What is Bitcoin?*

Craig Warmke

Abstract

Many want to know what bitcoin is and how it works. But bitcoin is as complex as it is controversial, and relatively few have the technical background to understand it. In this paper, I offer an accessible on-ramp for understanding bitcoin in the form of a model. My model reveals both what bitcoin is and how it works. More specifically, it reveals that bitcoin is a fictional substance in a massively coauthored story on a network that automates and distributes jobs normally entrusted to centralized publishing institutions. My model therefore falsifies a popular view according to which each bitcoin is a chunk of code.

1 Introduction

During the Great Recession, a person or persons under the pseudonym 'Satoshi Nakamoto' created bitcoin, a new form of digital money. In an announcement for bitcoin from February 2009, Satoshi [2009a] writes:

> The root problem with conventional currency is all the trust that's required to make it work. The central bank must be trusted not to debase the currency, but the history of fiat currencies is full of breaches of that trust. Banks must be trusted to hold our money and transfer it electronically, but they lend it out in waves of credit bubbles with barely a fraction in reserve.

Satoshi had seen how the banks partially responsible for the recession nevertheless received government bailouts. And he offered bitcoin as an antidote to these unfair systems in which institutions receive public funds to avoid the consequences of their irresponsible behavior.[1]

Now, a little over a decade later, governments worldwide have kicked their money printers into high gear to stave off a pandemic-induced depression. And, once again, they have bailed out companies with questionable

*For comments and discussion, thanks to audiences at Texas A&M (2019 Conference of the Society for Philosophy and Technology) and Pittsburgh (2019 Midwest Annual Workshop in Metaphysics), as well as Wassim Alsindi, Andrew Bailey, Nathan Ballantyne, Nic Carter, James Chiang, Judith Crane, Billy Dunaway, Alicia Finch, Rhys Lindmark, Matthew McKeever, Santiago Meijia, Martin Peterson, Bradley Rettler, Noël Saenz, Edmund Schuster, Erica Shumener, Kevin Vallier, Brandon Warmke, Vi Wysong, and students in my Fall 2018 seminar on idealism.

[1] I use the singular masculine pronoun since Satoshi chose a traditionally male name.

business practices. Through both fiscal and monetary policy, governments and central banks continue to debase our currencies and socialize the losses of undeserving corporations.

Although Satoshi has since disappeared without a trace, his creation remains. In addition to being highly divisible, easily portable, and, under best practices, effectively unseizable, bitcoin is also censorship-resistant because its peer-to-peer network transfers value without the intermediaries that close accounts, block transactions, and engage in financial surveillance.[2] Bitcoin has no CEO or central bank. No corporations with skilled lobbyists can situate themselves near its monetary spigot to reap unjust rewards. Nor can anyone effect a high rate of inflation, a practice that disproportionately helps the rich.[3] Instead, its permissionless system issues new bitcoin fairly to those who help secure the network along a fixed, disinflationary issuance schedule through the year 2140.

As governments and corporations begin to roll out their own digital currencies, bitcoin will remain an important alternative, especially to any such currencies used as tools of financial surveillance or those subject to negative interest rates.[4] Despite bitcoin's popularity and increasing importance, few understand how bitcoin works. This is unsurprising since it has many moving and mathematically complex parts with obscure labels.[5] Some need little more than a place to put bitcoin on their cognitive maps. But as bitcoin's profile rises in response to global events, many seek the answers to two questions, in particular:

Philosophical Question. What sort of thing is bitcoin?

Functional Question. How does bitcoin work?

Many questions about bitcoin belong to the domain of cryptography or computer science or economics. Bitcoin experts wield considerable knowledge in these fields and their answers to these questions illuminate bitcoin's fundamentals. But the Philosophical Question belongs to neither cryptography nor computer science nor economics nor to a triumvirate of all three. Although answering the question requires some knowledge of these fields, the question is essentially a philosophical one. So we also need the disciplinary tools of philosophy to reveal the answer.[6] Since we have yet to

[2] Cryptocurrencies are censorship-resistant to varying degrees. For a sustained discussion of cryptocurrencies generally, and their relationship to philosophy, politics, and economics, see Bailey et al..

[3] See Easterly and Fischer [1999], Li et al. [2002], and Albanesi [2007, 1090-1093].

[4] In a recent survey, the Bank of International Settlements [2020, 3] found that 80% of respondent central banks are "engaged in some sort of work" on developing their own digital currencies. For recent discussion on these developments, see Kiff et al. [2020], Lannquist et al. [2020], Auer and Böhme [2020], Meaning et al. [2018], and Davoodalhosseini [2018].

[5] Sometimes, descriptions of those parts obscure or stretch the truth. See Walch [2016, 2017].

[6] 'What is bitcoin?' is what Nathan Ballantyne [2019, 6] calls a *hybridized question*, one best "addressed and answered by combining evidence and techniques from two or more fields."

marshal these tools in this way, we have not yet fully understood what bitcoin truly is.[7]

Satoshi once opined that "writing a description for this [bitcoin] thing for general audiences is bloody hard. There's nothing to relate it to."[8] Ten years later, bitcoin still suffers from an open exposition problem—we lack a totally perspicuous explanation of what it is and how it works.[9] Such an explanation remains elusive partly because we've never marshaled the philosophical tools required to explain bitcoin. But we need more than the tools alone. We also need to use those tools within a particular method of explanation.

Why do we also need a special method of explanation? Even with the required tools, bitcoin's close relationship with its components makes it difficult to gain much traction on the Philosophical and Functional Questions. Since bitcoin's components help make it what it is, understanding bitcoin is difficult without understanding how it works. This line of thought suggests that we address the Functional Question first: maybe once we understand the components, we will understand bitcoin. Unfortunately, the jargon for the components obscures how they make bitcoin what it is.

So perhaps we should begin with the Philosophical Question. Indeed, I'm convinced that without an understanding of what bitcoin is in the first place, we lack the proper framework for understanding what exactly its components do. This line of thought suggests that we should seek to understand bitcoin first before we try to understand its complex machinery, contrary to our initial line of thought. So it looks like we're in a pickle: answering either the Functional or the Philosophical Question seems to require an answer to the other. We could easily spin our wheels approaching these questions one at a time.

We can gain traction, however, by approaching the Philosophical and Functional Questions more or less simultaneously. We can use what Michael Weisberg [2007, 209] calls a *modeling strategy*, "the indirect theoretical investigation of a real world phenomenon using a model." By modeling, we can come to understand a phenomenon without investigating it directly.[10] Us-

[7] Though some philosophers have written about bitcoin, none have done so primarily as an exercise in metaphysics, standardly conceived, as I do here. For example, Reijers and Coeckelbergh [2018] consider how blockchain technologies might shape our social world and frame a technology's societal effects as affecting this or that narrative in a wider ontology of narratives. An examination of the possible interpersonal and societal effects of a technology has great value. But it brings us no closer to understanding what bitcoin is in itself. Bjerg [2016] and Velasco [2017] also engage in different projects.

[8] Nakamoto [2010].

[9] I borrow this notion of an open exposition problem from Chow [2009]. Thanks to Matthew McKeever for helping me frame the issue in this way.

[10] Along similar lines, Swan and De Filippi [2017, 607] invite philosophers to use "conceptual metaphors" to help others understand the complicated nature of blockchain technology. In this vein, they briefly cast bitcoin as the Napster of money. However, Napster had a CEO and used centralized servers to pair file-sharers. Thus, it was exactly the kind of vulnerable intermediary with a central authority that bitcoin avoids by design. When Satoshi [2009b] announced the first release of the bitcoin software to a cryptography mailing list on January 8, 2009, he says that the system is "completely

ing Weisberg's account of modeling as a guide, I will construct a model, analyze its properties, and then show that many of those same properties apply to bitcoin. With simpler components and more descriptive labels, my model serves as an accessible on-ramp for understanding both what bitcoin is, metaphysically, and how it ultimately works.[11] The model involves a computer network designed to protect against internal abuse and external attack and on which many coauthors continuously write a simple fictional story.[12]

Although the model serves as an accessible on-ramp for understanding bitcoin, I don't solve bitcoin's open exposition problem. I'm writing, first of all, for other philosophers. I make controversial claims which require argument, and the arguments themselves require some philosophical background to assess. But I'm not writing for philosophers alone. I provide resources that may someday help others solve the open exposition problem. I also hope, perhaps in vain, that my exposition is perspicuous enough that those new to bitcoin will gain deep levels of understanding and appreciation for it, even if certain parts of the exposition prove obscure or difficult.

Let's begin with a preview. In the next section, I briefly explain the modeling strategy. In Section 3, I motivate aspects of my model by looking at an infamous case of coauthorship. In Sections 4 and 5, I describe and analyze the model. Then, in Section 6, I compare the model to the bitcoin network. The comparison reveals that bitcoin is a fictional substance in an on-going story that traverses a massively coauthored digital book otherwise known as the blockchain.

2 Modeling

Modeling both simplifies the study of a complex phenomenon and helps its practitioners sidestep false but compelling views about it.[13] The case of the modeling mathematician, Vito Volterra, exemplifies both benefits. During World World I, although fishing slowed in the Adriatic Sea, post-war populations of sharks and other predators increased while post-war populations of cod and other prey decreased.[14] The situation puzzled many. Instead of formulating and testing hypotheses directly, Volterra constructed and analyzed a simple model involving two populations, predator and prey, with certain stipulated mathematical properties. Volterra's mathematical model predicted that heavy fishing favored the prey and light fishing favored the predator, contrary to what many expected. Because Volterra's model and the aquatic life in the Adriatic were relevantly similar, he drew the surprisingly correct conclusion that pulling fewer cod from the sea led the

decentralized with no server or central authority."

[11] As I'll explain later, good models simplify aspects of the target phenomena. So we shouldn't expect to find every important aspect of the target phenomena in the model itself. My model serves primarily to answer the Philosophical and Functional Questions to the neglect of some other questions.

[12] For earlier approaches along similar lines, see Sztorc [2014] and McKeever [2018].

[13] See Godfrey-Smith [2006a], Paul [2012, 14], Weisberg [2007, 208].

[14] Here, I rely on the details in Weisberg [2007].

sea to have fewer cod and that a return to pre-war fishing levels would bring back the pre-war proportions of prey and predator.

In Weisberg [2007, 2013], Volterra's case illustrates the three main stages of modeling:

> **Stage 1**: **Construction**. The theorist does not investigate the target phenomenon directly but first constructs a model with certain stipulated properties.[15]
>
> **Stage 2**: **Analysis**. The theorist studies the model itself to determine what is true of it.[16]
>
> **Stage 3**: **Assessment**. The theorist assesses whether the model is sufficiently similar to the target phenomenon. If it is, the theorist maps aspects of the model onto the target phenomenon to reveal that some truths about the model are true about the target phenomenon.[17]

Thus, modelers can gain understanding of a phenomenon indirectly by investigating a simpler but relevantly similar system. But a word of caution: the very simplicity of a model that aids discovery also arises from a loss of information by abstraction. By design, successful models don't capture all aspects of their target phenomena. And my model also achieves simplicity through abstraction. When we later assess my model, I will highlight several ways in which it simplifies in order to answer the Philosophical and Functional Questions.

My target phenomenon is bitcoin, not fish populations in the Adriatic Sea. But bitcoin is equally ripe for modeling. First, instead of investigating bitcoin and its mathematically complex machinery, we can examine a simpler and highly idealized model. Second, by examining this simpler model, we will sidestep the initially plausible but wrong view that bitcoin is code.[18] Overall, then, we can use a model to bootstrap into a working knowledge of bitcoin and thereby enjoy both the ease of theft and goods of honest toil.[19]

I will devote a section in the paper to each stage, from Sections 4 through 6. My model is an imagined network of coauthors who write an on-going fictional story, chapter by chapter, about passing bread from basket to basket. Although I will stipulate many features of the model, we have good reason to stipulate these features beyond the fact that I obviously have an eye on using them to model bitcoin. For the network is designed to mitigate

---

15 Weisberg [2007, 209, 222-224].

16 Weisberg [2007, 209, 222-224].

17 Weisberg [2007, 209-10, 224-226].

18 I say more about the view that bitcoin is code in Sections 6 and 7.

19 One might object that, as a scientific tool, modeling is ill-suited for philosophy, in general, and metaphysics, in particular. Paul [2012], Godfrey-Smith [2006b], and Williamson [2017] would disagree. I find their positions plausible, but the adequacy of my modeling strategy here does not hang on whether they are right. I will use what Weisberg [2013, 19-25] might call a *hypothetical concrete model*, a concrete structure that I describe but never actually build. (Compare Giere [1988].) I will argue that, if my model were built, something in it would fall under a certain metaphysical category. Then, since that thing would be sufficiently similar to something in the target phenomenon, the thing in the target phenomenon also falls under the same category.

various challenges with coauthorship, peer review, and publishing. Before I present the model, let's review these challenges.

3 Coauthorship Problems

Once, someone invited Alexandre Dumas to coauthor. He reportedly replied, "Why should I wish to quarrel with you?"[20] Dumas knew all too well the challenges of coauthorship. To start, potential coauthors begin with no clear protocol for who does what. So they must first divvy up tasks, including the task of divvying up tasks. Now, as coauthors, we want to assign tasks fairly, but we also want to maximize the chances for success. And fairness and success sometimes pull in opposite directions, especially in our messy world of egos, feelings, and differing standards for both fairness and success.

Yet any number of reasons might prevent coauthors from completing tasks, no matter how carefully we distribute them. Even with stringent safeguards and a clear governance model, widening a project's network of coauthors would seem to increase the likelihood of disagreement, the involvement of incompetent or malicious actors, and, as a result, the likelihood of failure itself.[21] As you can imagine, then, mass coauthorship—or *hyperauthorship*, as it is called[22]—presents incredible challenges. We will soon design a system for hyperauthorship, a method for coauthoring on a massive scale. This system will serve as our model for understanding bitcoin. To further aid our design, we will now look at a case of coauthorship gone awry.

In 1800, the second edition of *Lyrical Ballads* appeared with some questionable changes from the first edition two years prior. In the first edition, neither William Wordsworth's nor Samuel Taylor Coleridge's name appeared on the cover, and the book opened with Coleridge's "The Rime of the Ancyent Marinere." Soon after, in a letter to the publisher dated June 24th, 1799, Wordsworth opines that Coleridge's opening poem had been "an injury to the volume."[23] In the second edition a year later, Wordsworth excised another one of Coleridge's poems, renamed and relegated the mariner poem to the penultimate spot, and described that poem in the endnotes as having "great defects."[24] Wordsworth kept the rest of Coleridge's poems simply for "variety" and attributed them generically to a "friend." The cover bore Wordsworth's name alone. This episode provides a few lessons—besides the obvious lesson that one should never collaborate with

[20] Matthews [1890, 169].

[21] Bozeman et al. [2015], Parker and Kingori [2016]. Relatedly, linguist Geoffrey Pullman [2016] has over 100 coauthored publications, enough to appreciate the difficulties of coauthoring without exaggerating them. He conjectures that the "worst-case difficulty of completing an academic work increases in proportion to D^n, where D is the degree of difficulty it would have had anyway and n is the number of authors."

[22] Cronin [2001].

[23] Barker [2002, 60].

[24] This comment appears on an unnumbered page in the end notes of Wordsworth and Coleridge [1800].

Wordsworth. The lessons will help us design a system for hyperauthorship less vulnerable to Wordsworthian malfeasance.

If poor Coleridge had issues with a single coauthor, how will we survive with hundreds or even hundreds of thousands? That many more people dramatically increases the likelihood of debilitating disagreement and the involvement of people who may harm the project. Since we need a low barrier of entry to permit hordes of authors, we can't use a high barrier of entry to prevent any Wordsworths from contributing. How, then, will we protect the project's integrity if almost anyone at all can contribute, including many of whom we may not trust?

To grease the skids, we might elect some central authority to delegate tasks and oversee their completion. But doing so would not secure the project unequivocally. Trusting central authorities injects new risk factors into the equation. Despite the advantages that central authorities often bring, centralized power is centralized vulnerability. The link in a network on which everything hangs is an attack vector vulnerable to both internal and external forces. A central power may fall prey to jealousy, natural disaster, theft, ignorance, laziness, bribery, nepotism, sexism, extended vacation, the spam folder, and assaults both digital and biological, among other things.

Suppose we reject centralized governance models to avoid these risks. Then how will thousands of coauthors coordinate effectively? Who decides whether any particular sentence makes it into the story? By vote? Well, then whose votes matter? A small body of special voters? If so, how would they have been chosen in the first place? Or does everyone vote? And, either way, which method would we use: plurality or majority rule, ranked-choice, quadratic voting, or some other?[25] On top of all this, how do we prevent voting fraud? Overall, then, how can arbitrarily many authors agree on the story's content without a central authority, especially when authors lack decisive reason to trust each other? We'll call this the *Coordination Problem.*

The Coordination Problem concerns the creation of our story. But once we've agreed on a chapter's final form, how do we secure its continued integrity? Wordsworth has taught us that if anyone has enough power over the project, even in a later edition, we risk surprise deletions, rearrangements, and petty endnotes. Since, by stipulation, our hyperauthorship system will continue its story indefinitely, one chapter at a time, our story will have no later editions. Yet we still want previous chapters to sit more or less immutably in their originally published state, even if some Wordsworth among us tries to tamper with them. And we want to protect the integrity of the project not only from untrustworthy coauthors but also from incompetent coauthors, hackers, and technological mishaps.

We don't want Fat-finger Freddy or Harry the Hacker inserting typos or deleting past chapters. We also don't want Microsoft AutoUpdate or a software bug in Dropbox to corrupt the file that encodes our story. To complicate matters, if we don't have any central authorities, we won't rely on any centralized service like Dropbox to store our official copy of the story anyway. But, then, in what sense can we have an "official copy" at

[25] For discusson of voting methods, see Pacuit [2019].

all, if it doesn't sit in a special place, like a queen on her throne? Here we encounter our second problem: how can we preserve the integrity of past chapters, and how can we make sense of the notion of integrity at all, without relying on centralized authorities to store an official copy? We'll call this the *Corruption Problem*.

Finally, and with just a single coauthor, Coleridge did not get the credit he deserved in the second edition. If the second edition had been the only edition, Coleridge would have had some difficulty later proving that he was, in fact, the "friend" that had authored some of the poems. We hope to assign credit properly for many thousands.[26] And two further considerations complicate matters. First, for a variety of reasons, authors sometimes don't want credit and prefer to operate anonymously or pseudonymously. Second, sometimes authors don't want credit initially but do later. Here we have our third, and final, problem for hyperauthorship: how will we attribute credit where it's due even when thousands contribute, without assigning unwanted credit, in a way that can be assigned reliably to any contributor, even posthumously, and, again, without relying on central authorities? We'll call this the *Credit Problem*.

Building a system for hyperauthorship which effectively resolves the Coordination, Corruption, and Credit problems is no small feat. But the components which enable such a system appeared across various academic literatures by the 80s and 90s.[27] We just need to bundle them together in the right way and explain how the resulting bundle resolves those problems. I do this in the next two sections.

4 Construction

In my model, coauthors write an on-going fictional story on a computer network using a special protocol. A *computer network* (or "network" for short) is a collection of computers linked to one another through communication channels. These channels allow computers on the network to exchange data. A *network protocol* is a system of rules encoded in software for the exchange of data among network participants. Our model's protocol governs a network of coauthors (writers), referees (those who ensure written submissions pass muster), and publishers (those who disseminate refereed submissions). The protocol ensures that submissions which meet certain criteria get published, and those which violate any rule remain unpublished. As I will explain shortly, the network participants running the protocol automate stages in publication typically entrusted to third parties like Reviewer #2 and central authorities like Elsevier.

[26] Attributing credit properly has posed a challenge for increasingly common hyperauthored academic papers. See Ioannidis et al. [2018]. But I don't mean to suggest that there aren't already effective solutions for assigning credit in cases of hyperauthorship. For discussion of the notion of credit in Wikipedia, see Forte and Bruckman [2005]. For discussion on the problems that arise in Wikipedia's governance model, see Kostakis [2010]. Importantly, Wikipedia's governance model permits the kinds of pockets of power that our system is designed to avoid.

[27] Narayanan and Clark [2017].

In replacing human referees and publishers with automated refereeing and publishing, we haven't simply exchanged one set of authorities for another with the same vulnerabilities. The protocol distributes the work normally entrusted to authorities in a way that protects from abuse and attack. So we spread our trust thinly over the whole network rather than let it pool in a few large honeypots. The protocol will also remain open-source: anyone can inspect its code to ensure that it doesn't benefit some at the expense of others. Hence, the method for thinly spreading our trust lies open for all to inspect and verify.

4.1 Bread

Our protocol governs the creation of a simple story in breadworld, an imaginary universe whose central character we will call *bread*. Bread is a fictional substance whose smallest unit is a *crumb*. One hundred million crumbs equals a *loaf*. A *quantity of bread*, as I use the term, is not just the measurement for some mass of bread, but the very mass of bread to which the measurement applies. The story also has a cast of baskets to give and receive quantities of bread.

Breadworld obeys two main rules:

Conservation. No basket gives more or less bread than it has.

Exclusivity. Two baskets never hold the same particular quantity of bread at once.[28]

The first rule implies that giving baskets empty their contents when they give any bread at all. But they can give up to and including their full amount right back to themselves. More importantly, Conservation ensures that baskets never give more than they themselves have received. So, unlike some 1^{st} century Palestinian baskets, baskets never increase the overall supply of bread when they give bread to others.

Now, Conservation does not imply Exclusivity. That basket's can't give more than what they have received doesn't mean that two baskets can't hold the very same quantity of bread at once. Similarly, the conservation of mass and energy doesn't prevent me from planting a single flag in different territories. The territories just have to overlap. Exclusivity, in effect, precludes baskets with "overlapping territories."

4.2 Submissions

The story follows bread's movements across a brigade of baskets. Every sentence in the story specifies one or more baskets to empty, one or more receiving baskets, and the amount each receives. But potential writers cannot write sentences willy-nilly and dump bread from any baskets they like. Each basket has a *password* without which one cannot write a publishable sentence about emptying the basket's contents. Each basket also has an

[28] This is not to say that baskets may never hold the same measured amount of bread. Two baskets may each have a loaf as long as they each have a different loaf.

address to serve as its unique and permanent name. Baskets, then, straddle the line between the real and the fictional: bread's fictional movements inside the story are anchored to password access outside the story.

Companies typically store their online users' account information along with their passwords. When you access an account to send an email, pay a bill, or buy a product, the company checks to see whether the supplied password matches the account's stored password. This kind of system is ripe for both internal abuse and external attack. First, companies can sell or leak users' personal information.[29] Second, since accounts require company permission, companies can problematically withhold or cancel that permission.[30] Third, account databases lure hackers who can gain access to millions of user accounts in one fell swoop.[31] Users have good reason to worry about privacy, censorship, and theft.

The breadworld network protects against these issues and doesn't store user passwords or any other personal data. So how can the network tell whether or not a writer has the appropriate password for a candidate sentence? The network uses two special *garblers*. Garblers garble: they take a message or string of symbols as input, and return a seemingly random string as output. They have two notable mathematical properties:

> **Consistency.** No input ever gives different outputs.
>
> **Irreversibility.** The best evidence suggests reverse-engineering from output to input is practically impossible.

To give a feel for how random a garbler's outputs appear, we've taken a quartet of words that bear a family resemblance and fed them to a popular garbler called SHA-256. Similar inputs give dissimilar outputs:

Figure 1. *The SHA-256 garbler takes similar inputs and gives wildly different outputs.*

[29] See Swire [1999] on financial privacy, in particular, and Solove [2004] on privacy more generally.

[30] See Aswad [2018] and Kesari et al. [2017] for recent discussion.

[31] Cheng et al. [2017].

With one extra letter, or even with the same letters differently arranged, this garbler returns wildly different results. Yet one and the same string of symbols always returns the same result. Despite this consistency, the garblers remain immune from attempts to reverse-engineer them. You might think that artificial intelligence could detect an emerging pattern in millions of input-output pairs and with that pattern predict the input of an arbitrary output. But the garblers within the breadworld protocol have withstood these attacks. As a result, finding the input from such a garbler's output would require a brute-force search by feeding the garbler an unimaginably large number of potential inputs until it generated the desired output. Even with our best technology, unless we were unbelievably lucky, such a brute-force search would take millions of years.

For many services, we choose a username or address and then generate a password independently. But, for breadworld, we begin with the password, and the *Password Garbler* generates its address. Because the Password Garbler is consistent, it never generates different addresses from a single password. Even so, we cannot reverse-engineer the Password Garbler from an address to its password. In summary, passwords generate addresses through the Password Garbler, and addresses appear in the story as names for baskets. But we cannot work backwards from an address in the story to its password.

Every published sentence in our book includes three ingredients:

(i) The addresses of one or more giving baskets.[32]

(ii) The addresses of one or more receiving baskets.

(iii) The amount(s), held by the basket(s) in (i), to be given to the basket(s) in (ii).

Each publishable sentence also bears a digital signature for each of the giving baskets. A digital signature provides proof that the writer has the password for a giving basket. Here, we meet our next garbler, the *Signer*. To produce a valid digital signature for a sentence, a writer privately feeds the Signer not only (ii) and (iii) but also the passwords for the giving baskets named in (i). In the simplest case of a single giving basket and a single receiving basket, the diagram below depicts the relation between a sentence's components (in the left column of yellow rectangles) and its signature:

[32] Those who already understand the details of bitcoin transactions may balk at my presentation here since I've chosen to use an account-based system rather than a UTXO-based system, like bitcoin's. But as Zahnentferner [2018] has argued, these systems are intertranslatable. So I've chosen to include the simpler and more familiar account-based system and will ease into bitcoin's UTXO-based system in Section 7.

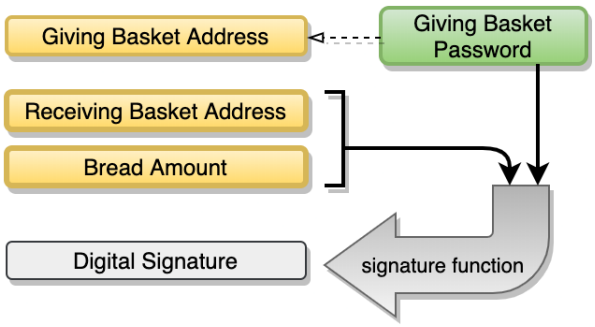

Figure 2. *How the Signer produces a digital signature.*

The signature then joins the sentence to form a *submission*, much like we see below:

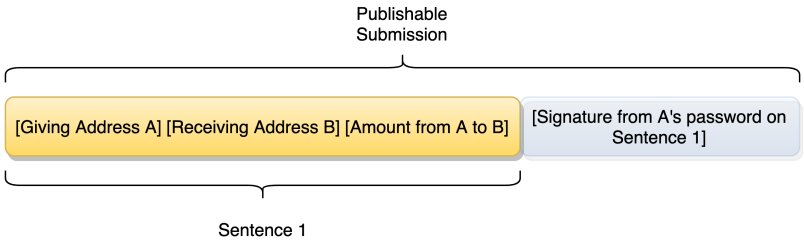

Figure 3. *A publishable submission with a valid signature.*

Like the Password Garbler, the Signer is both consistent and irreversible. It will return the same output from basket A's password and a candidate sentence every time. But we cannot use the candidate sentence and the output to compute A's password.

The Signer has one more feature, and it verges on the magical. The connection between between a basket's address and password enables a public verification of the signature via the *Notary*. Why is this verification so special? The Notary computes whether digital signatures have come from the appropriate passwords without ever seeing them. All the Notary needs is publicly available information: the sentence and the signature.[33]

The Password Garbler, the Signer, and the Notary together enable the breadworld network to block sentences written from those who lack the appropriate passwords without handling them. Since no central authority stores user accounts, the bread in a user's basket is as safe, in the story, as the user's password is secure, outside the story. Also, with no central authority to grant or deny access, anyone can acquire a password-address combo to submit a publishable sentence regardless of socioeconomic status, race, religion, age, gender, and so on. With internet access and a free, open-source breadworld writing app, one may generate a random password and

[33] The mathematics involves public-key cryptography, which enables someone to use a password to sign a message so that anyone, without the password, can verify whether signature came from the password.

its corresponding address for free. The basket corresponding to that pair will appear in the story once the network publishes a submission which says that the basket receives some bread.

In summary, each sentence includes the addresses of one or more giving baskets, the addresses of one or more receiving baskets, and the amounts of bread transferred. The sentence is publishable only if it comes with a signature from the password of each giving basket. But how does a submission transition from publishable to published? Next, we examine the stages leading to publication.

4.3 Referees

When someone submits a sentence to be published in the next chapter, the network propagates the submission to the *referees*. Each referee is a computer connected to the network running the breadworld protocol. The software encodes, among other things, standards for publishable submissions. Submissions that meet the standards are valid; otherwise, they are invalid. A referee's job is to judge whether submissions are valid or not.

Valid submissions meet two kinds of standards. First, they meet certain syntactical standards. Valid submissions specify one or more giving baskets, one or more receiving baskets, an amount of bread to give to each, and the correct signature via the Signer, all in the proper order with the appropriate symbols of the language. Second, valid submissions obey the fundamental rules of the breadworld universe. On this front, referees reject any submission that represents a basket as giving more bread than the story most recently says it has. Referees judge submissions as they pour in and forward valid submissions to the *publisher's queue* where they await publication.

4.4 Publishers

Like referees, publishers are computers on the network running the breadworld protocol. Publishers compete every ten minutes to compile sentences from the queue and publish them in the story's next chapter. We'll cover what chapters are first and then how publishers compete to publish them second.

Every chapter includes a title page along with a batch of submissions. The title page includes both a summary of its submissions and a pointer to the previous chapter. The summary is a garbled synopsis of the chapter's submissions from the *Summarizer*. And the pointer results from garbling the previous chapter's title page through the *Sequencer*. A chapter's pointer determines which chapter it should follow. Pointers therefore order chapters from the most recent all the way back to the first, which is embedded like stone in the breadworld software.

Publishers compete to solve a *Sudoku puzzle* to publish the next chapter.[34] To solve a Sudoku, one finds a string of numerals that, from top-left to

[34] I borrow the idea of using Sudoku puzzles as an explanatory strategy from Antonopoulos [2017, 26-27].

bottom-right, successfully fills in the Sudoku's blanks. Here's the catch. The title page has a slot for a random number, and the solution to the puzzle must come from the *Puzzle Garbler* after feeding it both a random number (to fill that slot) and the title page of the candidate chapter:

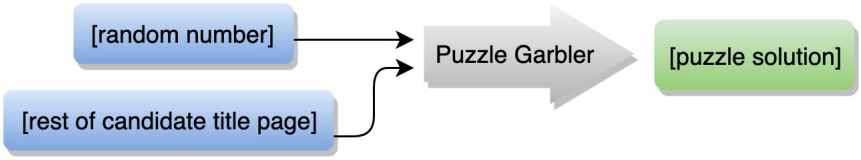

Figure 4. *The trial-and-error publishing puzzle.*

Since the Puzzle Garbler garbles unpredictably, the only effective strategy involves trying as many random numbers as quickly as possible until the Puzzle Garbler spews out a solution. When that happens, the publisher fills the slot in the title page with the winning random number.

Solutions are difficult to produce but easy to verify. A publisher who solves a Sudoku then broadcasts the candidate chapter and solution for all the referees to verify. Once verified, the referees add the winning publisher's version of the chapter to the end of their own individually stored version of the story:

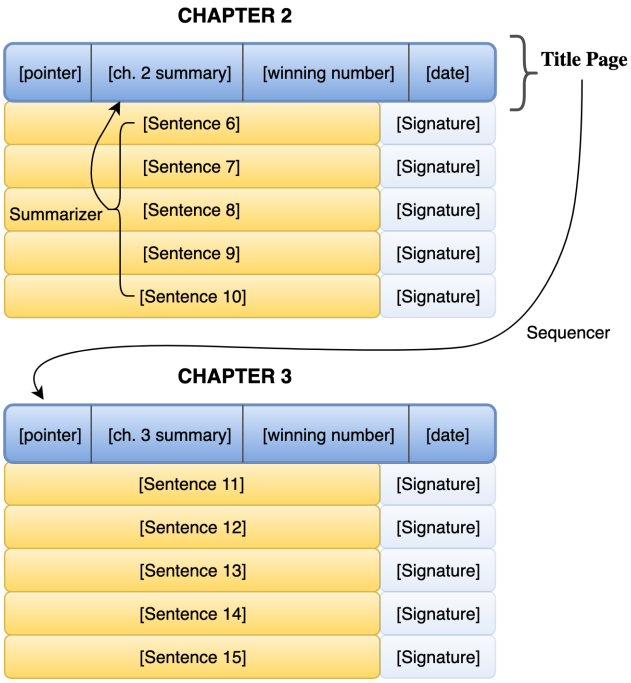

Figure 5. *Two published chapters, in order.*

Then the publishers begin to compete for the next chapter. If an offered solution fails, publishers will continue to compete until a broadcasted solution verifies.

Publishers compete to have their version of the chapter published because the rules allow each publisher to insert a sentence within its candidate chapter about receiving a fixed amount of bread in a basket whose password they have. So if a publisher's version of the chapter appears in the official story, so does a sentence which says that one of their baskets receives bread out of thin air, like digital manna from heaven. Publishers also have incentive to empty the queue. I've neglected to say that every sentence represents the giving basket as giving a little bread to a basket of the eventual puzzle-solving publisher. These are publication fees written into the story itself.

To gain a competitive advantage, some publishers may use more powerful processors to solve puzzles more quickly. But the network computes the average solution time every two weeks and the puzzle difficulty adjusts automatically. When, on average, solutions come much more quickly than ten minutes per puzzle, puzzle difficulty increases. Then, publishers must solve more difficult Sudokus for the next two weeks. Conversely, if average solution time increases substantially above ten minutes per puzzle, the difficulty decreases and publishers solve easier Sudokus for the next two weeks.

The chapter ordering afforded by the Sequencer enables referees to check whether the events described in a candidate chapter cohere with previous chapters. Suppose I submit two sentences in quick succession, one about emptying a basket of bread into your basket and a second about emptying it into another one of my own baskets. The story would violate CONSERVATION if it included both. Since referees keep track of which bread is in which baskets as they validate sentences, any referee which validates one sentence will reject the other.

Yet two publishers might solve puzzles at around the same time and then propagate competing chapters which disagree about whether my bread goes to your basket or my own. If this happens, referees keep both versions of the story going and eventually endorse the version with more accumulated *proof of work*—proof of having used computing power to produce puzzle solutions. The "proof" here resides in the solutions themselves. This is usually the version with more chapters since each published chapter requires its own puzzle solution. The referees then discard the version with less proof of work. In this way, the network achieves consensus about the story's contents without central authorities.

Because the network may take some time to reach consensus, the breadworld protocol also rejects sentences about sending bread won in any puzzle from the previous one hundred chapters. This prevents publishers from reaping rewards for orphaned chapters. However, the network doesn't completely orphan candidate chapters excluded for lack of proof of work. Previously validated sentences in failed chapters which remain consistent with the official storyline return to the publisher's queue for inclusion in the next chapter. The growing chain of agreed upon chapters stored individually by each referee constitutes the ever-expanding book of breadworld.

That's the system. To review, a network of computers running free, open-source software automates and decentralizes the refereeing and publi-

cation processes. Instead of trusting Referee #2, Quirky Editor, or Elsevier to behave responsibly, the network uses a transparent set of rules to validate individual submissions and forge consensus about the official storyline. With our model now complete, we may proceed to the next stage of modeling.

5 Analysis

According to Weisberg [2007, 209, 222-224], model analysis involves studying the model independently of its relation to the target phenomenon. Our analysis here comes in two parts. We will first evaluate how the breadworld network resolves the Coordination, Corruption, and Credit problems. Then, we will examine the ontology of breadworld.

5.1 The Coordination Problem

As you might recall from Section 2, the Coordination Problem concerns the effective coordination of arbitrarily many authors without a central authority. Breadworld authors can coordinate effectively because the stages leading to publication use an open-source protocol with simple rules about the form and content of written submissions on a decentralized and permissionless network.

First, because the protocol is open-source, authors have access to the rules governing every stage leading to publication and can judge the protocol's fairness. The protocol's simplicity enables authors to learn these rules relatively easily and submit sentences for publication accordingly.

Second, breadworld's network automates and distributes jobs we would otherwise reserve for humans in powerful positions. Automated referees verify submissions, and automated publishers compete to publish them. By automating these jobs, we greatly mitigate the vulnerabilities typically associated with human weakness. And by distributing jobs so that the network lacks a central clearinghouse, we greatly mitigate the risk of abuse or attack.

Third, the network is permissionless and so requires neither registration with nor permission from anyone to contribute. Hence, anyone with internet access can join as a coauthor or devote resources for refereeing and publishing. And yet the involvement of so many does not increase the likelihood that the project will fail. The protocol fairly adjudicates disagreement with its consensus mechanism and protects the project's integrity by rejecting submissions and candidate chapters which violate the protocol's rules.

5.2 The Corruption Problem

Whereas the Coordination Problem concerns the project's synchronic integrity, the Corruption Problem concerns its diachronic integrity. How does the network preserve the integrity of past chapters without using a central authority to store an official copy? Let's examine what happens when someone tries to alter a previously published chapter. Suppose we've

just published Chapter 10, and Harry the Hacker tries to alter Chapter 4 by editing, deleting, or adding a sentence. He can't do it by hacking into a central database because there is none. Each referee stores its own copy of the official story.

Yet hacking into one or even a few of these copies to change Chapter 4 won't work either. As you might recall from our discussion in the previous section, chapters are ordered linearly by their pointers. A chapter's pointer abbreviates data from the previous chapter, including the summary of all the previous chapter's breadworld sentences. So Harry's altered Chapter 4 will have a different summary, and, as a result, fail to provide the pointer already published as part of the version of Chapter 5 stored by all the other referees. Consequently, the network will reject Harry's Chapter 4 as the chapter prior to Chapter 5 and instead treat Harry's Chapter 4 as the beginning of an alternative storyline which, if it continues, will require an alternative Chapter 5, 6, 7, and so on.

By the protocol's rules, each referee "votes" for the version of the story with the most accumulated proof of work, the proof of having devoted computing power to solve puzzles. Harry's alternate storyline with fewer chapters has much less accumulated proof of work. So if Harry wants the network referees to endorse his version of Chapter 4, he will need to continue the alternate storyline until it has more accumulated proof of work than the original version, even as the original version continues to grow. This could require Harry to devote an enormous amount of energy because he would need to solve Sudokus on his own and at a faster clip than the rest of the network combined.

Harry's case illustrates that tampering with any past chapter requires re-writing and re-publishing all subsequent chapters. As long as enough publishers compete on the network, the computational power required to publish becomes cost prohibitive and makes chapters further back practically immutable. Thus, more recent chapters serve as digital sediment or amber whose gradual accumulation provides an increasingly protective layer for past chapters.

However, the book of breadworld is not completely secure. For instance, even if the network uses an enormous amount of energy, a malicious state-level actor could conceivably marshal enough energy to mount a successful attack. A different network with a different story could also draw publishers away from breadworld's own network and drastically weaken it, as a result. Finally, the breadworld network largely relies on the internet. Hence, the network inherits the internet's own vulnerabilities to cable-cutting, nuclear weapons, and electromagnetic pulse devices. So although the breadworld network effectively mitigates some vulnerabilities, it isn't completely without its own.

5.3 The Credit Problem

To resolve the Credit Problem, the network must attribute credit where it's due, without assigning credit where it's unwanted, in a way that can be assigned posthumously—all without relying on a central authority. Is this

possible?

Well, only someone with a basket's password can write a publishable sentence about giving bread from it. And that password generates a single address which serves as the basket's unique name. As a result, we can treat the address of a sentence's giving basket as a pseudonym for the sentence's author. Hence, each sentence with multiple giving baskets includes a pseudonym for each coauthor. No sentence goes uncredited, and no one's pseudonym gets credit it doesn't deserve.

Those authors who also want their credit transmitted to their public identities can prove at any time that they have the password associated with the pseudonymous address. To establish proof of authorship without giving away the password, an author can use the Signer (from Section 4.2) to sign an arbitrary message with the password. Then, with the message and the signature, anyone can use the Notary to verify whether the resulting signature arose from the correct password.

Authors also have some control when the verification happens, if ever. An author who wants credit for submissions under a certain pseudonym can sign an arbitrary message and post the message and the signature in a public place. Or if an author wants credit later, she can post that information later. But this can also be done within the story itself. I've not yet mentioned that each sentence has a tiny "memo" field in which one may write a small message. And this might be useful not only in cases in which one wants to prove who one is but also in cases in which one wants to prove who one is not, especially when someone publicly claims to be behind a pseudonym that, in fact, belongs to you. The network also allows someone to schedule a sentence far into the future. So someone may combine the schedulng and memo features to reveal an identity posthumously.

In summary, the breadworld network effectively solves the Coordination, Corruption, and Credit Problems. The system does have limitations, however. First, although the simplicity of the protocol allows many thousands of authors to coordinate, that same simplicity severely limits the story's arc. Second, although the proof of work system may prevent Harry the Hacker from tampering, the network may not withstand global catastrophe, attack from a state-level actor, or just a large-scale loss of interest. Third, the Password Garbler's irreversibility renders lost passwords practically unrecoverable. So users must hide passwords well enough but not so well that they lock themselves out forever.

5.4 Ontology

My model involves the creation of a massively coauthored story in which breadworld bread appears as the main character. Breadworld bread is a fictional substance, but recognizing it as such doesn't mire us in philosophical controversies surrounding fictional entities. Since fictional entities are so deeply controversial, though, I want to clarify what I do and don't mean when I say that breadworld bread is a fictional substance.

I don't mean to imply that breadworld bread is a fictional substance right now, on the same footing as Sherlock Holmes or Oliver Twist. Sherlock and

Oliver appear in actual fictional stories. Not so with breadworld bread. The book of breadworld has never been written and only appears in a model as the product of an imagined network. However, if we brought my model into reality and began writing the breadworld story, its main character would be a fictional substance.[35] When I say that breadworld bread is a fictional substance, I mean that breadworld bread is a fictional substance *in the model*.

Philosophers dispute both whether fictional entities exist or have being in some attenuated sense, and, if they do, what kinds of things they are. Following Kroon and Voltolini [2018] and the two-part division in Thomasson [1999], I'll call these the *ontological* and the *metaphysical* questions, respectively. To illustrate, I might answer the ontological question affirmatively and then answer the metaphysical question by identifying fictional entities like Sherlock Holmes with a certain kind of abstract object. In addition to these two questions, some philosophers also dispute whether certain things appearing in literary works qualify as fictional in the first place. I'll call this the *classificatory question*, and it is importantly different from the first two. For example, historical persons and places sometimes arguably appear in works of fiction, and some dispute whether the persons and places are the actual historical persons and places or merely fictional analogues of them.[36]

Yet few, if any, philosophers dispute whether Sherlock Holmes is fictional.[37] Sherlock is a prototypical fictional character in the minds of many. Even so, theorists may disagree on whether Sherlock, as a fictional character, has being or existence in some sense. Even though we refer to fictional characters and say true things about them, some theorists deny that Sherlock and other fictional characters exist or have any sort of being. These theorists' answer to the classificatory question sets 'em up ("*These* are the fictions"), and then their answer to the ontological question knocks 'em down ("Fictional entities don't exist").[38] That is, once we specify the do-

[35] There's an important difference between being a fictional entity and being a work of fiction. I claim that, in the model, bread is a fictional substance represented by a work of fiction. But since, outside my model, the digital book of breadworld corresponds to no real-world entity, some might also identify the book as a fictional entity, albeit a fictional entity that is itself a work of fiction. On the relation between modeled entities and fiction, see Godfrey-Smith [2006a] and Weisberg [2013, Ch. 4]. I take no stance in this debate. Some also endorse the argument in Kripke [1980, 24, 156–58] that fictional characters cannot be brought to life, as it were, no matter how closely some real-world entity corresponds to a character's profile in the fiction. I'm also neutral on this point. So I also don't mean to imply that if we made my model reality and began to coauthor a book about bread in the ways I've specified, then the resulting book would be *the* book of breadworld from the model. Nor do I mean to imply that the resulting book's main character would be *the* main character from the book described in the model. So, yes, there are philosophical controversies surrounding fictional entities and their connections to models, but my main claims here survive within a quite broad range of resolutions to those debates.

[36] Following Terence Parsons [1980, 57-59], we may frame the debate as being about whether the object appearing in the fiction is an *immigrant* from the real world or a *surrogate* for the corresponding real-world entity. See Motoarca [2014] for a recent examination of issues surrounding the debate.

[37] Some intellectuals play the so-called Sherlockian game and treat Sherlock and Watson as subjects of historical interest. For a founding text of this practice, see Knox [1920].

[38] See Kroon and Voltolini [2018, Sec. 2] for references and discussion.

main of fiction, we may then argue that the whole lot does or doesn't exist. We may also identify something as a fictional character but remain neutral about the more controversial ontological and metaphysical questions. Such is my stance here. When I say that, in the model, breadworld bread is a fictional substance, I simply mean to put it in the same category as Sherlock, whatever he is. I'm merely classifying, and I'm not making a controversial claim about the ontology or metaphysics of fictional entities.

We have good reason to classify breadworld bread and Sherlock similarly. Nothing external to the Conan Doyle stories has the properties that those stories ascribe to Sherlock. And nothing external to the breadworld story has the properties that the book of breadworld ascribes to bread. Breadworld loaves neatly divide into 100 million crumbs. Breadworld bread is tasteless and odorless, and has been passed across many digitally represented baskets in highly specific quantities in a certain order. We can't say the same of anything outside the book of breadworld. I don't mean to suggest that something couldn't be a fictional entity if something beyond the story in which it appears has the properties the story ascribes to it.[39] I'm merely suggesting that the reasons why most treat Sherlock as a fictional character apply equally well to breadworld bread.

Now, are there any reasons not to classify breadworld bread as a fictional substance? Could we plausibly argue that bread isn't a fictional substance because the breadworld coauthors aren't writing a *story*? I don't see how. Published breadworld sentences concern the movements of a quantifiable substance that doesn't exist outside the story and in a universe that operates in accordance with a few well-defined rules. The coauthors aren't lying, and they, too, understand that bread has no correlate in the world external to the story. The story may look boring to outsiders, but boring stories are stories, nonetheless. Besides, the initiated may enjoy the story's many interwoven subplots and exciting twists. Imagine the flurry of activity in the online breadworld message boards when a long-dormant basket holding the largest ever quantity of bread gives some to another basket without warning.[40] I've read many less exciting fictional stories, and any attempt to include them as works of fiction while excluding the breadworld story would likely have some implausible consequences.

Might we deny that bread is a fictional substance because the book of

[39] Kripke [2011, 59].

[40] Breadworld could plausibly achieve this level and intensity of participation, especially if amounts of bread were valued like a commodity, which is also plausible, given the development of a circular economy and a price floor initially set by the publishers. Questions about participation on such a scale raise the issue of *social scalability*, a notion defined in Szabo [2017]. Szabo writes:

> Social scalability is the ability of an institution—a relationship or shared endeavor, in which multiple people repeatedly participate, and featuring customs, rules, or other features which constrain or motivate participants' behaviors—to overcome shortcomings in human minds and in the motivating or constraining aspects of said institution that limit who or how many can successfully participate.

breadworld is still being written? Many hold that authors help create fictions.[41] If a fiction doesn't come to be until the creative process ceases or the story concludes, wouldn't breadworld bread fail to qualify as a fiction since the creative process continues indefinitely? Even if this were true, breadworld bread would at least be a merely intentional object in the unobjectionable sense: someone represents it in thought and nothing beyond the representation has the properties so represented.[42] It would also be in good company. For such a view arguably implies the same about, say, a slew of characters in the novels that Dickens published in serial form. Would we really want to deny that Mr. Grimwig was a fictional character until Dickens published *Oliver Twist* in its entirety, even though he appeared in monthly installments of *Bentley's Miscellany* up until that point? If not, we would have little reason to deny that breadworld bread is a fictional substance through the first installments of the breadworld story.

Whether fictional entities exist, or what they are if they do, is neither here nor there. If fictional entities exist, breadworld bread, as one of them, would exist. If fictional entities don't exist, breadworld bread, as one of them, wouldn't exist. No matter how we settle the more controversial debates about fiction, if we jumpstarted a breadworld network today, the main character in its digital book would be a fictional substance by any reasonable measure.

6 Assessment

Having completed the first two stages of modeling, we now begin the third stage—assessment. Here, we coordinate aspects of our model with aspects of the bitcoin network to assess whether the model is sufficiently similar to it. This coordination occurs according to what Weisberg [2007, 219-221] calls the *construal*, a description of what the modeler does and doesn't intend to model in the target phenomenon alongside the criteria for evaluating whether the model represents the target as intended.[43] I'll begin with a quick primer on bitcoin. As I go, I'll note some important differences with the model. Then I'll map aspects of the model onto the bitcoin network and argue that the more interesting things we've said about the model apply equally well to the bitcoin network.

6.1 The Bitcoin Network

On Halloween 2008, Satoshi Nakamoto posted a whitepaper online about how a peer-to-peer network could automate and distribute jobs normally entrusted to third parties like banks and credit card companies. He jumpstarted the network within months. And, several years later, it continues to facilitate transactions of its native digital asset and record those trans-

[41] Braun [2005], Goodman [2004], Kripke [2011], Salmon [2002], Searle [1975], Schiffer [1996, 2003], Voltolini [2006, 2015], Thomasson [1999], van Inwagen [1977].

[42] Compare Thomasson [1999, 89].

[43] Compare Godfrey-Smith [2006a, 733].

actions in a fully public ledger.[44] In what follows, I will examine how the bitcoin network both facilitates and records transactions.

6.1.1 Bitcoin

Unfortunately, 'bitcoin' is at least four-ways ambiguous in present usage. It can be used as a mass noun for a kind of stuff ("he has bitcoin"), a unit of measurement for that stuff ("his has 2.3 bitcoin"), a count noun ("he has two bitcoins"), and a name for the entire network, which is then sometimes capitalized. The smallest unit of bitcoin (in the mass noun sense) is a *satoshi*, in honor of bitcoin's pseudonymous inventor. 100 million satoshis equal a single bitcoin (in the count noun sense). I will usually use 'bitcoin' in the mass noun sense, and 'the bitcoin network' to refer to the network overall. Otherwise, the context should indicate whether I've used 'bitcoin' as a count noun or as a unit of measurement.

6.1.2 Transactions

The protocol governs the movements of amounts of bitcoin across *addresses* much like the breadworld protocol governs the movements of bread in its story across basket addresses. Each bitcoin address has both a private key and a public key. These keys work together in a cryptographic system called *public-key cryptography* that enables both message encryption and message authentication.[45] I can encrypt a message with your public key. But even if everyone knows your public key, if you alone have its private key, only you can decrypt the message. Separately, I can also use my own private key on a message to produce a digital signature so that anyone with only the message, signature, and my public key can verify whether I used the public key's paired private key to produce that signature. Bitcoin uses public-key cryptography for both message encryption and authentication but in a slightly more complicated way.

Bitcoin adds addresses to the usual mix of public-key cryptography's private and public keys. Here's how. Each private key generates a public key through what mathematicians call a one-way function. Since the function is one-way, we cannot practically run it in reverse to find a public key's paired private key. And since it is a mathematical function, the private key always generates the same public key. One-way functions are consistent and irreversible—they're garblers, in other words. The particular one-way function that uses the private key to generate its public key is elliptic curve multiplication (hereafter, *ECM*).[46]

The public key then generates an address through another one-way function called a *hash function*. The particular hash function that uses the public key to generate an address is actually a compound hash function, a function that garbles the public key and then takes the result and garbles it again. More specifically, the first function, SHA-256, garbles the public key, and

[44] Nakamoto [2008].

[45] For early overviews by two pioneers in cryptography, see Hellman [1978] and Diffie [1988].

[46] See Song [2019, Chapters 1-3] for an introduction to the use of elliptic curves in bitcoin.

then the second, RIPEMD160, garbles the result.[47] The diagram below captures the path of a private key to its address through a series of one-way functions:

Figure 6. *From a private key to an address.*

(For those with a technical background, let's pause here to address a philosophical issue about bitcoin addresses and their relation to the basket addresses in my model.[48] The "address" that emerges from RIPEMD160 appears in the blockchain in hexadecimal format. But we typically don't encounter addresses in hexadecimal. Addresses, as we typically encounter them, have been translated into the more concise and human-readable Base58 format. So if you look for your Base58-encoded bitcoin address within your raw transaction data in the blockchain, you won't find it. Reflecting on this observation, Andreas Antonopoulos [2017, 118-119] writes:

> Behind the scenes, an actual transaction looks very different from a transaction provided by a typical block explorer. In fact, most of the high-level constructs we see in the various bitcoin application user interfaces do not actually exist in the bitcoin system. ... In bitcoin, there are ... no addresses. [Among other things, addresses] are constructed at a higher level for the benefit of the user, to make things easier to understand.

If an address is just the Base58 string of characters, then Antonopoulos is right—addresses don't appear in the blockchain. But that's not the whole story. First, the hexidecimal strings that generate Base58 addresses appear in the blockchain's transaction outputs, the parts of transactions that specify which amount of bitcoin goes where. To spend the bitcoin from an output, one must have the private key that hashes into the output's hexadecimal string. So one could make the case that the hexadecimal strings serve as addresses, fundamentally speaking, and that the Base58 strings are merely addresses in a derivative sense. We use the Base58 strings outside the bitcoin blockchain to abbreviate the hexadecimal strings that actually function as addresses within the blockchain.

Second, both Base58 addresses and their hexadecimal counterparts name the very same mathematical object. Just as we have different names for the

[47] One can understand the notion of an address as either the hash of a public key (or a script hash, given *P2SH*, "pay-to-script-hash") or, more broadly, as the recipient in any transaction. In this latter sense, public keys once served frequently as addresses in the transaction format known as *P2PK* ("pay-to-public-key"). But even in Bitcoin 0.1.0, the software's first release, users could send bitcoin to the hashes of public keys in the transaction format known as *P2PKH* ("pay-to-public-key-hash). P2PKH gained prominence due to the extra layers of security provided by SHA-256 and RIPEMD160.

[48] Thanks to Matthew McKeever for raising this issue.

number 31 whether we write in decimal ('31'), hexadecimal ('1F'), binary ('1111'), or Base58 ('Y'), the number that goes by a certain Base58 bitcoin address appears by another name in hexadecimal within the blockchain's transaction outputs. Why is this important? Well, addresses serve as names for locations. Whereas both street addresses and coordinates name locations in physical space, Base58 addresses and their hexidecimal counterparts name abstract locations in mathematical space—numbers. A bitcoin transaction output isn't just code; each output says something with a meaning endowed by the practices and intentions of the bitcoin community. And what a bitcoin transaction output says is that some number of satoshis is tied to a number, a location in mathematical space.[49] The fictional baskets, in my model, correspond to these abstract locations in mathematical space. And the addresses of breadworld baskets correspond to the hexadecimal addresses appearing in bitcoin transaction outputs. So, by design, the breadworld model does not have something that corresponds to bitcoin's Base58 addresses. This is one way in which the model simplifies for the benefit of understanding.)

Now, suppose I have some bitcoin. How do I update the ledger to show that you receive some of it? Using one of many free, open-source applications, I must first compose a *transaction*.[50] Valid transactions have two parts: inputs and outputs. Each output locks an amount of bitcoin to an address and thus has two smaller parts:

(i) a recipient address

(ii) an amount to send to the recipient.

In the blockchain, each output has a unique identifier.[51]

A transaction output then remains unspent (during which time we call it a *UTXO*) until the network publishes another transaction with an input that unlocks its bitcoin. To unlock a UTXO, an input must contain:

(iii) an identifier for the UTXO in which the present spender's address appears as recipient,

(iv) the public key that generates the address in (iii), and

(v) a digital signature over the data in (i)-(iii), from the private key for the address in (iii).

Overall, then, transaction inputs unlock bitcoin previously sent to an address, and outputs lock bitcoin with a specification of how much of it should go where.

[49] Interestingly, the number written in hexadecimal and usually translated into Base58 is the result of hashing a public key, which is itself a number constructed from the x and y coordinates of a point on an elliptic curve. So bitcoin addresses name locations in mathematical space that themselves ultimately correspond to locations in geometric space.

[50] For a fuller account of transactions, see Warmke (ms).

[51] The identifier comes in two parts: an identifier for the transaction, and then a number for the output within that transaction.

So, again, how do I update the ledger to show that I've sent you bitcoin? Thankfully, wallet applications simplify the details for us. But, under the hood, the application composes a chunk of code that contains, first, a new output with your address and the appropriate amount, and second, an input that references a UTXO along with the information that unlocks that UTXO. The information that unlocks the UTXO includes my public key,[52] which hashes to the recipient address embedded in the UTXO, and a valid digital signature. The digital signature itself results after (privately) feeding a digital signature algorithm both the private key and other crucial information from the new transaction (excluding my public key and, of course, the signature itself). At the time I write, bitcoin uses the elliptic curve digital signature algorithm, or ECDSA.[53] The diagram below shows the interrelations among my transaction's components:

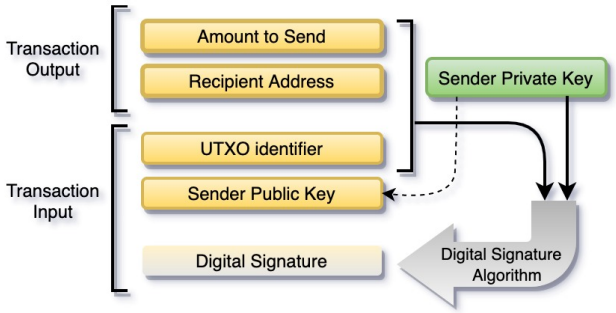

Figure 7. *The anatomy of a bitcoin transaction.*

No one sees the private key, but due to the aforementioned relation between it and its address, anyone can verify whether the appropriate private key helped produce the digital signature over this particular transaction. This feature will come in handy soon. The software application then concatenates all these pieces together and sends the result to the network.

Before we see what happens to my submitted transaction, we should undo a terminological knot and then note a difference with the breadworld network. First, the knot. Transactions are often described as including signatures. But signatures are also often described as being signed over the transaction. These two descriptions don't harmonize. Just as no person is his own parent, no signature results from being one of its own inputs into the signature algorithm. To avoid confusion, it is helpful to remember the distinction between breadworld submissions and breadworld transactions. Breadworld submissions compare with bitcoin transactions. Both include signatures. Breadworld sentences then correspond to *trimmed* bitcoin transactions: transactions minus signatures.[54]

[52] So, in an important sense, bitcoin transactions do not involve "from addresses." See https://en.bitcoin.it/wiki/From_address.

[53] A recent Bitcoin Improvement Proposal (or, "BIP") for Schnorr signatures stands a good chance of being adopted in the near future. See https://github.com/sipa/bips/blob/bip-schnorr/bip-schnorr.mediawiki.

[54] For more details, see Rosenbaum [2019, 127-138].

Finally, we should note that bitcoin transactions differ in an important way from breadworld sentences. Whereas bitcoin transactions generally reference one or more past transactions, breadworld sentences do not reference past sentences. Why not? The breadworld network fundamentally tracks the balances of addresses. With every new published sentence, the network debits an amount of bread from the sending address and credits that same amount to the receiving address. Our bank accounts in real life use this method. Both the breadworld story and our bank balances are account-based. But the bitcoin network does not fundamentally track fluctuating account balances. It tracks transaction outputs which remain unspent. So bitcoin transactions are not account-based but UTXO-based. Imagine if we never deposited checks but instead tracked them as we signed them over from person-to-person. We could record the overall value of checks each owns, but these balances would derive from the checks themselves. Similarly, the "balance" of a bitcoin address seen in a bitcoin wallet derives from the UTXOs in which that address has been the recipient. Since UTXO-based systems are less familiar, I chose to use a more familiar account-based system in the model for accessibility. As long as we note the simplification on the back end, the model still works as intended.[55]

Now, if you recall, I had submitted a transaction to the network to appear in the ledger. What happens next?

6.1.3 Full Nodes

My submitted transaction then awaits verification by the network's *full nodes*. Full nodes are computers connected to the network running free, open-source bitcoin software. They use the software to check whether submitted transactions follow the protocol's rules.[56] These rules concern the proper syntax for transactions and also rules similar to those governing breadworld's fictional universe. Full nodes reject transactions with improper syntax and attempts to spend already spent bitcoin. They also reject transactions without valid digital signatures and any transaction that attempts to spend more bitcoin than the UTXOs it tries to unlock.

The full nodes validate my transaction, and let me unlock bitcoin in a UTXO, only if the transaction contains (i) the public key that generates the UTXO's recipient address, and (ii) a valid digital signature on the new, trimmed down transaction. Without access to the private key, full nodes use the signature verification function to verify whether the signature came from the private key. Valid transactions survive these and other checks unscathed. Full nodes then forward them to a queue called the *mempool* to await bundling and eventual inclusion in the ledger's next *block*, a verified bundle of valid transactions.

[55] Zahnentferner [2018] shows that account-based and UTXO-based ledgers are actually intertranslatable. For a deeper explanation of the distinction, see Akcora et al. [2018, 2-3].

[56] The software is available at https://bitcoin.org/en/bitcoin-core/.

6.1.4 Miners

Bitcoin *miners* compete to have their bundled version of transactions from the mempool appear as the next block in the ledger. I'll explain what blocks are and how miners compete to produce them in turn.

Like breadworld chapters, each bitcoin block includes a summary of its contents and, with an exception for the first block, a pointer to its immediate predecessor. These appear in the *block header*. Here, a summary of the block's transactions exists in the form of a cryptographic digest called a Merkle tree root.[57] The root, which is just 64 hexadecimal characters, results from garbling pairs of transactions, then garbling pairs of garblings, and so on, until a lone garbling remains. The garbling here is done each time via double-SHA-256, a compound function that garbles an input through SHA-256 and then re-garbles the result through SHA-256. We'll see double-SHA-256 again shortly.

The block header also feeds into a hash function whose output is a digital fingerprint for it called a hash pointer. The function here, again, is double-SHA-256. The resulting hash pointer of each block then appears in the subsequent block's header. When full nodes receive a candidate block, its hash pointer determines which block it immediately follows. Hash pointers therefore order chapters linearly from the most recent all the way back to the first, which is embedded like stone in the bitcoin software.

Miners compete by trying to solve a mathematical puzzle. Solving the puzzle requires finding a number (a "nonce") which, when appended to the rest of the candidate block's header and fed to our good friend, double-SHA-256, gives a number beginning with so many repeating zeroes. (So the puzzle solution for a block neatly serves as its hash pointer in the subsequent block. Hence, in breadworld speak, the Puzzle Garbler *is* the Sequencer.) Because the function is one-way, solving the puzzle requires trying as many nonces as possible until the equation spits out a value beginning with at least the necessary number of zeroes. The puzzle is like an algebra problem where several solutions exist for x, but solving for x requires trial and error from an unimaginably large pool of numbers.

A miner who finds a lucky nonce then sends it and the candidate chapter back to the full nodes to verify the puzzle's solution. The nodes then verify that the block follows the protocol's rules before endorsing it as the most recent in a growing series of verified blocks. Each verified block appears in the full nodes' mutually agreed-upon but individually stored version of the ledger. The growing series of verified blocks secured by the network constitutes bitcoin's *blockchain*, a fully public but distributed ledger.[58]

Miners compete to solve puzzles because each block in the ledger includes a *coinbase transaction* providing bitcoin to the winning miner out of thin air. All bitcoin that ever exists will have come from such an award.[59]

[57] For accessible explanations of the cryptographic tools in this section, see Narayanan et al. [2016].

[58] You may browse the ledger at https://blockstream.info/.

[59] Satoshi mined the first block, and it is embedded in the bitcoin software. This aptly named "genesis block" contains a hidden message: "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks," which references a newspaper article from

Every four years, a "halving" occurs, and the amount of bitcoin awarded halves. By 2140, these rewards will cease, and the number of bitcoins in existence will remain permanently at just under 21 million. Bitcoin's provable scarcity has inspired many to treat it as a commodity whose potential value swamps the costs of running more powerful processors to gain a competitive advantage over other miners. But the bitcoin network assimilates more computing power without departing from its issuance schedule.

Every two weeks, the network computes the average puzzle solution time. If miners solve puzzles on average much more quickly (or slowly) than ten minutes, puzzle difficulty automatically increases (decreases). When puzzle difficulty increases, miners must find a nonce such that the input of it and the transaction bundle in the one-way function gives a number beginning with some computed number of extra zeroes. This restricts the pool of potentially lucky nonces, decreases the probability that any given nonce will solve the puzzle, and therefore increases the average solution time. Conversely, if average solution time increases substantially above ten minutes over two weeks, puzzle difficulty decreases to a target number beginning with some computed number of fewer zeroes. This increases the pool of potentially lucky nonces, increases the probability that any given nonce will solve the puzzle, and thereby decreases average solution time. But, on the whole, puzzle difficulty has steadily increased. The network's incentive structure draws miners in to compete in an arms race which, as we'll see, strengthens the network's own security.

Merkle tree roots and hash pointers together help the bitcoin network track unspent bitcoin. With a summary of transactions for each block and an order of blocks, the network can reject attempts to spend bitcoin that it judges to have been already spent. Yet these cryptographic tools alone cannot resolve the situation in which different miners solve a puzzle around the same time and propagate competing blocks. Like the referees of bread-world, full nodes keep both versions of the chain going but eventually endorse the version with more accumulated proof of work. So the further back in the chain a transaction appears, the more secure it is against an attempted double-spend.

Double-spending bitcoin would require altering the historical record by re-mining blocks in the ledger so that it says the bitcoin has not been spent. This kind of attack on the ledger would require an enormous amount of energy because the attacker must solve cryptographic puzzles at a faster clip than the rest of the network combined. This is cost prohibitive and makes blocks further back practically immutable as long as plenty of nodes and miners continue to operate on the network.

Satoshi designed bitcoin's consensus mechanism to steer through the Byzantine generals problem, a problem about reaching consensus among actors to avoid catastrophic failure when some of those actors are unreliable.[60] Bitcoin's distributed consensus concerns which addresses hold

the fallout of the 2008 financial crisis. The reference shows that the chapter wasn't written before 2009, which ensures that, unless Satoshi time-travelled, he has no secret stash of bitcoin created before 2009.

[60] An early statement of this problem appears in Pease et al. [1980] under an-

which amounts of bitcoin, and its proof of work system effectively prevents attempts to spend the same bitcoin twice without relying on central authorities. So bitcoin solves the Byzantine generals problem and the so-called *double spending problem* for digital cash in one fell swoop.[61] Satoshi effectively solved both problems by drawing miners into an arms race with the promise of value that the arms race itself secures.

Though the bitcoin network mitigates vulnerabilities associated with intermediaries and central authorities, it isn't bulletproof. Someone could conceivably crack the algorithm that generates public keys from private keys to gain access to any unspent bitcoin. An inflation bug could appear in a new version of the software.[62] Someone might open up a large short position to cover the cost of attacking the network. Central banks and politicians might even put together a record of responsible monetary and fiscal policy. And, of course, doomsday scenarios which would cripple the internet would severely weaken bitcoin, too. Although these scenarios seem unlikely, they are possible, nonetheless.[63]

6.2 Bread meets Bitcoin

Before we delve into more detailed comparisons between bread and bitcoin, let's look at a bigger picture question about whether a story, a literary creation, is apt for modeling the bitcoin blockchain, a ledger specifically made for transferring value. First, as I noted in Sections 1 and 2, we can successfully model without modeling every feature of a target phenomenon. In fact, models are meant to simplify, so if they modeled every feature of a target phenomenon, they would lose their effectiveness as models. The breadworld model is primarily meant to show both what bitcoin is, metaphysically, and, how the network functions. I will explain those aspects of the model soon. On top of these aspects, bitcoin also has various economic properties having to do with its being a store of value and medium of exchange. Does the absence of these features in the breadworld model count against it?

Not in my view. As long as the modeling successfully leads us to the various metaphysical and functional conclusions I draw below, the model

---

other name, and the coauthors themselves had some difficulty reaching consensus. In a reflective abstract of the paper (available at https://www.microsoft.com/en-us/research/publication/reaching-agreement-presence-faults/) coauthor Leslie Lamport recounts that he "wrote an initial draft, which displeased Shostak so much that [Shostak] completely rewrote it to produce the final version." The authors settled on the problem's name and popular description in Lamport et al. [1982], after realizing that a paper entitled "The Albanian Generals Problem" might cause offense. See https://www.microsoft.com/en-us/research/publication/byzantine-generals-problem/.

61 Recently, Eric Budish [2018, 6] has called the "double" part of the problem's name a "misnomer" since "the attacker can re-spend his Bitcoins arbitrarily many times." The problem is so named precisely because solving it prevents any $2 + n$ spending (where n is any positive integer). By definition, if I can't double spend, I can't triple spend, quadruple spend, etc.

62 For an explanation of bitcoin software development, see Lopp [2018].

63 For more discussion on some of these potential attacks, see Antonopoulos [2017, 253-256] and Ammous [2018, 241-251].

is no worse off for not helping us draw additional conclusions. But, even so, we could easily supplement the model in ways that would make it natural to value breadworld bread as a scarce commodity just like many now value bitcoin as a scarce commodity. First, a small circular economy might develop among enthusiasts where someone might sell, say, pizza for a specified amount of breadworld bread. Second, it costs money to publish breadworld chapters in the very same way that it costs money to mine bitcoin. This might help set a price floor in the market for breadworld bread. And, third, as people begin to respect its other properties (divisibility, scarcity, censorship-resistance), the small circle of enthusiasts might grow to include more enthusiasts, speculators, traders, and so on, increasing demand. So although the model officially includes none of these claims, it could. But their absence in the model doesn't count against it, given its original purpose.

The breadworld and bitcoin networks function similarly and produce a similar product. That product, in each case, is a story about a highly divisible substance with no real-world correlate. Often, what holds for bread holds for bitcoin. And, in many cases, the model breadworld network specifies a type of which something in the bitcoin network is an instance, a sign of successful modeling noted by Williamson [2017, 160]. I've specified the model's components generally by their roles without pinpointing specific role-players. We can map these roles onto specific players in the bitcoin network, which we do below:

Breadworld	Bitcoin
Password Garbler	ECM/SHA-256/RIPEMD160
the Signer	ECDSA
the Notary	signature verification algorithm
Puzzle Garbler	double-SHA-256
Summarizer	merkle tree
Sequencer	double-SHA-256

Table 1: From breadworld roles to bitcoin role-players

These parallels reveal that the differences in description between the breadworld and bitcoin networks often reside more in specificity than in anything else. In fact, the breadworld label for a component often better captures the essence of the corresponding component within the bitcoin network by describing its functional role. This is important because the bitcoin network can evolve by swapping in new components for older ones. The network survives these changes by preserving its components' functional roles. If bitcoin survives long enough, future users might even recognize bitcoin more easily in my model than in my description of bitcoin itself.

A number of important parallels remain. Both breadworld sentences and entries on the bitcoin ledger are chunks of language. Full nodes function as referees on the network, rejecting chunks of language which don't pass muster and validating those which do. Miners function as competitive publishers; they bundle chunks of linguistic material and disseminate it over

the network. And so on. The table below highlights the most important parallels between the real and imagined networks:

Category	Breadworld	Bitcoin
main character	bread	bitcoin
smallest unit	crumb	satoshi
smallest unit x 100,000,000	loaf	bitcoin
username	address	address
username access	password	private key
published item	submission	transaction
verified statement bundle	chapter	block
series of verified bundles	book	blockchain
verifier	referee	full node
distributor	publisher	miner

Table 2: Mapping from Breadworld to Bitcoin

Now, I don't mean to suggest that we may simply choose to apply the word 'referee' to full nodes, 'publisher' to miners, and so on like we might foolishly use 'God' to refer to the "triumphal march of history."[64] I mean quite literally that full nodes *are* referees because they referee and that miners *really are* publishers because they publish. The bitcoin blockchain *is* a digital book. The thousands of people daily "sending bitcoin" really are coauthoring on a massive scale without a central authority. It is by far the largest and longest-running case of hyperauthorship in human history.

With few exceptions, what goes for bread goes for bitcoin. This holds especially in the parallels between breadworld sentences and bitcoin transactions, despite the difference noted earlier between bitcoin's UTXO system and breadworld's account system. Just as a breadworld sentence describes the movement of bread and is not itself a movement of bread, a bitcoin transaction *describes* a transfer of bitcoin without *being* that transfer of bitcoin. Hence, in an important sense, calling this chunk of code a "transaction" conflates use and mention by identifying a chunk of language that describes an event with the event itself. In both the breadworld and bitcoin stories, we use chunks of linguistic items to represent movements of fictional substances that are not themselves linguistic items.

If the chunk of code in the bitcoin blockchain that describes a movement of bitcoin is not itself that movement, where does the movement, the transaction, take place? Nowhere—or everywhere, depending on how you think of it. If bread's movements are fictional, so, too, are bitcoin's movements across addresses as described in the digital book of bitcoin. Furthermore, nothing beyond our representation of bitcoin has bitcoin's properties. Since the bitcoin blockchain represents the fictional movements of a substance that is no more real than breadworld bread, we have every reason to treat bitcoin and breadworld bread similarly. Hence, like breadworld bread, bitcoin is a fictional substance or at the very least, a merely intentional object.

[64] Lewis [1986, 140].

Now, many falsely claim that bitcoin is code, which would imply that something beyond our representation of bitcoin does have the properties we attribute to bitcoin.[65] Saying that bitcoin is code is like saying that Sherlock is the name 'Sherlock' or that he is the class of sentences that mention him or something similarly confused. Sherlock is not a word or a class of sentences. Unlike any word or class of sentences, Sherlock is a British detective. Similarly, no portion of code has bitcoin's properties. If I send you five satoshis, I don't send five symbols or even five strings of symbols—that isn't how bitcoin works. There aren't even chunks of code with which we could identify each satoshi. Satoshis don't have traceable identifiers like vehicle identification numbers. The protocol only settles an address's amount of unspent bitcoin and the immediate source of that bitcoin. Nothing in the bitcoin blockhain corresponds to each bitcoin or satoshi. And whereas bitcoin divides into satoshis and not symbols, code divides into symbols and not satoshis. By Leibniz's Law, then, bitcoin is not code. Neither code nor anything else has bitcoin's properties. So we have every reason to treat bitcoin on a par with breadworld bread and Sherlock Holmes.

In review, our modeling strategy has revealed bitcoin to be a fictional substance in an on-going and massively coauthored book on a network that automates and decentralizes the stages leading to publication. Importantly, bitcoin isn't just a fictional substance, but a fictional substance valued as a commodity and whose ownership amounts to real life access to cryptographically secure private keys. So the story encoded in the bitcoin blockchain also serves as a trustworthy public ledger for unspent balances across all addresses. When we conceive of bitcoin in this way, the Coordination, Corruption, and Credit Problems for hyperauthorship take on new meaning.

The Coordination Problem concerns not just how we hyperauthor effectively without a central auhority, but how we achieve consensus about how much of a valuable commodity everyone has without trusting vulnerable central authorities. The Corruption Problem morphs into the problem of how to secure the integrity of that ledger against incompetence, malfeasance, and other potential risks. Finally, the Credit Problem no longer merely concerns how to assign credit for writing a valid chunk of code but now also how to assign ownership, or, in the accounting sense of the term, how to tell who has credited whom. This close relationship between authorship and ownership might tempt us once more into thinking that the coin that's owned is part of the code that's authored. But I hope that the pains I've taken in this paper will help us resist this temptation. Coin owned is not code authored.[66]

[65] Usually, authors say that bitcoins or satoshis are "chains of digital signatures." I say more about this claim in the conclusion. Here is a somewhat representative sample: Van Valkenburgh [2014, 9], Lastra and Allen [2018, 55], Akins et al. [2014, 30 n. 30], Wu et al. [2017, 3124], Kroll et al. [2013, 3], Gao et al. [2018, 27207], Zhang [2017, 560], Friedlmaier et al. [2018, 2].

[66] What, then, does it mean to own a cryptocurrency? This is an open and complicated question with important legal consequences. See Hinkes [2019], Sutherland [2019], and Stabile et al. [2020].

7 Conclusion

We began with questions about what bitcoin is and how it works. With a model meant to serve as an accessible on-ramp for understanding bitcoin's complexities, we have answered both questions more or less simultaneously. Bitcoin is a fictional substance that appears in a digital book, the blockchain, secured by a network that decentralizes the jobs of refereeing and publishing. Still, bitcoin has unexplored depths that we cannot possibly examine in a single paper. My account of bitcoin itself raises questions that we must save for another time.

First, since bitcoin mining is a form of publishing, questions about the freedoms of the press loom nearby. But bitcoin mining complicates these issues about the freedom to publish because the on-going publication generates a non-sovereign form of money. This connection between mining and the freedom of the press deserves further exploration.

Second, since bitcoin is a fictional substance and not code, it isn't any particular kind of code. Consequently, no bitcoin is a "chain of digital signatures," a conclusion which appears to contradict Satoshi's definition of electronic coins in the bitcoin whitepaper.[67] Unfortunately, this definition continues to appear with little or no qualification in a steady stream of academic papers, government documents, and arguments about which cryptocurrency is the *real* bitcoin. How, then, should we understand the definition?[68]

Finally, how, if it all, does bitcoin's fictional status differentiate it from fiat currencies like the U.S. dollar? As I argue elsewhere, I think not much.[69] Typically, what signifies isn't identical to what is signified. So a $20 bill is not itself the quantity of twenty dollars that the bill signifies. So a $1 bill is not itself the quantity of one dollar that the bill signifies either. But what are these things, the dollars, that bills signify? They are quantities of a fictional substance, in my view. Bitcoin differs from the dollar, however, in a crucial way. Whether a financial instrument successfully signifies dollars heavily depends on whether the U.S. government says it does. But no centralized body has similar control over bitcoin.

[67] Nakamoto [2008, 2].

[68] I answer this question in Warmke (ms).

[69] Warmke (ms). Thanks to Martin Peterson for discussion on this point.

References

Cuneyt Gurcan Akcora, Matthew F. Dixon, Yulia R. Gel, and Murat Kantarcioglu. Blockchain data analytics. *Intelligent Informatics*, page 4, 2018.

Benjamin W Akins, Jennifer L Chapman, and Jason M Gordon. A whole new world: Income tax considerations of the bitcoin economy. *Pitt. Tax Rev.*, 12:25–56, 2014.

Stefania Albanesi. Inflation and inequality. *Journal of Monetary Economics*, 54(4):1088–1114, 2007.

Saifedean Ammous. *The bitcoin standard: the decentralized alternative to central banking*. John Wiley & Sons, 2018.

Andreas Antonopoulos. *Mastering Bitcoin: unlocking digital cryptocurrencies*. O'Reilly Media, Inc., 2nd. edition, 2017. First published in 2014.

Evelyn Mary Aswad. The future of freedom of expression online. *Duke L. & Tech. Rev.*, 17:26, 2018.

Raphael Auer and Rainer Böhme. The technology of retail central bank digital currency. *BIS Quarterly Review, March*, 2020.

Andrew Bailey, Bradley Rettler, and Craig Warmke. Cryptocurrency: Philosophy, politics, and economics. Unpublished manuscript.

Nathan Ballantyne. Epistemic trespassing. *Mind*, 128:367–395, 2019.

Juliet Barker. *Wordsworth: a life in letters*. Viking, 2002.

Ole Bjerg. How is bitcoin money? *Theory, Culture & Society*, 33(1):53–72, 2016.

Codruta Boar, Henry Holden, and Amber Wadsworth. Impending arrival–a sequel to the survey on central bank digital currency. *BIS Paper*, (107), 2020.

Barry Bozeman, Monica Gaughan, Jan Youtie, Catherine P Slade, and Heather Rimes. Research collaboration experiences, good and bad: Dispatches from the front lines. *Science and Public Policy*, 43(2):226–244, 2015.

David Braun. Empty names, fictional names, mythical names. *Noûs*, 39(4): 596–631, 2005.

Eric Budish. The economic limits of bitcoin and the blockchain. Technical report, National Bureau of Economic Research, 2018. URL https://www.nber.org/papers/w24717.

Long Cheng, Fang Liu, and Danfeng Yao. Enterprise data breach: Causes, challenges, prevention, and future directions. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(5):1–14, 2017.

Timothy Y Chow. A beginner's guide to forcing. *Communicating mathematics*, 479:25–40, 2009.

Blaise Cronin. Hyperauthorship: A postmodern perversion or evidence of a structural shift in scholarly communication practices? *Journal of the American Society for Information Science and Technology*, 52(7):558–569, 2001.

Seyed Mohammadreza Davoodalhosseini. Central bank digital currency and monetary policy. *Available at SSRN 3011401*, 2018.

Whitfield Diffie. The first ten years of public-key cryptography. *Proceedings of the IEEE*, 76(5):560–577, 1988.

William Easterly and Stanley Fischer. *Inflation and the Poor*. The World Bank, 1999.

Andrea Forte and Amy Bruckman. Why do people write for wikipedia? *Workshop on Sustaining Community: The Role and Design of Incentive Mechanisms in Online Systems*, 2005. ACM conference on Groupwork, Sanibel Island, FL.

Maximilian Friedlmaier, Andranik Tumasjan, and Isabell M Welpe. Disrupting industries with blockchain: The industry, venture capital funding, and regional distribution of blockchain ventures. In *Venture Capital Funding, and Regional Distribution of Blockchain Ventures (September 22, 2017). Proceedings of the 51st Annual Hawaii International Conference on System Sciences (HICSS)*, 2018.

Yu-Long Gao, Xiu-Bo Chen, Yu-Ling Chen, Ying Sun, Xin-Xin Niu, and Yi-Xian Yang. A secure cryptocurrency scheme based on post-quantum blockchain. *IEEE Access*, 6:27205–27213, 2018.

Ronald N Giere. *Explaining science: A cognitive approach*. University of Chicago Press, 1988.

Peter Godfrey-Smith. The strategy of model-based science. *Biology and philosophy*, 21(5):725–740, 2006a.

Peter Godfrey-Smith. Theories and models in metaphysics. *The Harvard Review of Philosophy*, 14(1):4–19, 2006b.

Jeffrey Goodman. A defense of creationism in fiction. *Grazer Philosophische Studien*, 67(1):131–155, 2004.

Martin Hellman. An overview of public key cryptography. *IEEE Communications Society Magazine*, 16(6):24–32, 1978.

Andrew M Hinkes. Throw away the key, or the key holder? coercive contempt for lost or forgotten cryptocurrency private keys, or obstinate holders. *Northwestern Journal of Technology and Intellectual Property*, 16(4):225–264, 2019.

John P. A. Ioannidis, Richard Klavans, and Kevin W. Boyack. Thousands of scientists publish a paper every five days. *Nature*, 561:167–169, 2018.

Aniket Kesari, Chris Hoofnagle, and Damon McCoy. Deterring cybercrime: Focus on intermediaries. *Berkeley Tech. LJ*, 32:1093–1134, 2017.

John Kiff, Jihad Alwazir, Sonja Davidovic, Aquiles Farias, Ashraf Khan, Tanai Khiaonarong, Majid Malaika, Hunter Monroe, Nobu Sugimoto, Hervé Tourpe, et al. A survey of research on retail central bank digital currency. *Available at SSRN 3639760*, 2020.

Ronald A Knox. Studies in the literature of sherlock holmes. *New Blackfriars*, 1(3):154–172, 1920.

Vasilis Kostakis. Identifying and understanding the problems of wikipedia's peer governance: The case of inclusionists versus deletionists. *First Monday*, 15(3):162–192, 2010.

Saul Kripke. *Naming and Necessity*. Harvard University Press, Cambridge, MA, 1980. Originally published in 1972.

Saul A Kripke. Vacuous names and fictional entities. In *Philosophical Troubles. Collected Papers Vol. 1*, pages 52–74. Oxford University Press, 2011.

Joshua A Kroll, Ian C Davey, and Edward W Felten. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In *Proceedings of WEIS*, volume 2013, 2013.

Fred Kroon and Alberto Voltolini. Fictional entities. *Stanford Encyclopedia of Philosophy*, 2018. URL https://plato.stanford.edu/entries/fictional-entities/.

Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

A Lannquist, S Warren, and R Samans. Central bank digital currency policy-maker toolkit. In *Insight Report, World Economic Forum, Geneva*, 2020.

Rosa Maria Lastra and Jason Grant Allen. Virtual currencies in the eurosystem: Challenges ahead. *Brussels, Belgium: ECON Committee, European Parliament*, 2018.

David Lewis. *On the Plurality of Worlds*. Basil Blackwell, Oxford, 1986.

Hongyi Li, Heng-fu Zou, et al. Inflation, growth, and income distribution: A cross-country study. *Annals of Economics and Finance*, 3(1):85–101, 2002.

Jameson Lopp. Who controls bitcoin core?, 2018. URL https://medium.com/@lopp/who-controls-bitcoin-core-c55c0af91b8a.

Brander Matthews. The art and mystery of collaboration. *Longman's magazine, 1882-1905*, 16(92):157–170, 1890.

Matthew McKeever. What blockchains are and why you should care: A parable, 2018. URL https://medium.com/@mittmattmutt/three-reasons-leftish-people-should-like-blockchains-db79bd3d42bb.

Jack Meaning, Ben Dyson, James Barker, and Emily Clayton. Staff working paper no. 724 broadening narrow money: monetary policy with a central bank digital currency. 2018.

Ioan-Radu Motoarca. Fictional surrogates. *Philosophia*, 42(4):1033–1053, 2014.

Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. URL http://bitcoin.org/bitcoin.pdf.

Satoshi Nakamoto. Bitcoin open source implementation of p2p currency, 2009a. URL http://p2pfoundation.ning.com/forum/topics/bitcoin-open-source?id=2003008%3ATopic%3A9402&page=1#comments.

Satoshi Nakamoto. Bitcoin v0.1 released, 2009b. URL https://satoshi.nakamotoinstitute.org/emails/cryptography/16/#selection-9.0-9.21.

Satoshi Nakamoto. Re: Slashdot submission for 1.0, 2010. URL https://satoshi.nakamotoinstitute.org/posts/bitcointalk/167/.

Arvind Narayanan and Jeremy Clark. Bitcoin's academic pedigree. *Communications of the ACM*, 60(12):36–45, 2017.

Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. *Bitcoin and cryptocurrency technologies: a comprehensive introduction*. Princeton University Press, 2016.

Eric Pacuit. Voting methods. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. 2019. URL https://plato.stanford.edu/entries/voting-methods/.

Michael Parker and Patricia Kingori. Good and bad research collaborations: researchers' views on science and ethics in global health research. *PloS one*, 11(10):e0163579, 2016.

Terence Parsons. *Nonexistent Objects*. Yale University Press, New Haven, 1980.

Laurie A Paul. Metaphysics as modeling: the handmaiden's tale. *Philosophical studies*, 160(1):1–29, 2012.

Marshall Pease, Robert Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM (JACM)*, 27(2):228–234, 1980.

Geoffrey Pullman. To co-author, or not to co-author, 2016. URL https://web.archive.org/web/20170726032613/https://www.chronicle.com/blogs/linguafranca/2016/02/11/to-co-author-or-not-to-co-author/.

Wessel Reijers and Mark Coeckelbergh. The blockchain as a narrative technology: investigating the social ontology and normative configurations of cryptocurrencies. *Philosophy & Technology*, 31(1):103–130, 2018.

Kalle Rosenbaum. *Grokking Bitcoin.* Manning, Shelter Island, 2019.

Nathan Salmon. Mythical objects. *Meaning and truth: Investigations in philosophical semantics*, pages 105–123, 2002.

Stephen Schiffer. Language-created, language-independent entities. *Philosophical Topics*, 24:149–167, 1996.

Stephen Schiffer. *The Things We Mean.* Clarendon Press, Oxford, 2003.

John R Searle. The logical status of fictional discourse. *New literary history*, 6(2):319–332, 1975.

Daniel J Solove. *The digital person: Technology and privacy in the information age*, volume 1. NyU Press, 2004.

Jimmy Song. *Programming Bitcoin.* O'Reilly Media, Inc., 2019.

Daniel T Stabile, Kimberly A Prior, and Andrew M Hinkes. *Digital Assets and Blockchain Technology: US Law and Regulation.* Edward Elgar Publishing, 2020.

Abraham Sutherland. Cryptocurrency economics and the taxation of block rewards, parts 1 & 2. *Tax Notes Federal*, 165(6):749–771; 953–972, 2019.

Melanie Swan and Primavera De Filippi. Toward a philosophy of blockchain: A symposium: Introduction. *Metaphilosophy*, 48(5):603–619, 2017.

Peter P Swire. Financial privacy and the theory of high-tech government surveillance. *Wash. ULQ*, 77:461, 1999.

Nick Szabo. Money, blockchains, and social scalability, 2017. URL https://nakamotoinstitute.org/money-blockchains-and-social-scalability/.

Paul Sztorc. Long live proof-of-work, long live mining, 2014. URL https://www.truthcoin.info/blog/pow-and-mining/.

Amie L Thomasson. *Fiction and metaphysics.* Cambridge University Press, 1999.

Peter van Inwagen. Creatures of fiction. *American Philosophical Quarterly*, 14:299–308, 1977.

Peter Van Valkenburgh. Comments to the conference of state bank supervisors on the draft model state regulatory framework for virtual currency. 2014.

Pablo R Velasco. Computing ledgers and the political ontology of the blockchain. *Metaphilosophy*, 48(5):712–726, 2017.

Alberto Voltolini. *How ficta follow fiction: A syncretistic account of fictional entities*, volume 105. Springer Science & Business Media, 2006.

Alberto Voltolini. *Fictional Objects*, chapter A Suitable Metaphysics for Fictional Entities, pages 129–146. Oxford University Press, 2015.

Angela Walch. The path of the blockchain lexicon (and the law). *Review of Banking & Financial Law*, 36:713–765, 2016.

Angela Walch. Blockchain's treacherous vocabulary: One more challenge for regulators. *Journal of Internet Law*, 21(2):9–16, 2017.

Craig Warmke. Electronic coins. Unpublished manuscript.

Michael Weisberg. Who is a modeler? *The British journal for the philosophy of science*, 58(2):207–233, 2007.

Michael Weisberg. *Simulation and similarity: Using models to understand the world*. Oxford University Press, 2013.

Timothy Williamson. Model-building in philosophy. In Russell Blackford and Damien Broderick, editors, *Philosophy's Future: The Problem of Philosophical Progress*, pages 159–172. Wiley-Blackwell Hoboken, NJ, 2017.

William Wordsworth and Samuel Taylor Coleridge. *Lyrical Ballads*. T.N. Longman and O. Rees, London, 2nd edition, 1800. 1st edition published in 1798.

Qianhong Wu, Xiuwen Zhou, Bo Qin, Jiankun Hu, Jianwei Liu, and Yong Ding. Secure joint bitcoin trading with partially blind fuzzy signatures. *Soft Computing*, 21:3123–3134, 2017.

Joachim Zahnentferner. Chimeric ledgers: Translating and unifying utxo-based and account-based cryptocurrencies. *IACR Cryptology ePrint Archive*, 2018:262, 2018.

Yilu Zhang. The incompatibility of bitcoin's strong decentralization ideology and its growth as a scalable currency. *NYUJL & Liberty*, 11:556, 2017.