MITOCW | 22. Alternative Consensus Mechanisms

I'm going to talk about alternate consensus mechanisms. And there's a bunch of them. And some of them are variants on proof of work. Some of them are proof of stake-- all these different things.

So unique node lists-- this is something that Ripple and Stellar use. I'm not sure if there's others. Proof of stake is this huge research topic that lots of things fall under. There's lots of variance. One of them I'll talk about is delegated proof of stake.

Proof of space is an interesting thing that is basically a form of proof of work that doesn't use CPU as much. There's the idea of directed acyclic graphs, which IOTA is a great example of. And then one that I came up with a few years ago and is kind of fun-- proof of idol, which is sort of silly but some of these others are silly, too.

OK. And so disclaimer-- I'm OK with proof of work. So I think if you were at the class on Monday, it's a great segue into this, where, clearly, proof of work has, I don't know if you want to call it problems, but it's kind of crazy, right, all the stuff David was saying on Monday. It's not an egalitarian-- and I think in the Satoshi White Paper, he says one CPU, one vote.

Well, that's not really how it works. Define CPU. Maybe you have your own CPU. Someone else has a different view. And there's that huge industry of building all these chips.

So there's a lot of issues with proof of work. It's certainly not ideal.

That's probably not why most people don't like it. So I, generally, in academia, in a lot of business places, people don't like the Bitcoin proof of work mechanism-- not everyone, but it's a pretty widespread. Oh, Bitcoin is bad because it uses as much electricity as Denmark or something.

And I think that's a sort of early, I don't like it because it uses electricity. There's other reasons not to like it, which I think you have to go into much more depth. Like with David's talk yesterday, sure, it uses a lot of electricity. But it also uses a lot of fabrication plants, and it uses all these other distribution networks. So anyway, I get that there's a lot of-- one of the first things with Bitcoin is, OK, how can we improve upon this? How can we get rid of this huge energy sink?

On the other hand, I'm sort of OK with it, right? The first thing that I thought with Bitcoin is, wow, this is going to be huge. This is going to take a lot of electricity.

But it's kind of cool that it works that way.

So I'm sort of OK with people-- to me, you can't really get mad at people doing things you think are stupid because you'll just get mad all the time. People mine gold. People spend billions of dollars on diamonds. I think diamonds are really stupid. But hey, if you want to dig up diamonds, oh, whatever. Yeah. There's entire cultural areas that I just have no interest in. But hey.

OK. So I'll try to explain these other methods in a neutral way. But I do have my own biases, where I think like, hey, proof of work pretty much works. I think it's a cool system. And I'm somewhat skeptical of various different algorithms and mechanisms here.

And also, I'm most familiar with the proof of work system and all the intricacies of it. So like with David's talk yesterday, there was some new stuff, but I sort of knew because I've talked to these people for a long time, and this is how this system works.

Proof of stake-- I've seen it implemented in altcoins. But I haven't followed them that closely. So I don't know as intimately what goes wrong and all the weird details. Anyway.

OK. So the first one, UNL, or unique node list. This is, essentially, the consensus mechanism used by Ripple and Stellar. For history, Ripple started in 2013? No, 2011.

I don't know. It was a while ago.

AUDIENCE: [? Dennis ?] says [INAUDIBLE] January 1, 2013.

TADGE DRYJA: 2013, OK. So actually, the name Ripple and the idea of these debt obligations through this network actually predates Bitcoin. And then Jed McCaleb started the company.

He's also the guy who started Mt. Gox. So he started Mt. Gox, I believe, in 2011 or '10-- late 2010, sold it to the Mark Karpelés, started Ripple, hired a bunch of people, sort of got fired from Ripple.

There's a lot of bad-- everyone fighting. Because it was all these people who used to play Magic-- the Gathering and were friends. And now they're not friends anymore. And then, he started Stellar, which he called "Secret Bitcoin Project." And I was thinking of joining it, but he wouldn't tell me what it was. And also, I knew that other people knew what it was, like my friends, and if they weren't telling me because they're not supposed to, it was like, well, it must not be that cool. Because nobody can actually keep a secret if it's really cool.

[LAUGHING] So that was my, mm, I'll just go work somewhere else. And Stellar was, more or less, a copy of the

Ripple code, which was open source, so it was totally fine to do this. And then it diverged, and they added their own different parts of the consensus algorithm than Ripple did.

So it's account-based, somewhat like Ethereum. Transactions have senders, receivers. What's interesting is in the Stellar case, and possibly Ripple, there's a minimum balance. So I actually had some Stellar last year, or up until last year, because I went to the initial release party, and they gave out Stellar. And I had some.

And then, it wasn't much. But then, last year, I looked, and it was like, oh, this is like $200 worth. I'll sell it. And I sold it and got some bitcoins. But it was weird because they enforced a minimum balance, which seems weird to me. Because you had to retain some number of stellar units in every account, which, to me, seems sort of weird.

Because that means those stellars, they're gone, in a way, in that if you open a new account and it must have 50 stellar in it forever. Well, so Stellar effectively destroyed, it just seemed-- which is a fine thing to do. It's sort of a transaction fee. But from a computer science database standpoint, it's like, why not just have it cost money to create a new account instead of have a minimum balance? So just interacting with it, it was like, huh.

So there's no work. But the idea is that nodes sign transactions that they've seen. Everyone makes blocks. And you sign them, so you've got some kind of identity. Your node has a key. When you start a new node, you've got a public key/private key pair. You can sign off on transactions.

This is not the case in Bitcoin, right? In Bitcoin, you have a full node. Your full node might have a wallet associated with it with public keys and private keys, but the node itself does not. Possibly in the future, there's BIP 150, I believe, which the idea is to have encrypted communications between nodes, which would have some kind of authentication.

There's BIP and BIP 151. I might have them backwards. 151, I believe, is just encryption, so for privacy, confidentiality. And then, BIT 150 is for authenticity. So you connect to a node. You're like, oh, this is the same node at the same IP address that I connected to before. They're not implemented yet that I'm aware of.

So right now, in Bitcoin, when you connect to people, you just use their IP address.

You don't really know who they are. There's no authentication there. But in these, in Ripple and Stellar, you do know. OK. I'm connecting to this node. It's got a key. It's got an identity.

So to sync, instead of verifying all the work, you verify the signatures on blocks. And the question is, whose signatures, right? So if you assume, OK, well, if we have a majority that's honest and is not trying to create transactions that are invalidated or something like that, then it might work. But majority is hard to define when you're subject to civil attacks, right?

The obvious attack is I just make thousands of nodes, or maybe these nodes don't even really exist. I just make thousands of key pairs, have them pretend to be nodes in my own subnetwork, and they all endorse blocks that are different than the rest of the network is endorsing.

So majority is the tricky part. How do you know who's the majority if you don't know who these people are? You need some kind of way to identify humans or computers, even.

And this is a problem akin to certificate authorities.

So a lot of the problems that you see in blockchain, bitcoin, these kind of things, are not exactly new problems and have sort of already been, quote unquote, "solved." So does everyone know how CAs work, or do people have some idea? Or who's like, what's a CA?

OK. So it's kind of interesting. I can give a little demo.

AUDIENCE: Did you say CA stands for certificate authority?

TADGE DRYJA: Yes. Uh-oh. This doesn't work. Hold on. OK. So if you have a computer, I don't know where it is in Windows, but I think in Mac and Linux, it's in the same place.

You go to etc and then-- shoot, I forget where it is. Yeah. Where's the certificates? Cert-- tru-- AUDIENCE: CA certificates right under the mouse.

TADGE DRYJA: There it is. OK. So in your computer, and whether you have Linux or Mac, you've got a folder somewhere, etc/ca-certificates. It's something like that in Mac, as well.

And, oh, no, that's not it. Where are they? Shoot. Wait, wait, wait-- what is in update.dn?

And there's nothing there.

Hold on. I need to find this. Because this is kind of cool. Where are X? Ah, SSL-- is that-- there we go. OK. So sorry. etc/ssl, and then you've got certs. And this folder has a bunch of certs.

I trust all of these entities. Well, my computer does and, by extension, my browsers and my computer, which is crazy because I have no idea who these people are. TURKTRUST? I don't really trust Turkey. Swisscom-- Swiss people seem nice, I guess. I don't know. OK. Staat-- they're in Netherlands somewhere.

There's a bunch of, just things you've never heard of. Hellenic Academic and Research Institute, Hong Kong Post. Well, Hong Kong Post Office seems trustworthy. Yeah?

AUDIENCE: On the TURKTRUST one, you pointed out they have [INAUDIBLE] certificates with google.com when they're not supposed to. [INAUDIBLE].

TADGE DRYJA: And then, CNNIC is one which is like-- is that still in here? Maybe they got rid of it. And then some of them just have these numbers, and you don't even know.

You have to go in.

AUDIENCE: [INAUDIBLE] TADGE DRYJA: Yeah. China Internet Network Information Center EB certificates root. I actually really don't trust them to endorse-- pseudo arm-- there we go. OK, it's gone-- solved it, solved the distributed internet trust problem.

AUDIENCE: Now part of the internet doesn't work for you.

TADGE DRYJA: Yeah, well, I might get invalid certificate errors for some Chinese sites now. I don't go to many Chinese sites. Anyway, so that's the basic idea of certificate authorities.

And this started in mid-'90s, when they wanted to make public key infrastructure for secure websites.

One of the problems is how do you know-- so you've got-- oh, we didn't talk about Diffie-Hellman, or maybe I mentioned it. But you've got public keys, private keys. You've got ways to do signatures. Diffie-Hellman is a way to do key exchange. So if I know you're a public key, and I give you my public key, we can form a third, basically, key that we can then use for encryption and secure messaging.

This works, but how do you know, given a key, who is on the other end? So that's the idea of certificate authorities. You have this sort of root of trust. And these certificate authorities sign certificates in websites and let you know, oh, this is the person who you think it is.

So for example, Google-- secure connection. Really? How do I know? Verified by Google Trust. Well, Google has verified that Google is trustworthy, which is somewhat circular.

But no, that's sort of how it works. Yeah, so Google Trust Services is a certificate authority. And they signed off their own certificate for their own website. That's maybe not the best example.

OK, The New York Times, they are certified by Comodo CA, who also has had-- Comodo has gotten in a lot of trouble for signing the wrong things. So yeah. New York Times does not have an EB cert. I think Washington Post does. So there's also EB certs where-- you've got now The Washington Post, WP Company LLC.

That's like an extended validation cert, which, then, the CA is saying, we didn't just check that they had an IP

address and a domain name. We actually checked that they had some kind of legal corporate entity. And they sent it on their letterhead, and we, I don't know, emailed someone in Delaware to make sure they actually had a company. And this is verified by Entrust. OK. So Entrust, Comodo, those are all going to be in here. Yeah, there's Comodo.

So you've got these companies called CAs that have some kind of agreement with each other.

There's these standards where you make some kind of standards where, OK, don't sign the wrong thing. If you do, we're going to delete you. And then they all endorse entities' identity, usually for companies.

There's also stuff now-- Let's Encrypt, which does it for free and without really endorsing anything. This whole thing, it sort of felt like it was like the big scam of the late '90s because they made billions of dollars. But then again, we got Ubuntu out of it, right?

So yeah, Ubuntu was funded by Mark Shuttleworth, who started this one, I believe, and made a couple hundred million, went on the spaceship, went into orbit, had some fun. Eccentric billionaires are what fund a lot of technology development. And this is how we get them.

A lot of the people who work on Bitcoin and these systems feel that the current-- it's called X.509 is the name of this whole system with these certificate authorities-- and I also sort of feel that this is a suboptimal solution in that there's a lot of problems with certificates being signed inappropriately. It works, in a way. But it's not great.

So this is a problem. So the problem that these systems like Ripple and Stellar have to deal with are, in some ways, similar to the problem that is solved by CAs. So what they do is they say, we have a unique node list. We're not actually endorsing an identity from, here's a public key, here's a human name or a company name or something like that.

It's we're just saying that they're unique.

So it's more limited than a certificate authority. It's just saying we're certifying that these two keys belong to different people or different companies, which seems easier than the job of a CA, right? The CA has to verify that The Washington Post is actually-- whoever's presenting a public key is actually The Washington Post. So the actual certificate is just, basically, here's a pubkey, here's a name, and then the CA sign. And you can look at those and stuff.

OK. So for synchronization, you wait for a majority of nodes in your unique node list to sign and, if they've signed, accept. So there's some recent papers from Ripple. And in order to really get consensus, you need, basically, a 90% overlap in your own unique node list.

So if I have a unique node list of Alice and Bob and Carol, and you have a unique node list of Carol and Dave and Edna-- I don't know-- we might diverge. We might not agree on the same chain of transactions because we've got different people that were looking at their signatures. They may all be unique. But if we have different unique people we're looking to, it might not converge.

So they have a newer paper that reduces that to 60%-ish, where the overlap of what unique nodes everyone needs to look at is not as large but still quite large. And then, the real question is, OK, well, who provides the unique node list? Because that's not really a job I can do.

Maybe it is, in a way. If I know you, and we do some kind of web of trust thing where I'm like, oh, what's your Ripple nodes pubkey? OK. Cool, I know you. And what's your Ripple nodes pubkey? We meet in person. We sign each other's pubkeys. We do the whole PGP kind of thing, which I do. And the Bitcoin people actually do this.

So if I go to-- hold on. So I've got a bunch of keys, and everyone signed them. And there's Andrew Chow and Suhas and [INAUDIBLE] and all these Bitcoin people. And I've signed their keys and they've signed my keys because we meet in person and do that kind of thing.

So we've established not just a unique node list, but we've done a CA-free validation of each other's keys. We just meet in person, read each other's keys off.

So you can do that, in practice. It's really nerdy, and no one does that. And it would be cool if we could get it to be easier and maybe cooler somehow. But it's been an uphill battle.

So who provides the UNL? It's probably kind of obvious. Well, the Ripple company provides the UNL, right? So when you download Ripple, it has a list of unique nodes. Those are essentially de facto, the nodes that run Ripple. And at the current time, those nodes are run by the Ripple corporation, or Ripple apps.

Stellar, similarly, it comes with a default UNL. It sort of acts like the CAs, right?

It's sort of how when you start your computer, you've got this. This comes with your OS.

And you don't really have a choice. I just deleted the CNNIC one. But nobody looks at this stuff. Everyone just goes with the defaults. So that's the problem centralization-wise.

Yes?

AUDIENCE: You said Ripple or Stellar actually run the nodes? Because I understand that they might sign off on the UNL if you're saying that-- TADGE DRYJA: In Ripple's-- at least a year or two ago, and they may have

changed that, I think they're trying to-- but in Ripple's case, yeah. They ran their own Ripple nodes.

AUDIENCE: So it's not these bank partners or anything?

TADGE DRYJA: They may have some. They may now. But I know that two years ago, it was just Ripple, or majority Ripple. Because they also put in things like they can freeze funds.

Because Ripple got in trouble with FinCEN. And part of the thing is they modified their code to be able to freeze people's funds if they did something bad.

That's definitely not something that the Bitcoin developers, or even the Ethereum Foundation-- it's not as direct, anyway. The Ethereum Foundation did freeze funds and move them against the rules of the system in the case of the Dow. But that was, in a lot of ways, with support of a lot of the people running Ethereum. With Ripple, they can do it without much brouhaha.

They just freeze these funds.

So it's fast. There's no work involved. There's no worry about propagating blocks and miners getting advantages. But there's also known identities, and so it's more susceptible to subpoenas and things like that.

In these cases, all the coins existed at the genesis block. So in the case of Ripple, I think it was 100 billion or something. And they just started with all the coins and distributed them as they saw fit.

Same with Stellar. Stellar had this thing where they had this party, and they would give them away if you signed up on Facebook. A really interesting article about people in Manila getting thousands of used SIM cards from the US, registering for new Facebook accounts, selling those Facebook accounts to people in China, the people in China then registering with Stellar as unique people to get more stellars.

When you incentivize things, weird things happen, be it either giant warehouses full of SHA256 chips or giant warehouses full of 20-year-old girls in Manila putting SIM cards in cell phones all day. Yeah. I should link to that article. It's really interesting.

So anyway, this is the Stellar one. And Stellar also had this thing where they were going to give 20% of the stellar to people who held bitcoin, sort of an airdrop. I didn't get any from that. I was sort of hoping. I'm like, hey, cool-- air drop. I have bitcoin. I'll get some stellar. I didn't because you basically had to KYC with them.

They said, OK, send in your-- I don't know if it was the social security number-- but send in your documents and prove that you have these bitcoins, and we will credit some stellar to the same keys. And I'm not going to do that. Yeah. Some people did. Some people got some stellar that way. Yeah. So anyway. Yeah?

AUDIENCE: If Stellar's just freezing coins-- TADGE DRYJA: No. I don't know if Stellar has that capability. Ripple does. Ripple put that in. Stellar is a different-- it's similar software. They certainly argue about how different they are.

Stellar had a bug in 2015, I believe, where the consensus broke. And then they said, oh, it's Ripple's code's fault. This is a fault with all this code base. And then Ripple said, no, no, no. It's because you changed it. And you don't know what you're doing.

AUDIENCE: I was talking about, I guess, the minimums. You said there was a minimum.

TADGE DRYJA: Yeah, there's a minimum account balance.

AUDIENCE: And if there's a set number of coins, then eventually, accounts will just get-- TADGE DRYJA: Yeah. It's a lot, though. Also, there's enough forever. There's a lot of coins.

But yeah. But the idea is not that you have UTXOs, like in Bitcoin, and you make new addresses each time. The idea is you have an address. You keep that forever. So why would you have multiple different addresses, that kind of thing.

Anyway. So it's a pretty different system. In some ways, you could argue it's one of the first ICOs-- Ripple-- because it was pretty early. They just came up with all their coins and gave them out and then started selling them. And they continue to do that to this day.

Stellar is a nonprofit, which is also sort of weird. Because if you make a billion dollars off of something, is it really a nonprofit? I don't know. The organization didn't directly make the money and then pay it to the employees or anything. But the people who have all the stellar tokens made a lot of money. Interesting sort of ways to do it.

So I think both Ripple and Stellar argue that it's decentralized, distributed. But I think a lot of people say, well, maybe to some extent. But on the spectrum of completely decentralized versus one server handles it all, it's a lot further on the centralized side.

And Bitcoin is also kind of centralized, too, like we were talking about on Monday. But I would argue these are more so. So anyway. Any questions about Ripple, Stellar? Pretty fun.

OK. Next, the big one-- proof of stake. So this we've mentioned a little bit. It's a popular alternative where people really don't like the proof of work stuff. So instead of proving work, have the people who hold coins sign the blocks. So that bootstraps, in a way, your unique node list or your list of who's who.

Well, you don't really care who they are. But if they have coins, if you already have a set of who owns what coins, you can use that to say, OK, well, the people who own coins now signed. And they, presumably, have incentives not to destroy the network they have coins in.

So if you have a given genesis block with some kind of initial distribution, you can make it deterministic. You can say, I'm just going to go with whatever has the most stake, whoever has the most coins, signing off on the next block. And given two or three different histories, you can see, OK, from the start, which has the most total stakes signing off on these things? So that seems pretty cool.

Here's some issues. Stake grinding-- so for example, let's say you have a system where the signer of the next block is determined by the public key nearest to the previous block hash. So for example, OK, I'm looking at the current block hash. Who gets to sign next? Well, whoever's key is closest. Use X or just treat the hash as a number and treat the pubkey as a number and see which is closest.

You can do that. But the problem is if the next hash is determined by the current signer, that current signer can make a new signature each time and try to make sure that the block after is also going to be one where they are, again, the next signer. And they can make a couple hundred different accounts with their keys and then keep grinding stake.

And this basically means that it turns into proof of work. Because if you can influence who's going to be the next block signer-- yourself-- keep trying different ways to sign, OK, that's basically proof of work, but instead of hashing, you're signing.

And this has happened in many altcoins. One that I think was kind of interesting was Nxt, which is the people who later went on to make IOTA. They said, OK, well, we have a deterministic signature scheme, and so this can't happen, right? Given a message and a private key, there's only one signature that can be produced.

And this is true. So Bitcoin uses this. It's called rfc6979. Basically, the random nonce can be deterministically created from your private key and message.

The problem is-- I don't know if they realized, probably not-- but you can't enforce this.

It's a policy that says, OK, I'm going to make a deterministic signature. But given a signature, you can't tell if someone did this or not, right? So it's up to you to obey this rule. But you can't verify that anyone did it.

So that one also quickly devolved into proof of work and in a tricky way because a lot of people read the documentation and said, OK, well, there is no way to keep signing.

Their function that they wrote, there's only one way to sign. When you sign a message with a public key, you get a single signature.

But people who knew how it worked under the hood said, oh, I can change this code. It will still be accepted as a valid signature even though I'm using a different signing scheme. So this is one issue, stake grinding. There's ways to get around it.

So let's say you make it deterministic. You have some way to enforce that a signer can only create one message. There are certain signature schemes where you can verifiably make sure that a single key can only make a single signature on a message.

Other ways to do it is to, say, have this big lead time, where the people who are signing the next block are determined by entropy created days or weeks ago. So it's hard to know what's going to happen in a week because it's a long period of time's worth of entropy going into it.

OK. There's another issue with proof of stake called nothing at stake. So rehash-- rehash-- ha ha. So in proof of work, this happens, right? This happens just by accident, where OK, there's a block. And then two people are mining on top of it. Oh, they both got this and then sometimes even this. That almost never happens in Bitcoin, but sometimes, where you just randomly get two split chains coming out at the same time.

But then this happens, right? Someone builds on this one. OK, you get rid of those two.

And that's just because this is a random process. You can mine on whichever you want. One of them is going to happen first.

You can mine on both. You can split your hash bar. If you have a billion hashes per second, you can say, well, I'll do 500 million here and 500 million here. You can equivocate.

It's sort of equivocation that way. But one of them will finish first, with very high probability. So eventually, this happens. And then everyone just starts building off this 00a3.

Proof of stake-- so this happens. Also, notice that the hashes don't-- they don't start with zeros. So this happens. But then this happens. And then this happens. And then it just keeps going. Because why not sign both, right?

I don't know which one's going to win, right? As a proof of stake miner or staker, I don't really know which one's going to win, which is the same case and proof of work. In proof of work, I just flip a coin and pick, right? I don't know what's going to win.

I think in bitcoin, the default is go with the one I saw first. Because whatever. It doesn't matter who wins, right?

Basically, I win, whichever I pick. If I'm the next one making the block, I win.

But what I don't want is I don't want to be this this-- wait a sec. What I don't want to be in proof of work is I don't want to be 008a, right? I don't want to be the last one to make a block, and then something else comes out here.

So I'd rather be 00f2. But you don't know here, right? So you just pick one. OK. Well, we lost. In proof of stake, you don't have to worry about that. I'll just build both, right? It takes me no work to sign. So I just sign off on both of them and make parallel chain. And then everyone just keeps doing that. Because if I pick one, I might be wrong.

If I pick both, I can't be wrong.

So this is a problem called nothing at stake. You have no risk here. You're not actually doing any work. You might as well just keep going. So yeah. Faced with two blocks, why not sign both?

So the mitigations-- one of the ideas early in Ethereum was, OK, this thing called Slasher, where if you can prove that a signature from a different chain-- so the idea is on chain 00a3-- oh, wait-- oops. Sorry. On this chain, you say, hey, look. This guy equivocated.

He signed off on two blocks, one of which exists in my history that I can point to, one of which doesn't exist in my history, but I can provide that signature. I can say, look, here's two signatures at the same block height from the same person, but they're signing different things. Let's not invalidate this block, but let's take all the rewards from the person who signed.

OK. So yeah. This is called Slasher. I believe in Ethereum, like, 2014, they were saying, hey, we're going to do proof of stake. And then they tried all these things and sort of, oh, it's non-trivial. Let's start with proof of work, and then we'll move to proof of stake later. And that's currently their plan. So this was one of their early algorithms, their early ideas.

There's also issues with the mitigation because maybe it's hard to get this proof into the blockchain. Because the miners, or the stakers, are the ones who determine what gets in and what doesn't.

And so they certainly could say, look, if the person who created this block then sees in this block that there's a Slasher proof where, hey, I just now proved that you equivocated and destroy your account, well, the person who signed this doesn't build on this and instead forks off and builds their own, right? Why would I continue to mine on a chain where I just lost all my money?

So there's all sorts of weird-- and there's mitigations for that. So it's a little bit whack-a-mole, where there's all

these weird problems, and you have to try to fix them.

And then fixing them introduces new weird things. So there's a lot of people working on it.

Yeah. Long-range attacks is another big one. Let's say you go back to the genesis block, and you rewrite history. In Bitcoin, you can also do this. It's just really hard, right?

You could say, I'm going to get a bunch of miners, rewrite all nine years of Bitcoin's blocks, and we know exactly how long that will take.

sipa has a nice graph of it, I think. And sometimes it's not very long. Sometimes it's actually pretty easy to do that. Yeah, that's the other site. Hash rate-- this-- wait-- yes. Bitcoin network proof of work equivalent days. That's this year. This is since the beginning.

What this means is, OK, given the current hash rate in Bitcoin, how long would it take to rewrite the entire history of Bitcoin? Because the hash rate goes up, right? And sometimes it goes up in-- so this is the all-time hash rate. It basically always looks like this because it's exponential. So if you looked at it three years ago, it would look pretty much the same.

And then this is a log chart. So this actually does show some interesting detail, where, OK, this is when the GPUs came out. This is when nothing much happened in 2012. This is when ASICs came out. And then it just keeps going up. And this is log scale, where each number is 100 or 1,000 times. So it's big.

So when you have these really sharp upticks, like in mid-2010, the current power of the network is so high-- that probably corresponds to this, the lowest point there, because that's the steepest slope, where I don't know what this is, a week?-- you could rewrite the entire history of Bitcoin in a week and destroy everyone's currency and have all the coins yourself if you had enough power under your control, which you could probably do.

And today, it sits at around 200, a little less than 200, so a little over six months where, if you devoted all the current hash power, you could rewrite all the previous history. Still, six months-- it's a lot. And it's interesting how this changes. This is, I guess, when the ASICs first came out, and then the slope [INAUDIBLE].

So that is an issue in proof of work that can happen. It's not clear what would happen.

The software, at least in theory, the idea is if something like that happens, where there's a reorg that spans nine years, well, extend this out to 500,000 and say, OK, all the stuff you've been dealing with the last nine years, it's out. We've got this new history now with one person owning all the money, presumably.

The software will do that, probably, or it'll just crash or something. It's sort of seen as like, well, if there's a reorg

that spans weeks or months, the whole system's broken, kind of. But probably.

So this is an issue for Bitcoin. But it's a big issue in proof of stake because it seems possibly more feasible, right? In Bitcoin, there have been times when it's like, whoa, a week, yeah, that's feasible, whereas right now, it's like, OK, it would take all the hash power in existence maybe seven or eight months. So that means that Bitcoin would just halt for seven months, and then you'd see this reorg happen. And also, it assumes 51% of the miners are doing this.

However, proof of stake, it might not be that expensive, especially if you can get old keys.

So if you've got the keys from the genesis block in this proof of stake currency, you can rewrite a history from those with different transactions. So yeah. So a lot of the times, the solution is, OK, well, delete old keys and assume 50% honest.

That's tricky. Because old keys can be sold. And I know that people have asked, hey, can I buy your old Ethereum keys? Why do you want these? Well, if proof of stake happens, they might be worth something. Currently, if you have keys that are for addresses or accounts that don't have any money anymore, it's not very useful but possibly can be sold. Yeah?

AUDIENCE: Most proof of work schemes that I'm aware of use a [INAUDIBLE].

TADGE DRYJA: A proof of stake?

AUDIENCE: Yeah.

TADGE DRYJA: Yeah. So you checkpoint it. So basically, to prevent long-range attacks, the developers, usually, in GitHub, will just say, OK, well, the block hash here is this.

And so you can't reorg before that. But then, what's the mechanism for that? If it's the developers just sticking in a checkpoint, that's not really decentralized.

AUDIENCE: In the example of Aircoin, there's actually a checkpointing [? software ?] that, every few hours, forces a [INAUDIBLE].

TADGE DRYJA: OK. Well, that's even more not very decentralized. You've now got a pubkey hardcoded into the code, which then calls out to somewhere and says, OK, what's the correct block to be on? And then that key signs something, says, oh, go here. OK. Yeah.

That's pretty centralized.

So this is one that I think is-- it's hard, right? Because before these things happen, it's easy to dismiss as like, this is crazy. That's never going to happen. Who's going to go back and buy up all these old keys that people were supposed to have deleted? People probably did delete them. You can't assume that they haven't.

And even if they did and you made this big reorg, well, everyone would know in practice that that's not the right chain, similar to if there's a nine-year block reorg in Bitcoin, everyone would know, OK, come on. We all knew what the blocks were yesterday. You can't just tell me there's this entirely new history today.

And in practice, that's probably true, right? If there was a nine-year block reorg in Bitcoin, probably, people would just ignore it. Similarly, in these systems, probably, they would ignore it. But it's harder to reason about in these systems, I think.

But anyway. So proof of stake, there is a bunch of coins that use it. In a lot of cases, they have centralization that's hidden in certain ways. I like the term Greg Maxwell came up called trust laundering. It's not money laundering. It's trust laundering. You sort of move where you're trusting and try to hide it.

Yeah. So very common. But this also deals with, OK, who gets the initial coins? Because they have an enormous power over the system. So very common for altcoins-- less so now, I think-- was start with proof of work, then transition to proof of stake. Do they still?

Not as much.

AUDIENCE: Now you don't need to do that because you [INAUDIBLE].

TADGE DRYJA: Right. So you just do ERC-20. Yeah. But it used to be, like 2014, 2015, a lot of the coins would be, OK, proof of work for the first month, two months, whatever it was, with some weird algorithm and Comodo gravity well and all sorts of weird stuff.

And then it would switch at a certain point to proof of stake.

So that proof of work period builds out the list of who owns what in a sort of provable way. And then, from there, you can use proof of stake. Because you've got a, hopefully, fairly well distributed set of who owns what.

Yeah. Some things still do this. And some things have hybrid, where you still have to do work each block. But depending on how much stake you have, depending on how many coins you have, you have to do less work.

I think decred is something like that. That actually can make sense. Because you've still got work, and you can mix them in ways that negate some of the downsides of both.

Delegated proof of stake. Well, signing requires you to actually do something and be online.

So in a lot of proof of stake currencies, very few people actually stake because they don't care. And they leave money on the table, sure. But they don't want to run their own node. They don't want to deal with this stuff. They just buy it, keep it on exchange, or maybe keep it on their own computer.

So instead, you can endorse a leader by signing with your coins and saying, hey, I'm not going to actually sign off on blocks, but I'll sign off on a different key, who now can sign on my behalf. And so this can be called supernodes, masternodes. And then is it peer-to-peer, or is it client server is what it becomes.

So the current one that's pretty popular is called EOS, where I think they're doing this kind of thing. And they're like, well, there's 20 computers that run the network. And you can endorse one of them, and they'll give you some kind of profit back or something.

And you need a million dollars to run one of these computers. Because it's going to be super powerful, and you can do free transactions that way.

So this gets pretty far into client server territory, where you're not up here. You're just the client, and you ask them what is going on, which, in many cases, works, right?

But then again, it's not quite as interesting because that's sort of the system we already have with banks. So meh.

But in a lot of cases, yeah, that's what people want. I sort of joke that I should just make one called Central Coin, where there's just a nice server, and it just keeps track of who owns what, and then have an ICO. And people would use it. It's like, hey. So I think people would be into it. Anyway, so that's distributed proof of stake.

OK. So proof of stake, in general, it's hard to resolve conflicts using only the system itself. Proof of work has this really nice property where it's taking something from the external world, be it CPU time, be it chips, and using that to get a dead reckoning on the system itself. That's nice.

Proof of stake, the really difficult part is it's a closed system. You're trying to resolve conflicts within the system using no external data. That's really hard.

Another problem people complain about is that the rich get richer. I agree it's a problem.

It kind of sucks. But proof of work is the same. I don't feel like any of these systems will change the inherent Pareto distribution of wealth in the universe. That just seems to be how things work. It'd be cool if everyone had the same amount of money, I guess, maybe.

But I don't think it's going to happen.

Yeah. And then it relies on different assumptions. So the assumption in Bitcoin is 51% of miners can destroy the system. So in some cases, people say, well, you assume 51% honest in Bitcoin. Kind of. It's more like 51% rational. In Bitcoin, you could have miners with 51% attack the system, but they make more money by doing the right thing.

So there are certain cases in proof of stake where honesty is not as profitable. And if honesty is not profitable, well, maybe there's a lot more incentive to be not honest in the terms of the system, whereas in Bitcoin, it's like you've got this nice system where it seems like the honesty is not just enforced by people trying to be nice. It's enforced by people being greedy, which seems more common than being nice in the world.

So there's tons of research in proof of stake. Some of it's pretty interesting, pretty clever.

I'm not super hopeful. But the thing is it works until it doesn't. And so it's really hard to show security under different attacks.

So what I would worry about is you've got this proof of stake system. It seems to be working. And then something weird happens. And it's revealed as, oh, actually, this was a lot more centralized than it seemed to be, which is also a worry in proof of work. OK.

Any questions, proof of stake? Yes?

AUDIENCE: A quick one on the economics. Is the stake the nodes have. Yeah, right. So in essence, if it was on EOS, it's how many eos coins you have.

TADGE DRYJA: Yes, yes.

AUDIENCE: So then there's probably some economics where people who are not miners will lend their stake to get a return.

TADGE DRYJA: Basically, yeah. And a lot of coins, the majority of the coins, are held on different exchanges. And so that means the exchanges would possibly be able to then stake and get some kind of revenue.

AUDIENCE: Right. So they would be taking their customer funds that are in wallets-- they would have to be, probably, hot wallets-- and then the exchanges would have a leg up to be the miners.

TADGE DRYJA: Yes.

AUDIENCE: Or a [INAUDIBLE].

TADGE DRYJA: Yeah. And so that's also an issue, that there's all different ways to mitigate.

And you can say, oh, I'm in an exchange, but I get to sign and delegate my stake to someone else. And then I get some kind of revenue sharing from that.

But yeah. It's also tricky, how do you incentivize it in terms of how much new coins get issued to the people staking? If it's zero, then there's no real incentive to do it. You could have a lot of different chain ports.

If it's very high, then it's very, I don't know, Gini net curve goes way crazy because the rich really get richer. Because the more coins you have, you have an enormous amount of revenue. So you need a balance there that's also a bit difficult, economically.

That's also a problem in proof of work. I feel like one of the big issues with Bitcoin is half the coins came out in the first four years, where hardly anyone was aware of it.

And that makes people not want to adopt the system. Because it's like, well, wait. You guys just got all this coin for, basically, free. You were just running your computer in 2011. I'm not going to be part of that system.

So I feel like it would have been nicer if it was more of an S curve and less of a log curve. Because then, maybe it would ramp up to 2015, and then lots of coins were coming out. But anyway.

So Ripple or Stellar are sort of an extreme case of that, where all the coins were initially made by one entity. And so to me, I'm like, I don't want these. That's your money. You just made it up yourself.

I don't know. It just feels like kids being like, I have a bazillion dollars, and I write a bazillion dollars. It's like, OK. Well, you just wrote that. I'm not going to honor that.

Anyway. OK. So proof of stake, kind of interesting. Proof of space, there's a bunch of ideas here.

Some of them-- SpaceMint was some people here. I know Bram Cohen, who is the author of BitTorrent, right-- he made BitTorrent whatever 10, almost 15 years ago now-- he's now working on some of these kind of things.

He's working on a proof of space coin called Chia. And the idea, it's still similar to proof of work. But you're using some kind of memory or storage space, rather than your CPU. And the idea is the benefit is, well, maybe it doesn't use as much electricity.

And also, from talking to Bram with his, he's saying there's much more dead storage space than dead CPU. So every computer has a CPU, but the CPU here is not that powerful. It can't really compete with ASICs. But lots of people have empty hard drive space.

And you might have a terabyte of empty hard drive space, and one terabyte's as good as another. And it's also empty is what you need. Because you need to fill in space for these systems.

And so if you're like, well, I'm AWS, I've got tons of hard drives-- right, but people are using them. So you're not able to then just fill it in for these kinds of systems-- well, to some extent. They do have a lot of slack, and they can probably do it. But several ideas. Some are pretty cool.

So one example-- this is sort of an example idea. It's not fully fleshed out, but to give you an idea. So you buy a 10-terabyte drive, OK? You precompute 100 billion key pairs.

And this takes a long time. And this is work, right? You're doing 100 billion computations of coming up with the random private key, multiplying by G, getting the pubkey.

And you store it in a key-value store, like a database-- LevelDB or something-- on your hard drive. And the key is the pubkey. And the value is the private key. So you remember your key pairs. But you've sorted it in a way so that you can quickly go through the pubkeys, right? So you have logins, search time, some kind of binary tree-- whatever it is-- in your database.

And so then, the idea is, to create the next block, you want the key closest to the current block hash can sign. So a valid proof of work-- you have some threshold, maybe-- a valid proof of work is whoever's got a key that's very close to this, they're able to sign. And those keys do not exist on the network prior to the signing procedure. They exist just hidden on people's hard drives.

And so the idea is you could just try to keep computing keys till you find one and then sign with it. And that would just make it completely proof of work. So it is proof of work. But the idea is you can precompute. You can do all the work beforehand.

And then you can use that precomputed proof of work later on and possibly multiple times.

In practice, it's not going to be multiple times. Because you've got 100 billion, and you're never going to use the same key twice.

The idea is everyone does this, right? So you have trillions, quadrillions, however many keys out there that everyone's generated. And every block that comes out, someone will say, ah, I was lucky, and I made a key that can now sign the next block.

So it's work, but it's amortized over weeks, months, years. And it's basically how big is your storage. If you have a

lot of storage, you can precompute a lot of this stuff and then use it. So it's kind of a cool system.

There's other asterisks that actually get a little more complex because you need some timing mechanisms, as well. But I think it's a cool idea. Hard drives then maybe get more expensive instead of graphics cards getting more expensive. I don't know.

But the idea is like, yeah. You use the work but later on. So I don't know. Any questions about that one? Kind of cool. There's a SpaceMint-- it's a cool paper-- about how to do this cryptographically.

And then Chia is another one doing this kind of idea. I like it because it doesn't really change the assumptions of proof of work, right? It is still a form of proof of work. But instead of burning electricity, you're using hard drives. Kind of cool.

AUDIENCE: Wouldn't you still have the same issues, like you'd have to have ASICs just filling racks and racks and racks of hard drives for keys. Is it different?

TADGE DRYJA: You already have that, right? There's server farms where they're just storage farms, and they just have tons of hard drives. So that's already a thing.

AUDIENCE: But I'm saying it isn't any different. You'd still be-- it would be [INAUDIBLE]-- TADGE DRYJA: I guess the idea is the economies of scale are not quite as bad in that hard drives are already, basically, optimal at doing this. And if you can make a system that's better at doing this, you've just made a better hard drive, which, hey, great. You made a better hard drive. Everyone can use it. So I guess the idea is it's not as specific because it's just, hey, store a bunch of data and then [? seq ?] quickly through it.

AUDIENCE: It still seems CPU-bound, right, compute-bound.

TADGE DRYJA: No. Once you generate this-- OK, generating this is compute-bound, right?

You need to populate your 10 terabyte drive. But that might only take a few hours, right?

To make 100 billion key pairs, I'm guessing less than a day, right? Yeah. And you can do that in parallel. So you make it. And then once you've made it, you leave it there. And then you're going to be able to mine with that forever.

AUDIENCE: But see, the winner would be whoever can fill the most number of keys per second going forward.

TADGE DRYJA: But you need something to fill. I think the hard drive cost is much higher than the cost to fill it, right? Because a 10 terabyte hard drive is going to be, like, $200. And with a cheap CPU, you can fill a 10 terabyte

hard drive every few hours. So yeah, the CPU is a factor, but I think it's going to be smaller. Yeah?

AUDIENCE: At least from what I understand about SpaceMint, the idea is to make it not I/O bound [INAUDIBLE]. So if I have a better or faster hard drive, that's not [INAUDIBLE].

It's just [? another ?] space.

TADGE DRYJA: Because yeah. This is just a seq, right? And even a crummy hard drive, you just read through your binary trainer. You're like, oh, found it.

AUDIENCE: I think the point is that you're only supposed to do one with those pub [INAUDIBLE].

AUDIENCE: Yes. [INAUDIBLE].

TADGE DRYJA: So you look at, OK, here's the current block hash. What do I have that's close? Seq to that on my drive. Nope. I'm not going to win this one. OK. Let someone else do it. Or you seq, and you find it. You're like, hey, cool. I can sign. Yeah?

AUDIENCE: Is this also known as proof of space and time?

TADGE DRYJA: Yeah. So in Chia, they add this time component, which I'm not 100% clear on how it works. It's changed a little bit. But the idea of a time beacon is you want some function that cannot be parallelized so that if you have 100 computers doing it, it's not going to be any faster than one computer doing it. So basically, whoever's got the fastest single CPU will always win this. And that can be a time beacon for these space things.

Yeah?

AUDIENCE: Once you start to have a lot of storage, if you're in a search will find [INAUDIBLE], do you kind of go back to-- you're almost getting hurt. [INAUDIBLE] So what I'm saying, you're just doing the same thing again?

TADGE DRYJA: I think that's a [INAUDIBLE]. If you have tons of hard drives, and you want tons of key values, and you just want to find one or find the closest match, you can do that in log n time, even over a whole data center. Google, you just search, and it's like, hey. It comes up in a fraction of a second.

So that's doable. You need good software, and you need a good infrastructure to do it.

But I think it's doable. It scales pretty well, I think.

But yeah, it has the same issues where you just end up with people buying a whole bunch of hard drives. And what are the economies of scale there? There's arguments that it might be better than the economies of scale

with, like, ASICs, that we heard about on Monday. Maybe not. I don't know. We'll see if this takes off. It's fairly new. It's an idea that's been talked about for a few years.

AUDIENCE: It's [INAUDIBLE] space. And they've been around for years in this [INAUDIBLE].

TADGE DRYJA: OK. Well, so there's different ideas. I don't know. It seems like one of the more interesting. There's cool math and cool crypto involved. OK. So it'll work but amortized.

So and perennial MIT favorite, IOTA, they use a directed acyclic graph. And now, a chain is also a DAG. Because it's directed. There's no cycles. It's a graph. But it's a pretty simple one. But the ideas have multiple parents, right?

So if you have a block, instead of just pointing to one, you could say, well, now, this can happen in Bitcoin, right, where you've got two blocks, both supporting the same thing.

But this cannot happen in Bitcoin, where you say, oh, I'm coming off, and I'm referring to both of these. But in a directed acyclic graph, you can do that.

So Ethereum actually does it. I think Joe Bonneau was mentioning that. And so what happens is you endorse one as the correct one and one as the uncle. So you're saying, no, this is the real one. But I also saw this one. And give this one a little bit of money, right, maybe not the whole reward they were hoping for. But they get something.

And so then, something like IOTA says, OK, we have lots. And every node points to two previous. And so you get this whole map. They could even point to different heights. So I can point to here and here. I don't really know why you'd do this.

So there's some ways it could reduce latency, right? So one of the trade-offs in Bitcoin is 10-minute block time seems fairly arbitrary. It's kind of slow. Latency-- people don't like latency because they have to wait. Fine.

The more important metric, I think, is that it leads to miner centralization and miner pools. Because if there's only 144 blocks a day, like in Bitcoin, if you have a millionth of the hash power of the network, you're just never going to find anything, right?

If you're just mining on your own, over the course of a year, you're probably not going to find a single block, and you'll have wasted all of that effort. So instead, you join a pool and try to pool with other people to distribute those rewards.

However, if there's, like in Ethereum, a block every 15 seconds, and there's millions of blocks that come out, if you have a millionth of the hash power, you might find a block.

The blocks have smaller rewards, but chopping it up more finely is a nice way to do it.

So like P2Pool, which you were gone, is a way to try to make that. And it's important to get it more decentralized. But it's hard. Yeah?

AUDIENCE: P2Pool does that?

TADGE DRYJA: Yeah. Yeah. P2Pool sort of does this as a layer on top. So it could help latency.

And then, it could help reduce the amount of orphans in the blockchain, which then helps miner centralization. So there are interesting ideas for it.

But it doesn't help scalability at all, right? It actually hurts scalability. Because now I have to keep track of all these things instead of just one chain. And I'm going to have to keep track of all the data, anyway, if I've got the UTXO set. I can't just ignore parts of it.

So it doesn't help scalability at all. And in the case of IOTA, their custom ternary hash functions also don't help much, either. So that was what we wrote about last year.

They made all their own weird stuff. I don't know.

So there are interesting ideas between directly acyclic graphs. But it feels like when they say, hey, this is more scalable, I get very suspicious. Because it doesn't seem to help scalability. It does possibly help some other things. Yeah?

AUDIENCE: Is, effectively, Bitcoin refreshing [? test? ?] They've made it so every block has min difficulty. And so the transactions are instant because, right, it doesn't change-- TADGE DRYJA: What, in IOTA, or in-- AUDIENCE: In any of these stack coins. So it doesn't cost much to make a new block and make a transaction. And if everyone's competing to do that, then they're just another ton of orphans, and [INAUDIBLE].

TADGE DRYJA: Yeah, yeah. So also, in the case of IOTA, the idea is-- well, they don't call it this-- but it's the equivalent of saying every block must have one and only one transaction.

The transactions themselves have proof of work on them, and they point to each other, which, basically, is the same as blocks have one transaction, and you do the work yourself instead of paying a miner to do it.

I think that's not a great deal. And they say that that means there's no fees. I think there's still fees. Because you're just doing the work yourself. I think that's equivalent to saying, there is a refrigerator at Home Depot that doesn't use electricity. So hey, it's electricity-free. But there's a crank on the back, and you have to crank it to

make the refrigerator cold.

Yeah, you could make a refrigerator that way. I don't think people would want it. You're still doing the work. You just have to crank it instead of plugging in it. So the idea of you don't have to pay miners a fee to get into a block because you just mine the block yourself or, in this case, you mine the transaction yourself. Doesn't seem that useful.

Anyway, so IOTA's fun. We talked about them a while ago. OK. Last one, and it's fun-- proof of idle. It's an old idea that I wrote up four years ago, almost. It probably doesn't actually work that well. But a lot of these things don't work that well. Even if it works, it would just move the costs from opex to capex.

But what you do is you prove that you're not mining, and you get paid. And the other fun thing about this is it led to some of the ideas in Lightning Network. So the idea is, in Bitcoin, the difficulty adjusts so that blocks come out every 10 minutes, right? Every 2016 blocks, you look at the timestamps, adjust the difficulty.

So new miners coming in make it harder for the existing miners. So if you're a miner, you really don't want anyone else to start joining this network and mining. You really like it as it is. So if you have the first ASICs, you're good, and you don't want everyone to come in.

And this is why so many people buy mining equipment and lose money. Because they look at the current difficulty, and they say, hey, here's this device. Here's how much electricity it uses, how many coins it generates.

This is profitable. I'm going to buy this thing. And what they don't realize is that by the time they actually get it, or maybe a few weeks later, the difficulty is doubled.

And now it's making half as many coins for the exact same amount of electricity.

So yeah. And that's a fairly unique property of bitcoin that's not the case in, say, gold mining or other extractive industries. There's some kind of curve in gold mining where if you pay twice as much money to have twice as many people dig holes and mine gold, you won't get twice as much gold. Maybe you'll only get 10% more gold. But you will get some more, right? And if you drill for oil, you pay twice as much, you'll get some more. But yeah?

AUDIENCE: And it's worse than that. Because you're still obligated to mine. Because making $0.50, they're making half as much as they're going to make, it's still better than making nothing.

TADGE DRYJA: Yeah. So that happens, too. So yeah, in Bitcoin, two times the mining leads to 1X the coins mined, right? There is zero marginal product of labor in this system, which is weird and counterintuitive and doesn't really exist in normal life.

Most things have some kind of sublinear curve, where, yeah, the first low-hanging fruit-- if you're mining coal, and you start in Pittsburgh, and it's just sitting there on the hill, and you're like, hey, that was easy. And then, eventually, you have to start open pit mines and digging holes. And it's expensive, and you don't get as good. But you still, if you double your effort, you're going to get some extra stuff out.

Then again, there might be cases, I'm sure, in economics, where eventually, it goes negative, where if you continue adding members to an organization, eventually, they're just like deadweight, and they make it less efficient or something. But in Bitcoin, it doesn't matter how much you mine. You always get the same number of coins.

So the obvious thing there is, well, we should all just stop mining. And then we'll have zero electricity usage, and we'll still get the same amount of coins.

AUDIENCE: The example is Nobel prizes. It doesn't matter how many academics chase them, there's a unique number [INAUDIBLE].

TADGE DRYJA: OK, yeah. Yeah, that's another. Yeah. However good the science is, you just get one Nobel Prize. So OK. So let's say there's two miners. And there's, obviously, more.

And they're each mining with 2 gigawatts.

And they both think, well, wait. If we both turned our mining down 5%, we'd still get the same amount of coins, but we'd be using 5% less electricity. We should have a meeting, maybe some kind of cartel.

So this is sort of OPEC, right? This is a classic cartel. If we all restrict our output, we can raise prices and reduce our costs. In Bitcoin, it's the perfect cartel environment.

Because if we all reduce our output, we get the same output, sort of, right? If we all turned down mining 50%, we all still get the same number of coins. The network still works fine. Maybe nobody even knows we're doing this. This is ideal cartel scenario.

Problem for cartels, generally, is that cartels are hard to maintain, especially when there's not a lot of trust. Because there's so much profit in defecting and going against the rules of the cartel. And OPEC has this all the time, where one of the OPEC countries says, we're just going to pump a bunch of oil and sell it. Because everyone else reducing their output raises the prices. And now we can sell more with higher prices.

So in this kind of system, if all the miners said, hey, let's all turn down 50%, the one miner who then mines at full blast is going to make a lot of coins. So this is the problem.

In Bitcoin, nobody trusts anyone, right?

So the solution here is trustless collusion. And in the papers, I was like, well, is it collusion or cooperation? They both sort of mean the same thing. And collusion is just a bad word for cooperation. Cooperation's good. It's on Sesame Street.

OK. So the basic system is Alice pays Bob not to mine. So first thing, Alice needs to prove that Bob can mine. Because she doesn't want to pay Bob if Bob doesn't have any mining capacity that he's taking offline. She only wants to pay Bob for having the ability to mine and then not doing.

So A posts a block header at a specified time and says, OK, B, mine for 10 seconds, and give me your 100 best shots, right? So in Bitcoin, you have this fixed difficulty. And you just say, OK, anything below this block hash is valid.

But you could make one where you say, hey, give me your best 100, and I can then extrapolate, or interpolate, from that what your hash rate is. And you can get pretty accurate. So they have to do some work, right?

Bob does some work but for a brief period-- I don't know, 10 seconds. You don't want latency to be an issue, so maybe 10 seconds, maybe a minute-- whatever-- some small amount of work to prove that they have the capability to do the work. They respond with that. And Alice validates it and says, OK, you've got X amount of work capacity.

Alice then creates a 2 of 2 multisig transaction and sends one bitcoin to this address and builds two transactions with Bob. This is sort of like, how Lightning Network looks, Lightning Channels. But this predates it by a year or two.

So the idea is you've got this funding, and then you've got two transactions. They're both signed by both parties and held by both parties, although in practice the one that pays out to Alice, Bob doesn't really need to store. He doesn't like it. And the one that pays out to Bob, Alice doesn't need to store.

But the idea is they have conflicting locktimes. So in the transaction header, you can have a locktime, say, OK, this transaction is only valid after this point. So the one that pays Alice is height plus 144. So the current height of the blockchain plus 144, well, that should take about a day, right? If blocks come out every 10 minutes, 144 blocks is one day.

For Bob, Bob gets a coin with the current locktime plus 24 hours. So in Bitcoin, you can specify either Unix time or block height. And I think everything above 500 million is a time. Everything below 500 million is interpreted as a height, which means this whole system runs out in a couple thousand years.

So this is a race, right? There's two transactions. One of them can be broadcast first, depending on how fast blocks come up. So if blocks come out very quickly, this will be valid first-- 144 blocks. Maybe it only takes 20 hours, and these 144 blocks have come out. And Alice can post this transaction and get all of her money back, get her one coin back.

However, if blocks come out slowly and, after 24 hours, only 120 blocks have come out, Bob can post this transaction first. And it will be valid and confirmed, and Bob gets the coin.

So now Bob's incentives are, well, I'd like 144 blocks to take longer than 24 hours. Because then I'll get this coin.

And Bob has the means to influence this, right? Bob's a miner. So Bob can say, well, I'll just mine less. And if I mine such that, in 24 hours, only 130 blocks come out, I get the coin. Cool, right?

If blocks come out fast, on the other hand, Alice gets her money back. So Alice has no real risk here that Bob will run away with the money without doing his part of the job.

So Bob can slow down his mining in order to get the bounty coins. If Alice estimates incorrectly how much profitability Bob has, Bob just keeps mining. Alice gets her money back. This collusion didn't occur, at very little cost, right? Bob had to prove a little bit. There's some coordination costs. But basically, nobody loses the money.

So Alice can put whatever bounty she thinks is beneficial to her. OK, I'll pay you three coins not to mine as much. And you can chop this up. You can have a lot of different, smaller transactions and make a curve. If you mine a lot less, I'll pay you more-- things like that.

Yeah. So that's the idea behind proof of idle. It feels like it might happen, to some extent, long-term. It feels sort of like nuclear weapons, where everyone's got them, but they don't really use them. They just threaten.

So it feels like mining could be that kind of thing, where, well, we've all got all this mining capacity, but we just threaten each other with it, and we don't really mine. Because the mining's actually really expensive, and it's just the threat of mining that you need.

And if someone tries to do, say, a 51% attack and reorg, the existing mining infrastructure can spin up and say, oh, no. You thought you were 51%. You were actually 5.1%. And we're 90% offline. And we all come back online and reorg you out.

It's an interesting-- I think it's a fun idea. In practice, it doesn't work now because it's mostly capex. So this doesn't help if your main constraint is building the chips, which it sounds like it still is, from David's talk on Monday, right? Yeah, you need to get electricity, sure. But also, the really big problem is getting a supply of all these chips.

And this would just exacerbate that. This would make it so that I don't even need electricity.

I just get the chips. I don't really care how much I'm paying in electricity because most of the time, my chips sit off. I just need to occasionally prove I have them and have the ability to mine with them.

So it wouldn't solve the problem of proof of work in terms of people spending tons of money on it. But it would move it towards less electricity usage, more fabrication plant usage. Is that a good thing? Is that a bad? I don't know.

And in certain cases, it may be that it saves people money. It may be that, hey, if we can collude in this way, actually, we save money. And then we can use it to build more hash power.

So this might happen. I was sort of convinced in 2014. I'm like, oh, I think this is going to happen. I also thought that it would become predominantly opex in 2015 or '16. And it just didn't-- not even close. So who knows? But it's, I don't know, kind of a fun idea.

OK. So in general, lots of new ideas out there. Proof of work does seem to work. But I think one of the big issues is it's not really compatible with the Kurzweil/Roddenberry future idea.

Has anyone read Ray Kurzweil, like, Age of Spiritual Machines?

It's sort of this futurey AI is going to make everything great, and we're all going to live forever, and computers are going to be our best friends, and we're going to take over the universe. And Gene Roddenberry of Star Trek is sort of a popular, not quite as crazy-- maybe just as crazy-- view of that.

In Star Trek, they don't have money. They just sort of explore the universe and make the world a better place and stuff like that. And it's like, cool. And there's a lot of people who are into technology and research who are, maybe not consciously, but identify with these ideas, like, yeah, we're making the world a better place. Computers are going to help people, and it's going to be great.

And I think one of the issues with Bitcoin and proof of work is it doesn't fit into that, right? It's sort of dystopian, in a way. And then, it's like, wait. We're going to have giant server farms performing SHA256 for the next 50 years? Like how? How does the benevolent AI fit into this system?

So that's one of the reasons people-- the proof of idle idea was-- I was at University of Virginia, and Avi Shalot, who's now at Northeastern, he was like, I hate Bitcoin.

Because the whole point of SHA256 is that you can't find collisions. That was the design-- collision-resistant hash

function.

And now, you've built this giant system, which the whole point of the system is to find collisions.

Like, ah. That's the opposite of what it was supposed to be. And so it was just sort of inelegant and ugly. And that was where I'm like, well, maybe you don't really have to mine that much. And so I wrote this thing about proof of idle.

But anyway, that's one of the issues that people have with proof of work. And that's, I think, one of the big motivating factors for all this other research into different consensus algorithms. And further research is required, maybe.

The interesting thing is a lot of the people doing proof of work are fine with it, right?

So like David, Monday, he's not interested in proof of stake. Despite the proof of work ecosystem being so crazy, he's just like, nope, this is what I'm doing.

So none of the miners are interested-- I mean, miners, their job is to mine. So they're usually not interested in proof of stake. And then Bitcoin, in general, a lot of it's like, no, this seems to work. We're OK with it. This is the cost we're willing to bear.

But other people want to do other systems. So further research is required, or not. But it's happening. There's tons of research into different consensus mechanisms. I would say in academic research, that's the biggest thing.

What I would like to see more of is more academic research in proof of work. Because there's a little bit, but there's all sorts of interesting things with proof of work where it's like, oh, something like proof of idle or something like the stuff David was talking about on Monday. There's not much economics research into this.

And I think it's a really interesting question, where you've got, hey, I've got a device that prints money, and I want to sell it to you. What? How does that work in terms of economics?

How much should I sell it for? How much should I buy it for-- things like that. So there's a lot of new research into new proof of stake, different consensus mechanisms, economics that way and not as much into new forms or how proof of work works, which I think is, actually, really interesting.