshold determined?

TADGE DRYJA: Oh, OK, so there isn't really a threshold. It's just at any point, you're looking, and you're like, well, my minimum fee here is 30 Satoshis per byte. Given the block I'm trying to mine, what's my lowest fee? OK, 30 Satoshis per byte. So if I see one with less than that, I know right away I can ignore it. But yeah, it keeps changing every time a new transaction comes in that displaces an old one, now your minimum fee changes. So that's just sort of an easy way to look.

There is also a relay threshold which I believe most nodes now have at one Satoshi per byte, where if you're not mining and someone gives you a transaction which is less than one-Satoshi-per-byte fee rate, you will ignore it and not pass it on to your peers. That's sort of an anti-spam kind of thing. And it may be too high. It's also not a consensus rule, and you can set it to whatever you want in your config file. And nodes won't get mad at you. They won't ban you if you send something low.

So that's sort of from the miner's side. From the miner's point of view, you keep seeing all these transactions, you sort them, you have to deal with the sort of descendent transactions.

But you try to get a good block, try to mine it. And you don't really care what people are doing, you just want to make a lot of money. You just want to maximize for fees.

From the person transacting, on their side, you want to minimize fees. You don't want to pay these miners. They already make gazillions of dollars anyway from all the new bitcoins they're generating-- bunch of jerks. So you set your fee to one Satoshi per byte. So you know it'll be relayed, you sign it, and you broadcast. Easy, right? So what's the problem here? Nobody--

AUDIENCE: [INAUDIBLE]

TADGE DRYJA: Doesn't confirm, all the miners ignore it. This is actually a huge-- well, OK, no, probably the biggest sort of UI problem in Bitcoin is that people have, for years and years, lost all of their coins due to theft, or loss, or whatever. That's the big one. But other than that, this is the biggest UI problem. This is the biggest problem in Bitcoin for users. I made a transaction, it hasn't confirmed, what the heck? And I sort of had, on Reddit, BTC fees, WTF.

[CHUCKLING]

Like, why are BTC fees so high? There's constant people complaining, what the heck, why-- are you kidding me? Trezor says $50 in fees. Am I missing something? People get mad. And then there's also, I made a transaction last week, it hasn't confirmed. Bitcoin doesn't work. Bitcoin sucks. I hate this thing.

So huge UI problem, really poor user experience. A lot of it is, also, the software is bad. Wallets-- OK, a couple of years ago, wallets would always just set a zero fee. There weren't fees in 2012. You just didn't pay any-- 2013, right? I never paid fees. It was awesome. Because there was this block-size limit of one megabyte. But no one was ever running up against that limit.

The miners would just say, look, I'm not even going to sort anything because there's 200 kilobytes worth of transactions in the mempool. I just mine all of them, whether the fees are zero, whether the fees are one Satoshi, whatever-- whatever I can get, just mine it all.

Now you've got this constrained space, and so you're optimizing-- the miner is. But years ago, you weren't. And so the wallets, a lot of times, they would either have zero fee, completely, a fixed fee per transaction, which is also really suboptimal-- they just say, OK, I'll pay 100 hundreds Satoshi per transaction. Some transactions have a lot of inputs, are much bigger, so that's a highly variable fee rate which the user can't control.

Or they'd have a fixed fee rate, where they just say, I'm going to set five Satoshis per byte, and that's the fee rate in this wallet, and you don't allow the user to change it at all. That's also really suboptimal. Once the fee market starts changing and people start paying more, this wallet software will not work.

Or you can have the user choose a fee rate, which seems great, but users are like, I don't know. You're using Bitcoin, what fee rate do you want, 10 Satoshis per byte, 20, 40, 70? What's a Satoshi? What's a byte? This is not a great problem to hand off to the user. How do I even determine this?

There are sites, and the sites aren't very good, either. Where was it-- yeah, here is what used to be 21.co, with all the Raspberry Pi guys. And then this is their fee rate thing, which is a really hard-to-understand chart. What does this mean, unconfirmed transactions, transactions today? What are these numbers? I still don't really understand this site.

And at the bottom it says, fastest and cheapest transaction fee is 50 Satoshis per byte, shown in green at the top. So I guess this, or this. Thing is, huge overestimate. I think one Satoshi per byte is probably OK today-- maybe five.

This is a better site-- also complex-- where you can say, OK, well, here's the mempool size in megabytes, sorted by fee rate. And you can see that, well, at 8 o'clock, mempool was basically empty. At 8 o'clock, you could have paid zero and it would have gotten through, but people are paying more now. And it's also kind of crazy to see that some people are paying 200 Satoshis per byte, even though, generally, over the course of a few hours, zero Satoshis per byte will also confirm. Are they really time-sensitive? Probably not. Probably, their software is crummy, and their software just estimates 200.

So it's easy to make fun of people and be like, these guys are idiots. It's actually not that easy to estimate what fee you should issue a transaction with. It's also sort of asymmetric in that a lot of exchanges, someone says, I want to withdraw my 10 bitcoins. I traded, this is a lot of money, withdraw, click. It doesn't confirm for six hours. The customer gets super mad and starts calling customer support.

Whereas if the exchange is like, look, we're just going to issue all of our transactions with 200 Satoshis per byte, they always go through the next block. Yeah, we lose money. But maybe we get the customer to pay for it. Yeah.

AUDIENCE: Do you think it could be people using maybe percentage of their transactions, which obviously doesn't make any sense, but--

TADGE DRYJA: I don't think there's any software that does that

AUDIENCE: --finance firm, and that's just what they're used to. That's how they're used to paying fees, [INAUDIBLE]. I don't think so.

TADGE DRYJA: I've never seen any software that does that. I've seen some software that does really dumb stuff with regard to fees. I haven't seen that. That would be a new thing. So I don't think so, but maybe.

AUDIENCE: I presume that it's the exchanges or wallet, companies just feel like their customers are happy are happier and they're charging their customers. It's just passing through the cost to customers. But a quick question-- why not just do an algorithm that takes the last six to 10 blocks, maybe time-weights it, and do an algorithm base, so an actual experience in the marketplace?

TADGE DRYJA: Yeah, that is what the current Bitcoin D algorithm does. Even that can be off sometimes. And it allows sort of a estimate-- OK, how quickly do you need this? Do you need this in the next hour, do you need it in the next six hours, the next half hour? So yeah, the one in Bitcoin D is better. But it's not-- it's easy to say, oh, this should be easy.

The thing is, it's a weird market. You're bidding for something that you don't quite know-- and you don't know what other participants are going to pile in. Sometimes-- so something like this, oh, a block didn't come out for half an hour, and a bunch of other people came and submitted higher bids after you. And so now you got out-- in practice, well, yeah, someone who submitted a bid here, a transaction here, a block came out there, and-- let's zoom in for an example. Well, that's too low.

But I submitted something here. I think it will get in. But then all these other people came after me, and then the block didn't confirm mine. It did later, another few minutes later. But it's hard-- there's never any determinism in how it's going to work. Yes.

AUDIENCE: So if fees are always lower around 8:00 AM or 4:00 AM, why not just pool transactions for around that time?

TADGE DRYJA: They're not, because it's all over the world. They do tend to be lower on the weekends. People don't make as many transactions on the weekends. And so, yes, sometimes people will say, hey, I've got these transactions. In practice, if you say, I don't care how long it takes, just set a low fee and wait. And it will stay in the mempool for weeks. So there have been people who said, I made a transaction, I put a super-low fee, it confirmed a month later.

The thing is, if you-- I should go, yeah, this is something of a tangent. But if you look over six months, it's really highly variable. It used to be sort of nothing, and then-- part of it is the-- it looks most dramatic here, because you're sorting by-- this shows the fee rate. It's like nothing. You don't have to deal with it. And then like, whoa, OK.

And people were complaining, hey, it's days where nothing confirmed. Here it was weeks, months, like all of December and January. And it was correlated with everyone in the media, everyone wanting to buy bitcoins, the price skyrocketed. So it's very sort of peaky. And then I think interest has died down a bit, and the fees have gone down with it.

So this is a hard problem to deal with. And it wasn't something people were dealing with until a few months ago. Most of the time, you could get by with just setting a low fixed fee or something, and it would work. But now we're really seeing the fee market develop.

Also, people disagreed that this should even be a thing. So there's Bitcoin Cash, there's all these forks, the idea of, if you start hitting the block-size limit, increase the limit. Why should we have to pay fees? The miners are already getting plenty of money to mine with, we should just have as much transactions that we want. So there's people that say that. Yes.

AUDIENCE: Sorry, if you're not getting them confirmed, you could just issue another transaction with higher fees.

TADGE DRYJA: That's what I'm going to get to next. So not always. So one, your software might not do that. Most software in the last few years just said, look, once the transaction is sent, it's stuck. It's just like, yeah, I sent it, waiting for it to confirm. And you have to actually code something to deal with changing the transaction ID, invalidating the old one. That's some software.

So one way a wallet could deal with this is to say, OK, well I'll do Child Pays for Parent. This is actually a little bit easier, software-wise, because you don't have to invalidate anything. You could just say, I'm going to send a transaction, spending one of those new outputs that's not confirmed, with a very high fee. Now, the average of the two will be high enough, maybe it will get in.

You can also do this when you're on the receiving end. So if someone sent you a coin, but they sent that transaction with a low fee. And maybe you can't contact them anymore, maybe they're unresponsive or they don't care. And you're like, well, I really need this money, I want it confirmed, I can then respend that output with a high fee to confirm both of them. It's kind of annoying, but possible.

Downsides of this-- Child Pays for Parent is very inefficient. You're making an extra transaction. And it's exacerbating the problem you're trying to solve. The whole problem is, there's too many transactions, there's not enough space. And you're increasing your priority in the queue by wasting even more space. So if everyone does Child Pays for Parent, it's really bad. Now every transaction is going to take two or three transactions.

And the dependency graphs are kind of complex. You can have-- I don't remember exactly how. There are certain cases where Child Pays for Parent lowers the priority of a transaction. I didn't understand how. But someone explained it to me, and I'm like, oh, shoot, that's bad. Where someone else spending an output can-- if there's a whole branch of tons of different outputs and tons of different Child Pay for Parent, there are certain parts where, OK, if someone makes a low fee transaction, that can actually lower the priority of the parents. I don't remember. Anyway, so it's kind of a mess, it's complex, and it's also just very inefficient.

OK, so another way which seems obvious, why don't I just double-spend it? I made a transaction with a low fee, why doesn't my wallet just say, look, I'm going to replace that transaction with one with a higher fee? And I just decrease the change output amount. That's simple, right?

[CHUCKLING]

Another problem-- the default relay behavior is to ignore double-spends. And this is not just a default that's for fun. This is actually-- there's good reasons for it. So if you see a transaction spending output A, output B, and then you see another transaction over the wire, spending one of those outputs that you've already seen, you will ignore it. You won't ban-- won't get mad at the person who sent it to you, but you'll just ignore it. Because you're like, no, I already saw a conflicting transaction. So the default is, first thing you see is the thing you go with.

This is not a consensus rule, right, because we're not dealing with anything that's in the blocks yet. But any ideas of why this is an important rule to have? What are some attacks that you could perform? If you just said, yeah, anything-- I'm not going to stick with the first-seen transaction. I will update anytime I see a new one. I'll just go with the-- I'll go with the last-seen transaction instead of first-seen transaction.

What would be an attack that would be enabled by that? Any ideas? You got to think adversarially. I'm trying to bring down Bitcoin. If you change your code to last seen instead of first seen, how do I bring down the network?

AUDIENCE: [INAUDIBLE]

TADGE DRYJA: Right, yeah, I just keep spending the same outputs, 100 times a second. And they all have the same fee rate. I can just keep doing that indefinitely, and I just keep flooding the network. And everyone's like, oh, here's this new one, oh, here's this-- total mess.

However, there is a way to prevent that kind of flooding attack, or at least significantly mitigate it, which is, OK, we'll relay conflicting transactions, but we require an increase in the fee. And the required increase is equal to our minimum relay fee of one Satoshi per byte. So every time I could say, OK, I'm going to replace it because it has a higher fee. So now it's sort of constantly incrementing.

You can't keep doing that indefinitely. Every time you want to send a transaction out to the network and use everyone's network resources, you are sort of potentially paying something for it. Yeah, you will continue to ignore transactions with same or lower fee. Oh, I did the why. Yeah, the why happened there. Anyway, yeah, the DDoS attack is make lots of transactions with the same fee, clog the network.

OK, so this seems simple, right? We have replace by fee. If you have a higher transaction fee, you replace the transactions, wallets implement it, no problem. Well, people don't like it. People said, no, we don't like replace by fee. This was a controversy around 2015, when a bunch of the developers were like, yeah, this seems reasonable, let's put it in. People said, no, this hurts the security of unconfirmed transactions.

And to some extent, they do have a point, right? If every node on the network says I'm going with my first-seen transaction, if there's a conflicting transaction later, I ignore it. That means it's hard to double-spend, even before it gets into a block.

You spend-- you say, OK, I'm going to send money to Alice, and here's my change output.

And then you replace it with, I'm going to send money to Bob instead, and my change output. You can potentially make it appear that someone's going to get paid-- it still says unconfirmed, but they say, yeah, you sent me a transaction, it'll get confirmed soon, and then switch it out and try to make it so that you pay it back to yourself instead.

When everyone on the network ignores those new transactions, it gets harder. You'd have to contact a miner directly and say, hey, I have this transaction on the network, here's a conflicting one, please include it in your next block, to double-spend, to defraud this person.

The thing is, in practice, it's not that hard to do this. And so a lot of the developers are like, look, unconfirmed transactions really have no security with them. And yeah, it might seem like we're reducing the security here, but you really should not depend on unconfirmed at all. It's not in the blockchain. There's no proof of work. The whole point of the system is to put the proof of work to really get consensus. And you cannot rely on network-level consensus, because there isn't any.

And yeah, so to some extent, replace by fee does make the double-spends easier to do. That's the point. But the thing is, it's not secure anyway. So anyway, it's also a UI issue-- should you even show unconfirmed transactions? Almost all of the wallets do, even the SPV ones. And the SPV ones have no idea if it's even valid or has a valid signature since they can't even check signatures.

Unconfirmed transactions are not meaningless. If you see it on the network, and you're running a full node, you see, yes, this is a transaction that could be confirmed. They've provided a valid signature. Any miner can put this in. There's no conflicts. This can work.

If you're SPV and you see an unconfirmed transaction, you have no idea, because you don't even know if it's spending outputs that exist because you don't have a UTXO set. You can't verify the signatures. You don't even know what the keys are supposed to be. But most SPV wallets still show unconfirmed transactions because users want it. They want to know.

And it's like, sometimes they'll put it in, like, red text, like, hey, it's unconfirmed, or put a little question mark next to it. I would rather just not show it at all. I think it gives a misleading sense to users. But this is an issue, right? Users are like, I want to know if the sender has actually signed and sent. And then yeah, afterwards, it gets in the block. Yeah, now it's confirmed, now it's safe. But I want to know. So SPV can sometimes connect multiple nodes and ask.

But this is a controversial thing that people have argued about. My personal opinion is that security is more important than UI and usability in this case, and you really should not rely on anything that has unconfirmed transactions. And also, the RBF-- the Replace By Fee policy is not a network consensus-level rule. You don't know what policy people have. If people are running-- if nodes are running "I just stick with the first-seen transaction," or "I replace with incrementing fees," or "I replace no matter what," or who knows, "I'll go with three or four updates and then I'll ignore." You just don't know what people are doing.

So the compromise that's currently in the Bitcoin software is called opt-in replace by fee. You flag a transaction by changing the sequence number in the input to not be FFFFF. And then you indicate, hey, I'm flagging my transaction as this can be replaced with an increasing fee. It's kind of ugly in my opinion. It's

Like-- it's not really opt-in. You can't even opt out. It's weird to say to people, oh, yeah, you opt into RBF. Because there's no way to prevent it. You can always contact the miners, or anyone can just run it. And there's code out there that's a couple of lines different that's like, here's the full RBF version of Bitcoin, where it'll just ignore this flag and treat everything the same.

But we do now have the option-- so if you run a wallet, and I believe that the Bitcoin Core Wallet, with the new version last week or last month, now, by default, flags every transaction as replace by fee, so that you can then bump the fee later on. And this is a much better way to deal with fees. Because then you're not stuck.

And then, to some extent, you don't even need to look at the mempool and what fees everyone else is paying. You can just say, well, I can also time-lock transactions, and I can say, OK, well, at the current block, I'll pay one Satoshi per byte. I'll also send out a transaction where it's not valid until the next block, and I pay two Satoshis per byte. And then the block after that, I'll pay four, and block after that, I'll pay eight, or whatever sort of fee schedule you want.

And then the miners now have the option to look and say, well, two Satoshis per byte is too little, I'm not going to take it. But then a block comes out-- OK, now it's four. OK, I'll take that one. And so that's a really nice sort of fire-and-forget from the user's wallet, where I don't even know what fee I should pay, well, look, I'll just keep incrementing it. And hopefully it confirms soon at a low rate. As it takes longer and longer, my rate goes up, and eventually I'll get it in. Yeah.

AUDIENCE: Are there any limits to how many times you do that?

TADGE DRYJA: Not really. OK, so one limit is you cannot actually broadcast the transaction that is valid in a future block. The network will ignore it. It's a similar reason, denial of service thing, where, hey, here's a transaction that's valid next week. Well, why'd you tell it to me today? So you'd have to be online to do that. But your wallet can do it automatically. It can just do it in the background.

And in practice, I don't know any wallets that do that right now. The Core Wallet will allow you to in the UI. You sort of right-click and you say, bump-- increase fee, and then it will rebroadcast with an increased fee.

OK, so yeah, it's a kind of a mess, but it's getting there. We're getting better at this. Any questions before we have a break? Yes.

AUDIENCE: So the one [INAUDIBLE] they'd pay [INAUDIBLE] their own confirmation of transaction.

TADGE DRYJA: Well, eventually they will, right?

AUDIENCE: [INAUDIBLE]

TADGE DRYJA: Will they eventually?

AUDIENCE: Once it gets the confirmation, but normally they accept transactions without confirmation.

TADGE DRYJA: That's silly anyway.

AUDIENCE: [INAUDIBLE]

TADGE DRYJA: OK, well yeah, so some sites will accept zero-conf without RBF, but not with. My thinking is they shouldn't be accepting zero-confirmation transactions, regardless. So whatever.

But yeah, and people-- there was a thing, Peter Todd stole $10 from Coinbase a couple years ago because he was trying to prove a point. It's so easy to double-spend before it's in a block. And all these different sites are saying, oh, yeah, here's a transaction that's not confirmed yet, but it will be soon. And then they dispense with the product.

And then I remember Peter Todd was like, look, I just got $10 from you on Steam or whatever, do you want it back? And they didn't reply. Because it's really not that hard to double-spend before it's in a block. The whole point of the blockchain, all the mining, is to prevent the double-spend. OK, so the last part is going to be about--

AUDIENCE: [INAUDIBLE]

TADGE DRYJA: Oh yeah, I already said this. One thing, though, is further research is needed. This has all been changing in the last year. If you had this talk a year ago, it would sort of be like, yeah, I guess you could do replace by fee, I guess fees could work this way. But we had never really had this sustained, long fee ramp-up. This hadn't happened. This had never happened. There had been cases where sometimes blocks had been full, but we'd never seen this ramp-up.

It's also really interesting how inelastic it was, in that people started paying thousands of Satoshis per byte, $20, $30 a transaction. And people got mad about it, but what was interesting is people were paying. If everyone refuses to pay these rates, the rates go down. But someone's paying that much. But a lot of it is exchanges and stuff.

OK, yeah, problems-- exchanges overpay. Bitcoin D Wallets overpays. Nobody cared for seven years. It's sort of like it's also, efficiency-wise, in terms of your transaction size, it kind of reminds me of 2008 when everyone was driving Hummers, and then gas goes to like $4.00, and then everyone gets a Prius.

That's happening now, where it's like, OK, let's batch our transactions. Let's use compressed pub keys. There's all sorts of techniques to reduce the size of your transactions so more can fit in the block which are now still being implemented. It's sort of after the fact. It takes this, hey, you're spending thousands of dollars a day on this to really kick people into action to write it. I remember looking into December, though, at Coinbase's transactions and being like, you should hire-- you guys are probably losing $1 million a day. I could fix this for you. You should-- no? OK.

[CHUCKLING]

But it's sort of frustrating to see. And probably, their system is a huge mess from so many different layers and things on top of each other that it's hard to fix. But looking at-- Gemini also is an exchange. They use uncompressed pub keys. So the pub keys are 65 bytes instead of 33. There's no advantage there. It's just an extra 32 bytes. Which they've probably spent, I don't know, a couple hundred K on just that weird aspect of their software.

I don't know if they're aware of this, even. You might want to be like, hey, guys, you're spending too much money. Because look, your transactions are an extra 32 bytes per input because of this. But yeah, it's interesting to see.

I've also paid. I used to not pay.

Oh, one thing that used to be the case, transactions had priority, which was based on the age of the outputs multiplied by the amount of the outputs. That was a thing until two years ago. And it was great because it meant if you were rich, you didn't pay fees.

[CHUCKLING]

So if you had a bunch of bitcoin, it was kind of a maybe not-so-great social idea that, hey, if you have an output that's got 100 coins in it, you can make a zero-fee transaction. It actually worked from a theoretical perspective because it prevented spam. Because the idea is, if you have 100 coins, OK, you can spend that every day or something, with no fee. If you have only one coin, maybe have to wait 100 days to be able to spend it with no fee. So there's this sort of priority, age-based, to limit the velocity of these payments based on how big they are.

They got rid of that because it made sense from a network level, "let's protect the network from spam" idea. It did not make any sense for the miners to prioritize these transactions. I'm a miner, who cares if you have a bunch of old outputs that you haven't spent in a long time. It does nothing for me. It does help the network in general. But so I was a little disappointed, because I don't have a ton of coins, but they're all really old, so I could get away without paying fees.

Long term though, this is sort of an interesting thing to look at. I'm not sure that's-- oh, shoot, is that number right? Wait, James, do you know? Is it 210,000? I think it is. That number might be wrong. I made a mental note to confirm it. I think it's right.

Anyway, the mining reward drops in half, every four years, approximately. And eventually all the coins are mined. So it's dropped in half twice. First it was 50 coins a block, then 25, and now it's 12 and 1/2. In two years it's going to be 6.25 or something.

So eventually all the coins are gone, you mined them all. That takes 100 years. So everyone's like, I'll be dead-- I'll be gone, you'll be gone. But actually the rewards become negligible a lot sooner than you think. In 20 years, there's way less than a coin in rewards, because it keeps getting chopped in half every four years. so yeah, there's this sort of long tail where you've got like 50 years of a couple Satoshis per block. But it's negligible, right?

So the rewards going to near zero actually happens definitely within our lifetimes, pretty soon. And a lot of people looking at Bitcoin are sort of thinking of it long term. Bitcoin's only worth a lot of money now because people think it's going to be worth a lot of money in 20 years, right? If everyone knew it was going to collapse in 10 years, I think people would lose interest.

So what are some weird long-term problems? OK, well if there's no new coins to mine, why do you mine. Well, the title of this piece.

However, it's not that simple. There's a lot of problems with the fees being the only incentive, in that TX fees are very variable. Without a backlog, the fees tend to go to pretty close to zero. And if the fees are zero, there's no reason to mine. All the new coins have been generated, there's no fees in the mempool, why am I running my-- maybe you have a megabyte worth of zero-fee transactions in the mempool.

But as a miner, why am I spending electricity to mine this? I get nothing. So the miners just turn off. That would be what they should do. And you can see, yeah, it's super variable. Maybe miners are like, oh, this is great, we're making a ton of money. And then here they're like, I guess wrap it up, close down the mining facility. And that could be a problem, even on much shorter time scales, such as the time scale of a single block.

So for example, here's what you do. You're a miner. There's no fees in the mempool. There's no reward. There's no reason to mine, right? Turn off your chips, and then turn them back on once the mempool starts filling up, because it will eventually.

The idea is, OK, for the next couple of minutes, I'm just idling. You could do that, or you can be clever and say, well, I've got all these chips, the opportunity cost, a lot of it is CAPEX. A lot of is the capital expense of the chips and not the electricity. So if I'm just turning off my chips, that's a huge loss for me. I want to run. I don't want to run with an empty mempool, because I get nothing. What should I do? What's a fun attack? Yeah.

AUDIENCE: Could you make a bunch of transactions where the fees are high, and the new people will leave when the fees are high?

TADGE DRYJA: Very good idea. I'm pretty sure miners are doing that, or were in November. That doesn't help you-- that helps you sort of medium-term, right? In the very short term, where there's an empty mempool, what do I do? There is another attack you can do. Yeah.

AUDIENCE: You can just mine in empty blocks.

TADGE DRYJA: Yeah, but I don't get any money. You could, but that's not a fun attack. Yeah.

AUDIENCE: You could build up a bunch of non-transactions.

TADGE DRYJA: Yeah, but I want to get money. I want to get paid, there's nothing in the mempool, where do I go? Yeah.

AUDIENCE: Output to another currency.

TADGE DRYJA: OK, so yes. So in Bitcoin's case, the algorithm-- well now there's Bitcoin Cash, and so there's sort of two chains that share a single algorithm. And in many alt coins, they share either algorithms or they're using a GPU, which can be quickly switched between different mining algorithms. In Bitcoin's case, though, you're sort of stuck. It's a big one, and you're using Sha256. So OK, good.

AUDIENCE: You could mine off the previous--

TADGE DRYJA: Yeah, you go back to the past. You mine the past transactions. OK, someone mined a bunch of transactions here, and they got two bitcoins. Those are two bitcoins I could've mined.

[CHUCKLING]

So that last block at a couple of coins in fees. I'm going to remine it. I'm not getting anything if I mine-- if I continue the chain and make progress, I get no money. But if I try to snipe the last guy's mining stuff, I might be able to take his money. If you find two blocks in a row, which happens, you get all the fees. So that's sort of an optimal thing to do. I've got all this mining power, maybe I turn it off because I'm mostly OPEX instead of CAPEX. But if I've got all this mining power, I'm going to use it no matter what, the optimal thing to do is try to take a reorg the blockchain and take the last person's fees.

So for example, instead of sequentially building, you fight over it. So here's this blockchain. OK, Alice got two, Bob got three, Carol got two, Dave got one. These are fees within the block. And now the current mempool, it's zero. Dave got it. Dave swept it. It was sort of declining, sort of going up. Dave completely emptied the mempool. There's nothing in there. And there's no new coins coming out.

So I'm what, Eve, I'm the attacker. Eve is always the bad person, right? So what's also even better, I say, OK, I'm going to build-- I'm going to try to reorg out Carol. I can take all the fees here. I can take the two bitcoins here and the one bitcoin here, and I can mine it into this block. This block is not valid yet, right? This is higher. But I might build another block here. And then I might build another block here. And by now there might be some coins. So now I get four coins. I could have mined here and gotten nothing. If this happens and I pull it off, "puh-toom," OK, Carol and Dave's coins are now mine. So this is a--

And to some extent, you might think, well, whatever, it's a little messy, but this is not the worst thing in the world. You still make progress. It's actually really bad. Because you're now trading this sort of memoryless process where everyone's competing on a level playing ground where whoever can mine fastest is always going to win here. So now if you're a miner that's twice as big as the next guy, you're always going to be able to do this-- not always, there's still some-- but this really reduces the variance and really incentivizes the miners to consolidate. And you're going to end up with a monopoly in this kind of a scenario, because small players will just always get a reorged out.

Yeah, so the question-- is this an attack? It seems like they're just trying to get paid, right? The whole idea is you're not trusting the miners. You're just assuming that they're acting in their economic interests. To some extent, you could say, well, if they keep doing this, they reduce the utility of Bitcoin, and so it's bad for their holdings. Yeah, maybe, but it looks dangerous, right?

However, this isn't a problem. We don't see this problem today. There have actually been several cases where we should have, where it would have been optimal for the miners. So someone made a fee that was like 1,000-- no, it was 400 bitcoins or something. They just screwed up. They made a transaction, and they were spending like 1,000 bitcoins and they only sent like 600 out or something. So the fee was 400 bitcoins, which is like, hey, that's way more than a block. That's millions of dollars. It wasn't when it was millions of dollars, but it's still a ton of money.

And the miners, if they were rational, they should have all just stopped and all tried to fight each other over that one block and that one transaction, because it's so much more valuable than progressing the chain. However, they didn't, because they were just running their software. They weren't anticipating this kind of thing.

However, if this becomes, long term, the general state of the network, they'll definitely anticipate it, and they'll write the software to do this. So it's not a problem when there's low reward variance from block to block. If every block-- if the next block coming out has about the same fees as the last one, then you just keep building on it and you're not trying to snipe other people in the past. But that means there's going to be a backlog. You need this really big mempool with a large backlog of transactions to ensure that you have low variance in the rewards and to ensure that, long term, you've got a stable system.

So this is tricky. This is a hard problem. Because you've already got these sort of scalability balances in Bitcoin. One of them is related to your hardware. So if you say, OK, we're going to make the block size really small, we're going to make the block size 10 kilobytes, so you can only have a few transactions every 10 minutes, if it's too small, then very few people can actually use Bitcoin. There's just not that many transactions a second.

Most people will have something like Coinbase, where someone else holds the coins for them. That's no fun. The point is to have your own UTXO, so really be in control of your own money and have your own private keys.

However, if the blocks are too large, then not many people can run it. If the blocks are a gigabyte each and you've got these petabyte-sized blockchains, I can't run that. I need a data center. And then you can't really verify and validate that the system is operating the way you think it will. You're just going to have to trust, again, some kind of big institution to do it for you. So they both sort of turn into the same thing when you're at the extremes. If it's too small, well, I can't really own my own coins, I got to have someone else do it for me. If it's too large, I guess I can own my own private keys, but I have no idea if they're actually-- if there's an actual transaction in the blockchain because I can't download it.

So these are sort of balances based on hardware. And we're thinking, well, as computers get more powerful and networks get more powerful, we can sort of get into the larger side and people will still be able to run it. And we have seen that. So like with SegWit, the block size has been increased through software. So it now takes a bit more space and more network traffic to use.

However, the long-term fee sniping thing, it's not hardware related. Even if you had perfect computers that were sort of like, oh, this network can send any amount of data instantly, we've got unlimited drive space, these are computronium-- that's a thing, right, the theoretical maximum computer. So even if you have this amazing computer and the hardware limits are not a limit at all, you still have to balance the block size. Because if you make your block size too large, you're going to have that problem long term, where there's no real fee market because there's no competition to get into the blocks. And so the whole system kinds of halts, and then you've got the largest miner just keep winning.

So this is a mess. There's a paper. What's it called? Like, Bitcoin Is Not Stable Without a Fee or something. It's Princeton guys, right? "On the Instability of Bitcoin Without the Block Reward." And to be fair-- yeah, these are Princeton guys. To be fair-- and they write about this issue. But it's a little weird, because one of their assumptions is that-- mining strategy simulator.

Anyway, one of their assumptions is that every miner completely wipes out the mempool at every block. And it's like, wait, that assumption-- given that assumption, yes, you can show that it's unstable and there's all these problems, like I was sort of showing. But that assumption is a huge assumption. And a lot of the people working on Bitcoin are like, no, you can't make that assumption. Because we're aware that there's these instability problems, and the solution, to the extent that it's a solution, is always have a backlog. There's a really big demand for transactions, and there's not enough space, and so people have to compete based on their fees.

In that case, you can have a stable system, long term, where the miners get paid. But without it, these guys show it. And so yeah, it's true, but it's sort of weird because one of their assumptions is that there are no fees, long term, essentially.

So yeah, this is a weird issue, and it's also how, timing-wise, do we need to worry about this now? Maybe we can just punt the problem for a decade and try to get lots of adoption, lots of people using Bitcoin, and we'll figure it out once it's really well established. I don't think that works. I think it's harder and harder to change these systems as more people use it. So maybe we're stuck.

And it's almost certain that one megabyte or four megabytes is not the optimal size. We have no idea what the optimal size is, right? Long term-- because also, long term, how many people are going to using this, and for what, and for what kind of transactions? So it's definitely an open problem.

Yeah, so we're sort of at the dawn of the fees. A year ago, we didn't have all these things.

We're starting to understand the fee markets and see how it works. What's interesting is that the elasticity seems like people really want their transactions in when they want them in, which I guess can make sense. If the price of Bitcoin spikes to $18,000, and you're like, I want to sell one, it's going to crash, and you're like, yeah, I'll pay $50 to move it, to sell to someone, because I'm selling millions of dollars worth. And so it seems really inelastic. You also have people, companies, wasting millions of dollars on fees that you can just be like, why'd you do-- oh, OK. You're wasting on space, wasting on fees.

It is a fun research area. And it's definitely research. And so I'm like, yeah, I hope this works. We don't really know, long term, if this whole system is stable and works. I think it can. But it's not something to base the global economy on just yet, because we're not quite sure it all really works long term, but we think it does.

OK, any questions about this sort of long-range attack stuff? Yes.

AUDIENCE: What's the case with a bunch alt coins points that don't have full members?

TADGE DRYJA: Alt coins?

AUDIENCE: The rest of the cryptocurrency space that has sparse mempools.

TADGE DRYJA: They generally don't have fees, or the fees are enforced-- you can enforce a fee. You can say, hey, the consensus rule of this system is that every transaction has to have a 50-Satoshis-per-byte fee. There's problems with that, because then you can have out-of-band fees where, if you try to make it a consensus rule, it doesn't really work. It's sort of like price controls, where you can say, hey, you have to sell milk for $2 a gallon. And it's like, well, that's not what the market says. So people can go out of band. And then they're saying, OK, the consensus rule is I have to pay this fee, but really, how about you give me some of it back?

Or-- so generally the rules they put in are minimum fees, which you can try to work around that way. But in general, most of the other alt coins-- with the exception of Ethereum, Ethereum definitely has similar issues-- but most of the alt coins have a block size or a network capacity that exceeds their uses. So there isn't really competition to get into the next block.

Ethereum, though, similar to Bitcoin, has-- well, I don't-- where did it go? This kind of thing. I don't know, if you graphed Ethereum, it probably wouldn't look quite the same. But there are also times in Ethereum where the fees skyrocket. A lot of times with ICOs, everyone wants to get in their bids really quick, and so the network will get super congested for a day, and no one can use it for anything-- for normal uses.

So Ethereum also has this problem. It's not quite the same in that it's not a block size, it's a gas per block kind of thing, so how much computation or storage it's using. But in practice it's a similar thing. So yeah, those are sort of the only two that I'm aware of that really have hit this issue so far. Any other questions about this, the long term? Yes.

AUDIENCE: Could you comment on blockchains that have no transaction fee, particularly blockchains that are designed for things like IoT, and which pay to be able to handle microtransactions and transactions at face value without any fee.

TADGE DRYJA: Should I take advice from counsel before commenting on this?

[LAUGHTER]

Yeah, so you could Google-- there's one, I don't know if you're talking about it specifically-- there's one called Iota that we looked at last summer. My mom tells me, if you don't have anything nice to say, don't say anything. But we did say quite a bit in our report.

That one specifically, I don't think it really works that well. Actually, they have a fee, they just pretend they don't. But they very much have a fee. So in Iota's case, they say, you have to do work on your transaction. You have to mine your transaction. And they say that's not a fee. Well, OK, it's on a fee you're paying to some other miner, but you're still doing work. That has a cost. You can say it's a cost, not a fee. I don't know.

But it feels like saying, hey, this refrigerator doesn't need electricity to run, it's just got a hand crank in the back that you have to-- yeah, OK, I don't have to plug it in, but I still have to do all the work. So it's an anti-spam measure which you have to do work on. So it's the same new thing.

But there are some that have no fee whatsoever, and they're very susceptible to spamming. If there's no fee, what stops me from just sending out a bazillion transactions and having everyone else store them?

AUDIENCE: Generally well MANO is one of them that claims to be instant no fees, et cetera. But what that means is, to get around that, they've made it so that you don't need global consensus-- until you do need global consensus, at which point you do proof of state and [INAUDIBLE]. But since there are no fees, it's free to just calls for state to happen every round, [INAUDIBLE] a bunch of conflicting transactions.

TADGE DRYJA: Huh. Yeah, there's a lot of really smart people, way smarter than me, working on these things. And it's a hard problem, and they seem to think it's a hard problem. So I'm very skeptical when someone, especially with a financial interest in getting me to believe that they've solved this problem, says, hey, I've solved the problem, zero fees, infinite scalability. I'm just very skeptical when I see that. Because, OK, what did you do? Like, we've been looking at this. Maybe there's something there. But I haven't seen any really great stuff.

There's Lightning Network, which I'll talk about in a few weeks, and I think that's pretty cool, and I sort of work on that. There's other different ways to do this. But in general, there's got to be a cost. There's no free lunch. And you're going to have to-- if you're requiring other people to store a lot of data and process it, there's got to be some kind of limit there.