METAPHILOSOPHY

METAPHILOSOPHY
Vol. 48, No. 5, October 2017
0026-1068

BLOCKCHAIN IDENTITIES

NOTATIONAL TECHNOLOGIES FOR CONTROL AND MANAGEMENT OF ABSTRACTED ENTITIES

QUINN DUPONT

Abstract: This paper argues that many so-called digital technologies can be construed as notational technologies, explored through the example of Monegraph, an art and digital asset management platform built on top of the blockchain system originally developed for the cryptocurrency bitcoin. As the paper characterizes it, a notational technology is the performance of syntactic notation within a field of reference, a technologized version of what Nelson Goodman called a "notational system." Notational technologies produce abstracted entities through positive and reliable, or constitutive, tests of socially acceptable meaning. Accordingly, this account deviates from typical narratives of blockchains (usually characterized as Turing or state machines), instead demonstrating that blockchain technologies are effective at managing digital assets because they produce abstracted identities through the performance of notation. Since notational technologies rely on configurations of socially acceptable meaning, this paper also provides a philosophical account of how blockchain technologies are socially embedded.

Keywords: blockchain, notation, Goodman, identity, philosophy of technology, philosophy of computing.

Digital technologies are useful in large part because they (a) create representations by abstracting away complex properties of objects and then (b) use these newly formed identities for control and management of entities. Typically, this computational process is then used to control and manage "real world" entities (the targets of representation), which consequently do work in the world. In fact, this process of abstraction and subsequent control has long been one of the greatest strengths of digital technologies. Consider, for example, the formative years of proto-computing: systems and machines such as Francis Bacon's bi-literal encryption that used binary "characters" to "represent anything by anything," Gottfried Wilhelm Leibniz's stepped reckoner that developed in the shadow of his notation for combinatory mathematics, and Herman Hollerith's tabulator that sorted and calculated digital "punch card" data. Each of these machines relied on an (often unacknowledged) process of abstraction to form identities that would be inscribed in their media and perhaps even mechanized. The innocuous characterizations of data, abstraction, and representation, however, obscure the tremendous philosophical complexity and genuine conceptual innovation that was necessary to bring to life these inventions.

More recently, we have seen considerable impact from networked digital technologies with, worrisomely, each new device further controlling an aspect of daily life previously considered "authentic" and human. Like their "offline" predecessors, these digital systems order, sort, and calculate and are often combined with technologies that then control and manage real people and goods. Today, these systems are critical for moving physical goods through supply chains, reallocating "flexible" workforces, transferring finances across national borders, and controlling computer-aided manufacturing machines, to name just some obvious examples. Again, these networked computational systems rely on their ability to abstract and manage identities, a philosophically complex process underlying an extensive range of computing technologies today. Perhaps these technologies are most potent when configured as what we might call online "identity systems" (if the term was not already occupied), which would include hyperlinks on the World Wide Web (Universal Resource Identifiers), data and software source code versioning systems (git, RCS, and so on), and recently, blockchain technologies (the technology powering cryptocurrencies, such as, originally, bitcoin).

I argue that this process of abstraction and identity management is a necessary property of digital computation technologies in general, and blockchain technologies in particular. This line of thinking is a continuation of my earlier work on cryptocurrencies (DuPont 2014) and algorithms (DuPont and Takhteyev 2016), where I attempted to provide philosophical groundwork for computing technologies in the wild. Being "digital" is one characterization of what I am trying to get at, and is a major part of the process. I shall, however, shy away from this term, as the associated concept is unable to explain how these machines work, and besides, the term is unhelpfully polysemic in contemporary use.

My construal of this particular dimension of computing identifies a process of abstraction and identity reliant on the syntactic and semantic characteristics of "notation," in the sense identified by Nelson Goodman in his *Languages of Art: An Approach to a Theory of Symbols* (1976 [1968]). In developing Goodman's theory of notation, I argue that digital computation technologies are necessarily, but not sufficiently, notational technologies. Today, the notational aspects of digital technologies are perhaps the most critical ones and ought to be recognized as the key to understanding contemporary computing. In particular, blockchain technologies as they have developed from their original use for the cryptocurrency bitcoin operationalize notational properties for much of their characteristic functionality. To show the utility of this line of thinking, I describe the philosophical underpinnings of the Monegraph platform, a blockchain system tailored for managing and monetizing digital art. I conclude that the Monegraph platform relies on the notational properties of digital art for management and monetization, and therefore is a notational technology par excellence and a useful technology to measure the conceptual efficacy of my construal against.

Goodman's Theories of Notation

Nelson Goodman is probably best known for his work on logic and language, and for his infamous modern rehabilitation of nominalism. While he was more famous in the field of philosophy for his work on these topics, it is his late work on aesthetics, *Languages of Art* (1976 [1968]) that contains a sophisticated theory of notation, which I believe is foundational for understanding modern digital technologies, and in particular, blockchain technologies. Unfortunately, Goodman's complex analytic prose and tendency to advocate for nominalism (which never really caught on in the philosophical mainstream) seem to have overshadowed this book's stunning descriptions of digitality. Moreover, it was written before the widespread use of digital technologies (especially personal computers) and is ostensibly about art: it is no wonder, then, that it is not essential reading today for computer scientists. Nonetheless, Goodman himself offers the example of digital computers when describing his theory of notation, marking his recognition of the theory's applicability.

To develop his specific account of representation for notation systems, and therefore computers, Goodman first reflects on the "naive view" of representation, sometimes also known in literary fields as mimesis (see also DuPont and Takhteyev 2016). Goodman rejects this view because it relies on the psychological character of representation. He argues that *resemblance* (the primary character of the naive view) is symmetric (A is as much like B, as B is like A), which, among other issues, causes a kind of phenomenological recursion in interpretation and therefore makes it unsuitable for a robust theory of notation.

According to Goodman, resemblance fails to offer coherent normative grounds for what (aesthetically) good resemblance would need to be. He argues that resemblance is always resemblance "as," or resemblance from some specific and usually unspecified vantage point (1976, 6–10, 27–31). Indeed, this *conceptual vagueness* is resemblance's sin of theoretical omission. Goodman also points out that resemblance is *conventional and mutable*. The ways in which something might resemble something else change over time; for example, the invention of artistic perspective in the Renaissance meant a radical change to people's views about how things look, and therefore changed how things were thought to resemble one another. So, over time (as views about perspective changed), notions of "accurate" or "good" resemblance potentially became contradictory, and therefore, Goodman argues, a theory of resemblance makes a poor philosophical grounding for representation (10–19). "Effective representation and description" Goodman suggests, "require invention"—not merely imitation (33). The world, at least as we know it, does not come free of interpretation. Goodman stresses his nominalist view: "Nature is a product of art and discourse" (33). On less nominalist interpretations, we might admit a degree of ontological realism and still recognize the role of social and technical construction.

Reference and Performance

Computing is an interesting case where it appears quite clear (to me, at least) that resemblance has very little to do with how computers work. Computers are, in this sense, very distinct from older forms of media, such as pictures, television, radio, and so on. Nonetheless, I also argue that computers are representational (I admit that this is a contested point). To develop a construal of computing that makes sense of representation without resemblance, I target the properties of "notation." Other media might be assessed and characterized by their ability to *look like* their subjects, and this is certainly important for the ways that computers portray their subject matter (we are likely to prefer a digital image that, according to some changing standard, *looks more like* its subject than some poor-quality digital image). But, by and large, for media that are characterized by notation, at least for their notational uses (discussed below), there is little insight to be gained through an analysis of resemblance. This tension between notation and resemblance is a key element of my case study of the Monegraph platform (presented below).

The way that the represented world is made varies between subjects, contexts, times, cultures, and personal predilections; in other words, representation is socially constructed. For notation, we can see that some expressions are already ideally suited to the kind of representation that is necessary, whereas others may require a great deal of preparation or alteration (a form of abstraction). Abstraction is not a

Platonic procedure (the way numbers are usually thought to be). Rather, the degree to which a subject may require transformation in preparation for notational representation is context specific (that is, is vague) and is judged by implicit personal and cultural standards (that is, is conventional and mutable). Consider, for example, a realistic-looking trompe l'œil painting. While it is possible to render the painting in some crude digital (notational) format (imagine a digital image captured by an early digital camera), or even a sophisticated high-quality digital format (with a digital camera today), in both cases there does seem to be something missing from the painting's digital representation beyond its mere materiality. The digital image *lacks* an important dimension of the painting, even if it is difficult to pinpoint what exactly we feel this lack might be.

This same question is taken up by John Haugeland in his analysis of digital and analog devices (1981). While Haugeland disagrees with a few key points of Goodman's analysis, he agrees in essence with Goodman's view, that in most cases a second-order digitality is possible. In other words, Haugeland asked, "Is every analog device second-order digital?" (223). Yes, he concludes. But he demurs: in many cases the performance of digital computation or simulation of a natively "analog" world would be "preposterous" (and practical computation or simulation is the nut of the exercise, he seems to believe) (224). For instance, digitally simulating individual molecules would far exceed current capabilities (224). Or, to use a more germane example, the fundamentals of contemporary cryptography are considered secure because the possible guesses needed to brute force attack the key quickly spirals beyond any current or *possible* traditional computational analyses (quantum cryptanalysis potentially complicates this). The possible guesses become impractical due to the "combinatorial explosion" that occurs in cryptanalysis of long keys (that is, there are more possible guesses needed to crack a typical strong encryption than there are atoms in the universe). In addition, Haugeland argues that claims about second-order digitality are about "macroscopic phenomena," or the way that "photographs, linear amplifiers, and analog computers" can or cannot be digitally simulated (224). Claims about second-order digitality are not about "fundamental physics" (224).

Haugeland seems to miss the philosophical point here, however, or is simply unwilling to accept the "preposterous" conclusions. He is surely right that some digital simulations are pretty preposterous; consider the Discrete Integrated Circuit Emulator (DICE) software. This ambitious project attempts to emulate old video games (such as Pong and Atari Breakout) by simulating the actual transistor propagation delays (the analog microphysics) for each and every circuit. The result is a breathtaking commitment to video game "authenticity" but is also so enormously computationally intensive that only the very simplest games can be emulated with top-of-the-line contemporary personal computers. Do the microphysics of transistor propagation delays bubble up into the macroscopic experience of playing these old games—making for a more authentic experience? Perhaps, or perhaps not—but it doesn't stop people from trying.

Or, to be completely preposterous, imagine the computer that simulates the comedian Louis C.K.'s entire known universe, as he explains it to his daughter: "Some things are, and some things are not. Why? Well, because things that are not can't be. Why? Because then nothing wouldn't be. You can't have fucking nothing isn't, everything is! Why? Cause if nothing wasn't, there'd be fucking all kinds of shit, like giant ants with top hats dancing around" (Santos et al. 2006). It is rather simple—Parmenidian, even: there is, and there is not. Inside this abstraction—"is" and "not"—there can be all kinds of things (but perhaps not giant ants with top hats). But, at this preposterous level of second-order digitality, the issue is not that the system cannot be simulated or that it devolves into a microphysics, the issue is plainly that (probably) nobody would much value a simulation with only a single binary modality. The digital simulation of the entire known universe is preposterous!

In general, those objects that we take to be more natural, authentic, and creative are the ones we have the hardest time abstracting and accepting in notational (or "digital") terms. Notational versions of these objects, we usually say, lack naturalness and authenticity, and might not have the correct material origin. On the other hand, an already digital representation—perhaps a data point that tracks a mail-order package from its distribution center to a home—probably "makes sense" to most people and does not appear to be unusually unnatural, inauthentic, or without the correct origins. That is, when the digital, logistical representation is just a row in a database, we tend to accept this representation and do not deem anything particularly important to be missing. Distilling logistical tracking to a mere data point does not lack in ways that a digital rendering of a beautiful trompe l'œil painting might (in extreme cases, the digital rendering may even seem morally wrong or false to some people). For most people, especially those who enjoy art, the technological representation of "traditional" art seems to miss something important about *art* itself.

This feeling of lack may extend to all digitized art, but as we shall see, some kinds of art (born digital art, in particular) does not seem to produce such a strong feeling that something is missing. When digital art stays digital or is transformed into another digital format, less (or perhaps *nothing*) is lost. As we shall see, this dimension of digital authenticity and our individual and cultural feelings about it raise serious issues about storing representations of art on a blockchain.

Whether such abstractions are seen to be acceptable depends on how they are used and what people think about this kind of notational representation.

One of the key distinctions here is whether the artwork in question (for example, a digitized trompe l'œil painting) has the appropriate origin. The test is: Is a painting's origin relevant for its identity, and if so, does this particular painting have the appropriate origin? If the painting in front of me is an authentic Rembrandt in the sense that Rembrandt literally painted it so many years ago, then we usually value its appropriate origin (a counterfeit version is usually unacceptable). If the painting in question is a digital copy (even a very good one), then it usually fails the test, and we typically register a feeling of lack.

Goodman offers a more precise analytical characterization for this question about origins of identity. First, there is a class of expressions that are "unfakable," or "allographic" (for example, musical scores). Second, there are also "fakable" expressions, called "autographic," which have a "significant" distinction between originals and forgeries (for example, painting) (1976, 112).

Goodman considers the counterargument that the distinction is really a matter of whether or not the expression is "one stage" or "two stage." For instance, musical composition is two stage because usually it is composed (stage one) and then played (stage two). Painting, on the other hand, is one stage because paint is applied directly to the canvas. These counterexamples, however, are defeated by Goodman. Consider literature, which is allographic (unfakable), although one stage (Goodman 1976, 114). Goodman concludes that the distinction between autographic and allographic arts is not the number of stages in its production. Rather, allographic expressions require the "sameness of spelling" between two copies—any sequence of marks that "corresponds to a correct copy" is notational (even if the correct copy is itself a type of forgery—perhaps a fake of an author's manuscript). In cases of allographic arts, "nothing" Goodman notes, "is more the original work than is such a correct copy" (116).

One of the prerequisites of the criteria for "sameness of spelling," then, is that the work must be composed of "certain signs or characters," that is, be in a "definite notation" (Goodman 1976, 116). This definite notation must be "constitutive" of the work (116). Even in cases of two-stage arts, such as music, the performance is allographic (and not autographic) because the "constitutive properties demanded of a performance" are *prescribed* in the score (117). Nonconstitutive (that is, contingent) properties of the music may change from one performance to the next (such as variations in loudness or tempo), but so long as a certain adherence to the score is maintained, the performance remains allographic.

To continue with the example, there must, however, be a test for music in order to "correlate appropriate sounds with visible signs in the score" and to determine compliance of sequence and order (Goodman 1976, 117). This test is important because a score—indeed, all such notation—has the primary function of authoritatively identifying a work from one *performance* to another *performance*. Consider classical music: compliance to (and divergence from) the score determines two different symphonies (for example, *Brandenburg Concerto* versus Orchestral Suite No. 1). Similarly, compliance to a score is why a rough high-school recital of a Bach *Brandenburg Concerto* is *formally* the same as Glenn Gould's version. Even if the two renditions differ immensely in the "quality" of playing, they are both *expressions* of the same *work*. Given the nature of such a test, however, there may in fact be some (or even many) times when making a determination of correspondence is difficult (perhaps telling the difference between a quietly played C versus a C-sharp). But, critically, making a determination of correspondence must never be impossible: the test must be possible *in theory*.

Haugeland offers a similar account. A digital device is:

i) A set of types;
ii) A set of feasible procedures for writing and reading tokens of those types; and
iii) A specification of suitable operating conditions; such that
iv) Under those conditions, the procedures for the write-read cycle are positive and reliable. (1981, 215)

Haugeland, however, is concerned only with the *procedures* needed to determine if an expression is digital (notational). This focus on procedure alone is because, unlike Goodman, he does not believe "digital devices are ... necessarily representational or symbolic" (225). Therefore, for Haugeland, the performance of a digital device extends to its *practical* utility but not its essence, as it were. I disagree with this line of argument: I argue that digital devices are representational in the sense of an interplay between their mechanism and meaning (Smith 2010), and therefore their performance qua digital is critical. As I shall discuss in the context of blockchain technologies, the ability to "round trip" a digital representation—from performance to re-presentation—is critical to managing and controlling representations and therefore is critical for these technologies to do real work in the world.

Symbol Systems: Notational Schemes and Systems

The performance of notation, or digital symbols, marshals the shift from syntax to semantics. Goodman himself charts this move by offering two theoretical construals of notation. The first is called a "notational *scheme*" and has only syntactic requirements. The second is called a "notational *system*" and includes all the requirements for notational schemes plus additional semantic requirements (that is, is a superset of notational schemes). With performance, the notational scheme is able to reach out into the world and create meaning: "The relationships obtain ... between notational scheme and *application*" (Goodman 1976, 130; my emphasis). To reiterate, a notational system requires syntactic notation and its performance.

A notational *scheme* consists of characters. A character is any mark or utterance that can be "freely exchanged for one another without any syntactic effect" (Goodman 1976, 131). This is the sufficient condition of marks being "true copies" of each other, according to Goodman; or, to user more familiar terminology, the notational scheme is composed of tokens of a set of types (the set of types with a positive and reliable test, as per Haugeland's construal). Since each token can be replaced with another of the same type, they are character "indifferent," which means they must be "disjoint" from one another and "finitely differentiated" (133–35). Their being disjoint means the tokens must be syntactical copies in the sense that no two types contain mutual tokens (that is, no syntactically identical tokens between types). Their being finitely differentiated means there must be some in-principle test to determine if some token does not belong to any two types (similar to the test for judging differences in performance, described above).

A notational *system* is a superset of a notational scheme that is "correlated with a field of reference" (Goodman 1976, 143). What this means is that the notational system has some certain "compliance-class," or "extension," or simply "performance" that it is *related* to, or "complies" to. An additional requirement of notational systems is that they cannot be ambiguous or have "different compliants at different times or in different contexts" (145). This unambiguity extends from the marks, utterances, or inscriptions to the notations themselves (so, for example, both the musical performance and the notes in the score must be unambiguous). Moreover, notational systems must be semantically disjoint and semantically finitely differentiated. This means that the compliance-class, or performance, must also be disjoint and finitely differentiated.

These semantic requirements for inclusion in the set of notational systems, are, in fact, remarkably high bars for inclusion. Whereas most familiar notations, such as the alphabet, binary, or musical notation satisfy the criteria for notational schemes, many of these schemes do not (typically) have *performances* that meet the stipulated semantic requirements. For instance, when the alphabet is used for natural language (in the sense that the alphabet inscribes patterns that construct words referring to, or representing, some objects in the world), the alphabet in this case is only a notational *scheme* (composed of disjoint and finitely differentiated symbols).

For use as a carrier of natural language, the alphabet's performance fails to meet the criteria of being semantically disjoint and semantically finitely differentiated. Consider Goodman's example: a notational system cannot contain both "doctor" and "Englishman" (as the English language plainly does), because in a notational system you cannot permit the existence of a single object that satisfies the *meaning* (semantics) of both "doctor" and "Englishman" (Goodman 1976, 152). In addition to their being semantically *disjoint* (not doctor and Englishman), you must also be able to, in theory, determine if the meanings are *differentiated*. If the effort to determine differentiation between marks or utterances is ever finer and more precise, with no theoretical limit to be able to conclusively determine compliance of performance, then the marks fail the requirement of being a notational system. For example, if the test for differentiation between lines drawn on a page requires ever more powerful microscopes to zoom in further and further, with no way to ultimately determine if a mark or performance is of one type or another, then the notation fails the criteria of semantic finite differentiation (there is no in-theory test to determine if a given mark represents one type or another). Only systems that have tests that can determine differentiation in actual fact, and do not have objects shared among characters or classes, can count as notational systems.

Goodman summarizes the requirements of a notational system thus: "A system is notational, then, if and only if all objects complying with inscriptions belong to the same compliance class, and we can, theoretically, determine that each mark belongs to, and each object complies with inscriptions of, at most one particular character" (1976, 156).

From Notational Systems to Notational Technologies

Today, we are in general remarkably aware of the role of code. It is becoming increasingly necessary to have at least a working vocabulary of the distinctions between binary, machine code, and software code, and we increasingly appreciate that computing technologies work with "encoded" materials. We are less familiar, however, with the idea that these special codes count as a kind of writing. Accordingly, we often fail to grasp the parallels and disjunctures between, say, English alphabetic writing and the Javascript programming language. We assume an ontological chasm between alphabetic writing (meaningful, for humans) and Javascript (executable, for computers). To keep to the same example, we also typically assume that there is a straightforward and unproblematic translation between Javascript and binary (ostensibly the "language" used by computers). None of these distinctions is clear or uncontested; rather they require, for starters, a good definition and delimitation of language itself.

Notational schemes draw a continuum between the inscriptions that are needed for natural language, source code, and compiled binary but do nothing to dissolve the complexity of language. Instead, notations work as though by fiat: these systems constrain the domain of use to performances that are semantically isomorphic with their symbols or characters. This is distinct from natural language, but not radically so. People like Francis Bacon and John Wilkins, and many others during the sixteenth and seventeenth centuries' philosophical and universal language planning activities, imagined a perfect isomorphism between systems of writing and the essence of things. These projects failed—as languages—although they were important precedents for the kinds of representational and notational machines that later succeeded.

I also draw a distinction between what we typically call "computers" and what I am here calling notational technologies. Computers are famously difficult to pin down philosophically. There are many opinions about the substance of computers and computation. Beyond internal debates, in the early days of artificial intelligence and cognitive science, for example, this was a persistent topic of discussion. Consider the work of cognitive science in the twentieth century as explored by John Searle (1980), Hubert Dreyfus (1972; 1992), Marvin Minsky (1982), and Fred Dretske (1997), who each drew rather different connections and disjunctions between the workings of human brains and machines (for a critique and summary, see Smith 2010). Or consider some less philosophically plausible but nonetheless popular ideas about computers: the computer as formal symbol manipulator, Turing machine, information processor, or state machine. Not one of these ideas has gained any philosophical consensus, and perhaps one of the key reasons is because computers can be and are used for so many different things and have so many capabilities.

I have developed the neologism "notational technologies" to refer to the set of technologies *broader* than contemporary computers, which includes all technologies that entail the performance of notation for their characteristic use—such as telegraph systems, scored music, encryption devices, and all digital technologies. This construal focuses on the notational dimensions of these technologies and does not attempt to explain their richness and multiple, varied uses. For example, the ability to perform mathematical calculations is an extremely important dimension of contemporary computing, and in many respects this is why computers are interesting and powerful today. But calculating has little to do with notation and therefore is beyond the scope of my construal (although it should be noted that mathematics has its own history of notation, which is a critical if underappreciated dimension of its development [see Wolfram 2000]). By focusing on notation, and especially the ways in which notation is wrapped up with the ability to abstract away properties and form new, artificial identities, the role of control and management by sophisticated algorithms is brought into sharp relief and ought to be part of an informed social critique of contemporary computing.

Goodman's theory of notational systems offers a description of the ways that notation can be activated in performance. This is an activation or *execution* of a specially designed writing system: the performance of a notational scheme to produce abstract entities (again, abstract in the sense of complying with a socially acceptable test, not in a Platonic or pure sense). In practical terms, in order for these systems to produce a stable (and isomorphic) identity out of the froth of the world, they must manage complexity and eliminate confusion. Haugeland calls this the "primary motivation ('payoff')" for designing computers (1981, 219). The apparatuses necessary for notational technologies, therefore, must be, to borrow Haugeland's terminology, positive and reliable.

Consider the naive materialist: she is constantly dealing with the froth of the world and sees computers as mere mechanisms. Ideally, she will be willing to admit that a great deal of work—in the sense of societies and institutions—goes into computing technologies to stabilize materially complex electromagnetic pulses as they run through complex systems of control circuity. Overcoming the many challenges to make all of these physical properties work together is the study of computer engineering.

This picture of materiality, however, is bereft of the reasoning as to why certain complex electromagnetic pulses are deemed <1> or <0> (to focus on the conventional representation of binary), or why other electromagnetic pulses are deemed noise. Goodman's gesture with notational systems provides a plausible answer: inside a contemporary computer is a socially constructed and ultimately arbitrary, yet reliable and positive, test for *compliance* between the material performance and the intended notation. This test of compliance is not just material but rather draws from the interplay between meaning (symbol) and mechanism (the mechanical performance of digitality). This interplay is realized, for example, when the computer symbolizes <1> (or "on," or "high," and so forth): the symbol complies with a particular electromagnetic pulse, and vice versa. The test to determine whether a given performance complies with the notation is key, and is what separates mere notation, in the sense of an abstract ideality, from notational *systems*. Of course, in real working systems edge cases may occur where it is difficult to determine whether a particular electromagnetic pulse is <1> or <0>, but, so long as there is an in-principle test to determine the compliance-class between electromagnetic pulse and notation, the performance (say, the running calculation) therefore meets the criteria for a notational system.

The interplay of performance and notation going on inside a computer is exactly parallel to the performance of music and its corresponding notation. When a given musical note (a notational scheme) is read and then played on a musical instrument, it produces a certain acoustic vibration. This vibration—the performance—has a test of compliance, as per the notational system. For example, a certain emitted sound vibration corresponds to (or complies to) a middle C note. The musical instrument and the computer each have certain "programmings" that they must follow. The difference is that the instrument produces acoustic vibrations while the computer produces electromagnetic pulses. At the level of notation, therefore, the performance of a musical score is identical to the "performance" of binary instructions.

For uses that are characterized by their notationality, the same semantic isomorphism holds between the generated symbol and the representation. That is, there is a conceptual isomorphism between performances and representations, or a positive and reliable test of compliance. This is why when we hear a note played on a piano we can, if sufficiently trained and attuned, identify its notation; or this is why the parcel sitting in the back of a delivery van corresponds to, or is tracked by, a database entry in logistics software. The compliance between an electromagnetic wave and a single binary bit must carry "upward," into the world, and back down too. Complex systems such as blockchain platforms are notational in the same way, in that there are tests of compliance between their multifaceted objects and their notational origins. There are, of course, a great number of levels of transformation between each stage, including recording, ordering, permuting, transcribing, and reencoding, but notational technologies are powerful precisely because they hold the chain of transformation stable across these many semantic bounds. They do so by abstracting complexities and (arbitrarily) constructing socially accepted tests of compliance.

Blockchains as Notational Technologies: The Example of Monegraph

Blockchain technologies are one of the clearest examples of notational technologies: they rely on notation to abstract properties to form identities because the object and representation are isomorphic (in the sense that there is a test of compliance for the notational scheme, its system, and its semantic representation in the world).

My construal of blockchain technologies departs from the typical characterization of blockchains. In the technical literature, blockchain technologies are usually described as Turing machines or state machines.

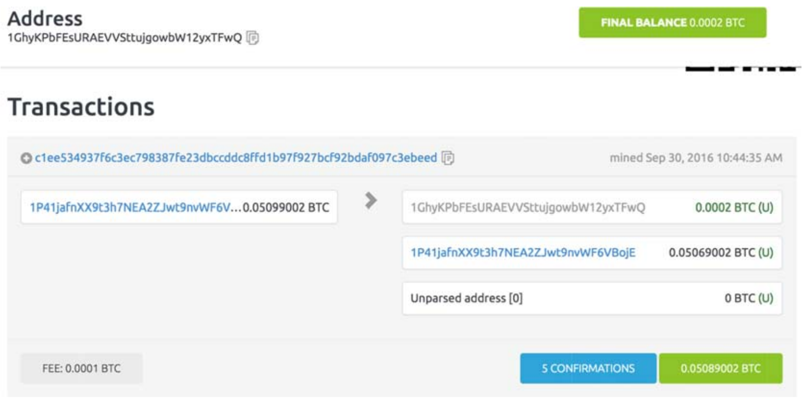
Blockchain technologies such as Ethereum are said to be "Turing complete," in the sense that they can be programmed to effectively simulate any other machine (that is, they are "universal"). (In fact, Ethereum is not technically Turing complete, due to the necessity for a finite supply of ether "gas" to run computations.) Similarly, blockchain technologies are said to be state machines because the ledger maintains state, usually transaction inputs and outputs (or optionally, code and data). These descriptions, however, fail to explain the relationship between ledger and application, or how blockchains work. To better understand how and why blockchain technologies are so effective at managing and controlling digital assets, and then subsequently controlling their "real world" counterparts, I shall briefly introduce and explore the example of Monegraph.

Monegraph is, at its core, a digital asset registry using blockchain technology. In the few short years since its launch, it has grown into, or at least has aspirations to be, a "complete monetization platform" ("Monegraph" 2016).[1] Monegraph allows copyright holders to register and manage assets, which includes the ability to track changes to the asset, set contractual parameters on the object, and manage the sale and distribution of the object's rights. Currently, the Monegraph platform only supports a small range of digital image formats focused on digital art, but in principle the system could be extended to any digital file. Such rights are represented by a digital "fingerprint" of the digital object (a SHA-256 hash signature) and a rights contract hash (the contract is written in the Open Digital Rights Language, or ODRL, a "rights expression" language), which is subsequently stored on the blockchain.

When it was originally conceptualized and launched by Anil Dash and Kevin McCoy (at the Rhizome-sponsored 2014 "Seven on Seven" event), Dash and McCoy considered using the Namecoin fork of the Bitcoin blockchain, but they later settled on the mainline Bitcoin blockchain for implementation. Although bitcoin was originally conceived as a digital currency, bitcoin developers and enthusiasts quickly realized that the underlying decentralized ledger technology could store arbitrary data, making for a public, immutable, and censorship-resistant ledger, or database. (One technical restriction is that in order to keep the blockchain of a manageable size—since all transactions are stored on it—the amount of data that can be stored is very limited, which is why Monegraph only stores hash data on the Bitcoin blockchain.) The Bitcoin blockchain (and therefore the Monegraph hash data) is replicated across a large system of "mining" computers, ensuring that (so long as the miners continue to mine) no data can be lost or censored by a single actor (miners are economically

---

[1] For the purposes of research, I have registered an account with the Monegraph platform, and I maintain a small fund of bitcoins and other cryptocurrencies solely for the purpose of experimentation and research. I have no financial connection with Monegraph, Inc.



Address
1GhyKPbFEsURAEVVSttujgowbW12yxTFwQ

FINAL BALANCE 0.0002 BTC

Transactions

c1ee534937f6c3ec798387fe23dbccddc8ffd1b97f927bcf92bdaf097c3ebeed

mined Sep 30, 2016 10:44:35 AM

1P41jafnXX9t3h7NEA2ZJwt9nvWF6V...0.05099002 BTC

1GhyKPbFEsURAEVVSttujgowbW12yxTFwQ 0.0002 BTC (U)

1P41jafnXX9t3h7NEA2ZJwt9nvWF6VBojE 0.05069002 BTC (U)

Unparsed address [0] 0 BTC (U)

FEE: 0.0001 BTC

5 CONFIRMATIONS

0.05089002 BTC

FIGURE 1. Blockchain explorer representation of hash signature of image stored on bitcoin blockchain with Monegraph platform [color figure can be viewed at wileyonlinelibrary.com]

incentivized to mine through bitcoin "minting" and transaction fees, and they require *significant* collusion to censor or attack the blockchain data). In addition, due to the cryptographic features of blockchain authentication and verification, all transactions are immutable, in the sense that transaction history cannot be changed and all valid transactions must be registered. One consequence of using the Bitcoin blockchain as a decentralized ledger is that all transactions can be publically verified, which also, at the time of writing, costs Monegraph Inc. between US$0.06 and US$0.45 in fees for each digital image registered (see figure 1).

Representations of digital objects are stored on the Bitcoin blockchain, and associated contractual (copyright) data are publically visible and verifiable. The object data (such as an image's JPG binary file), however, are not stored on the Bitcoin blockchain, which means, in principle, the actual item to be managed or sold could be hidden from view, while rights are managed publically (in practice, Monegraph enables its users to display a public "catalog" of work, which has an e-commerce component for customers to directly make purchases from). Therefore, quite a bit of copyright and sales flexibility is enabled by the Monegraph platform, including the ability to simply register a work (establishing provenance); sell exclusive or limited editions; and register Creative Commons–like rights, such as permitting resale or remixing. Rather unique features of the Monegraph platform include its ability to publically track and verify sales, track changes to the object or its licensing, and even (if the stipulated contract allows it), resell the digital object. In fact, the very promise of reselling a *digital* object is the raison d'être of Monegraph and is what makes the system so unique—

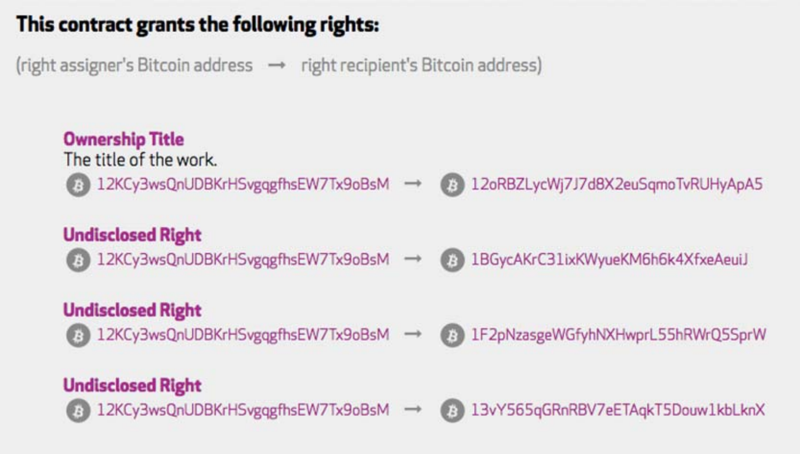
different from all the other kinds of content distribution, management, and sales platforms already available.

In an unusual twist, one of the supposedly greatest features of digital computers and the Internet—the ability to infinitely copy objects as though there were no material constraints—is here inverted: Monegraph makes digital images unique, and therefore precious (and therefore valuable), because it establishes new identities for digital objects *as though they were material.*

In a roundtable discussion sponsored by *Dis Magazine*, McCoy said, "[Blockchain technology] wasn't just a database, ... it was a transaction mechanism that had a conception of value" (Sacks et al. 2016). Moreover, he continued, "There was scarcity to it[:] ... this weird contradictory possibility of ubiquity and scarcity at the same time" (2015). In articulating the "weird contradictory" nature of ubiquity and scarcity, McCoy recognized that scarcity is an important dimension for creating value, yet, problematically, Internet-connected things tend toward ubiquity. Due to the possibility of infinitely copying digital files, assets are potentially ubiquitous (and not scarce) and therefore have little to no value (in the traditional terms of economic value). This interplay between scarcity and ubiquity is a consequence of the way Monegraph handles digital assets. As an example of a notational technology, performances of the digital art (in the sense that it can be viewed) are ubiquitous and therefore cheap (or even free), but the notational identity is scarce and therefore valuable.

When a digital file is registered with the Monegraph system, it is first "fingerprinted." This fingerprinting mechanism is a common cryptographic technique that produces a "hash." Hashing a digital file reduces a large binary representation to a small (fixed-size), unique representation. One of the key features of cryptographic hashing is that if even a single bit in the source binary file changes, the hashed product will vary widely and noticeably. This ability to highlight change is key to making the hash system practical, since rather than a series of files being slightly different, they are represented as radically unique. For example, while a slightly corrupted (or "glitched") JPG-compressed image may *look* (in viewer software) only slightly (or imperceptibly) different from its original, uncorrupted version, its hash signature will be completely unique (that is, its representation is without resemblance). In addition, the process works the other way too: it is possible to "authenticate" an image against its hash (but note that the notational system requires only compliance, not a sense of "authentic"). By running the hash function on a file, one can determine if it is identical by comparing the hashed output to the original hash data—where the hashes match, there is an assurance of identity.

Hash functions can guarantee that two files are identical, but for digital content management systems this is nothing new. To make a



This contract grants the following rights:

(right assigner's Bitcoin address → right recipient's Bitcoin address)

Ownership Title
The title of the work.
₿ 12KCy3wsQnUDBKrHSvgqgfhsEW7Tx9oBsM → ₿ 12oRBZLycWj7J7d8X2euSqmoTvRUHyApA5

Undisclosed Right
₿ 12KCy3wsQnUDBKrHSvgqgfhsEW7Tx9oBsM → ₿ 1BGycAKrC31ixKWyueKM6h6k4XfxeAeuiJ

Undisclosed Right
₿ 12KCy3wsQnUDBKrHSvgqgfhsEW7Tx9oBsM → ₿ 1F2pNzasgeWGfyhNXHwprL55hRWrQ5SprW

Undisclosed Right
₿ 12KCy3wsQnUDBKrHSvgqgfhsEW7Tx9oBsM → ₿ 13vY565qGRnRBV7eETAqkT5Douw1kbLknX

FIGURE 2. Details of an entity's rights contract stored in hash format on Bitcoin blockchain using Monegraph system [color figure can be viewed at wileyonlinelibrary.com]

digital file both "authentic" and provably "unique" (in a legal sense), and therefore bound by real identity constraints, Monegraph registers the file's hash signature on the Bitcoin blockchain, which can then be publically verified (and cannot be erased). So, if I sell a digital file to you, once the hash signature has been registered by Monegraph on the Bitcoin blockchain, you (or anyone else, such as a future buyer) can verify legal possession of the file by comparing your hash signature with your blockchain address (which is actually a cryptographic public/private key pair managed by Monegraph) to the one on the public blockchain (see figure 2). In the specific design of Monegraph, the file that you possess may not actually be unique, in the sense that you are the only person in the world possessing the material inscription of that unique set of bits (because the file you download from Monegraph is an "original" unchanged version), but you can *prove* legal ownership. It is precisely the ability to prove legal ownership—a kind of social compliance test against the hash signature—that gives the Monegraph platform its unique capability.

Therefore, Monegraph enters into the political economy of copyright and its troubled interaction with the digital. However, the set of copyright tools enabled by the system might be insufficient to actually protect artists, as Martin Zeilinger (2016) has pointed out. What is more substantial, though, is that Monegraph has conceptually narrowed the possibilities of the digital, especially with respect to digital arts. According to Zeilinger, the Monegraph platform fails to realize the forms of resistance and critique that are implied by digital arts. For example, Zeilinger points to two exhibitions (*The Human Face of Cryptoeconomics* and *Neoliberal Lulz*) for evidence of the ways that the identity system produced through hash signatures can be used to create pure aesthetic value, instead of economic value (which is liable to be captured by typical capital flows). On the one hand, I am sympathetic to Zeilinger's critique, especially with respect to the ways that the Monegraph platform potentially enters objects into the circuitry of control and management. I have elsewhere critiqued the political economy of blockchains within the context of the control society (DuPont 2014). These concerns raise important ethical questions, beyond art and digitality, and need to be further explored. On the other hand, the ability to harness the constraints of identity—creating unique and manageable representations—affords interesting possibilities beyond the narrow scope of economics and copyright.

Conclusion and Discussion

In terms of notational technology, Monegraph is interesting because it is fundamentally a system for establishing the identity of some digital object (through the hash function), and then maintaining the technical and social infrastructure for managing and controlling that identity. With this infrastructure, there are an array of performances and compliance tests, from the most basic low-level bit and electromagnetic pulse compliance up to a robust system of hashes being compared to cryptographic public/private key pairs. That is, the complex interplay of meaning and mechanism results in a notational identity (a hash signature) that complies to, or is a performance of, a syntactically and semantically constrained digital image. The system relies on legal rights to buttress questions of intellectual property, but its ability to manage representations is provided by a technological system of identities, formed by the compliance between abstract representations, which are fundamentally notational.

In activating a notational system, many of the artistic and visual aspects that traditionally would make a digital image valuable are abstracted away, and the "essence" of the image then becomes the registered hash signature. One can think of the Monegraph system as either challenging Goodman's division between allographic and autographic arts or, as I prefer to see it, utilizing the allographic forms to socially overdetermine (through management and control) the autographic forms. This requires acceptance of the notational form, through social (and legal) apparatuses. Inherently, there is nothing "valuable" about a hash signature, but we imbue it with value because of its compliance with a deeper sense of identity—the identity of a work of art.

By managing identity, the digital file is then (potentially) made rare through its notation, which means it can be bought and sold like traditional (material) art, and can potentially command similarly high prices. Moreover, as the art is bought and sold, these transactions are represented (and tracked) on a publically verifiable blockchain, adding to the capabilities of identity and exchange.

Monegraph and other blockchain systems are not the only notational technologies (far from it), but they are interesting ones because they deal so centrally with the parameters of identity and notationality. Indeed, as blockchain platforms continue to develop in the coming years, the successful ones will realize the ways in which they are, fundamentally, trading on notational identities, and therefore will exploit this to their advantage. This capability permits the sophisticated control of objects, both digital and physical, in a wide range of applications.

In this article, I have argued that many so-called digital technologies can be construed as notational technologies. I have described notational technologies as the performance of syntactic notation within a field of reference, which is socially constructed. That is, a notational system is a constitutive determination as to whether the given notation renders the referent in an acceptably meaningful way. By understanding modern digital technologies and computing devices in terms of notation, the role of identity and identity management (or control) has been highlighted. The role of identity and identity management has been explored through the example of Monegraph, a blockchain-enabled art monetization platform. My construal is distinguished from typical characterizations of blockchains as either Turing machines or state machines. By focusing on the conceptual and practical affordances of notationality, the social embeddedness of blockchains is highlighted, but it remains a complex terrain still in need of further study, especially philosophical and ethical study.

UW Information School
Box 352840—Mary Gates Hall, Ste. 370
Seattle, WA 98195-2840
USA
isaac.q.dupont@gmail.com

References

DuPont, Quinn. 2014. "The Politics of Cryptography: Bitcoin and the Ordering Machines." *Journal of Peer Production* 1, no. 4:1–29.

DuPont, Quinn, and Yuri Takhteyev. 2016. "Ordering Space: Alternative Views of ICT and Geography." *First Monday* 21, no. 8 (July 23).

At http://firstmonday.org/ojs/index.php/fm/article/view/6724. (Last accessed August 25, 2017.)

Dretske, Fred. 1997. *Naturalizing the Mind*. Cambridge, Mass.: MIT Press.

Dreyfus, Hubert L. 1972. *What Computers Can't Do: A Critique of Artificial Reason*. New York: Harper and Row.

———. 1992. *What Computers Still Can't Do: A Critique of Artificial Reason*. Cambridge, Mass.: MIT Press.

Goodman, Nelson. 1976 [1968]. *Languages of Art: An Approach to a Theory of Symbols*. Second edition. Indianapolis: Hackett.

Haugeland, John. 1981. "Analog and Analog." *Philosophical Topics* 12, no. 1 (Spring): 214.

Santos, Steven J., Sue Wolf, Robin Shlien, David Grossman, and John Fortenberry. 2006. *Louis C.K.: One Night Stand*. HBO Studios.

Minsky, Marvin. 1982. "Why People Think Computers Can't." *AI Magazine* 3, no. 4 (Fall): 3–15.

"Monegraph." 2016. At https://monegraph.com/. (Accessed September 30, 2016.)

Searle, John R. 1980. "Minds, Brains, and Programs." *Behavioral and Brain Sciences* 3, no. 3:417–24.

Smith, Brian Cantwell. 2010. "Age of Significance." At http://www.ageofsignificance.org/. (Last accessed August 25, 2017.)

Sacks, Steven, Kevin McCoy, Zoë Salditch, Adriana Ramić, Saul Ostrow, and Mike Pepi. 2016. "Monegraph and the Status of the Art Object." *DIS Magazine*. At http://dismagazine.com/discussion/73342/monegraph-and-the-status-of-the-art-object/. (Accessed September 30, 2016.)

Wolfram, Stephan. 2000. "Mathematical Notation: Past and Future." In *MathML and Math on the Web*. Urbana-Champaign, Ill. At http://www.stephenwolfram.com/publications/mathematical-notation-past-future/. (Last accessed August 25, 2017.)

Zeilinger, Martin. 2016. "Digital Art as 'Monetised Graphics': Enforcing Intellectual Property on the Blockchain." *Philosophy and Technology* 30, no. 1:1–27.