A Unified Theory of Code-Connected Contracts

Carla L. Reyes*

Smart contracts and their promise of automatic performance capture legal and entrepreneurial imaginations. But the excitement around the technology led to some confusing legal responses. Several legal scholars use chronology to help reduce the confusion and place smart contracts within what is already familiar about computational contracting. According to this line of thinking, blockchain-based smart contracts simply represent the next technological advancement in a long history of computable contracting technologies. However, other scholarly work suggests that such a chronological explanation under-simplifies the nature of the linkages between smart contracts and other forms of code-connected contracts. This Article offers a unified theory of code-connected contracts as a tool for guiding risk allocation and design trade-off discussions when considering whether to use one or more forms of code-connected contracts. Specifically, this Article argues that the many variations of code-connected contracts should be viewed along an axis of state transition complexity. Doing so brings to the forefront the fact that the issues facing smart contracts used to merely automate performance of contractual obligations differ in terms of both magnitude and novelty from algorithmic decision-making tools used by parties to fill gaps in contractual terms and other computational contracting tools.

In other words, the state transition complexity theory of code-connected contracts set out in this Article offers an analytical tool for anticipating legal and business risk when using computational contracts. Ultimately, the state transition complexity theory of code-connected contracts demonstrates that getting to the core legal issues presented by code-connected contracting requires an analysis of the details of each specific implementation. As with most legal questions, proper analysis depends on facts and circumstances. Nevertheless, many of the core legal issues will arise because emerging technology, like all technology, is social technology. Thus, the implications of the state transition complexity theory of code-connected contracts are that many of the legal issues are not terribly new, and we should not forget to look to existing jurisprudence and scholarly work in contract law and corollary disciplines.

---

* Assistant Professor of Law, Southern Methodist University Dedman School of Law; Affiliated Faculty, Indiana University Ostrom Workshop on Cybersecurity and Internet Governance; Faculty Associate, University College London Blockchain Research Center. My thanks for tremendously insightful and helpful feedback to the participants of the Arizona State University Sandra Day O'Connor College of Law Governance of Emerging Technologies 2019 Conference and the Wayne State University School of Law 2019 Blockchain Symposium. I am grateful for the work of the kind, thoughtful, and diligent editors of the *Journal of Corporation Law*.

I. INTRODUCTION

The current hype-cycle around smart contracts encourages us to believe that contract law as we know it will soon end.[1] Marketers, code developers, and legal scholars alike see the potential for smart contracts to usher in a new era of perfect, automatic enforcement of contractual obligations.[2] Several legal scholars attempt to bring order to some of the hype-fed frenzy around smart contracts by placing them in the context of a longer history of computable contracting.[3] According to this line of thinking, blockchain-based smart

1. *See, e.g.*, Ameer Rosic, *Smart Contracts: The Blockchain Technology That Will Replace Lawyers*, BLOCKGEEKS, https://blockgeeks.com/guides/smart-contracts/ [https://perma.cc/T9MY-B5H6] ("[L]awyers will transfer from writing traditional contracts to producing standardized smart contract templates, similar to the standardized traditional contracts that you'll find on LegalZoom."); Gideon Greenspan, *Why Many Smart Contract Use Cases are Simply Impossible*, COINDESK (Apr. 17, 2016, 7:00 AM), https://www.coindesk.com/three-smart-contract-misconceptions [https://perma.cc/JG7V-URGN] ("When smart people hear the term 'smart contracts', their imaginations tend to run wild.").

2. For marketers and developers, *see, e.g.*, Rosic, *supra* note 1. *See also, e.g.*, Greenspan, *supra* note 1. For legal scholars, *see, e.g.*, Anthony J. Casey & Anthony Niblett, *Self-Driving Contracts*, 43 J. CORP. L. 100 (2017) (working toward new contract law theory for smart contracts—"self-driving contracts" that use big data and artificial intelligence to autonomously "fill[] in the contract details to achieve the parties' designated outcome."); Alexander Savelyev, *Contract Law 2.0: «Smart» Contracts as the Beginning of the End of Classic Contract Law* (Higher Sch. of Econ., Working Paper No. WP BRP 71/LAW/2016), https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2885241 [https://perma.cc/AUG3-4KZH] ("One of the most promising areas of implementation of Blockchain technology is using it for creating fully automated contracts—agreements, which are performed without human involvement.").

3. Kevin Werbach & Nicholas Cornell, *Contracts Ex Machina*, 67 DUKE L.J. 313, 320–24 (2017); Jeremy M. Sklaroff, Comment, *Smart Contracts and the Cost of Inflexibility*, 166 U. PA. L. REV. 263, 286–91 (2017).

contracts simply represent the next technological advancement in a long chronology of computable contracting technologies.[4] However, other scholarly work relating to the perceived drawbacks of using smart contracts in the context of relational contracting suggests that such a chronological explanation under-simplifies the nature of the linkages between smart contracts and other forms of code-connected contracts.[5] The lack of a unifying theory of code-connected contracts inflames the hype-cycle, obscures the true usefulness of smart contracts, confuses lawyers, and sows seeds of concern among legislatures and courts regarding existing law's ability to properly address any potential exceptional qualities of smart contracts. Instead of feeding the hype cycle by examining the legal enforceability of smart contracts under certain circumstances, lawyers should examine, first and foremost, the social context within which smart contracts reside.

All technology is social technology;[6] all technology is created, used, and applied within a broader social context.[7] Law is a technology.[8] Thus, law is a social technology—it is created, used, and applied within a broader social context. Contracts, as a sub-system of law,[9] therefore, represent a technology within a technology—they offer a tool for privately ordering the rules applicable to a specific social context, the transaction between the parties.[10] Code-connected contracts may layer another level of technology onto the legal technology stack for private ordering, but they continue to represent functional equivalents[11] to their low-tech contract counterparts. Unpacking that functional

4. Werbach & Cornell, *supra* note 3; Sklaroff, *supra* note 3.

5. Karen E. C. Levy, *Book-Smart, Not Street-Smart: Blockchain-Based Smart Contracts and the Social Workings of Law*, 3 ENGAGING SCI. TECH. & SOC'Y 1, 2 (2017); Jeffrey M. Lipshaw, *The Persistence of "Dumb" Contracts*, 2 STAN. J. BLOCKCHAIN L. & POL'Y 1, 8 (2019); Lauren Henry Scholz, *Algorithmic Contracts*, 20 STAN. TECH. L. REV. 128, 132 (2017).

6. Carla L. Reyes & Jeff Ward, *Digging into Algorithms: Legal Ethics and Legal Access*, 21 NEV. L.J. 325, 344–45 (2020).

7. *Id*.

8. John O. McGinnis & Steven Wasick, *Law's Algorithm*, 66 FLA. L. REV. 991, *passim* (2014). As McGinnis explains, "Law thus works necessarily in part as an information technology—a tool for the distribution of information to the world that may itself change through the infusion of more information from the world." *Id.* at 993. Further, "[h]umans are both creators and creatures of technology. Everything we do is vitally connected to the tools we develop, and the law is no different. Law itself is in part a tool and an information technology, but its effectiveness depends on the larger domain of material technologies in which it nests." *Id.* at 1000.

9. Law can be seen as a system, through the lens of systems analysis. Lynn M. LoPucki, *The Systems Approach to Law*, 82 CORNELL L. REV. 479, 481 (1997). "'Systems analysis' is a methodology developed in the fields of engineering, business information systems and computer programming specifically to manage complexity." *Id.* at 481. The term "system" is defined differently across many disciplines, yet, as one scholar notes: "[T]hese definitions share several common, persistent elements: A system has an objective or goal (in other words, it has defined work to be accomplished). A system contains multiple components (or Resources). A system's components work together, each performing defined functions, to enable the objective or goal to be achieved." JEFFERY RITTER, ACHIEVING DIGITAL TRUST: THE NEW RULES FOR BUSINESS AT THE SPEED OF LIGHT 133 (2015). A separate component of systems includes the rules that govern the system's behavior. *Id.* at 145. "Systems are composed of subsystems. Subsystems are themselves systems, which in turn have their own subsystems." *Id.* at 487. *See also* Tamara Belinfanti & Lynn Stout, *Contested Visions: The Value of Systems Theory for Corporate Law*, 166 U. PA. L. REV. 579, 599–600 (2018) (discussing the core forms and characteristics of systems).

10. Contracts are generally defined "as embracing all promises that the law will enforce." Alan Schwartz & Robert E. Scott, *Contract Theory and the Limits of Contract Law*, 113 YALE L.J. 541, 543 (2003).

11. Throughout my scholarship, I tend to think about "functional equivalents" in the comparative law sense of that term. *See* Michael S. Gal, *The Cut and Paste of Article 82 of the EC Treaty in Israel: Conditions for a*

equivalence enables more efficient use and interpretation of code-connected contracts and prevents the law from absorbing the technology hype-cycle and related misunderstandings.

This Article argues that the key to unpacking the functional equivalence between code-connected contracts and their traditional paper counterparts lies in understanding the linkages between approaches to risk mitigation in the traditional contract context and the management of state changes by code-connected contracts. In particular, this Article offers a theory of state transition complexity as a way to bridge the gap between technologists and lawyers in their understanding of different types of code-connected contracts and the trade-offs between computational contracting and traditional methods of contracting.

This Article unpacks the state transition complexity theory in three parts. Part I examines many of the variations of code-connected contracts and describes their treatment in the legal literature, pointing out that both the unrelenting focus on legal enforceability and the insistence on organizing different code-connected contracts along a time linear progression undertheorizes the role of smart contracts in use today. Indeed, these two emphases in the literature leave those using smart contracts—whether as part of a legally enforceable contract or as part of their products and services—without an analytical tool to help them assess the legal and business risks attendant to their activities. Part II attempts to reorient the discussion toward an analytical tool the legal field can use in conversation with those building code-connected contract systems by organizing the variations of code-connected contracts along an axis of state transition complexity. Part II argues that the state transition complexity theory of code-connected contracts reveals whether and how code—in the form of smart contracts, data-oriented or computable contract terms, or algorithms—serves as a functional equivalent to elements of traditional contracts. Perhaps more importantly, even, the state transition complexity theory makes clear where the code is not intended to serve a legally enforceable contract related function at all. Part III argues that when viewed through the lens of the state transition complexity theory, code-connected contracts are not terribly exceptional. Indeed, each time a new technological tool that might be used in code-connected contracts becomes available there is an industry-wide reconsideration of the same contract law questions from earlier iterations of the technology. Instead, the state transition complexity theory reorients the focus from the detailed elements of contract doctrine toward bigger lessons for lawyers operating at the intersection of law and emerging technologies.

II. THE MANY VARIATIONS OF CODE-CONNECTED CONTRACTS

Although the legal frenzy around smart contracts leads many to think otherwise, the

Successful Transplant, 9 EUR. J.L. REFORM 467, 469 (2007) (discussing the theories of transplanting laws from one jurisdiction to another). One of the core principles of the comparative functional method is that elements of a legal system must be considered in light of the function that they serve in responding to a societal problem. Ralf Michaels, *The Functionalism of Legal Origins*, *in* DOES LAW MATTER? ON LAW AND ECONOMIC GROWTH 21, 23 (Michael Faure & Jan Smits eds., 2011). This in turn requires recognizing that other systems may use functional equivalents to address the same societal or legal problem differently. Ralf Michaels, *Comparative Law by Numbers? Legal Origins Thesis, Doing Business Reports, and the Silence of Traditional Comparative Law*, 57 AM. J. COMP. L. 765, 778 (2009). In this approach, comparativists consider functional equivalents to include: "other institutions, legal or non-legal, that perform the same function . . .." *Id*. *See also* Tom Ginsburg, *Lawrence M. Friedman's Comparative Law, with Notes on Japan*, 5 J. COMP. L. 92, 102 (2010) ("[D]ifferent legal rules can play similar functions in different societies.").

term "smart contract" is not a new invention of the blockchain-age. Rather, Nick Szabo first introduced the concept of smart contracts in 1994.[12] Szabo's work contributed to long-standing discussions related to connecting contracts to code—making contracts computational. Scholars tend to use this history of code-connected contracts to argue that the contract law issues for smart contracts are not that different from other forms of electronic contracting. They also use the history of code-connected contracts to bring order around the technology by offering a time linear history of the many forms of electronic contracts. This Part briefly reviews computational law and the place code-connected contracts occupy in that field. It then provides a (very brief) primer on smart contracts (both in Szabo's sense and the blockchain-based smart contract sense) and concludes by highlighting some future trends in code-connected contracting. In its review of the code-connected contracting landscape, this section highlights several important lessons. First, although many assume parties use smart contracts in place of traditional contracts, many simply use smart contracts to achieve more efficient performance of traditional contractual obligations. Moreover, putting smart contracts into a timeline of technology developments does not help lawyers or clients assess which of the many variations of code-connected contracts might be most useful to adopt in light of business and legal risk considerations. Lastly, the lawyerly emphasis on the inflexibility of smart contracts is misplaced, particularly in light of the expected trajectory of the technology.

A. Earlier Forms of Code-Connected Contracts

A look at the historical development of law, from Greek and Roman law all the way up to the common law, reveals an intricate and longstanding relationship between the development of law and technology.[13] With the advent of modern computer science, scholars began exploring the ability to code legal systems and legal sub-systems.[14] This exploration led to the field of computational law.[15] Computational law, generally speaking, seeks to enable the application of computational techniques to law by representing law in formal logic.[16] The ultimate goal is to predict the consequences of law when it is applied

12. S. ASHARAF & S. ADARSH, DECENTRALIZED COMPUTING USING BLOCKCHAIN TECHNOLOGIES AND SMART CONTRACTS: EMERGING RESEARCH AND OPPORTUNITIES 45 (2017).

13. McGinnis & Wasick, *supra* note 8, at 1000–06.

14. *See generally, e.g.*, Anthony D'Amato, *Can/Should Computers Replace Judges?*, 11 GA. L. REV. 1277 (1977); Tom Allen & Robin Widdison, *Can Computers Make Contracts?*, 9 HARV. J.L. TECH. 25 (1996); William McGeveran, *Programmed Privacy Promises: P3P and Web Privacy Law*, 76 N.Y.U. L. REV. 1812 (2001); Anthony J. Bellia Jr., *Contracting with Electronic Agents*, 50 EMORY L.J. 1047 (2001); F. Allan Hanson, *From Key Numbers to Keywords: How Automation Has Transformed the Law*, 94 LAW LIBR. J. 563 (2002); Erik F. Gerding, *Contract as Pattern Language*, 88 WASH. L. REV. 1323 (2013) (all assessing the myriad of ways information technology may be poised to augment or supplant legal practice); Harry Surden, *Machine Learning and Law*, 88 WASH. L. REV. 87, 88 (2014) ("In a similar vein, this Article will suggest that there may be a limited, but not insignificant, subset of legal tasks that are capable of being partially automated using current AI techniques despite their limitations relative to human cognition."); Mark D. Flood & Oliver R. Goodenough, *Contract as Automation: The Computational Representation of Financial Agreements* (Off. Fin. Rsch., Working Paper No. 15-04, 2015), https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2648460 [https://perma.cc/8UDF-US99].

15. "Computational Law is that branch of legal informatics concerned with the codification of regulations in precise, computable form." Michael Genesereth, *Computational Law: The Cop in the Backseat*, STAN. L. SCH.: CODEX (Jan. 13, 2016), https://law.stanford.edu/2016/01/13/michael-genesereths-computational-law-the-cop-in-the-backseat/ [https://perma.cc/G576-PN5P].

16. *Id.*

to specific facts.[17] Interest in computational law stems from interest in enabling narrower legal rules, decreasing costs, and reducing risks.[18] Computational contracting occupies a unique place in the broader computational legal field as one of the longest standing branches of computational law.[19] Indeed, computational contracting, including through smart contracts, continues to capture legal academic interest.[20] Computational contracting assumedly suffers from two significant barriers preventing advancement and adoption: adjudication and language.[21]

In a leading article considering the problems of adjudication and language, Professor Harry Surden examined the potential for what he termed data-oriented and computable contracts.[22] As defined by Professor Surden, a data-oriented contract is one in which the parties "express[] some part of their contractual arrangement as computer-processable data."[23] Meanwhile, a computable contract exists when the parties have arranged for a computer to make automated, prima-facie assessments about compliance or performance.[24] Surden argued that data-oriented and computable contracts act as a form of legal transplant of contract law into code.[25] Namely, as the code reduces transaction costs by fulfilling the role of some law, then the effective scope of the law may change.[26] The effect of those

17. *Id.*; Jeffery Atik & Valentin Jeutner, *Quantum Computing and Algorithmic Law* 10, (Loy. L. Sch., L.A. Legal Stud., Rsch. Paper No. 201938, 2019), https://papers.ssrn.com/sol3/papers.cfm?abstract_id =3490930 [https://perma.cc/B7K8-WYAG].

18. Thorsten Kaeseberg, *The Code-ification of Law and Its Potential Effects*, 2 STAN. J. BLOCKCHAIN L. & POL'Y 232, 233 (2019); James Grimmelmann, Note, *Regulation by Software*, 114 YALE L.J. 1719, 1748–750 (2005); Surden, *supra* note 14, at 87; McGinnis & Wasick, *supra* note 8.

19. Inquiries into computational contracting can be identified in the academic literature as early as the 1950s. Megan Ma, *Writing in Sign: Code as the Next Contract Language?*, MIT COMPUTATIONAL L. REP. (Aug. 14, 2020), https://law.mit.edu/pub/writinginsign/release/1 [https://perma.cc/KZ93-QFF8].

20. *See, e.g.*, Harry Surden, *Computable Contracts*, 46 U.C. DAVIS L. REV. 629 (2012) (expressing why lawyers might prefer computable contracting), Werbach & Cornell, *supra* note 3, at 320 (stating that "in many cases, [smart contracts have] raised significant legal and policy questions"); Sholz, *supra* note 5, at 131 (asserting that "it is contract that gives algorithms the power to change our world by enabling businesses to exchange resources and services, with the law's power backing the transction"); Sudhir Agarwal et al., Toward Machine-Understandable Contracts, Paper Presented at the Workshop at the 22nd European Conference on Artificial Intelligence 1 (Aug. 30, 2016), http://www.ecai2016.org/content/uploads/2016/08/W2-ai4j-2016.pdf#page=5 [https://perma.cc/DB3S-L8J5] ("The impetus for . . . Computational Law . . . is a vision where computers or people with the help of computers, are able to rapidly understand the implications of contracts in their legal context.").

21. Ma, *supra* note 19, at 4 (citing Kingsley Martin, *Legal Technology Barriers—Understanding Language and Exercising Judgment*, THOMSON REUTERS: LEGAL EXEC. INST. (Sept. 24, 2015), https://legalexecutiveinstitute.com/legal-technology-barriers-understanding-language-and-exercising-judgement/ [https://perma.cc/D8JZ-JP7D]).

22. Surden, *supra* note 20, at 635–36.

23. *Id.* at 635–36.

24. *Id.* at 636.

25. *Id.* at 696–98.

26. *Id.* at 698–99 ("To the extent that laws or justifications rest upon assumptions of transaction costs associated with mass-contracting based upon prevailing levels, and to the extent that technological advances allow for computing technology to reduce certain transaction costs broadly, the relative substantive scope of legal doctrines may alter as transaction costs levels change. . . . When transaction costs levels change, the scope of a law can change even if the doctrine or text remains constant. Thus, any technology which broadly changes prevailing transaction costs levels—such as computable and data-oriented contracting—can potentially change the substantive scope of even seemingly unrelated laws.").

changes may be beneficial, according to Surden, so long as parties elect the use of data-oriented and computable contracts with an understanding that they make a contractual design trade-off: receiving *ex ante* certainty in exchange for a measurable loss in *ex post* flexibility.[27] Surden also cautioned that the appropriate use of either a data-oriented or computable contract would depend upon the nature and subject of the contract the parties seek to automate.[28] Thus, from very early on in computational contracting literature, it is clear that, like any contract, the appropriate mix of computable contracts, data-oriented contracts, and traditional natural language contracts depends upon the social context in which the contract will be put to use.

B. Today's Code-Connected Contracts: Smart Contracts

At its most basic, a smart contract is computer software that causes something to happen upon the fulfillment of pre-determined conditions.[29] In other words, smart contracts are computer code that says "if data is received that X has occurred, Y will execute."[30] In this sense, smart contracts are quite passive.[31] Smart contracts cannot reach out to find data evidencing an event, "x," has occurred.[32] Rather, the smart contract must be triggered (i.e., sent a signal) that an event, "x," has occurred.[33] The signal that triggers execution of the smart contract, certifying that "x" has occurred, can be internal to the blockchain (i.e., coming from other smart contracts), or the smart contract can receive the signal, and the data specific to it, from an outside source.[34] This concept of smart contracts expands on

27. Surden, *supra* note 20, at 681 ("[T]his article is not *advocating* for the adoption of *ex-ante* efficiency and computable terms over other considerations. Rather, it is noting that there is a genuine tradeoff in terms of *ex-post* flexibility, accuracy, and ability to take into account other valuable and relevant information.").

28. *Id.* at 682 ("The computable contracting approach is limited to a subset of contracting scenarios. Because the framework uses automated comparisons to assess compliance, it is implicitly limited to contexts in which such straightforward evaluations are sufficiently useful to meet the contracting needs of the parties.").

29. HENNING DIEDRICH, ETHEREUM: BLOCKCHANIS, DIGITAL ASSETS, SMART CONTRACTS, DECENTRALIZED AUTONOMOUS ORGANIZATIONS 167 (2016) ("A smart contract is decentralized code that moves money based on a condition.") [hereinafter DIEDRICH, ETHEREUM]; ARVIND NARAYANAN ET AL., BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES: A COMPREHENSIVE INTRODUCTION 264–65 (2016) ("In Ethereum, a contract is a program that lives on the block chain. Anybody can create an Ethereum contract, for a small fee, by uploading its program code in a special transaction. This contract is written in bytecode and executed by a special Ethereum-specific virtual machine, usually just called "EVM." Once uploaded, the contract will live on the block chain. It has its own balance of funds, other user can make procedure calls through whatever API the program exposes, and the contract can send and receive money.").

30. Richard Gendal Brown, *A Simple Model for Smart Contracts* (Feb. 10, 2015), https://gendal.me/2015/02/10/a-simple-model-for-smart-contracts/ [https://perma.cc/2RLK-XE8P] ("A smart contract is an event-driven program, with state, which runs on a replicated, shared ledger and which can take custody over assets on that ledger.").

31. ANDREAS M. ANTONOPOULOS & GAVIN WOOD, MASTERING ETHEREUM: BUILDING SMART CONTRACTS AND DAPPS 128–29 (2018) ("All smart contracts in Ethereum are executed, ultimately, because of a transaction initiated from an EOA. A contract can call another contract that can call another contract and so on, but the first contract in such a chain of execution will always have been called by a transaction for an EOA.").

32. *Id.* at 129 ("Contracts never run 'on their own' or 'in the background.'").

33. *Id.* ("Contracts effectively lie dormant until a transaction triggers execution, either directly or indirectly as part of a chain or contract calls.").

34. DIEDRICH, ETHEREUM, *supra* note 29, at 167–70. When smart contracts receive data from outside sources, those outside sources are often referred to as "oracles." WILLIAM MOUGAYAR, THE BUSINESS BLOCKCHAIN: PROMISE, PRACTICE, AND APPLICATION OF THE NEXT INTERNET STRATEGY 43 (2016) ("Oracles

the vision of smart contracts introduced by Nick Szabo.[35] For Szabo, the goals for using smart contracts were limited "to satisfy[ing] common contractual conditions (such as payment terms, liens, confidentiality, and even enforcement), minimiz[ing] exceptions both malicious and accidental, and minimiz[ing] the need for trusted intermediaries."[36] Although many fixate on the common description of smart contracts as "self-executing," their core function centers on performance—performance of obligations imposed elsewhere, via contract or statute.[37] Smart contracts incentivize performance, rather than obliterate enforcement mechanisms.[38]

What becomes clear from even this brief study of the nature of smart contracts is that the word contract is not used in the legal sense of legally enforceable contract. Rather, smart contracts encompass a far greater range of computer programs running on blockchain technology.[39] Although some smart contracts *may* intend to create or represent a legally enforceable contract, generally speaking, smart contracts are not by default "contracts" in the sense of the legal term of art referring to offer, acceptance, and consideration.[40] Considered in this light, smart contracts could be used as the program that enables the data-oriented contract term or that performs the computable contract term after other programming has rendered a decision within Professor Surden's framework of data-oriented and computable contracts. The point here is simply that smart contracts can be

---

are data sources that send actionable information to smart contracts.").

35. ASHARAF & ADARSH, *supra* note 12.

36. Nick Szabo, *Smart Contracts: Building Blocks for Digital Markets* (1996), https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html [https://perma.cc/66RQ-UCE2].

37. HENNING DIEDRICH, LEXON BIBLE: HITCHHIKER'S GUIDE TO DIGITAL CONTRACTS 47 (2019) [hereinafter DIEDRICH, LEXON BIBLE] ("Playing to the strengths of blockchain smart contracts requires a focus on providing incentives for performance rather than emphasizing legal remedies in the event of a breach.").

38. HENNING DIEDRICH, LEXON DIGITAL CONTRACTS 6 (2020) ("Digital contracts, in so far as they are blockchain smart contracts, cannot coerce any action. They can send money and log statements. They cannot otherwise force anyone to do anything. They typically operate on incentives instead and utilize staking to broaden the applicability of this principle: you pay something in that you will lose if you don't perform your role.").

39. ASHARAF & ADARSH, *supra* note 12, at 48; DIEDRICH, ETHEREUM, *supra* note 29, at 169; DIEDRICH, LEXON BIBLE, *supra* note 37, at 20.

40. CARDOZO BLOCKCHAIN PROJECT, "SMART CONTRACTS" & LEGAL ENFORCEABILITY 4 (Oct. 16, 2018), https://cardozo.yu.edu/sites/default/files/2020-01/smart_contracts_report_2_0.pdf [https://perma.cc/YE4X-7J2J] ("Currently, the use of smart contracts in the context of legal arrangements, falls on a spectrum."); J. DAX HANSEN ET AL., MORE LEGAL ASPECTS OF SMART CONTRACT APPLICATIONS 6 (Oct. 2018), https://www.perkinscoie.com/images/content/1/9/v5/199672/2018-More-Legal-Aspects-of-Smart-Contract-Applications-White-Pa.pdf [https://perma.cc/PW4T-UZBY] ("While the analysis of smart contracts as varying forms of legal contracts offers both useful and productive insights into the changing legal landscape, many of the current use cases for smart contracts involve proprietary platforms offered in the manner of software as a service, and do not purport to serve as a proxy for a traditional legal contract."). Many have taken to making this distinction in smart contract terminology itself, using the term "smart contract" or "smart contract code" to refer to computer programs that perform Y upon confirmation that X occurred, and using "smart legal contracts" or "programmable legal contracts" to refer to smart contracts that fulfill the requirements of a legally enforceable contracts. For a more nuanced approach to terminology at the intersection of smart contracts and legally enforceable contracts, *see* DIEDRICH, LEXON BIBLE, *supra* note 37, at 20–22 (introducing a taxonomy of "digital contracts," including: "digitally enhanced" contracts—"a legally enforceable contract is in part automated by a smart contract;" "digitally expressed" contracts—the smart contract is the legally enforceable contract;" and "digitally produced" contracts—"a smart contract running on the blockchain initiates a multitude of legally enforceable contracts, one with each person interacting with the smart contract.").

used for any number of things, as can data-oriented and computable contracts.

Indeed, many of the most promising use cases for smart contracts focus on their use as tools to automate or incentivize performance of contractual or regulatory obligations, to build the business logic of a decentralized venture, or to build back-end software for applications.[41] For example, OpenLaw developed a platform that allows parties to create an employment agreement (in words) with embedded smart contracts that automate the performance of the employer's payment obligations, including the payment of payroll taxes.[42] Further, the bank consortium R3 used its proprietary distributed ledger software, known as Corda, to create regulatory technology ("RegTech") for banks in England.[43] Working with England's banking regulator, the FCA, R3 used Corda to create an automated program for compliance with reporting requirements related to mortgage transactions.[44] The smart contract-based application allowed the FCA to view the compliance data in real-time.[45] In both examples, the parties used smart contracts to perform some obligation. In the OpenLaw context, the employer used the smart contract to perform its payment obligations to the employee for work completed and to perform its legal obligations with respect to paying payroll taxes to the Internal Revenue Service. Meanwhile, banks used Corda to perform its obligations as functionally regulated entities dealing in money and monetary instruments. In other words, smart contracts enable automatic performance of contractual obligations, regulatory obligations and other activities.[46]

Despite these rather concrete examples of smart contracts as tools for automatic performance of underlying contractual obligations, the hype around smart contracts would have us believe that contract law as we know it will soon end.[47] The reality, as Professors Kevin Werbach and Nicholas Cornell explain, is that "promise oriented disputes and grievances will not disappear, but their complexions will shift."[48] Parties will worry less about scenarios requiring significant *ex post* flexibility, and instead will engage in disputes

41. *See, e.g.*, Lin Pang & Guy E. Glynn, *Wyoming Takes a Step Ahead to Clarify the Legal Status of Decentralized Autonomous Organizations*, DLA PIPER: PUBLICATIONS (Mar. 22, 2021), https://www.dlapiper.com/en/us/insights/publications/2021/03/wyoming-takes-a-step-ahead-to-clarify-the-legal-status-of-decentralized-autonomous-organizations/ [https://perma.cc/X9AC-M82B] (describing smart contract based business entities and a new Wyoming law designed to accommodate them); Carla L. Reyes, *Creating Cryptolaw for the Uniform Commercial Code*, 78 WASH. & LEE L. REV. (forthcoming 2021) (developing smart contract based back-end software for the Article 9 filing system).

42. Aaron Wright, *OpenLaw—Code as Law—Tax Law Demo*, YOUTUBE (May 28, 2018), https://youtu.be/HPbgR4gG_4E [https://perma.cc/KM9K-YV4S].

43. Stan Higgins, *The U.K.'s Chief Financial Markets Regulator Has Helped Develop a New App Using Blockchain Consortium R3's Corda Distributed Ledger Platform*, COINDESK (Sept. 12, 2017, 3:00 AM), https://www.coindesk.com/uk-financial-regulator-builds-blockchain-app-r3s-corda [https://perma.cc/2QNS-8JLA].

44. *Id.*

45. *Id.*; *see also* Maria Terekhova, *FCA and R3 Develop Blockchain-Based RegTech Solution*, BUS. INSIDER (Sept. 13, 2017, 8 :44 AM), https://www.businessinsider.com/fca-and-r3-develop-blockchain-based-regtech-solution-2017-9 [https://perma.cc/ZVQ6-K9VD].

46. Indeed, by default, smart contracts are not legally enforceable contracts. They can be legally enforceable, but they are not designed to be legally enforceable as a default matter.

47. Werbach & Cornell, *supra* note 3, at 315 ("Enthusiasts of various stripes believe that smart contracts offer the potential to displace the legal system's core function of enforcing agreements.").

48. *Id.* at 318.

over whether the smart contracts adequately performed under *ex ante* specifications and expectations.[49] As a result, Werbach and Cornell view smart contracts as simply the latest in the technological evolution of contracting.[50] This time-linear progression views the starting point as electronic contracting, which evolved to data-oriented contracting, which then evolved to computable contracts, now culminating in smart contracts.[51] Indeed, because smart contracts could enable automatic performance of data-oriented or computable contracts, it is tempting to think of them chronologically as the next technological development in the history of computational contracting.[52]

However, other scholarly work relating to the perceived drawbacks of using smart contracts for relational contracting suggests that such a chronological explanation under-simplifies the nature of the linkages between smart contracts and some of its code-based predecessors. For example, Jeremy Sklaroff argues that the inherent lack of flexibility in smart contracts represents a significant drawback of using smart contracts as legally enforceable agreements.[53] Specifically, in natural language contracts, parties use standards like "commercially reasonable efforts" to allow for flexibility in performance obligations when the parties expect change in the broader context of the transaction or in the relative position of the parties over the duration of the contract term.[54] Parties use such standards to minimize *ex ante* costs of drafting and negotiating precise terms.[55] Parties also use such standards to enable enforcement flexibility *ex post* and thereby reduce anticipated litigation costs.[56] Such flexibility is less possible to achieve using single smart contracts than using data-oriented or computable contracts.[57] As such, the narrative that smart contracts simply represent next-generation computable contracts falls short of fully explaining the relationship of smart contracts to its predecessor code-based contracts.

---

49. *See id.* at 364 (detailing this scenario); *see also* Maren K. Woebbeking, *The Impact of Smart Contracts on Traditional Concepts of Contract*, 10 J. INTELL. PROP., INFO. TECH. & E-COMM. L. 105, 108 (2019) ("In a nutshell, with smart contracts the drafting stage of the contract ex ante, leading to an automatic execution, will become more important than subsequent law enforcement ex post.").

50. Werbach & Cornell, *supra* note 3, at 320–24.

51. *Id.* at 322 ("The evolution from electronic, to data-oriented, to computable contracts embodies a trend toward greater machine autonomy."). Professors Werbach and Cornell are not alone in their view of smart contracts as fitting into a time linear progression of electronic contracting. Jeremy Sklaroff also views smart contracts as the next evolution of electronic contracting, an evolution starting with electronic data interchange ("EDI"). Sklaroff, *supra* note 3, at 278.

52. Werbach & Cornell, *supra* note 3, at 324 ("Szabo's vision, the full automation of forming and performing contracts, was ahead of its time. . . . The development that made Szabo's vision of smart contracts more than a mere curiosity was Bitcoin, a digital currency not reliant on governments, banks, or other intermediary institutions.").

53. *See* Sklaroff, *supra* note 3, at 263, 279–80 (demonstrating how traditional semantic contracts allow parties to easily change terms and avoid high drafting and renegotiation costs); *see also* Levy, *supra* note 5, at 1 ("The technology of smart contracts neglects the fact that people use contracts as social resources to manage their relations. The inflexibility that they introduce, by design, might short-circuit a number of social uses to which law is routinely put.").

54. Sklaroff, *supra* note 3, at 281.

55. *Id.* at 282–84.

56. *Id.* at 284–86.

57. Werbach & Cornell, *supra* note 3, at 123–24 (achieving smart contract flexibility requires trade-offs in efficiencies and decentralization); Surden, *supra* note 20, at 681 ("[T]here is a genuine tradeoff in terms of ex-post flexibility, accuracy, and ability to take into account other valuable and relevant information.").

C. Newer Inquires in Light of Industry Developments

Recent interest in several high-profile uses of smart contracts triggered new proposals for understanding the relationship between the smart contract code and natural language contract. For example, the use of smart contracts to create tokens used in initial coin offerings ("ICOs")[58] prompted consideration of whether the smart contract could form part of a "contract stack" together with related natural language promises that courts can rely on to "make sense of these new forms of commercial exchange."[59] At least one scholar argued that contract law must adapt to directly address "algorithmic contracts": "contracts in which one or more parties use an algorithm to determine whether to be bound or how to be bound."[60] Another inquiry undertaken by legal scholars considers theories of "code-deference," an approach to limiting the risk of using smart contracts as tools for building legally enforceable code-connected contracts by requiring parties to agree in advance to be bound to the outcome of smart contract execution.[61] Finally, the potential for linking data-oriented contract terms, including those powered by smart contracts, with artificially intelligent sources of data that triggers their execution, generates interest at the intersection of smart contracts and personalized law.[62] These approaches continue to consider how courts might enforce, or not, contracts at least partially memorialized in code. It remains unclear what linkages can be said to exist between forms of code-connected contracts, or how those using one form of code-connected contract or another (or several combined) should consider the relative risk among code-connected contracting techniques.

III. UNIFYING CODE-CONNECTED CONTRACT VARIATIONS ALONG AN AXIS OF STATE TRANSITION COMPLEXITY

What is clear from the existing literature is that many view smart contracts as merely another step in the chronological development of technology that enables computable contracting. Although this observation is not inaccurate, it is also not particularly helpful in terms of a tool for guiding clients through the legal and business risks attendant to using smart contracts in products and services. Specifically, a time-linear approach to ordering code-connected contracts indicates the need to anticipate the rise of new legal issues in technologically advanced forms of contracting, but does not leave us with much in the way of a framework for anticipating or mitigating those challenges. Indeed, it leaves lawyers without a linguistic framework within which to advise clients, stirs confusion among lawyers and lawmakers about what smart contracts are and what they can achieve, and leads legislatures to adopt unfortunate laws "accommodating" smart contracts. This Part introduces the possibility of a unifying theory for code-connected contract variations other

58. "ICO participants buy an asset—a 'token'—that enables its holder to use or govern a network that the promoters plan to develop with the funds raised through the sale." Shaanan Cohney et al., *Coin-Operated Capitalism*, 119 COLUM. L. REV. 591, 593 (2019). ICOs "exist on the internet, embodied in software code. The key forms of software are known as 'smart contracts'—automated, 'if-this-then-that' rules that coders can design to govern the functionality of the digital 'crypto' assets sold in ICOs." *Id.* at 594.

59. Shaanan Cohney & David A. Hoffman, *Transactional Scripts in Contract Stacks*, 105 MINN. L. REV. 319, 327 (2020).

60. Scholz, *supra* note 5, at 134.

61. Andrew Hinkes, *The Limits of Code Deference*, 46 J. CORP. L. 869, 869 (2021).

62. Casey & Niblett, *supra* note 2.

than one that is time-dependent. In particular, this Part offers a theory that aims to bridge some of the terminology gap that leads lawyers to talk past each other and fuels the hype-cycle. Namely, when plotted out along an axis of state transition complexity, design trade-offs between the various code-connected contracts become clear. Such clarity enables parties and courts to better anticipate the potential range of disputes that might arise in contractual relationships that incorporate some form of computable contracting, including the use of smart contracts as performance mechanisms (rather than as a contract term as such). Indeed, the state transition complexity theory enables the identification of smart contracts as simply an element of a product or a service, rather than an element of a legally enforceable contract. Such uses may result in disputes of a type entirely distinct from the considerations under contract law that so clearly dominate the early smart contract literature. Only when parties can adequately anticipate the nature of potential disputes can they appropriately mitigate related risk. Thus, the state transition complexity theory of code-connected contracts offers parties and lawyers a tool for building risk mitigation systems in an increasingly computable legal system.

A. Some Code is Better Suited to Handle Complex State Transitions

In computer science, the "state" of a computer program refers to the content of program data stored in a computer's memory at any given point in the program's execution.[63] A blockchain protocol like Ethereum is a "stateful protocol"—one in which the state is the data carried over from the last block and retained in the protocol's memory (in the case of Ethereum, the Ethereum Virtual Machine's memory).[64] The ledger metaphor that people so frequently use to describe blockchain technology refers to the fact that blockchain protocols track the transitions in state between blocks.[65] Some code is highly state-dependent, in so far as the program involves many variables and allows for many possible actions involving those variables.[66] Here, self-driving vehicles offer a strong

63. DICTIONARY OF COMPUTER SCIENCE, ENGINEERING, AND TECHNOLOGY 467 (Phillip A. Laplante ed., 2000) ("[S]tate: in any computation, the current set of values being operated upon.").

64. NARAYANAN ET AL., *supra* note 29, at 269 ("Specifically, every block contains a digest of the current state (balance and transaction count) of every address as well as the state (balance and storage) of every contract."). More broadly, a server is "stateful" if it "keep[s] track of which clients are using files and in what ways." CONCISE ENCYCLOPEDIA OF COMPUTER SCIENCE 327 (Edwin D. Reilly, ed. 2004). Meanwhile a "stateless" server "retains no knowledge about the files client machines are using" upon reboot. *Id.*

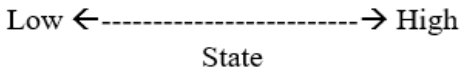
65. A state transition is a change in the information in a computer's set of registers. CONCISE ENCYCLOPEDIA OF COMPUTER SCIENCE, *supra* note 64, at 222; *see also* DICTIONARY OF COMPUTER SCIENCE, ENGINEERING, AND TECHNOLOGY, *supra* note 63 (defining state transition as "in a state machine, the event related to a change of state. Generally, the transition is performed when a certain condition becomes true while staying in the departing state of the transition."). In that sense, blockchain protocols can be thought of as state machines because they "control a process by stepping from state to state as a function of its inputs." DICTIONARY OF COMPUTER SCIENCE, ENGINEERING, AND TECHNOLOGY, *supra* note 63.

66. The variable values at any point during a computation define the state, and all possible combinations of variable values define the state space. . . . Mathematically, an action is a binary relation over any state, an action describes a set of successor states. If the successor set of state s has exactly one state, the effect of the action is to transform s to its unique successor. . . . If the successor has more than one state, the current state is transformed to any of its successors; such an action is called nondeterministic. If the successor set is empty, the action is not enabled in the given state. If an action is executed in a state in which it is not enabled, it has no effect; *i.e.*, the state does not change.

reference point.[67] When self-driving vehicles operate entirely on their own, "a computer is steering the car, accelerating, braking, and following traffic signs or signals," and they are doing so by "tak[ing] in information from sensors that analyze the road and the nearby surroundings to make automated decisions about where and when to drive and stop."[68] Still, other code may involve relatively few variables and potential actions on those variables.[69] For instance, a car odometer "may be regarded as a system with six variables, one for each position; each of these variables may, independently, assume a value between 0 and 9. . . . For the six-digit odometer, there is usually one action whose effect is to add 1 . . . to the current state."[70] When a program operates on the basis of many inputs, state transitions can be complex, while fewer inputs result in more simple state transitions.

Clearly then, the complexity of state transition supported by specific software results in different uses appropriate for that software. A passenger of a self-driving vehicle, for instance, would not want the vehicle to be operated by the code that updates the vehicle's odometer. In other words, even within the same system—a self-driving car, different software programs achieve different actions. The software program that automatically updates the odometer might be described as being powered by code with a low level of state dependency in that it has a small number of variables and only one potential action. Advanced driver-assistance systems, such as parking assist,[71] might be described as operating with moderate state dependency, in so far as it must execute one of a variety of potential actions in reliance on a higher number of variables. And at the far end of the spectrum, fully autonomous vehicles involve a high level of state dependency, insofar as they involve an exponentially greater number of variables and potential actions. This spectrum of software state dependency is visualized in the simple diagram in Figure 1, below.

Figure 1: Spectrum of Software State Dependency



B. Some Contracts are More State Dependent Than Other Contracts

We can apply the analogy of state dependency to legal technology as well. Contracts negotiate the rights and responsibilities between parties based on anticipated state changes. Where a transaction may be impacted by a single variable, there will be low potential variance in the outcome of the transaction. So, we might say, that where the potential for change in the state of the world around the transaction and the relative positions of the parties is low, the potential variance in the outcome is also low. The outcome of the

JAYADEV MISRA, A DISCIPLINE OF MULTIPROGRAMMING: PROGRAMMING THEORY FOR DISTRIBUTED APPLICATIONS 14 (2001).

67. Harry Surden & Mary-Anne Williams, *Technological Opacity, Predictability, and Self-Driving Cars*, 38 CARDOZO L. REV. 121, 131 (2016) (explaining what it means to say a self-driving car is autonomous and making plain the complexity of the computation required).

68. *Id.* at 135.

69. MISRA, *supra* note 66.

70. *Id.*

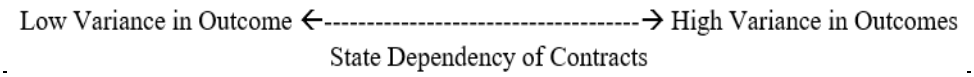
71. Surden & Williams, *supra* note 67, at 134–35.

transaction will likely materialize the way that the parties expect it to, barring exceptional circumstances. A simple escrow agreement, even in the context of an important transaction like the purchase of a home, serves as an example here. In an escrow agreement, two parties agree to provide consideration to an escrow agent and empower the escrow agent to disburse the consideration to the other party only upon the fulfillment of certain conditions. In the common home purchase, those conditions are few. For the buyer, the title must come back clean, the inspection must be satisfactory, and the appraisal must satisfy the lender. For the seller, the full amount of the purchase price of the home must be funded to the escrow agent. Either these conditions are met, and the funds are released to the seller and occupancy and title of the home given to the buyer, or not. In other words, the escrow transaction involves very few variables, and very few actions operating on those variables.

On the other end of the spectrum, when the parties anticipate a high number of changes in the relative position of the parties, there may be high variance in the outcome of the transaction. This may, for example, be particularly true in the context of relational contracting, where the parties create a long-term relationship, during the duration of which changes in the world and in the position of the parties are all but inevitable.[72] In such scenarios, we might say that where the potential for change in the state of the world around the transaction and the relative positions of the parties is high, the potential variance in the outcome is also high. The outcome of the transaction governed by the contract is highly difficult to precisely predict, although both parties expect the contractual relationship will ultimately be beneficial despite the risk brought by outcome uncertainty. A sophisticated corporate acquisition agreement serves as an example here. The purchase of one corporation by another usually involves an extensive and detailed contract containing an extensive and detailed list of conditions to closing. The conditions to closing may be interdependent, may depend on third parties, and may change in status over time as the deal is finalized. Further, the deal may end in any number of ways. Of course, the acquisition could close on time and as planned, or the buyer or seller could walk away, as in any asset purchase. But other options also remain available to the parties. The buyer may waive certain unsatisfied conditions and move forward with the deal, or new information related to closing conditions may alter the nature of the deal entirely. In other words, a corporate acquisition involves many variables, and many potential actions on those variables, with the parties unable to anticipate the nature of some potential actions at the beginning of the deal.

We might visualize the spectrum of contract outcome variance as existing along an axis of contract state-dependency, where the certainty of the outcome depends upon the number of variables, the potential for change in those variables during the negotiation or life of the contract, and the number of potential actions that might be carried out on the variables. In other words, the level of outcome certainty depends upon the level of expected change in the state of the world surrounding the transaction and the relative position of the parties over time.

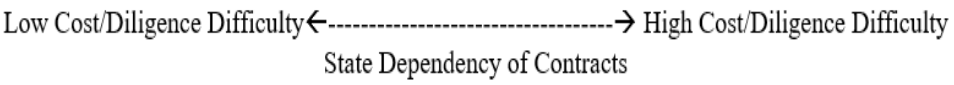
Figure 2: Spectrum of Contract State Dependency



72. Sklaroff, *supra* note 3, at 292.

When a contract centers around a transaction with low variance in potential outcomes, it is likely that due diligence and negotiation also reflect a relatively low level of difficulty in information gathering and lower negotiation costs. In the context of the home purchase, for example, the buyer must contract an inspector, and pay for the title search and insurance. The buyer might negotiate with the seller for additional concessions depending on the outcome of the inspection. The seller, for her part, considers the financial position of the buyer and mitigates risk through earnest money and the use of an escrow agent. In total, the costs of due diligence and negotiation is quite low in the common home purchase. On the opposite end of the spectrum, however, when a contract centers around a transaction with high variance in potential outcomes, it is likely that due diligence and negotiation reflects a high level of difficulty in information gathering and negotiation cost. This is clear in the context of the corporate acquisition. Due diligence can involve hundreds of thousands of pages of documents, and significant costs in attorneys' fees. Negotiation of the details of the contract is complicated and drafting roughly agreed upon terms into contract language both parties can live with requires further negotiation and attorneys' fees. And if any variable changes any of the potential actions the buyer or seller might take, the due diligence and negotiation costs increase. Thus, the above image of the spectrum of contract state dependency might also be conceived as:

Figure 3: The Relationship Between State Dependency, Cost and Due Diligence



So far, then, the principles of the state transition complexity theory of code-connected contracts instruct us that: some code is better suited for high state transition complexity than others, and some contracts involve higher state transition complexity than others. Further, in the context of contracts, at least, the higher the state transition complexity, the higher the cost and relative level of due diligence difficulty. In other words, the higher the contract state transition complexity, the higher the risk. These principles help develop an analytical tool for assessing which type of code-connected contract to use in a given scenario, and whether that scenario involves a coded attempt to substitute for a legally enforceable contract at all.

C. The State Transition Complexity Theory of Code-Connected Contracts

The three principles of the state transition complexity theory serve as an analytical tool for identifying which code-connected contract variation, if any, will help mitigate risk rather than introduce new risk. In the context of legally enforceable contracts, the state transition complexity theory helps lawyers, law-makers, legal academics, and enterprises make sense of when to use one simple smart contract, a combination of smart contracts, a combination of smart contracts and other code-connected contracts (e.g., data-oriented, computable, algorithmic), a combination of code-connected contracts and other technology, or a regular contract in which the parties agree that one or more of them can perform certain obligations via computer code. For example, contracts in which the social context is highly state dependent with attendant high cost and due diligence difficulty might require a combination of a code-connected contract and some other technology (including

the plain old legal technology of a natural language written contract). This might be the case, for example in the context of the corporate acquisition agreement. In such a high stakes context, the parties might prefer the traditional written contract to mitigate risk, and if the parties are ambitious or see some related business value from doing so, might consider using a code-connected contract variation, if any, will help mitigate risk rather than introduce new risk.[73] A low state transition complexity contract, however, could maximize efficiency without increasing risk by using a low state transition code-connected contract tool. In the escrow example, a smart contract could serve the parties well, particularly if the traditional escrow agent served as the oracle that triggers the smart contract upon fulfillment at closing.

In between these two extremes lie opportunities to incorporate computable contracts and algorithmic contracts, and to explore the relationship between the entire field of computational contracting tools. For example, the lessons from Professor Surden regarding data-oriented and computable contracts apply with equal force to smart contracts. Many of the questions regarding the legal enforceability of smart contracts could be addressed in ways similar to Professor Surden's proposals for data-oriented and computable contracts. For example, one way to give smart contracts shared meaning is through a data-meaning threshold agreement.[74] Such agreements "are traditional, written language documents that parties agree to before engaging" in a transaction involving smart contracts.[75] Further, threshold procedural agreements can "set up the procedural foundation" for future dispute resolution.[76] Such agreements essentially say that the parties agree to contract electronically and "the data records should be considered as their contractual expression."[77] This technique is used all the time in code-based contracts.[78] For example, in derivatives, parties adopt provisions from a standard agreement called the ISDA Master Agreement.[79] Smart contract enthusiasts have advocated for something similar in the form of "code deference," but the links between code deference and the threshold agreements proposed by Professor Surden remains underexplored. This type of connection between code-connected contracting tools goes beyond a time linear progression understanding of the technology. The connection can be seen because the state transition complexity theory made the links and similarities more obvious. This is one of many ways the state transition complexity theory can be expected to illuminate the variety of code connected contracts and help tie them to legal and business risk considerations.

The state transition complexity theory also serves as an analytical tool for translating between the legal field and smart contract developers. Because blockchain protocols

73. *Intelligent Contract Founders: Clause.io*, ARTIFICIAL LAW. (Jan. 1, 2017), artificiallawyer.com/2017/01/01/intelligent-contract-founders-clause-io/ [https://perma.cc/NQN4-KCZA] (describing Clause.io's development of a data-augmented natural language supply agreement that used smart contracts triggered by sending them external information).

74. Surden, *supra* note 20, at 651. Note: the terms threshold agreement, data-meaning threshold agreement, and threshold procedural agreements are all terms coined by Professor Surden. *Id.*

75. *Id.*

76. *Id.* at 655.

77. *Id.*

78. *See generally* Surden, *supra* note 20, at. 639–42 ("To the extent that contracting parties want computers to process the substance of their obligations, they must reorient their contractual expression away from ordinary language and toward highly-structured data—a form more amenable to computer processing.").

79. *Id.* at 656.

represent specialized software that track state transitions, many observers assume that all blockchain protocols are suitable for the operation of highly state dependent programs. In reality, however, many blockchain protocols currently remain better at tracking low complexity state transitions.[80] However, new protocols seek to improve the capacity of blockchain protocols to track more complex state changes. Indeed, the Ethereum protocol represents an improvement over the Bitcoin blockchain in this regard.[81] This explains the enduring gap between industry uses of smart contracts to power performance of contractual obligations or as components of products and services, and the legal field's focus on whether a court would treat a smart contract, standing alone, as a legally enforceable contract. The focus of the legal field, including lawmakers, centers on a use of smart contracts not yet widely supported by the technology. Advances in smart contract technology and computational contracting more broadly may reduce this gap over time, however, the state transition complexity theory will remain an important tool for translation between the legal field and the developers of smart contracts as they work together to mitigate legal and business risk.

IV. LESSONS FROM THE STATE TRANSITION COMPLEXITY THEORY FOR THE INTERSECTION OF EMERGING TECHNOLOGY AND LAW

Although the state transition complexity theory of code-connected contracts is designed as an analytical tool for use in applying law to code, inquiries at the intersection of emerging law and technology rarely only look one way. Viewed through the lens of the state transition complexity theory, code-connected contracts offer broader lessons for the intersection of code and contract law, and for emerging technology and law more broadly. This Part unpacks three such lessons for consideration. First, among code-connected contracts, smart contracts are not so exceptional that they require new rules on legal enforceability and interpretation that other forms of code-connected contracts have not previously required. Second, the tendency of the legal field to rely on technology buzz words without taking a deep dive into how the technology works can lead to bad outcomes, such as enactment of laws to "accommodate" smart contracts which actually have the opposite effect. Third, like most things, if a description of technology sounds too good to be true, it probably is. As a result, lawyers cannot simply rely on hype-filled technological descriptions when evaluating the legal and business risks that adoption of the technology may pose to a client. Rather, the key to assessing legal risk remains, as always, in the socio-technical context—the use to which the client puts the technology and the mechanics of how the client achieves the use case.

A. Among Code-Connected Contracts, Smart Contracts Are Not Exceptional

If viewed as the next chronological technological advancement in computational contracting,[82] one might be tempted to assume that smart contracts pose unique questions not raised by computational contracting, and thus that a procedural threshold agreement or

80. NARYANAN ET. AL., *supra* note 29, at 263.

81. *Id*. at 263–64.

82. Werbach & Cornell, *supra* note 3, at 322 ("The evolution from electronic, to data-oriented, to computable contracts embodies a trend toward greater machine autonomy.").

some other common contracting technique will not work in the smart contract context. When smart contracts are viewed as simply one type of code-connected contract organized along a spectrum of state transition complexity, however, it becomes clear that smart contracts are not exceptional.[83] A technology might be considered exceptional when it so fundamentally disrupts the existing social order that new, technology-specific law may be necessary to adequately address the new challenges the technology poses.[84] Whether a technology rises to the level of exceptional and requires a new legal approach depends upon that technology's essential characteristics—the characteristics that distinguish a new technology from prior or constituent technology.[85] When viewed through the lens of the state transition theory of code-connected contracts, smart contracts do not disrupt existing contracting practice or contract doctrine so fundamentally that new legal rules represent the only way to restore social values around contracts.

Smart contracts are just one among many variations of code-connected contracts. In fact, smart contracts might not even be a contract. The escrow example again offers instruction here. In the context of a home purchase, the key contract is the contract to purchase the home. Any agreement with the escrow agent is ancillary. Both are written contracts in the traditional sense. If a smart contract were used by the escrow agent to securely hold the buyer's funds until all closing conditions have been satisfied, the escrow agent is merely performing her obligations under the escrow agreement using a technological tool. Just because the tool is called a smart contract, and just because the smart contract might incentivize the seller to fulfill any conditions for which she is responsible (because she knows the money is locked in the smart contract waiting for her), does not alter the fact that the smart contract is a tool used to help efficiently and securely perform obligations imposed by a traditionally-styled escrow agreement.

Recognizing this makes it easier to understand that smart contracts do not necessarily present a novel set of legal issues. They simply offer a new technology for use in computational contracting. Or not. Indeed, when viewed as a technological tool—a type of computer program—it becomes clear that smart contracts can exist and be used in contexts that do not touch on contract law at all, or they can sit within a contract governed entirely by traditional contract law principles, or they can push the boundaries of contract law such that the parties seek to protect themselves with plain English contracts that inform potential future dispute resolution forums of their original contracting intent for the code (data meaning threshold agreements and procedural threshold agreements). As in the context of any deal-making, the nature of the deal dictates the relative level of legal and business risk. As that risk increases, the tools at hand change, including the code-connected contracting tools. The state transition complexity theory aims to offer lawyers and technologists a common analytical tool that draws on concepts from both disciplines to help them cooperatively, constructively, and creatively assess the legal and business risk attendant with whatever use of code-connected contracts adopted.

---

83. Ryan Calo, *Robotics and the Lessons of Cyberlaw*, 103 CALIF. L. REV. 513, 552 (2015) (defining exceptional technology as a technology that, when introduced into mainstream society, "requires a systematic change to the law or legal institutions in order to reproduce, or if necessary displace, an existing balance of values.").

84. *Id.* at 550.

85. *Id.* at 532.

B. Core Lessons at the Intersection of Code-Connected Contracts and the Law

The core lessons found at the intersection of code-connected contracts and contract law reflect core insights into the intersection of emerging law and technology more broadly. Inquiries at the intersection of emerging technology and law often begin with questions about whether and how the law applies to the technology. Thus, it is no surprise that the inquiries at the intersection of smart contracts and contract law began with questions about whether and how courts would enforce and interpret smart contracts. However, emerging technology often serves as a magic mirror, such that when the technology is held up to an area of legal practice, lessons emerge about the law more generally. In other words, inquiries at the intersection of law and technology are rarely only one-way endeavors. The state transition complexity theory not only holds lessons for navigating the use of code-connected contracts in commerce, but also for the continuing collision between law and emerging technology.

Commonly, at the intersection of law and technology, lawyers, media, and academics use buzz words to refer to or describe characteristics of emerging technology. Buzz words used to refer to a particular type of technology, like smart contract or algorithm, often actually encompass several different variations of technology.[86] Although using buzz words is easier than trying to understand the technology and its many variations, doing so creates a variety of problems. For example, in the area of smart contracts, focus on whether contract law will recognize smart contracts as legally enforceable ignores what people are actually doing with smart contracts[87] and prevents discussion around how to avoid and mitigate legal and business risks associated with actual uses. Further, without deeper investigation into both the reality of the technology and the legal demands of the social context in which the technology will be used, participants in the legal and law-making system risk continuing generalizations and perpetuating myths that exacerbate the extent to which law lags behind or, worse, compounds the risks related to technology.[88]

Most emerging technology that promises "automation" or other seemingly magical properties are best suited to achieve those properties for simple things. For example, when parties seek to fully draft a contract in code that automatically executes, a simple escrow

86. For example, in the context of artificial intelligence, people often use the term "AI" but really mean unsupervised machine learning. Amanda Levendowski, *How Copyright Law Can Fix Artificial Intelligence's Implicit Bias Problem*, 93 WASH. L. REV. 579, 590 (2018) ("When journalists, researchers, and even engineers say 'AI,' they tend to be talking about machine learning, a field that blends mathematics, statistics, and computer science to create computer programs with the ability to improve through experience automatically."). The same is true for the term smart contract. Cohney & Hoffman, *supra* note 59, at 322–23.

87. DIEDRICH, ETHEREUM, *supra* note 29, at 169 ("A smart contract is not necessarily between two parties, and in reality, almost never, so far, the mirror image or replacement of a legal contract."); *see also id.* at 175 ("One smart contract will almost never replace exactly one legal contract.").

88. Take, for example, state laws purporting to "accommodate" smart contracts in contract law by amending the Uniform Electronic Transactions Act. *See, e.g.*, H.B. 2417, 53rd Leg., Reg. Sess. (Ariz. 2017); NEV. REV. STAT. § 719.090 (2017). Doing so is not only unnecessary, but also potentially achieves the opposite of the intended purpose: excluding smart contracts from UETA's coverage rather than "accommodating" smart contracts. Uniform Law Commission, *Guidance Note Regarding the Relation Between the Uniform Electronic Transactions Act and Federal ESIGN Act, Blockchain Technology and "Smart Contracts"* (2019), https://www.uniformlaws.org/HigherLogic/System/DownloadDocumentFile.ashx?DocumentFileKey=d2026984-1040-3c6f-62c8-a676b12d7bff [https://perma.cc/JF7D-KWW2].

transaction remains the core example.[89] More complex applications often require a combination of technology, if the magical properties promised by the hype are to be achieved at all. Indeed, it is more likely that even if a contract can be formalized in computer logic, people will need to trigger the execution of its provisions and perform certain of its obligations. Take, for example, decentralized autonomous organizations ("DAOs") built out of a complex web of smart contracts. Many DAOs automate many coordination activities among their participants, but do, in fact, rely on human or other legally recognized persons to participate in the governance and operation of the DAO.[90] Thus, to get to the core legal issues, we must look at the details of the implementation—as with most legal questions, proper analysis and risk assessment depends on facts and circumstances. Lawyers only need to worry about whether a smart contract is legally enforceable if, in fact, the client intends to use the smart contract as a replacement for a legally enforceable natural language contract.

Ultimately, many of the core legal issues around smart contracts, and all variations of code-connected contracts, arise because emerging technology, like all technology, is social technology. Thus, many of the legal issues really are not that new, and the legal field should not forget to look to past applications of law to the broader field of relevant technology for instruction on what to expect. Further, this means, lawyers possess the tools they need to assess many of the legal issues presented, even if they do not understand the code.[91] All that remains necessary is a common analytical tool the legal field can use to standardize an approach to those legal and business risks. Such analytical tools need to acknowledge the relevant design tradeoffs and the related risk spectrum as primary considerations. The state transition complexity theory of code-connected contracts offers a starting point in that regard for smart contracts and the many other forms of code-connected contracting tools.

V. CONCLUSION

Smart contracts always form part of law and blockchain discussions. Inevitably, as seen in the legal academic literature and related symposia, the question of legal enforceability of smart contracts arises. For lawyers and the legal community, the discussion is always complicated by the fact that the second word in smart contract is "contract." Lawyers think they own the word "contract" as a term of art and immediately and unwaveringly go from smart contract to smart offer, acceptance, and bargained-for exchange. In other words, lawyers and the legal community immediately want to think about smart contracts in terms of legal enforceability. The legal community and lawmakers influenced by the legal community need to resist this urge.

As this Article seeks to clarify, smart contracts are just computer programs that can be used for any number of things. Parties might be able to digitally represent a legally enforceable contract via smart contracts, yes, but they might also simply opt to use a smart contract to perform part of their obligations under a traditionally documented contract, or

89. Stuart Levi, *An Introduction to Smart Contracts and their Potential and Inherent Limitations*, JDSUPRA (May 8, 2018), https://www.jdsupra.com/legalnews/an-introduction-to-smart-contracts-and-98025/ [https://perma.cc/6MKD-M79M].

90. Carla L. Reyes, *Autonomous Business Reality*, 21 NEV. L.J. (forthcoming 2021), https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3574988 [https://perma.cc/P7E8-R9ET].

91. Reyes & Ward, *supra* note 6.

to build decentralized software applications that have nothing to do with legally enforceable contracts. As such, the issue of legal enforceability is not necessarily the issue to focus on. Indeed, leading developers and computer scientists in the field plead with lawyers to stop focusing on legal enforceability of smart contracts because it prevents the legal community from seeing other applications, uses, and issues. Essentially, the fixation on legal enforceability prevents lawyers from being creatively helpful to the vast majority of businesses using smart contracts in their products and services as something other than a legally enforceable contract. The focus on enforceability prevents the legal community from devising tools that help business ventures assess the legal and business risks associated with using smart contracts generally.

The state transition complexity theory represents an attempt to fill this gap by offering an analytical tool for assessing the design trade-offs businesses make when they use smart contracts and other code-connected contracts in their business models—as a legally enforceable contract, part of one, or as something else entirely. The state transition complexity theory reminds the legal community that ultimately, like any technology, what a smart contract is, for legal analysis purposes, depends on how it is used—it depends on the socio-technical context. Further exploration of the place of each type of code-connected contract on the state change complexity system and the extent of any links between them is warranted. And as the technology continues to advance, with natural language programming potentially enabling new uses of code-connected contracts, the state transition complexity theory analytical tool may need to be adjusted. In both cases, however, the aim of the state transition complexity theory is to reorient the legal field away from the 'is this new development enforceable' question, and towards a deeper understanding of the technology and its uses.