IN BLOCKCHAIN WE TRUST?

MARTIN GLAZIER

Institute of Philosophy
University of Hamburg

Draft; comments welcome

The late 2010s saw a wave of enthusiasm about blockchain, the 'distributed ledger' technology that underlies bitcoin and other cryptocurrencies. Between January and May of 2018, for instance, blockchain startups raised more than $1.3 billion in venture capital.[1]

But serious questions about the technology are now coming to the fore. We don't know whether cryptocurrencies will ever be widely used. And although many other applications of blockchain have been proposed, in domains ranging from medicine to logistics to government, we don't know how successful any of them will ultimately be. In this paper, I try to determine which potential applications of blockchain deserve the most attention as we continue to explore the technology.

My guiding assumption will be that the most successful applications of any technology are likely to be those that exploit its distinctive potential—those that take advantage of what the technology does best. For example, a French press can be used to make coffee, or it can be used as a paperweight. But it's only the first application that exploits the technology's distinctive potential.

As for the French press, so for blockchain. We should focus our attention on those applications of blockchain that take advantage of what blockchain does best.

But what is that? Many people have thought the answer to this question somehow involves *trust*. But they've had conflicting views on how exactly trust is related to blockchain.

Satoshi Nakamoto, the inventor of blockchain (and bitcoin), sees the technology as, in a sense, opposed to trust. He took himself to have 'proposed a system for electronic transactions without relying on trust'.[2] For Nakamoto, blockchain makes trust unnecessary.

But other people have seen things differently. They've stressed blockchain's ability, not to get rid of trust, but to create it. On its website *Blockchain: The New Technology of Trust*, the investment bank Goldman Sachs writes that blockchain 'provides a simple, secure way to establish trust for

[1] *The Economist*, 'Nailing It', Vol. 428, Iss. 9107, (Sept. 1, 2018): S10-S11.
[2] Nakamoto (2008, 8).

virtually any kind of transaction'.[3] And the *New York Times* columnist Andrew Ross Sorkin writes:

> The blockchain is ultimately about solving society's ultimate challenge: trust. Or rather, lack of trust. Blockchain is about using technology to create a shared sense of trust by a group of disparate participants.[4]

It's fair to say, then, that blockchain's relationship to trust is less than perfectly clear.

Still, I think there *is* a strong connection between the two. In what follows, I argue that blockchain is a 'trust transformer'. It turns a kind of generalized trust in a community as a whole into a particularized trust in a specific database. But different applications of blockchain generate different levels of trust in their databases. Some applications only generate trust in the *integrity* of the database (§1). Other applications generate trust not just in the database's integrity but in its *accuracy* too (§2). It's this second group of applications that I think deserves the bulk of our attention in the future (§3).

1. INTEGRITY

Before we can understand why blockchain is a trust transformer, we have to understand what blockchain itself is.

It is sometimes described as a quintessential digital-age technology, inseparable from the internet and impossible without bleeding-edge cryptographic algorithms. But in fact the technology is not a distinctively digital one. It is just a method, or system, for preserving data. The system allows a community to collectively create and maintain a database or 'ledger' that no individual member of the community has the power to alter. Such a system *can* be implemented digitally, but it can also be implemented in other ways. To get a clear understanding of how the system works, we'll keep computers and cryptography out of the picture at the beginning.

We'll start by looking at a non-blockchain system for preserving data. This system is simple but requires a lot of trust from its users. By modifying the system so that it no longer requires this level of trust, we'll arrive at a blockchain system.

So: imagine a community of book collectors who acquire and trade rare books. One's status in the community depends on how impressive one's collection is. And so each member of this community wants to be able to prove to everyone else which books they have. How can they do that?

Here's one way. Any time someone in the community wants to prove they have a given book, they can go door-to-door showing it off to everyone. But let's suppose that these books are so old and so fragile that they can't be exposed to light very often. The community needs another way.

---

[3]Available at https://www.goldmansachs.com/insights/pages/blockchain/.

[4]'Demystifying the Blockchain' by Andrew Ross Sorkin. Available at https://www.nytimes.com/2018/06/27/business/dealbook/blockchain-technology.html.

So let's suppose the community instead creates a *ledger* to record which books each of them has. Let's say they hold a special, one-time-only conference. At the conference, they come to consensus on who has which books, and they write that down in the ledger. Then they entrust the ledger to an *administrator* for safekeeping.

But these antiquarians don't just sit on their collections. They trade with each other. And so they want to keep the ledger up to date; they want it to reflect the current state of everyone's collection. Here's how they can do that. Let's suppose Alice wants to transfer a certain rare book—a Gutenberg Bible, say—to Bob. When the transfer occurs, Alice writes on a slip of paper 'Alice transfers the Gutenberg Bible to Bob'. Then she signs it (in an unforgeable way) and submits it to the administrator of the ledger.

Now the administrator checks the ledger to make sure it satisfies one of two conditions. First, the ledger says that Alice had the Gutenberg at the one-time-only conference and *doesn't* contain a statement saying that she transferred it to someone else. Or, second, the ledger contains a statement saying that someone transferred the Gutenberg to Alice and no *later* statement saying that she transferred it to someone else. If Alice's statement meets either of those conditions, the administrator deems it *valid* and adds it to the ledger. Otherwise it's deemed *invalid*.

If all goes well, this system will keep the ledger up to date. But it requires the community to place a huge amount of trust in the ledger administrator. What if the administrator overlooks one of the statements that is submitted to her? Or what if she misjudges one as valid when it is really invalid?

These shortcomings can be mitigated by making the ledger, and all submitted statements, publicly viewable. If everyone can see the ledger and all the statements that have been submitted, then each person can verify for herself that the administrator has been careful and thorough. Each person can verify, that is, that all and only the valid submitted statements have been added to the ledger.

Still, this system is open to abuse. Suppose that the administrator of the ledger is Alice herself. And suppose that, after transferring the Gutenberg Bible to Bob, Alice decides to boost the status of her friend Carol by making it appear that *she* has the Gutenberg. She can write down on a slip of paper 'Alice transfers the Gutenberg Bible to Carol', sign it, and submit it to the ledger administrator—that is, to herself. In the ledger, Alice can then *replace* the earlier statement, the one that says she transfers the Gutenberg to Bob, with this new fraudulent statement.

If Alice does this devious deed, the average member of the community won't be able to tell. Of course, anyone can inspect the ledger, and if they do that they will see that the submitted Alice-Bob statement was not included in it. But this will appear entirely appropriate, since by the standards of the fraudulently altered ledger, the Alice-Bob statement is invalid. After all, the altered ledger says that Alice already transferred the Gutenberg to Carol!

Because of its potential for abuse, it makes sense for the community to adopt this system only if they really trust the ledger administrator. Now

there might be someone in the community widely thought to be so upstanding as to be beyond suspicion, in which case the system would be fine. But there also might not be anyone like that. If the community is large enough, there might not be anyone who is even *known* to everyone else, let alone trusted by them. So it's worth considering alternatives.

If the community wants to avoid trusting any *one* person to properly administer the ledger, they can make it not only publicly viewable, but publicly administered. Instead of leaving the sole copy of the ledger in one person's hands, the community can distribute copies of it to everyone. And when someone wants to submit a new transfer statement, they can just send copies of it to everyone in the community. Under this *distributed* system, everyone is expected to check each new submission against their own ledger to judge whether it is valid or invalid. If a submission is valid it goes in; if not, not.

But inevitably, not everyone will be a perfect administrator of their own ledger. Someone will lose a statement that was sent to them, or they'll judge a statement to be valid when it is really invalid, and so on. And this will lead to problems.

Suppose Bob wants to impress another antiquarian, David, by proving to him that he has the Gutenberg Bible. Short of displaying the book itself, what can he do? He can ask David to consult his ledger, but what if David lost, or never even received, the statement that says that Alice transferred the Gutenberg to Bob? Then his ledger won't show that Bob has it. Of course, Bob could show David his own ledger, but David has no reason to trust that Bob has carefully, or honestly, administered it. For all David knows, Bob transferred the Gutenberg to someone else without ever including the corresponding statement in his ledger.

Another thing Bob could try is to appeal to the community at large. He could ask everyone in the community to show David their ledgers. Of course, there might be some other delinquents who, like David, haven't properly administered their ledgers. So maybe not *every* ledger will show that Bob possesses the Gutenberg. Still, Bob might think, the *majority* of ledgers will show this. Surely, he might argue, most members of the community are responsible, or responsible enough, and so whatever the majority of ledgers show will reflect whatever transfer statements have been circulated.

But even if most members of the community are responsible, that does not mean that the majority of ledgers will reflect the statements that have been circulated. After all, the community might contain malicious actors like Alice. Suppose that, like before, Alice alters her ledger to include a fraudulent statement to the effect that she transferred the Gutenberg to Carol. She might then create lots of copies of this fraudulent ledger—so many copies that the majority of ledgers in the community end up being fraudulent.

Her scheme could be blocked *if* there were a way to enforce a 'one person, one ledger' rule. But this might not be possible, especially if the community interacts largely online. After all, Alice could just create lots of

accounts, each of which claims to be an independent member of the community whose ledger just happens to agree with Alice's own.

In a way, this system is even worse than the previous one. The earlier system required the community to trust one person to administer the ledger. This new system no longer requires *that* kind of trust, but the community now has to trust everyone not to run a scheme like Alice's, something that was impossible under the old system.

But there is a way to remove the need for this trust. We start with the distributed ledger system we just described. But now we require every ledger to satisfy two additional conditions.

(1) If any statement in a ledger is altered, that automatically destroys all the statements that come after it.
(2) Adding a statement to a ledger takes a long time.

These two conditions are the key to blockchain.

Don't worry right now about how these conditions are to be enforced. First just suppose they hold, so that we can see what the point of them is. Imagine that Alice and Bob have identical ledgers consisting of 100 transfer statements. Let's suppose that statement number 50 says that Alice transferred the Gutenberg Bible to Bob. And now suppose that, like before, Alice alters her ledger by replacing this statement with a fraudulent statement that says that she transferred the Gutenberg to Carol.

By condition (1), this action automatically destroys statements 51 through 100 in her ledger. So now Alice's fraudulent ledger is half the size of Bob's legitimate one. This disparity in size is a smoking gun: it shows that Alice has altered her ledger. If not for her devious deed, her ledger would have been just as large as Bob's.

Condition (2) now makes the smoking gun hard to dispose of. Suppose Alice tries to hide the evidence of her fraud by enlarging her ledger so that it is once again the same size as Bob's. To do this she'll somehow have to add 50 statements to her ledger. But by condition (2), this will take a long time. And during this time, Bob will be receiving more statements from other members of the community and will be including the valid ones in his ledger. So his ledger will continue to grow beyond its current stock of 100 statements. Of course, condition (2) means that adding statements will be slow for him too. But he has a head start on Alice. It will be hard for her to catch up.

What this shows is that anyone who tampers with their ledger is likely to end up with a smaller ledger than those who were honest with their ledger. And *that* means that it is likely that the largest ledger in the community has *not* been tampered with. Let's put this in a punchy way by saying that it is likely that the largest ledger in the community will have *integrity*. This is the core insight behind the blockchain method.

Now let's return to the question of how these two conditions could actually be enforced. Take condition (1) first. If statement number 50 is altered, we don't need to literally destroy statements 51 through 100. It would be enough to somehow force there to be a clear mismatch between statement

50 and the later ones. Then it would be obvious to everyone, or everyone who looked, that the ledger had been tampered with. The only way to avoid suspicion would be to remove statements 51 through 100 from the ledger. So altering statement 50 would—effectively, if not literally—destroy statements 51 through 100.

Here is a simple way to achieve this. Whenever a statement is added to the ledger, just include in that statement an exact copy of every previous statement. So statement 2 contains a copy of statement 1, statement 3 contains a copy of statements 1 and 2, and so on. Statements 51 through 100, then, will each contain a copy of statement 50. So if the *original* statement 50 is altered, the mismatch with the later statements will be obvious to everyone who looks. (Note that this simple method uses a huge amount of storage space, and so for that reason it's not employed in practice. Actual blockchains employ sophisticated cryptographic techniques to achieve the same end with a minimum of storage space.)

Now take condition (2). How can we make it so that adding a statement to the ledger takes a long time? Well, we just have to require that when you add a statement, you include with that statement something that takes a long time to produce. This thing could in principle take a number of forms. It could be a ship in a bottle, for instance, or an original pen-and-ink illustration, or a sweater knitted by hand. But if we want to implement our blockchain system digitally, we'll need something digital. Here is, in essence, what actual blockchains do.

A *one-way function* $f(X)$ is a certain kind of mathematical function. Given X, it is easy to calculate $f(X)$. But given $f(X)$, it is very hard to work backward and find X. A good example is multiplication, if we require X to be a set of prime numbers. It's easy to find the product of several prime numbers, but it's much harder to find a number's prime factors.

Now suppose we have some particular way of digitally (and so mathematically) encoding ledger statements. We can then define a one-way function f that takes as input a ledger statement together with an integer called a *nonce*. The function yields as output another integer. And when someone wants to add a statement to the ledger, we require them to find a nonce that satisfies the following condition. The function f, applied to the new ledger statement together with the nonce, must yield as output an integer lying in a relatively small range—say, between one thousand and one million. How can you find a nonce that works? It's not so easy. Since f is a one-way function, you can't just work backward from the range of output you want in order to find a nonce that will give you an integer in that range. The only way to find it is by trial and error, and even with a fast computer, this takes a long time. (But once a working nonce is found, it is easy for anyone to verify that it works.)

This nonce—the one that yields the right value for f—is just what we need in order to enforce condition (2). Like a ship in a bottle, this is a 'product' that takes a long time to create. But unlike the ship, it takes a

digital form. A ledger will be above suspicion only if it includes a working nonce for each of its statements.

That's how a community can enforce conditions (1) and (2). And if they do that, then as we saw earlier, that makes it likely that the largest ledger that exists in the community will have integrity.

The community can make this even more likely by working together. Let's suppose that when a new valid transfer statement *S* is circulated throughout the community, everyone begins working on adding *S* to their ledger. Now assuming the community is implementing the blockchain method digitally, this requires finding a working nonce. But let's suppose that when someone manages to find one, she circulates her updated ledger, with the newly added statement *S* and its nonce, throughout the community. And let's suppose that the other members of the community, if they see that her ledger is larger than theirs, simply adopt hers as their own. (Or at least, they adopt hers as their own after checking to make sure all her nonces work and all her statements match, as conditions (1) and (2) require.) If they do this, then they can stop working on finding a nonce for *S*, since it's already been found, and start working on finding one for the *next* circulated transfer statement.

Let's call the members of the community who work together in this way the *honest* members. Because of their teamwork, the honest members' ledgers will grow faster than Alice's fraudulent ledger. It will be hard for her to overtake them.

Her only possible strategy will be to ape the teamwork of the honest members. In just the way that the honest members work together to find working nonces for new ledger statements, Alice will have to rely on the help of some accomplices to generate working nonces for her fraudulent statements. And Alice's syndicate will have to do this faster than the honest members. But if most members of the community are honest, then it will be very hard for the syndicate, even working together, to outpace them.

One thing the syndicate might try is to buy a very powerful computer, one that can test many, many nonces each second. In fact, since the members of the syndicate, by working together, are in effect pooling their computing power, they might *each* try to buy powerful computers. These acquisitions would help them grow their fraudulent ledger faster. But the honest members are also pooling their computing power. If most members of the community are honest, then it will be very hard for the syndicate to acquire more computing power than all of the honest members put together.

So by working together, the community can protect itself against malicious actors like Alice. But this kind of teamwork also brings a further benefit: it increases the extent to which any two ledgers agree. Since the honest members of the community are constantly circulating ledgers among one another, their ledgers will tend to agree, or very nearly agree. The largest ledger, then, will not only have integrity, it will also be widely shared.

We will call this method, or system, the *blockchain system*. The largest ledger in such a system we will call the *blockchain*. We've so far described

the blockchain system as a system for preserving data about rare book holdings. But in principle, this system could be used to preserve data about lots of things. We'll consider some other possibilities in the rest of the paper.

We're now able to understand part of blockchain's relationship to trust. A certain level of trust is a prerequisite for the blockchain system. After all, the largest ledger or blockchain is guaranteed to have integrity only if most members of the community are honest. You don't have to trust anyone in *particular* to be responsible or to avoid attempting fraud. But you do have to trust that the community as a whole is on the level.

But although the blockchain system *requires* trust, there is also a way in which it *creates* trust. If most members of the community are honest, then the blockchain is guaranteed (or almost guaranteed) to have integrity. You can trust that it hasn't been tampered with.

So the blockchain system is a kind of trust transformer. It takes a generalized trust in the community as a whole and transforms it into a specific trust in the integrity of a particular blockchain. As long as there is trust in the community as a whole, the system guarantees the existence of a blockchain whose integrity can be trusted.

2. ACCURACY

The fact that the integrity of the blockchain can be trusted only goes so far. Just because the blockchain has *integrity* does not mean it is *accurate*.

To see why, think again about the rare book community. Let's suppose Alice circulates the statement 'Alice transfers the Gutenberg Bible to Bob'. And let's suppose this statement is judged valid and ends up being included in the blockchain.

Now imagine that years pass. Thousands of other books change hands. The blockchain grows. Assuming the community is mostly honest (I won't bother to make this assumption explicit from now on) the blockchain will have integrity: it cannot be tampered with. So even years later, we can be sure that it will still contain the statement 'Alice transfers the Gutenberg Bible to Bob'.

But for all that, it might be that Alice in fact never physically transferred the Gutenberg to Bob. She just circulated a statement saying she did. The book might still be sitting, carefully preserved, in Alice's storage facility. If that's the case, then the blockchain contains an inaccurate statement. It represents Bob as possessing the Gutenberg when in fact he doesn't. Nothing in the blockchain method prevents this kind of misrepresentation. Although the method guarantees the integrity of the blockchain, it does nothing to guarantee its accuracy.

This point might seem obvious, but it's often overlooked. Why is that? I suspect it's because the original and best-known application of the blockchain system is one in which inaccuracy is impossible, or nearly impossible. This is the application to cryptocurrency. But to see why inaccuracy is impossible there, and where else it might be impossible, we need to consider another case first.

2.1. **Vaulted gold.** Imagine a community that uses gold coins as currency. Let's keep things simple by imagining that every coin has the same weight and purity and that every coin has a different serial number.

As many people who have thought about these issues have pointed out, this currency system requires the community to, in a sense that I won't try to spell out here, *recognize* these gold coins as valuable, just like Americans recognize dollars as valuable and Mexicans recognize pesos as valuable.[5]

There's another thing that fewer people have pointed out. Not only does the community have to recognize that the coins are valuable, it also has to recognize what counts as *holding* a coin. It has to provide an answer to the following question: what do you have to do in order to count as having some money?

A natural answer for the community to give is that someone holds a coin if, and only if, they physically possess it. You could have a community, for instance, in which everyone carries around a coin purse for everyday transactions. When a transaction occurs, the buyer takes some of the coins in her purse and gives them to the seller, who places them in her own purse and hands over the purchase.

But this isn't the only way of holding a coin. Imagine that the community hates having to always carry coins around. Suppose they hold a meeting to designate a trusted member of the community as their *banker*. They have the banker write down each person's name and the serial numbers of the coins in her purse. Then they store everyone's coins in an underground vault. Once the coins are in the vault, there is no longer anyone who physically possesses any of them: everyone's purse is empty. But the community could still recognize people as holding various coins on the grounds that the banker's book lists the serial numbers of various coins next to their names. Under this new system, a transaction no longer involves physically moving coins from the buyer's purse to the seller's. Instead, the buyer tells the banker to cross out certain serial numbers next to her name and write them next to the seller's name instead.

This example illustrates a concept that's going to be crucial for understanding what makes a blockchain accurate. Focus on the relationship between these two conditions:

(1) You hold a certain gold coin.
(2) That coin's serial number appears next to your name in the banker's book.

Under this vaulted-gold currency system, these two conditions will always go together. If you hold a coin, then its serial number will be next to your name, and vice versa. And it's not an accident that they always go together: they *have* to go together. That's in part because (1) just *consists in* (2). All there is *to* holding a coin is having its number next to your name in the

[5]See Searle (1995) and Passinsky (2020) for two attempts to say what this kind of 'recognition' amounts to in more precise terms.

banker's book. You hold that coin *because* its number is next to your name.[6] Put another way, that inscription in the banker's book is what makes it true that you hold that coin. Let's put this in a punchy way by saying that the banker's book *dictates* the facts about which coins you hold.

Now let's see how this sheds light on blockchain. Let's consider a variant of the banker's-book system where the book is formatted more like the blockchain ledger of §1. Suppose that at the initial meeting, the banker writes down sentences like 'Alice initially holds #1234' to record the fact that Alice physically possessed the coin with serial number 1234. Then the coins are all vaulted away. After that, if Alice wants to transfer one of her coins, say coin 9876, to Bob, she writes down 'Alice transfers #9876 to Bob' on a slip of paper and give this to the banker. The banker checks this submission against her book to see whether (i) the book has the entry 'Alice initially holds #9876' and does not have an entry like 'Alice transfers #9876 to *X*', or (ii) the book has an entry like '*X* transfers #9876 to Alice' and has no later entry like 'Alice transfers #9876 to *Y*'. If one of these conditions is satisfied, then the transfer statement is said to be *valid* and the banker adds it to her book.

This currency system requires a lot of trust in the banker, but this can be avoided by using a blockchain system instead. Instead of having the banker administer the ledger, each member of the community can administer their own ledger, just like in §1. When Alice wants to transfer a coin to Bob, she just circulates a transfer statement to the community as a whole. If the statement is valid, it will be added to the largest ledger. And the community can recognize Alice as holding a given coin if, and only if, the largest ledger—that is, the blockchain—satisfies either condition (i) or condition (ii) with respect to that coin.

We saw in the case of the banker's-book currency system that the banker's book dictates the facts about who holds which coins. In the case of the blockchain currency system, it is the blockchain that dictates these facts. You hold a given coin just because the blockchain says you satisfy condition (i) or (ii) with respect to that coin.

The blockchain currency system has a remarkable feature: when a statement is added to the blockchain, it is guaranteed to be accurate. Suppose, for example, that the statement 'Alice transfers #9876 to Bob' is added to the blockchain. Given the definition of a valid statement, it must then be that, just before the statement is added, the blockchain satisfies either condition (i) or condition (ii). But that's exactly what it takes for Alice to hold coin 9876. And just after it is added, the blockchain will have the entry 'Alice transfers #9876 to Bob' and no later entry like 'Bob transfers #9876 to *Y*'. But that's enough to make it true that Bob holds coin 9876. So just before the statement is added, Alice holds the coin. And just after the statement is

[6]Philosophers call this a relationship of ground. The fact that the serial number *n* appears next to your name (together with some facts about the community's recognition of what counts as holding a coin) *grounds* the fact that you hold coin *n*. For more on ground see Raven (2015).

added, Bob holds the coin. So Alice does indeed transfer the coin to Bob. The statement is accurate!

The underlying explanation for this accuracy is that the blockchain dictates the facts about who holds which coins. That's why we can be sure that just before the statement 'Alice transfers #9876 to Bob' is added, Alice holds the coin, and that just after the statement is added, Bob holds the coin.

So far we've argued that a statement is always accurate at the time it is added to the blockchain.[7] If the blockchain could be altered or tampered with later, of course, it might not continue to contain only accurate statements. But what we saw in §1 is that this can't happen. The blockchain is guaranteed to have integrity: it cannot be tampered with. So even after some time has passed, the blockchain is guaranteed to contain only accurate statements. Let's say that a blockchain is *accurate* when it satisfies this condition.

2.2. **Cryptocurrency.** So: when we apply a blockchain system to keep track of who holds which coins in the underground vault, the resulting blockchain is accurate. We can now show that cryptocurrency blockchains are also accurate.

To see why, start by imagining that one day the vault becomes totally inaccessible. It collapses, or a landslide blocks the entrance, or whatever. If the community is using the blockchain system from §2.1, this development wouldn't affect them in any way whatsoever. They could just carry on using the coins in daily transactions by updating the blockchain to reflect each new transfer from buyer to seller. The physical inaccessibility of the coins wouldn't matter at all.

Now we're assuming each coin has a unique serial number. (To keep things simple, let's suppose the serial numbers range from 1 to 1,000,000.) So here's something that might happen—maybe gradually over a long time, maybe all at once in an explicit decision. The community might come to recognize these *numbers*—the integers from 1 to 1,000,000—as valuable instead of the coins.

If the community changes what they recognize as valuable in this way, then they'll also have to change how they understand the entries in the blockchain. When the blockchain says 'Alice transfers #9876 to Bob', they can no longer understand that to mean that Alice transfers the coin whose serial number is 9876. Instead, they have to understand it to mean that Alice transfers the number 9876 itself. Under this new system, people hold numbers, not coins. Alice holds a number, such as the number 9876, if either (i) the blockchain has the entry 'Alice initially holds #9876' and does not have an entry like 'Alice transfers 9876 to *X*', or (ii) the book has an entry like '*X* transfers #9876 to Alice' and has no later entry like 'Alice transfers #9876 to *Y*'.

[7]The blockchain contains not only transfer statements but also statements about who *initially* holds various coins. But these are also guaranteed to be accurate at the time they are added. After all, at that time, there are no other statements in the ledger, so condition (i) is automatically satisfied. And this is enough to make these statements true.

This number-based system is in one way very different from the earlier coin-based system. Now the currency—the 'locus of value'—is numbers, not coins. But it's clear that the systems are in another way very similar. They have the same structure; they work the same way. Since it's clearly possible for a community to adopt the coin-based currency system, it's possible for them to adopt the number-based system too.

What this shows is that it's possible to have a currency system in which the currency itself is not a material thing, but rather a kind of abstract object. In this case, the abstract objects in question are certain numbers. But there are also currency systems that use other kinds of 'abstracta'. Here's an example. Think of the earlier coin-based system, but suppose each coin bears, not a unique serial number, but a unique hieroglyph-like shape. In principle, a community could use a blockchain method to keep track of who holds each coin. And they could transition to a system in which they recognize as valuable, not the coins, but the hieroglyphs themselves.

We're now ready to talk about cryptocurrency. Like the number or hieroglyph based systems, cryptocurrency systems are abstracta-based currency systems—a little more complicated than our earlier examples, but essentially the same.

Which abstracta are recognized as valuable under a cryptocurrency system? It depends on the cryptocurrency. The best-known cryptocurrency, bitcoin, uses as its currency certain parts of its own blockchain. The parts in question are chains of statements. Simplifying things a bit, suppose we create a new bitcoin blockchain. And suppose statement number 1 in that blockchain is 'Alice initially holds the chain beginning with statement 1'.[8] This statement is also a chain of statements—a chain that is one statement long. Alice, if she wants, can transfer this chain to Bob by adding to the blockchain the statement 'Alice transfers the chain beginning with statement 1 to Bob'. Let's suppose some other statements have been added in the meantime, so that the Alice–Bob statement is statement number 5. Now our chain of statements has grown. It used to consist just of statement number 1; now it consists of statement number 1 together with statement number 5. Bob, if he wants, can transfer this chain to Carol by adding to the blockchain the statement 'Bob transfers the chain beginning with statement 1 to Carol'. This addition grows the chain yet again, and now Carol is the one who can transfer it to someone else.

These chains are called unspent transaction outputs, or UTXOs. They are the currency in the bitcoin system. The person who *holds* a UTXO is just the recipient of the last transfer in the chain. In our little example, for instance, Carol holds the chain beginning with statement 1. Clearly the bitcoin blockchain dictates the facts about who holds which UTXOs.

The community of bitcoin users recognizes such UTXOs as valuable. And indeed, it's natural to say that bitcoins simply *are* UTXOs. A bitcoin,

[8]Strictly speaking, statements in the bitcoin blockchain involve account numbers rather than users' names. This fact is important for the anonymity of bitcoin, but it can be ignored here.

that is, is nothing other than a chain of transaction statements beginning with a statement of the form '*X* initially holds the chain beginning with statement *n*'.[9]

There is one aspect of the bitcoin system that is a little strange. Although I don't think it indicates any real problem, it is good to be explicit about it just to avoid confusion. When you pay for something with bitcoin, or with any UTXO-based cryptocurrency, what you transfer to the seller is a chain of transaction statements. We usually think of a transaction statement as something that is *about* a transfer and not as the thing that is itself transferred. But in a UTXO-based cryptocurrency it is both.

Although many cryptocurrency systems use UTXOs as their currencies, not all of them do. In the *ethereum* cryptocurrency system—again simplying things—the blockchain contains a sequence of balance 'snapshots'. Each snapshot contains a statement of how much 'ether' everyone holds. If Alice circulates the signed statement 'Alice transfers one ether to Bob', then a new snapshot will be added to the blockchain that decreases Alice's ether balance by one and increases Bob's ether balance by one.

This balance-snapshot framework avoids the strangeness of the UTXO framework. Instead of transferring a chain of transaction statements like bitcoin users do, ethereum users just transfer some ether. The balance snapshots themselves are never transferred; they just keep track of how much ether everyone holds.

But this raises the question of what *does* get transferred. The answer, we said, is ether—but what is that?

The most natural answer, I think, is that ether consists of abstract 'tokens'. We could compare a transfer of ether to a move in correspondence chess.[10] Suppose that, in a game of correspondence chess, White sends the message 'Be5' to Black. By sending that message, White moves her bishop to square e5. But what is it that actually moves? Nothing physical. It's correspondence chess: there's no physical chessboard, no physical bishop. Instead, what 'moves' is an abstract piece, on an abstract chessboard—one that both Black and White know the state of. That's how I think we should look at ether transfers. When Alice transfers one ether to Bob, she transfers an abstract token. It's these tokens that the community of ethereum users recognize as valuable. And holding a certain quantity of these tokens is just a matter of the latest balance snapshot saying you do. The ethereum blockchain, then, dictates the facts about how much ether everyone holds.

Like in the case of the gold-coin blockchain we discussed before, the statements contained in these cryptocurrency blockchains are guaranteed to be accurate. This is easiest to see for the case of balance-snapshot currencies like ethereum. Each snapshot states how much ether everyone holds at a certain time. But when a snapshot is added to the blockchain, the facts about how much ether everyone holds at that time are also dictated by the

[9] I'm ignoring the complications that arise from the existence of UTXOs of different denominations and from the possibility of splitting and combining UTXOs.

[10] Compare the discussion in Smith (2008).

blockchain. So each snapshot is guaranteed to be accurate at the time it is added. And since the ethereum blockchain, like all blockchains, is guaranteed to have integrity, this means that it will always contain only accurate statements.

The same goes for the somewhat more complicated case of UTXO currencies like bitcoin. The bitcoin blockchain contains statements saying that a certain UTXO, or chain of statements, is transferred from person *A* to person *B*. In order for a statement like this to be added to the blockchain in the first place, *A* has to hold the UTXO in question. And once the statement is added, the UTXO expands to include that transfer as the last link in the chain. Since *B* is the recipient of this transfer, now *B* becomes the person who holds the UTXO. So *A* really does transfer the UTXO to *B* and so the statement is accurate at the time it is added to the blockchain.[11] Integrity then ensures that the blockchain will always contain only accurate statements.

We pointed out earlier that, in a cryptocurrency system, the currency itself is an abstract object. And now we've just seen that cryptocurrency blockchains are accurate. So these are examples of abstracta-based currency systems whose blockchains are also accurate. But these aren't the only kinds of currency systems whose blockchains are accurate. After all, in §2.1 we saw that a currency system based on physical gold coins could also have an accurate blockchain.

Still, there is a way in which abstractness promotes accuracy: abstracta-based currency systems avoid a certain *threat* to accuracy that arises for physical currency systems.

To understand the threat, think again about the vaulted-gold currency system from §2.1. Imagine someone breaches the underground vault and emerges with a bunch of coins. Let's suppose they start going to shops and trying to spend them. It's reasonable to think that at least some shopkeepers will accept the physical coins as payment. But if so, then these shopkeepers do not really recognize someone as holding a given coin if, and only if, the blockchain says they do. Their actions reveal that they also recognize the person standing in front of them, handing them a coin, as holding that coin. And so the blockchain does not really dictate the facts about who holds which coins, and so the statements it contains are not guaranteed to be accurate.

Nothing like this is possible in the case of abstracta-based currency systems. There is no physical manifestation of the currency that might compete with or undermine the 'authority' of the blockchain. And so people will be more likely to take the blockchain of an abstracta-based currency to be the final word on who holds the currency. What alternative is there?

[11] The bitcoin blockchain also contains statements like '*A* initially holds the chain beginning with statement *n*'. At the time this statement is added to the blockchain, it forms a UTXO that is one statement long. Let's suppose that the community recognizes a UTXO of this form as being held by *A*, so that this statement too is accurate at the time it is added to the blockchain.

2.3. **Where accuracy comes from.** We've now seen several examples of accurate blockchains. Cryptocurrency blockchains are among these. Let's now take a step back and ask: in general, what is it about the accurate blockchains that *makes* them accurate?

The examples we've looked at show that accuracy flows from two features.

(1) The blockchain dictates the very facts that its statements are about.
(2) The blockchain has integrity.

The first feature ensures that, when a statement is added to the blockchain, it is accurate at the time it is added. For example, the statements of the gold coin blockchain are about who holds which coins, and the blockchain also dictates the facts about who holds which coins. The second feature ensures that the blockchain can't be tampered with later, and so it will always contain only accurate statements.

These two features of accurate blockchains are not completely independent of each other. The first feature depends on what the community recognizes. But the community's recognition depends in turn on the second feature. The gold coin blockchain, for instance, dictates the facts about who holds which coins precisely *because* the community recognizes someone as holding a certain coin if the blockchain says they do. But it only makes sense for the community to do this if the blockchain has integrity. Who in their right mind would be willing to take the blockchain to be the final word on their assets if it could easily be tampered with?

We've also seen some examples of non-accurate blockchains—blockchains whose statements are not guaranteed to be accurate. The rare book blockchain is one of these. Non-accurate blockchains, like all blockchains, have integrity. So they have the second of the two features listed above. But they lack the first. The rare book blockchain, for example, cannot be tampered with, but its statements are about who has which books. And the facts about who has which books aren't dictated by any blockchain. You can't make a book have a certain location by updating a database, no matter how sophisticated that database might be.

So what accounts for the difference between the accurate blockchains and the non-accurate blockchains? Not integrity; all blockchains have that. What accounts for the difference is that only the accurate blockchains dictate the facts that their statements are about.

We're now in a position to say more about blockchain's relationship to trust. We saw in §1 that the blockchain system is a trust transformer. It takes as 'input' a certain generalized trust in the community as a whole, and creates as 'output' trust in the integrity of a particular blockchain. We can now see that, when a blockchain has the first feature listed above, it creates further trust. It creates trust, not only in the integrity of the blockchain, but in its accuracy too.

3. APPLICATIONS

Call an application of blockchain technology *accurate* if it would involve an accurate blockchain; otherwise call it *non-accurate*. I think that as we continue to explore this technology, we should focus our efforts on accurate applications.

To see why this focus makes sense, let's first think about *non-accurate* applications. Like the rare book blockchain of §1, these are applications whose blockchains don't dictate the facts that their statements are about. A lot of the applications are being explored right now are non-accurate, including applications to supply chain management,[12] to health records,[13] and to certificates of birth, marriage, and death.[14] No blockchain can dictate the facts about where a certain widget is or what medical condition someone has or when someone died.

There is some reason for general skepticism about the value of non-accurate applications. In a non-accurate application, the blockchain is a tamper-proof database—yet there is no guarantee that the statements in the blockchain were accurate upon their creation. Now implementing a blockchain system requires a lot of hard work, or at least a lot of computational resources. So it only makes sense to do it if you really need the kind of tamper-proof database that it provides. And so it only makes sense to do it if whatever data you are trying to store is likely to be subject to tampering. But then wouldn't those same forces be likely to distort the data upon its creation? If so, then the blockchain is likely to contain many inaccurate statements. It won't be a reliable database.

There might be some specialized non-accurate applications that avoid this problem. Imagine a lottery organization that sells numbered tickets and randomly picks a certain number as the winner. Say they sell tickets numbered 1 to 1,000,000. Now if you buy one of these tickets, you might worry that, even if you win, you won't get paid. Suppose you buy ticket number 500,000, and that ends up being the winning number. The organization could just claim that they never actually sold ticket number 500,000. Of course, if you had a physical ticket you could show it to them to prove your purchase, but what if you bought the ticket online? You could produce a screenshot, but the organization could claim you doctored it.

A non-accurate blockchain might be useful here. After all, after the lottery draw has occurred and the winning number is known, there will be many people with a huge incentive to tamper with any existing records of who bought which tickets. But before the lottery draw, at the time of those records' creation, there is no reason to falsify or distort them. So it might make good sense for the community of lottery players to use a blockchain to record which tickets each of them bought. After the lottery draw, the winner

[12]https://www.ibm.com/industries/retail-consumer-products/supply-chain.

[13]https://www.forbes.com/sites/robertpearl/2018/04/10/blockchain-bitcoin-ehr/#17c7f1c79e77.

[14]https://www.newscientist.com/article/mg23531454-500-blockchaininspired-project-means-you-are-who-you-say-you-are/.

can point to the blockchain as good evidence that she really did purchase the winning ticket. It's not conclusive evidence, of course, since nothing guarantees that the blockchain record was accurate upon its creation. But why *wouldn't* it be accurate? At the time of its creation there would have been no reason to claim to have purchased that ticket number in particular. And the integrity of the blockchain guarantees that if the record was accurate then, it remains so after the winner has been chosen.

Although this case is somewhat interesting, it is pretty specialized. It seems likely that many, even most, non-accurate applications of blockchain will face the general skeptical worry raised above. What's more, a lot of non-accurate applications have the potential to cause significant harm. The harm comes from the difficulty of error correction. Since there's no guarantee that the statements of a non-accurate blockchain will be accurate, in all likelihood some of them will not be. But then how do you correct these inaccuracies? As we've seen, it is very hard to alter the statements in the blockchain. So the inaccuracies will tend to persist. Ironically, what might have seemed to be a virtue of blockchain technology—the creation of a tamper-proof database—turns out, in non-accurate applications, to be a vice as well.

More promising, I think, are the accurate applications, the applications whose blockchains dictate the facts that their statements are about. Cryptocurrency is the best-known accurate application, but it's not the only one. Another very interesting accurate application concerns voting.

Just like any community that uses currency has to recognize what counts as valuable, any community that holds elections will have to recognize what counts as a vote. There are lots of options available. For example, a community could recognize Alice as voting for Bob if she writes 'Bob' on a slip of paper, or if she punches a chad from a card next to the name 'Bob', or if she prints out a certain barcode from a certain computer, and so on. Here's another option. A community could recognize Alice as voting for Bob if, and only if, she signs a statement saying 'Alice votes for Bob' and adds it to a special blockchain.[15]

Let's suppose a community recognizes votes in this way. Then this blockchain will dictate the facts about who casts which votes. The fact that the blockchain contains the signed statement 'Alice votes for Bob' will make it true that Alice votes for Bob. And so this blockchain will be an accurate one.

In the United States, each state has broad leeway to administer elections however it wants. Because of this state-level control, it might be relatively easy for the US to implement a blockchain-based voting system in at least some jurisdictions. Indeed, in the 2018 elections, West Virginia experimented with letting overseas voters cast absentee votes by contributing to a

[15]I'm ignoring complications associated with secret ballots.

blockchain.[16] And the ongoing pandemic provides strong public health reasons to develop methods for voting at a distance, not only in elections but in legislative bodies as well.

Certain other accurate applications are also interesting, but will be more difficult to implement. Think about property titles, for instance. What counts as owning a plot of land? In principle, a community could recognize Alice as owning plot *P* if a certain blockchain contains a statement of the form '*X* transfers *P* to Alice' and no later statement of the form 'Alice transfers *P* to *Y*'. If the community took the blockchain to be the final word on who owns what land, then this application of blockchain would be accurate.

But in practice, this blockchain system would probably be unworkable. Land ownership is constrained by centuries of law and custom. Because of this tradition, it's hard to believe that a community would take the blockchain to be the final word on who owns what land. In the case of a conflict between the blockchain and the tradition, tradition would surely win. But that means that the blockchain would not really dictate the facts about who owns what land, and so it would not be accurate.

A similar example is given by copyright and other intellectual property. A community could try to set up a blockchain to record statements about the transfer of various intellectual property rights. But given the long tradition of intellectual property law, it's hard to believe a community would take the blockchain to be the final word on its members' intellectual property. And so this blockchain would not be accurate.

4. Conclusion

Blockchain is an exciting new technology. Although in its application to cryptocurrency it has already had some measure of success, it is still not completely clear what other applications it might have. In this paper, I've suggested that as we continue to explore this technology, we should focus our efforts on what I've called *accurate* applications. These are the applications that best exploit blockchain's nature as a trust transformer, something capable of turning generalized trust in a community as a whole into trust in the integrity—and in the best cases accuracy—of a particular blockchain.[17]

References

Nakamoto, S. 2008. Bitcoin: A peer-to-peer electronic cash system.

Passinsky, A. 2020. Social objects, response-dependence, and realism. *Journal of the American Philosophical Association 6*(4): 431–443.

Raven, M. J. 2015. Ground. *Philosophy Compass 10*(5): 322–333.

Searle, J. R. 1995. *The Construction of Social Reality*. New York: Free Press.

[16] https://sos.wv.gov/elections/Pages/MobileVote.aspx.

[17] I am grateful to Lewis Cohen, Gary Miller, Bradley Rettler, and Craig Warmke for helpful comments and for discussion of these issues.

Smith, B. 2008. Searle and de Soto: The new ontology of the social world. In *The Mystery of Capital and the Construction of Social Reality*, 35–51. Chicago: Open Court.

Email address: martin.hemenway.glazier@uni-hamburg.de
URL: www.mglazier.net