OFR WORKING PAPER
OFFICE OF FINANCIAL RESEARCH

15-04 | March 26, 2015

Contract as Automaton: The Computational Representation of Financial Agreements

Mark D. Flood
Office of Financial Research
Mark.Flood@treasury.gov

Oliver R. Goodenough
Office of Financial Research and Vermont Law School
Oliver.Goodenough@treasury.gov
OGOODENOUGH@vermontlaw.edu

events, as later described. It also has powerful implications for the computational complexity of the contractual machinery.

Table 3: Contract States (Q)

Table 4: Event Alphabet (Σ)

Table 5: Transition Function (δ)

Just as we simplified the graphical representation of the DFA in Figure 1, we abbreviate the transition function in Table 5 by suppressing the "stay-in-place" transitions that return the system to the same state in response to an event. These event occurrences are "irrelevant" to a given state if a self-transition — i.e., ignoring the event, rather than provoking a rejection — is an appropriate response.[54] Unlike a parser, for example, which has a responsibility to reject character sequences that are unacceptable, a financial agreement has the flexibility simply to ignore most of the superfluous event occurrences. This creates a proliferation of self-transitions that would otherwise clutter Table 5 with relatively trivial entries. In other words, the application context for financial agreements is less tightly controlled than for programming-language parsers, and contracts should be relatively permissive and robust to irrelevant event occurrences. Table 5 also suppresses omnipresent events such as the filing of a lawsuit (event D) that could occur and be relevant in any state. For completeness, we also present in Table 6 the full transition function as a matrix, where the rows correspond to the states in Table 3, the columns correspond to the event alphabet in Table 4, and the state listed in in each cell is the result of the transition function applied to the combination of the initial state (identified by the row) and event (column) values. Unlike Table 5, the full transition function in Table 6 does not suppress irrelevant and omnipresent events. To condense the matrix to a single page, Table 6 does, however, omit the natural language queries and correlates included in Table 3 and Table 4.

Table 6: Full Transition Matrix

Regular Expression Representation

[54] Knowing when to ignore events, rather than reject (for example, by triggering a back-office investigation), is an important judgment call for contract drafters and implementers. For example, expiration of the statute of limitations with respect to the obligations of the borrower (event E in Table 4) before delivery of the principal (event F) is a physical impossibility, so there is little point in developing error-handling procedures for this case. On the other hand, multiple occurrences of event F in quick succession would be a plausible clerical error, and a rejection procedure might be appropriate to handle this case.

Our final representation of the DFA is a regular expression. A regular expression — sometimes called a "regex" — is a compact shorthand notation focusing on the event sequences that the DFA recognizes. The basic idea of a regular expression will be familiar to any computer user who has ever deployed a wildcard expression to refer to a set of filenames. For example, the command "DEL *.PDF" to delete a collection of documents at a single stroke includes a simple regular expression, "*.PDF". This regex defines a set of character sequences (interpreted as filenames) to be included in the delete action.[55] In the context of a DFA representing a financial contract, the event alphabet is different, but the shorthand principle is the same. We emphasize that the regular expression presentation, although cryptic, captures the same structural information available in the tabular and graphical representations above (minus the textual labels that describe the states and transitions). Indeed, there are standard procedures for converting to the regular expression representation from the other representations, and vice versa.

More specifically, a primary upshot of a finite automaton is a declaration of the set of all strings — concatenated sequences from the event alphabet — that the machine will accept. These are the event sequences for which the contract has scripted some appropriate behavior of the counterparties, and which leave the automaton in one of its "accept" states. For example, a given contract might accept a happy-path event sequence of "sign"-"pay"-"quit," meaning sign the deal, then make the promised payments, then terminate the relationship. In contrast, a sequence of "quit"-"pay"-"sign" would be nonsensical, and the contract's DFA should declare its inability to process this sequence of events. In the case of our streamlined contract, we can see that the event sequence "**ACFGQPR**" defines the happy path, while the shortest event sequence resulting in a final state is "**AB**."

Extending the metaphor of an alphabet of events, these collections of all acceptable (by the automaton) event sequences are called the "language" recognized by the DFA. In short, the DFA defines a "grammar" for the language. As noted above, a DFA is one of the simplest computational formalisms and only supports some of the least expressive languages, known as the "regular" languages. A useful feature of the regular languages is that they have a shorthand notation that can capture an entire regular language with a single snippet of event-sequence patterns, known as regular expressions.

The regular expression shorthand that describes a particular regular language is stated entirely in terms of the characters from the event alphabet, perhaps interspersed with a few control characters and wildcards that are special to the regular expression notation itself. The DFA that governs the regular expression is implicit rather than explicit. There are programmatic techniques for generating regular expressions from DFAs. However, just as "cannot" "can not" and "can't" are all character strings expressing the same concept, a given DFA can have many regular expressions. Fortunately, it is usually easy to justify that the shortest regular expression for a given language is the "best" one to use, and it is possible for the programmatic generators to produce this unique, optimal version. Perhaps more surprisingly, there are programmatic techniques to move in the other direction and recover the DFA implicit behind the regular expression for a language. Here again, the implicit DFA is not unique, but

[55] For an introduction to regular expressions, see Friedl, Jeffrey E. F. *Mastering Regular Expressions*. 3rd ed. Sebastopol, CA: O'Reilly, 2006., at: http://regex.info/book.html. For a more technical introduction, see ch. 10 Aho, Alfred V., and Jeffrey D. Ullman. "10." In *Foundations of Computer Science*. New York: Computer Science Press, 1995, at: http://infolab.stanford.edu/~ullman/focs.html.

there are rules for choosing the best from among the many equivalent possibilities.[56] The result of all this is a set of tools for converting our DFA into a regular expression representation and back again.

We use the method of state elimination to convert the DFA represented in Figure 1 into an equivalent regular expression shorthand for the set of event sequences the DFA accepts. The key operation in state elimination is to replace the event arrows in the DFA with arrows describing event sequences, while still preserving the state-transition logic of the full graph. For example, consider this snippet from Figure 1:

{***Pmt. 1 due***} –[**B**]–> {***Default (borr.) $ miss***.} –[**K**]–> {***Borrower notified of payment default***}

One can eliminate the central state, "Default (borr.) $ miss." without disrupting the overall state-transition logic by replacing the elided state with at joint transition arrow, labeled as an event sequence:

{***Pmt. 1 due***} –[**BK**]–> {***Borrower notified of payment default***}

This is a particularly simple example of state elimination, but the procedure extends in a straightforward way to more involved states in the network.[57] The regular expression shorthand has a few syntactic tools to allow it to express some common DFA structures with compact notation. We use a basic version of the Unix notation here. There are numerous competing shorthand notations for regular expressions. For example, Hopcroft, et al., use the "+" symbol to indicate a "union" or branching pattern, where the path can follow either one of several branches. Sisler uses the "∪" symbol for the same purpose.[58]

- Parentheses indicate association, meaning that the enclosed regular expression should be evaluated first, before considering the rest of the pattern.
- The "*****" wildcard indicates that the preceding event (or parenthetical pattern) can appear any number of times, including zero times.
- The "**?**" wildcard indicates that the preceding event (or parenthetical pattern) can appear zero or one time only.
- A vertical bar between two states, such as **A|B**, indicates that the path can follow either event **A** or event **B**.
- A concatenation of states enclosed in square brackets, **[ABC]**, indicates that the path can follow any of those events. That is, **[ABC]** is logically identical to: **A|B|C**.

For example, the streamlined contract depicted in Figure 1 has several pathways from the start state that lead to a relatively rapid demise of the relationship: **AB** or **ACBE** or **ACBD**. We can capture all three in a single regular expression, **A(B|CB[ED])**. The happy path for the streamlined contract is given by the event sequence: **ACFGQPR**.

---

[56] For an introduction to the translation between DFAs and regular expressions, see Sipser (2006), section 1.3. For a deeper discussion, see Rosenberg (2010), especially section 5.2.

[57] For a more detailed introduction to the state-elimination method, see Hopcroft, et al. (2001) section 3.2; or Sipser (2006), pp. 66-76. Hopcroft, et al. (2001) describe two general methods for the DFA-to-regular expressions conversion, namely path induction and state elimination. The two methods are equivalent in the sense of producing equivalent regular expressions.

[58] See Hopcroft, et al. (2001), section 3.3, for an overview of the Unix syntax for regular expressions.

Note that the regular expression for a given DFA is not unique. For example, it is easy to see that the two expressions, **A(B|CB[ED])** and **(AB|ACB[ED])**, recognize the identical event sequences and are functionally equivalent. We find it convenient to organize the regular expression for our streamlined contract as the union of four key segments, corresponding to: (a) rapid-demise paths; (b) the happy path; (c) unhappy paths (payment and nonpayment defaults) around the first payment date; and (d) unhappy paths around the second payment. Eliding the derivation of these four expressions, the overall regular expression representation of the DFA is:[59]

A(B|CB[ED])|	*Rapid demise*
ACF(G(BK)?)QPR|	*Happy path*
ACF([HIJ]LN)*(GBK|[HIJ]L)O(S|B[DES])|	*Unhappy* 1
ACF(G(BK)?)Q([HIJ]LN)*(PBK|[HIJ]L)O(R|B[RED])	*Unhappy* 2

Note that the final segment, "unhappy 2," follows the happy path up to state **Q** (payment 2 accruing) and then diverges into the various ramifications of payment and nonpayment default from that state. The happy path segment here includes a wildcard sub-segment, **(BK)?**, covering the possibility of a missed payment that is quickly cured. Similarly, the two unhappy segments include a wildcard subsegment, **([HIJ]LN)***, indicating that the contract can tolerate an arbitrary number (including zero), *, of any of the three nonpayment defaults, **[HIJ]**, as long as they are cured in a timely fashion, **LN**.

At first glance, the regular expression representation of the contract may seem like a hopeless bit of gobbledygook, but like a sort of contractual DNA, it is actually a powerfully compact distillation of two full pages of legalese. In addition to its theoretical benefits, regular expressions are also enormously practical, as users of the venerable Unix command-line utility, "grep" (the global regular expression printer), can testify. In particular, the regular expression's string of symbols provides a simple and intuitive measure of the complexity of the contract, namely the length of the string. One might object that, because a DFA's state transition-network and its regular expression are generally not unique, the complexity score is arbitrary. This objection is ill-founded, however, because there are programmatic techniques to reduce any DFA to a theoretical minimum state-transition network and standard techniques for representing any given DFA as a regular expression. The complexity score should measure the length of a standardized regular expression for the minimized DFA. In the example here, the complexity score is 109.

A contract should be as simple as possible, but no simpler. Note that the bulk of the contract's complexity (75.2 percent, to be precise) arises in the two nexuses of unhappy ramifications. The two

[59] The derivation of the regular expression from the state-transition network via state elimination is mildly tedious, but a useful exercise for deepening one's understanding of the DFA machinery. Conversely, it is also instructive to follow each of the four segments of the regular expression here as a set of directions through the maze of the state-transition graph in Figure 1 to verify that the regular expression describes all possible pathways through the network.

unhappy substrings, while dealing with potentially unlikely events, account for 82 symbols in total, or 82/109 = 75.23 percent of the overall string length. Unsurprisingly, much of the hard work of managing a relationship occurs not when things are going well, but rather when the process starts to deviate from the happy path. Much of the value of good contracts and good lawyering derives from the seemingly tedious planning for all the ways that a relationship might run off the rails.

Note too that financial risk and valuation models — the core of financial engineering — tend to ignore the complications of the unhappy nexuses, focusing instead on probabilistic models of the happy path. Implicitly, this seeming negligence relies on an assumption that all unhappy relationships are idiosyncratic, so that the manager of a well-diversified investment portfolio can safely ignore these high-maintenance details. Holdings in many cases, such as bank commercial loan portfolios, consist of a relative handful of large, specialized relationships; this concentration of risk exposures denies the portfolio manager the luxury of simple diversification. Alternative mechanisms, such as securitization and syndication have evolved to spread the risks in these situations. Formal modeling of these complicated portions of financial relationships as DFAs may make them more measurable and manageable. If so, such modeling has the potential to create significant value by reducing overall financial drag.

VI. Discussion and Directions for Further Exploration

The foregoing provides evidence for our proposition that the structures embodied in financial contracts are state-transition systems. The key is in recognizing that the state-transition structure is sufficiently fundamental to a financial agreement that we can represent it using the standard computational formalism of a deterministic finite automaton without disrupting the contract's organizing principles. If this is true, why does it matter? By identifying the DFA that undergirds a contract, we expose the entire edifice to the tools and techniques developed in the computational and linguistics communities to work on finite automata.

For our streamlined contract, the preceding section presents three canonical representations — graphical, tabular, and regular — of the underlying DFA. Taken together, they are equally valid embodiments of the process set out in natural language in the streamlined contract, with the added benefit of being expressly computable. By this we do not mean that the DFA restates the natural language contract. Instead, each is a valid embodiment of a logic that exists as an independent algorithm waiting for description.

This possibility for multiple forms of expression resembles the way that an algebraic proposition can be described both as a word problem and as a numeric formulation. And as with algebra, representing the proposition in the formal language of mathematics opens the proposition to a number of tests, applications and manipulations that are much more difficult to access when it remains a word problem described in natural language. For example, it is possible to craft the three representations in such a way that they are precisely formally equivalent. A key to establishing this mutual equivalence lies in the application of techniques for minimizing the finite automaton. One of the contributions of the Myhill-Nerode Theorem is that there exists a *unique* smallest finite automaton that will accept a given language of event sequences defined by the regular expression.

We did not apply this sort of formal minimization logic in the construction of the streamlined DFA, preferring to maximize instead the common-sense legal semantics of the DFA. It will be instructive to see how far our legally optimized presentation is from the theoretical minimum. This gap is likely to widen as we apply the approach to more realistic agreements. It is important to recognize that legal contracts are ultimately devices for coordinating human activity and much of their effectiveness derives from their enforceability. The lender is willing to relinquish the principal, in part because she knows authorities exist to enforce repayment if necessary. These authorities involve human interpreters — judges, juries, arbitrators, etc. — who must be able to parse the agreement in the crucial tasks of dispute resolution. Otherwise, the contract has failed to meet one of its most important requirements.[60]

We started with the natural-language representation (old lawyering habits die hard), and retrofitted the computational model. The process of representing the state-transition logic, however, clarified requirements and assumptions, leading us in turn to amend the natural language version. The final streamlined contract is the outcome of iterative approximations to both the natural language and the DFA, so that neither form has full precedence. Indeed, creating the DFA helped us to refine the logic of the transaction as a whole and to clarify the role of standard elements as warranties and defaults. A natural next step will be to perform a similar exercise on a more ambitious agreement, perhaps a standard master agreement taken from the realm of actual financial practice. We can also envision a process of building a useful master agreement of our own in computational terms from the ground-up. Perhaps the next iteration of ISDA will be drafted in just such a fashion.

As we pursue that direction in our research, one interim goal will be to assemble some of the existing tooling — much of it described in textbooks and existing software packages — into a suite of functionality to make these transformations as a set of programmatic, push-button tasks.[61] This toolkit will be instrumental as we begin to apply the state-transition approach to more realistic contracts.

As the analysis moves to more complex and realistic examples, we will want to expand the toolkit beyond DFAs to include nondeterministic finite automata (NFAs). NFAs are a more sophisticated class of automata that allow multiple alternate transitions from a state in response to an event.[62] This is a representational convenience that affords significant simplification of the state-transition graph in many cases. Note that NFAs are semantically equivalent to DFAs. They support the same set of regular grammars, and there exist standard techniques for translating between DFA and NFA representations.

---

[60] One can imagine a science-fiction future in which contract enforcement is itself fully automated and delegated to machines for efficiency's sake. The ethical ramifications of such an institutional arrangement could be extensive.

[61] Examples of relevant software packages include Berkeley YACC, available at: "BYACC – Berkeley Yacc – Generate LALR(1) Parsers." BYACC – Berkeley Yacc – Generate LALR(1) Parsers. Accessed December 13, 2014. http://invisible-island.net/byacc/byacc.html; and GNU Bison, available at: "Bison - GNU Project - Free Software Foundation." Bison - GNU Project - Free Software Foundation. Accessed December 13, 2014. https://www.gnu.org/software/bison/. Textbook sketches of many of the programmatic transformations are available in, for example, Sipser (2006), Rosenberg (2010), and Aho, Alfred V., Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. *Compilers: Principles, Techniques, & Tools*. 2nd ed. Boston: Pearson/Addison Wesley, 2007.

[62] Multiple responses to the same event appear to be a contradiction in terms. However, the multiple transitions are not meant to be taken literally. Instead this is a modeling abstraction that still produces the same final behavior in the contract or other system, but which can typically be expressed more concisely. See Sipser (2006, ch. 1) for further discussion.

Any state-transition system with a DFA representation can be converted to an equivalent (in terms of its acceptable sequences of events) NFA and vice versa. It is already clear from our preliminary forays into standard financial master agreements that this sort of flexibility will be useful.

The DFA captures the central legal logic of the contract. However, the DFA by itself does not capture the entirety of the agreement. There are, in addition, two import semantic conversions that round out the picture. First, a "measurement" or "feature extraction" process is defined by the agreement to convert from the salient events and occurrences in the real world to the tidy, finite microcosm of the DFA. For example, the borrower represents that his assets exceed his liabilities under generally accepted accounting principles. The DFA requires this simple determination — yes or no — whether the borrower is solvent, encoding this fact as an alphabet element. The measurement process to reach this decision will typically involve a multitude of assumptions, interpretations and judgments of accounting, but the bottom line will have the full clarity of a discrete, binary variable. Requiring the counterparties to maintain this clarity is a valuable function of the contract. The translation of external occurrences into the event alphabet is still mostly handled by the natural-language definitions in the contract. The details and nuances of this measurement task have been excluded from the scope of the present paper, but they are an important area for follow-on research. The work on ontologies described above is one example of some of the steps this process.

Second, there is a question of the semantic interpretation of the states and transitions in the automaton. For example, when the automaton is in state "xL", this fact about the DFA has important implications for the parties back in the real world; one of the parties is likely to be filing litigation in a specified jurisdiction, requiring documentation of claims, etc. Some interpretative mechanism is required to understand that all of these messy contextual details are entailed by the simple marker, xL. This explicit mapping from the DFA to the external legal context is a sort of formal semantics that requires additional attention in subsequent research.[63] Some of the old challenges — and perhaps advantages — of ambiguity and lawyer brains creep back into the process through the use of natural language queries as the basis for specifying an alphabet triggering event.

As an aside, we note that one standard technique for building state-transition systems is to focus on the entity-specific states that might describe each participant individually in a system of interacting entities, such as contractual counterparties. The state of the relationship is then assembled from all the possible combinations of the individual states. This is called a "product automaton."[64] For example, each spouse in a marriage might individually be in the state happy (H) or unhappy (U), so that the state of the marriage is described by one of the product states: HH, UH, HU, or UU. The work we present here suggests that the law does not work this way. That is, a financial contract does not attempt to describe the state of the counterparties directly and then combine these counterparty-specific states into their various interactions. Rather, the contract reifies the relationship as a distinct thing. The states and

[63] For an introduction to computational logics and formal semantics, see Huth, Michael, and Mark Ryan. *Logic in Computer Science: Modelling and Reasoning about Systems*. 2nd ed. Cambridge, U.K.: Cambridge University Press, 2004.

[64] See Hopcroft, J., R. Motwani, and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. 2nd ed. Boston: Pearson/Addison Wesley, 2001. 38-35.

transitions between states that structure the contract then describe the relationship object itself. Abstractly, one might instead establish a structural equivalence between the states of the relationship and those of a product automaton. For example, one might re-label the product states suggested above to say that the marital relationship is in a "successful" state if the product state of the spouses is HH. As a practical matter, however, attempting to model the state-transition structure of a financial contract as a product automaton is awkward and ultimately counterproductive. For example, it does not generally matter to a contract if one party is secretly flirting with default, or the other is engaged in undetected activities that might (if caught and prosecuted) provoke mistrust. Modeling these sorts of party-specific states is typically a distraction. The agreement defines its own universe of relationship-specific states and events — which may include party-specific states where needed, for example as covenants, representations, and warranties.

The availability of a fully developed and well understood formalism such as the DFA for capturing the deep structure of financial agreements is likely to further energize the ongoing trend toward contract automation, previously discussed in Section III. The fact that humans will retain a role, perhaps in the interpretation of input events or the state of the relationship, does not mean that the boundary between manual and automated tasks will not shift. Traditional tasks of lawyering will continue the march toward automation.[65] In this evolution, we should reserve for humans those tasks for which we retain an absolute or comparative advantage, such as those requiring nuanced judgment over legal or ethical dilemmas or strategic financial goals.

In contrast, although parsing the logic of a complex financial contract is traditionally among the hardest of the tasks of interpretation, the availability of a robust computational model will expose parts of that task to automation. For example, one might develop automated reasoners to discern whether a draft agreement is complete in the sense of being able to accept certain key event sequences. Similarly, one might develop objective criteria for automatically scoring the computational or transactional complexity of a legal agreement, on either a relative (to other, similar agreements) or an absolute scale.

We emphasize that the DFA is a high-level system diagram setting out the logical structure of the contract. It is not intended to be a procedural flowchart for automating the relationship. [66] Indeed, the DFA would be a relatively cumbersome form for a computation engine, as each step must be set out in

[65] Engineering for human-in-the-loop systems is an established field, and there is an extensive literature on human-computer interaction (HCI). See for example M.G. Helander, T.K. Landauer, and P.V. Prabhu, eds., Sears, Andrew. *The Human-computer Interaction Handbook Fundamentals, Evolving Technologies, and Emerging Applications*. 2nd ed. New York: Lawrence Erlbaum Associates, 2007, or the articles in the ACM Transactions on Computer-Human Interactions (TOCHI), at: "Home." TOCHI. Accessed December 13, 2014. http://tochi.acm.org/. More generally, Hamid Ekbia and Bonnie Nardi refer to human-machine interdependence as "heteromation; see Ekbia, Hamid, and Bonnie Nardi. "Heteromation and Its (dis)contents: The Invisible Division of Labor between Humans and Machines." *First Monday* 19, no. 6 (2014). available at http://firstmonday.org/ojs/index.php/fm/article/view/5331).

[66] At the implementation level, many contracts will also involve some calculation chores, such as determining precise payment amounts, sorting through holiday calendars and day-count conventions, etc. The streamlined agreement elides these considerations by explicitly stating precise dates and dollar amounts. In general, these sorts of calculations would and should typically be the implementation details of some delegated subsystem and need not detain us here.

order, without the availability of memory or recursive operations to reduce the complexity of the representation. Actual implementation of more complex financial contracts will use these shortcuts. Nonetheless, the DFA is a good starting point for highlighting the conceptual link between contracting and computation.

An especially important extension of the state-transition model applies to financial agreements that interact through the events they consume. A finite-state transducer (FST) is an enhanced DFA that allows transitions to *emit* events. That is, contracts are not always mere consumers of events; contracts can generate events as well. Moreover, the output events for one agreement may be input events for another. A canonical example is a cross-default clause, which specifies that one contract should "listen" for transitions to default states as they occur in another agreement. Cross-default clauses can have systemic implications, because they typically trigger payments acceleration, creating a legal mechanism for the propagation of default across a network of contracts. A standard FST is a 6-tuple augments the basic DFA by equipping it with an output alphabet, often represented as Γ. The transition function, δ, is similarly augmented, so that a transition is associated with a character from both the input alphabet, Σ , and the output alphabet, Γ.[67] The FST model also may help represent how to deliver information about particular transitions to the parties themselves, even when there are not systemic consequences.

VII. Implications and Conclusions

There are a number of potentially significant implications that flow from this exercise in computational contract specification. At the most basic level, the exercise of stating even a simplified contract as a DFA serves as a proof of concept, demonstrating that we can describe legal rule and consequence structures directly in computational terms. Of course, success here does not prove that this modeling technique would apply to all contracts, including those of much greater complexity. At this stage, however, we see no conceptual barrier to such a task, provided the contracts are reasonably bounded in their terms.

A practical complication that we consciously excluded from the scope of our streamlined contract was any sort of internal governance mechanism to manage the relationship in the face of future uncertainty. Most financial contracts do not have internal governance provisions, and so can be expected to be susceptible to DFA representation. We note that the litigation exit specification in our example in effect moves the result determination out of the contract itself when faced with a court proceeding. This is a necessary step; such a proceeding constitutes just such a governance mechanism. As discussed below, tackling more complex contracts is a logical next step; our work here both suggests methods for doing that and provides hope that the effort involved will be time successfully invested.

[67] Transducers are widely used in control systems, including computer hardware design, and in computational linguistics. The control applications are closer to our case of computable contracts; see Mueller, Silvia M., and Wolfgang J. Paul. *Computer Architecture Complexity and Correctness*. Berlin: Springer, 2000. There are two general classes of FSTs, called Moore machines and Mealy machines, which differ essentially in whether outputs can (Mealy) or cannot (Moore) depend on the input event that triggered the transition. See Moore, Edward F (1956). "Gedanken-experiments on Sequential Machines". Shannon, Claude Elwood, and John McCarthy. *Automata Studies*. Vol. 34. Princeton: Princeton University Press, 1956. 129-153.; and Mealy, George H. "A Method for Synthesizing Sequential Circuits." *Bell System Technical Journal*, 1955, 1045-079. It remains a topic for future research which of these structures would best suit the analysis of computable contracts.

The exercise of representing contracts as DFAs can help us better understand how contracts work. Seeing the graphical representation of the paths that flow from breaches of warranties and covenants, for instance, have helped us to conceptualize the role they play in financial contracts, effectively shifting the deal from a chain of original expectation and timing to a new agreement that is generally time- and value-collapsed. We do not claim that such insights could not be gained by looking just at the natural language version, but they leap out from the graphical version.

The requirements of technologically based progress present implications for legal education. Law schools of the future would be well advised to teach legal coding as well as legal writing. Acquiring such knowledge seems daunting to many of us now in the law, but it is probably less challenging and more logically consistent than having to learn the intricacies of Blue Book citation.

The DFA exercise also gives formal expression to our categorization of contractual functions into actions (states), events (alphabet), calculations and chains of logical application (transformations). This division helps to clarify where specificity and completeness is in fact a virtue, and in the process helps to resolve some of the apparent conflicts between ambiguity-rooted discretion and legal automation.

Taken a step further, using the DFA as a starting point to conceptualize and design a contract provides the drafter with a more controlled pallet for representing the logic of a transaction. Those of us who grew up before the spread of personal computing can recall the struggle to master the operating systems and user interfaces of the new machines.[68] Mastery of basic skills, such as launching programs and saving files, quickly progressed to higher-level challenges, such as configuring programs and organizing file system directories, etc. A similar transition awaits the contract drafters of the future, as such tools as promise, default, and acceleration become deployable as explicit steps in a chain within a state machine. This vision of starting the drafting of contracts in formal computational representation suggests a different line of research from those looking to use learning algorithms or other parsing tools to abstract computable meaning from word based formulations.

Although embodying financial contracts in software has the potential to provide significant benefits, we also recognize that the increases in speed, accuracy and flexibility that this development will provide will have the potential to create problems as well. In an automotive analogy, a significant increase of power and acceleration in the motor probably needs to be balanced by increased efficiency in the brakes. The use of the use of one of the simplest computational formalisms (the DFA) in our streamlined example is intentional in this regard. There is a danger of the "Sorcerer's Apprentice" problem, that the unwise application of powerful computational tools could encourage inexperienced drafters with only limited understanding of the issues involved to create contracts of unmanageable complexity.[69]

One of the more significant implications of this exercise is that the link between law and formal computation is not a one-way street. If we can restate a natural language contract as a DFA as we have done here, then we may reasonably conclude that the natural language original was itself a statement of

[68] The machines also struggled to adapt. The same generation of users will recall Microsoft Windows' infamous "blue screen of death," which signaled a total operating system crash, typically due to an untrapped exception.
[69] Computer systems designers worry actively about the problem of state space explosion (i.e., proliferation). See for example Baier and Katoen, supra note 21, esp. ch. 2.

a DFA. The sometimes tortured prose of legalese, from this standpoint, can be viewed as enabling the use of natural language to create a word-based automaton. If this is so, perhaps we can turn the analytic tools of computation theory onto legal systems as they exist now and not just on the formal representations that may contain our laws in the future. This extension of computational approaches into our natural language formulations suggests that computational analysis may extend beyond the law, into further aspects of human cognition, expression, and behavior. Computational neuroscience, computational psychology, and computational economics are already fields of study, and computational law may have many companions as an applied science of thought and behavior.[70] Work in the existing field of natural language processing often borders on such research, but focuses more on algorithms for extracting meaning from natural language (a version of the feature extraction problem cited above) rather than on viewing natural language itself as a computational model.[71]

Representing a contract in computational terms also has connections with research in game theory and institutional design. It is possible to view the role of contracts and of many forms of legal, cultural and biological institutions more generally, as means of re-denominating the game structure of potentially cooperative interactions so as to move from predatory to mutually beneficial outcomes. [72] If this is so, and if a contract can be restated as a DFA, then we may also be able to represent the architecture of strategic games more generally in computational terms. [73] As we have suggested, contracts are a subset of institutions, in the sense of rule sets that re-denominate the strategic landscape of cooperation problems. The fact that at least some contracts can be restated as DFAs suggests that institutions more broadly — and perhaps the strategic structures of game theory generally — may be susceptible to similar treatment. An equivalence between game theory and linear programming has long been

[70] See for example Bower, James M. *20 Years of Computational Neuroscience*. New York, NY: Springer, 2013.; Sun, Ron. *The Cambridge Handbook of Computational Psychology*. Cambridge, UK: Cambridge University Press, 2008.; Tesfatsion, Leigh, and Kenneth Judd. *Handbook of Computational Economics*. Amsterdam: Elsevier, 2006.); and, generally, the series Schmedders, Karl. *Handbook of Computational Economics*. Amsterdam: North Holland, 2014., available at http://www.sciencedirect.com/science/handbooks/15740021, and "Computational Economics - Springer." Computational Economics - Springer. Accessed December 13, 2014. http://link.springer.com/journal/10614.

[71] See for example Cambria, Erik, and Bebo Whitel. "Jumping NLP Curves: A Review of Natural Language Processing Research." *IEEE Computational Intelligence Magazine* 48 (2014).available at http://sentic.net/jumping-nlp-curves.pdf. But see, Liu, Hugo, and Henry Lieberman. "Metafor: Visualizing Stories as Code." 2005., part of the proceedings of IUI'05, January 10–13, 2005, San Diego, California, USA, available at http://web.media.mit.edu/~hugo/publications/papers/IUI2005-metafor.pdf

[72] See for example Zak, Paul J. *Moral Markets the Critical Role of Values in the Economy*. Princeton: Princeton University Press, 2008.

[73] Nisan, Noam, Vijay V. Vazirani, Tim Roughgarden, and Eva Tardos. *Algorithmic Game Theory*. Cambridge: Cambridge University Press, 2007. http://www.cambridge.org/journals/nisan/downloads/Nisan_Non-printable.pdf; Dütting, Paul, and Andreas Geiger. "Seminar Report: Algorithmic Mechanism Design." May 9, 2007. Accessed December 13, 2014. http://www.cs.uu.nl/docs/vakken/msagi/mech_design.pdf. and Nisan, Noam. "Algorithmic Mechanism Design: Through the Lens of Multi-unit Auctions." January 21, 2014. Accessed December 13, 2014. http://www.cs.huji.ac.il/~noam/amd-mua.pdf.

recognized.[74] This would help validate the approach and would be a potential mine for design elements.[75]

The analysis of a streamlined agreement initiates a larger project of automating financial instruments. An obvious next step will be to undertake such a description for existing agreements actually used in financial markets, such as the 2002 ISDA Master Agreement and the 1997 International Foreign Exchange Master Agreement. While the proof necessarily awaits the exercise, a preliminary review of these agreements suggests that the challenges of restating them as DFA are those of time and patience in the face of complexity, rather than of fundamental differences of kind. Such a step would be a precursor to creating a software version of these instruments; it could also suggest re-drafting opportunities. We can imagine an updated "2016 ISDA Master Agreement" that is completely computable. In addition to restating existing natural language contracts as DFA, another plausible next step would be a project of new drafting, where the goal is to embody transactional structures straight into the formalism of computation and software, without relying on a natural language precursor. Computational drafting is likely to be a necessary skill for some lawyers of the future.

The DFA representation of financial contracts should also assist in additional projects seeking to make financial firms and markets more robust in the face of risk and disruption. For instance, the Office of Financial Research (OFR) of the U.S. Department of the Treasury is working to create a library of contingency clauses commonly used in financial instruments. The frequent function of such clauses, particularly those linked to events of termination or default, is to change the pathways of result from one anticipated chain of calculation and performance to another. DFA representations, particularly the graphical approach, can help us clarify these pathways and increase our understanding.

Another possible application of the state-transition approach is to restate other legal formulations such as statutes and regulations in DFA terms. One possible target is Regulation D under the Securities Act of 1933. Reg. D sets out a safe harbor from the registration requirements of the 1933 Act for private placement transactions. It is relatively self-contained, and it appears that the cross-references with the larger world of SEC regulation and beyond can be handled through event definition shortcuts in the feature extraction exercise. A broader target might include the Federal Rules of Civil Procedure; it is interesting to speculate on whether the procedural aspect generally of a formal litigation is itself describable as a state machine. In this sense, a summary judgment proceeding or even a full trial might also be seen as a computation, with the states and transitions perhaps defined by the elements of the

---

[74] Brickman, Louis. *Mathematical Introduction to Linear Programming and Game Theory*. New York: Springer-Verlag, 1989.; Thie, Paul R., and G. E. Keough. *An Introduction to Linear Programming and Game Theory*. 3rd ed. Hoboken, N.J.: Wiley, 2008. Harold W. Kuhn, in the introduction to the 2004 reissue of the Theory of Games notes, "By the end of the summer [of 1948], we had established that, mathematically, linear programming and the theory of zero-sum two-person games are equivalent." Neumann, John, and Oskar Morgenstern. *Theory of Games and Economic Behavior*. 60th Anniversary ed. Princeton: Princeton University Press, 2007. http://press.princeton.edu/chapters/i7802.pdf.

[75] Fernando Bevilacqua notes that, "aspects of agent-based behavior have been modeled in state machine terms." See "Finite-State Machines: Theory and Implementation - Tuts Game Development Tutorial." Game Development Tuts. October 24, 2013. Accessed December 13, 2014. http://gamedevelopment.tutsplus.com/tutorials/finite-state-machines-theory-and-implementation--gamedev-11867.

cause of action and the rules of evidence and procedure, and with the alphabet provided by the evidence presented.

As this discussion demonstrates, the application of computational approaches to legal formulation, whether in contracts or more broadly, is not just a matter of metaphor. The tools of computation theory have direct application in the specification of a system of ordered and defined rights and obligations. Putting computational tools to work in the context of law and justice has the potential to revolutionize aspects of the legal system, with significant consequences for markets, government, and society. Law is being “Turing’d”; it is time to recognize the fact and to work to make the process as beneficial as possible.

TABLES

Table 1: Ingredients of a Typical Nonequity-style Contract

Counterparties	A specification of the parties to the agreement and the date of its formation and effectiveness. Specified means of communication between the parties.
Basic obligations	One or more obligations for payment, transfer or other financial performance, together with the calculations that specify the amount or extent of this obligation. A fixed date, or means for fixing the date, for the execution of the obligations for payment, transfer or other financial performance. A calculation of interest, exchange, or some other rate of return. Covenants, which provide promises of things which a party is obligated to do or obligated to refrain from doing, generally not including the underlying financial obligations described above. Warranties, which provide a promise about a state of facts; these have the effect of creating factual parameters for the transaction and of allocating the risks for a failure of the facts to conform to those parameters. The provision of security, either through a guarantee by another entity or through the dedication of some asset as a source of value if the transaction goes into default.
Default provisions	A procedure for declaring a default, which can be based on a failure to meet the obligations of performance, including the covenants, a failure of a warranty to be accurate either when made or on a continuing basis, or other specified events, such as bankruptcy or the default under other obligations (cross-default). Additional rights granted the "innocent" party in the case of a default, including the acceleration of obligations and the authorization of protective measures (these rights in effect re-denominate the contract). The imposition of penalties, including increased and continuing rates of interest, in case of a default. Avenues for proceeding against the security of collateral or guarantee. Procedures for going forward if certain specified or unanticipated events disrupt the transaction (force majeure).
Enforcement	A procedure for resolving disputes between the parties (courts, arbitration, etc.). A procedure for enforcing the obligations between the parties. A procedure for making the nondefaulting party whole (indemnity) for the costs of invoking those procedures. A specification of what legal regime will govern the transaction (choice of law). A procedure for waiver, amendment, renegotiation and agreed termination of the agreement.

Table 2: A Streamlined Loan Agreement

Agreement

This loan agreement dated June 1, 2014, by and between Lender Bank Co. ("Lender") and Borrower Corp. (Borrower), will set out the terms under which Lender will extend credit in the principal amount of $1,000 to Borrower with an un-compounded interest rate of 5% per annum, included in the specified payment structure.

1. The Loan:
At the request of Borrower, to be given on June 1, 2014, Lender will advance $1000 to Borrower no later than June 2, 2014. If Borrower does not make such a request, this agreement will terminate.

2. Repayment:
Subject to the other terms of this agreement, Borrower will repay the loan in the following payments:

(a) Payment 1, due June 1, 2015, in the amount of $550, representing a payment of $500 as half of the principal and interest in the amount of $50.

(b) Payment 2, due June 1, 2016, in the amount of $525, representing a payment of $500 as the remaining half of the principal and interest in the amount of $25.

3. Representations and Warranties:
The Borrower represents and warrants, at the execution of this agreement, at the request for the advance of funds and at all times any repayment amount shall be outstanding, the Borrower's assets shall exceed its liabilities as determined under an application of the FASB rules of accounting.

4. Covenants:
The Borrower covenants that at the execution of this agreement, at the request for the advance of funds and at all times any repayment amount shall be outstanding it will make timely payment of all state and federal taxes as and when due.

5. Events of Default:
The Borrower will be in default under this agreement upon the occurrence of any of the following events or conditions, provided they shall remain uncured within a period of two days after notice is given to Borrower by Lender of their occurrence (such an uncured event an "Event of Default"):

(a) Borrower shall fail to make timely payment of any amount due to Lender hereunder;

(b) Any of the representation or warranties of Borrower under this agreement shall prove untrue;

(c) Borrower shall fail to perform any of its covenants under this agreement;

(d) Borrower shall file for bankruptcy or insolvency under any applicable federal or state law.

A default will be cured by the Borrower (i) remedying the potential event of default and (ii) giving effective notice of such remedy to the Lender. In the event of multiple events of default, the first

to occur shall take precedence for the purposes of specifying outcomes under this agreement.

6. Acceleration on Default

Upon the occurrence of an Event of Default all outstanding payments under this agreement will become immediately due and payable, including both principal and interest amounts, without further notice, presentment, or demand to the Borrower.

7. Choice of Law:

This agreement will be subject to the laws of the State of New York applicable to contracts entered into and performed wholly within that state.

8. Amendments and Waivers:

Any purported amendment to, or waiver of rights under, this agreement will only be effective if set forth in writing and executed by both parties.

9. Courts and Litigation:

Any legal action brought to enforce, interpret or otherwise deal with this agreement must be brought in the state courts of the State of New York located in New York County, and each of the parties agrees to the jurisdiction of such courts over both the parties themselves and over the subject matter of such a proceeding, and waives any claim that such a court may be an inconvenient forum.

10. Time of the Essence; No Pre-Payment

Timely performance is required for any action to be taken under this agreement, and, except as may otherwise be specifically provided herein, failure to take such action on the day specified will constitute a binding failure to take such action. Payments shall only be made on or after the dates specified in Section 2 or on or after such other date as may be required under Section 6; pre-payments made on earlier dates shall not be accepted.

11. Notices

Notices provided for in this agreement will be given by an email to the email addresses set out below and will be effective upon receipt.

[Lender email here]

[Borrower email here]

Accepted and agreed:

LENDER BANK CO.	BORROWER CORP.
By: _____	By: _____
Title: _____	Title: _____

[NOTE: Statute of Limitations on debt obligations in NY is 6 years]

Draft of July 23, 2014

Table 3: Contract States (Q)

State	Label	Natural Language Consequences and Correlates (Λ)	Sec
start ☺	Start	Contract is fully specified; key information (payment dates, notice addresses and procedures, choice of law, and dispute process) delivered	7, 9, 11
q0 ☺	Active contract	Contract is fully signed/executed	
q1 ☺	Principal requested	Borrower's has requested and awaits $1,000	1
P1 ☺	Payment 1 accruing		
P1d ☺	Payment 1 due		
P2 ☺	Payment 2 accruing		
P2d ☺	Payment 2 due		
Dl	Default (lender)	Lender has failed to deliver principal	5
Acc1	Payments 1 and 2 accelerating	Accelerated payment due is $1,075	6
Acc2	Payments 2 accelerating	Accelerated payment due is $525	6
Db0_1	Default (borrower) payment missed	Borrower has failed to make first payment on time and should be notified	5
Dbcv_1	Default (borrower) covenant	Borrower violates covenant(s) and should be notified	5
Dbrw_1	Default (borrower) representations/warranties.	Borrower breaches representations or warranties and should be notified	5
Dbbkr_1	Default (borrower) bankruptcy	Borrower files for bankruptcy or insolvency and should be notified	5
Nb0_1 Δ	Borrower notified of payment default	Borrower has two days to pay, or all payments accelerate	5
Nbnpd_1 Δ	Borrower notified of general default	Borrower has two days to pay, or all payments accelerate	5
Db0_2	Default (borrower) payment missed	Borrower has failed to make first payment on time and should be notified	5
Dbcv_2	Default (borrower) covenant	Borrower violates covenant(s) and should be notified	5
Dbrw_2	Default (borrower) representations/warranties	Borrower breaches representations or warranties and should be notified	5
Dbbkr_2	Default (borrower) bankruptcy	Borrower files for bankruptcy or insolvency and should be notified	5
Nb0_2 Δ	Borrower notified of payment default	Borrower has two days to pay or all payments accelerate	5
Nbnpd_2 Δ	Borrower notified of general default	Borrower has two days to pay or all payments accelerate	5
xT †	TERM	Contract is fulfilled in accordance with its terms	
xL †	LIT	A legal action is brought to enforce, interpret, or otherwise deal with the agreement in the state courts of the state of New York located in New York County that the results of this action will replace the computation of the contract	9
xC †	CANC	Contract is canceled due to the passing of time beyond the statute of limitations or canceled because of modification or termination by mutual agreement of the parties	8
Crisis1	Crisis1 — accelerated payments not made	Payments accelerated, but borrower has not responded	6

Crisis2	Crisis2 — accelerated payments not made	Payments accelerated, but borrower has not responded	6
☺ *States on the "happy" path of the contract lifecycle* Δ *Default states* † *Terminal states*			

Table 4: Event Alphabet (Σ)

ID	Label	Natural Language Event Specification	Section
A	Contract signed	Contract is signed to bind all parties	
B	1 Day passes since last event	June 1, 2014, passes	1
C	Money requested	Borrower gives request for loan of $1,000	1
D	Lawsuit	A legal action is brought to enforce, interpret, or otherwise deal with the agreement in the state courts of the state of New York located in New York County.	
E	Statute of limitations	June 1, 2020, passes — the Statute of Limitations on debt obligations in New York is six years	
F	Principal advanced	Lender advances $1,000 no later than June 2, 2014	1
G	June 1, 2015, passes	Payment 1 due on June 1, 2015	2(a)
H	Representations/warranties	The borrower's assets exceed its liabilities as determined under an application of the FASB rules of accounting	3, 5(b)
I	Covenant	The borrower fails to make a timely payment of an amount of state or federal tax	4, 5(c)
J	Bankruptcy	The borrower files for bankruptcy or insolvency under any applicable federal or state law	
K	Notice given	Notice given to borrower of a failure to make timely payment of an amount due to lender under this agreement	5
L	Notice given of general default	Notice given to borrower of an event of default other than a failure to make timely payment of an amount due	5
M	Payment default cured	A payment-related event of default is cured	5
N	General default cured	A nonpayment related event of default is cured	5
O	2 Days pass since last event	Two days elapse since last event/notice	5
P	June 1, 2016, passes	Payment 2 is due on June 1, 2016	2(b)
Q	Payment made $550		
R	Payment made $525		
S	Payment made $1075		
T	Cancel or modify	Contract in this form is canceled because of modification or termination by mutual agreement of the parties	8

Table 5: Transition Function (δ)

Initial State	Event	Resulting State
start ☺	A	q0
q0	B	xT
q0 ☺	C	q1
q1	B	Dl
Dl	D	xL
Dl	E	xC
q1 ☺	F	P1
P1 ☺	G	P1d
P1d	B	Db0_1
P1	H	Dbrw_1
P1	I	Dbcv_1
P1	J	Dbbkr_1
Db0_1	K	Nb0_1
Dbcv_1	L	Nbnpd_1
Dbbkr_1	L	Nbnpd_1
Dbrw_1	L	Nbnpd_1
Nb0_1	Q	P2
Nbnpd_1	N	P1
Nb0_1	O	Acc1
Nbnpd_1	O	Acc1
Acc1	B	Crisis1
Acc1	S	xT
Crisis1	E	xC
Crisis1	D	xL
Crisis1	S	xT
P1d ☺	Q	P2

(Continued)

Initial State	Event	Resulting State
P2 ☺	P	P2d
P2d	B	Db0_2
P2	H	Dbrw_2
P2	I	Dbcv_2
P2	J	Dbbkr_2
P2d ☺	R	xT
Db0_2	K	Nb0_2
Dbcv_2	L	Nbnpd_2
Dbbkr_2	L	Nbnpd_2
Dbrw_2	L	Nbnpd_2
Nb0_2	R	xT
Nbnpd_2	N	P2
Nb0_2	O	Acc2
Nbnpd_2	O	Acc2
Acc2	B	Crisis2
Acc2	R	xT
Crisis2	E	xC
Crisis2	D	xL
Crisis2	R	xT

☺ *Transitions along the "happy" path of the contract lifecycle*

As noted above, only the transitions that result in a change of states are noted here; all un-noted transitions result in the state being unchanged.

FIGURES

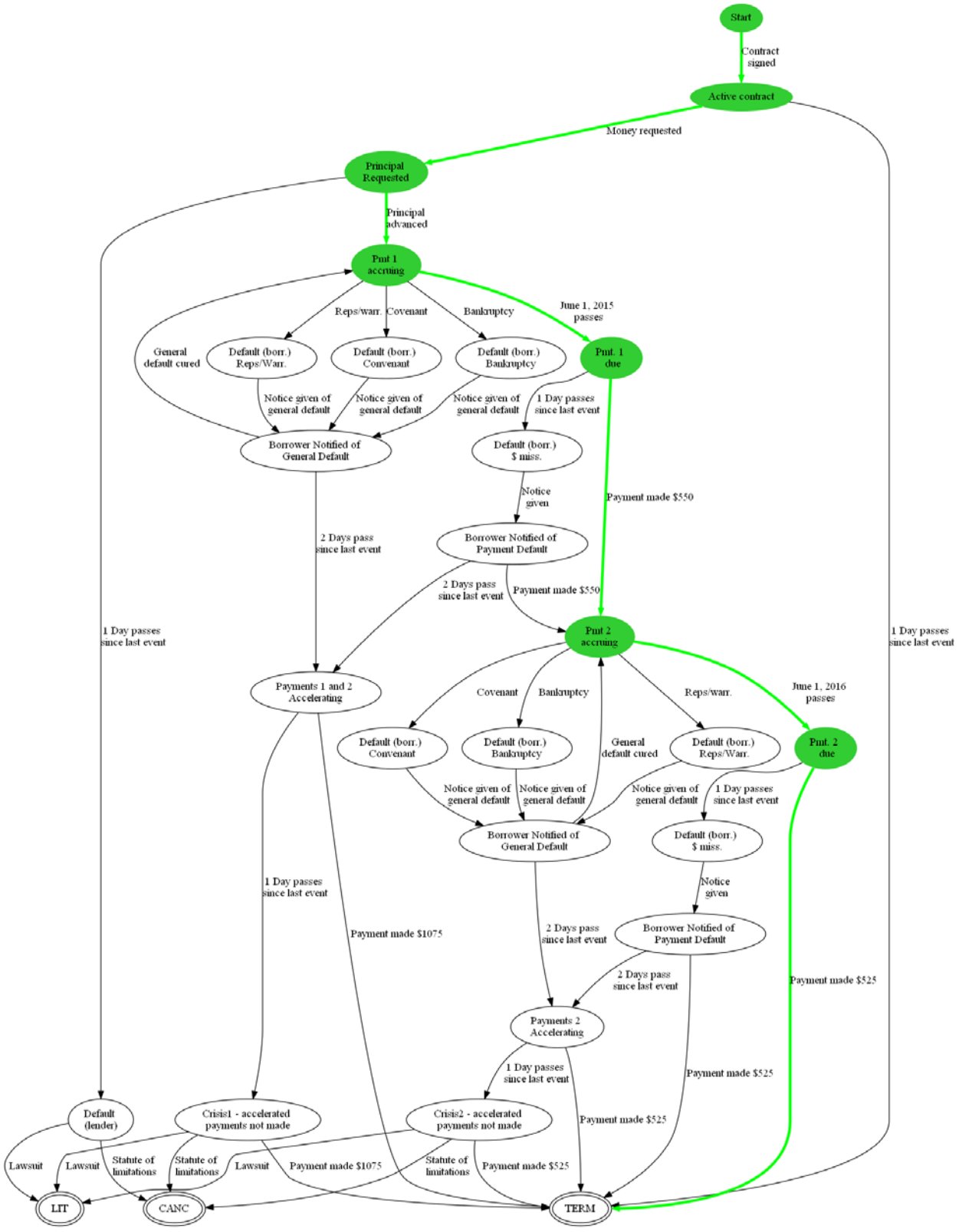

Figure 1: Graphical Representation of the DFA for the Streamlined Contract

References

Abreu, D. & Rubinstein, A. (1988), 'The Structure of Nash Equilibrium in Repeated Games with Finite Automata', *Econometrica*, 1259-281.

Aho, A. V.; Lam, M. S.; Sethi, R. & Ullman, J. D. (2007), *Compilers: Principles, Techniques, and Tools, 2nd edition*, Pearson/Addison Wesley.

Aho, A. V. & Ullman, J. D. (1995), *Foundations of Computer Science*, Computer Science Press. http://infolab.stanford.edu/~ullman/focs.html.

Allen, L. (1957), 'Symbolic Logic: A Razor-Edged Tool for Drafting and Interpreting Legal Documents', *Yale Law Journal* 66, 833-79.

Association for Computing Machinery (2014), *ACM Transactions on Computer-Human Interactions (TOCHI)*, ACM. http://tochi.acm.org/.

Baier, C. & Katoen, J. (2008), *Principles of Model Checking*, MIT Press.

Bank for International Settlements (2013), 'Triennial Central Bank Survey Foreign Exchange Turnover in April 2013: Preliminary Global Results', Technical report, BIS, Accessed December 13, 2014. http://www.bis.org/publ/rpfx13fx.pdf.

Bench-Capon, T.; Araszkiewicz, M.; Ashley, K.; Atkinson, K.; Bex, F.; Borges, F.; Bourcier, D.; Bourgine, P.; Conrad, J. G.; Francesconi, E.; Gordon, T. F.; Governatori, G.; Leidner, J. L.; Lewis, D. D.; Loui, R. P.; McCarty, L. T.; Prakken, H.; Schilder, F.; Schweighofer, E.; Thompson, P.; Tyrrell, A.; Verheij, B.; Walton, D. N. & Wyner, A. Z. (2012), 'A History of AI and Law in 50 Papers: 25 Years of the International Conference on AI and Law Artificial Intelligence and Law', *Artificial Intelligence and Law* **20**(3), 215-319.

Bevilacqua, F. (2013), 'Finite-State Machines: Theory and Implementation - Tuts Game Development Tutorial', *Game Development Tuts*, Accessed December 13, 2014. http://gamedevelopment.tutsplus.com/tutorials/finite-state-machines-theory-and-implementation--gamedev-11867.

Binmore, K. G. & Samuelson, L. (1992), 'Evolutionary Stability in Repeated Games Played by Finite Automata', *Journal of Economic Theory*, 278-305.

See Blaszkowsky, David M., and Reed, Matthew P. (2012), “Meta-What? Lawyers, Legal Training and the Rise of Meta-Data for Digital Securities and Other Financial Contracts,” in: O. Goodenough and M. Lauritsen, eds., *Educating the Digital Lawyer*, New Providence, NJ, LexisNexis.

Bolton, P. & Dewatripont, M. (2005), *Contract Theory*, MIT Press.

Bower, J. M. (2013), *20 Years of Computational Neuroscience*, Springer.

Brammertz, W. (2009), *Unified Financial Analysis: The Missing Links of Finance*, Wiley.

Brickman, L. (1989), *Mathematical Introduction to Linear Programming and Game Theory*, Springer.

Brose, M. S.; Flood, M. D.; Krishna, D. & Nichols, B. (2013), *Handbook of Financial Data and Risk Information Vol. I: Principles and Context*, Cambridge University Press. http://www.cambridge.org/sg/academic/subjects/mathematics/mathematical-finance/handbook-financial-data-and-risk-information-i-principles-and-context-volume-1.

Brose, M. S.; Flood, M. D.; Krishna, D. & Nichols, B. (2013), *Handbook of Financial Data and Risk Information Vol. II: Software and Data*, Cambridge University Press. http://www.cambridge.org/sg/academic/subjects/mathematics/mathematical-finance/handbook-financial-data-and-risk-information-ii-software-and-data-volume-2.

Cambria, E. & Whitel, B. (2014), 'Jumping NLP Curves: A Review of Natural Language Processing Research', *IEEE Computational Intelligence Magazine* 48. http://sentic.net/jumping-nlp-curves.pdf.

Center for Computer-Assisted Legal Instruction (2014), 'A2J Author', CALI, Accessed December 13, 2014. http://www.cali.org/content/a2j-author.

Clark, C. (2011), 'Whither Equity Volumes?', January 31, 2011. *NYSE Exchanges*. http://exchanges.nyx.com/cclark/whither-equity-volumes.

Computational Legal Studies (2014), 'Computational Legal Studies', Accessed December 13, 2014. http://computationallegalstudies.com/.

Cordell, L.; Huang, Y. & Williams, M. (2011), 'Collateral Damage: Sizing and Assessing the Subprime CDO Crisis'(11-30), Technical report, FRB of Philadelphia. http://www.philadelphiafed.org/research-and-data/publications/working-papers/2011/wp11-30.pdf.

Crawford, S. E. S. & Ostrom, E. (1995), 'A Grammar of Institutions', *The American Political Science Review*, 582-600.

Dickey, T. E. (2014), 'Berkeley Yacc (byacc)', *Invisible Island*, Accessed December 13, 2014. Generate LALR(1) Parsers. http://invisible-island.net/byacc/byacc.html.

Drobac, J. A. & Goodenough, O. R. (2015), 'Exposing the Myth of Consent', *Indiana Health Law Review*.

Duffie, D. (2001), *Dynamic Asset Pricing Theory. 3rd ed.*, Princeton University Press.

Dütting, P. & Geiger, A. (2007), 'Seminar Report: Algorithmic Mechanism Design', Accessed December 13, 2014. http://www.cs.uu.nl/docs/vakken/msagi/mech_design.pdf.

Ekbia, H. & Nardi, B. (2014), 'Heteromation and Its (dis)contents: The Invisible Division of Labor between Humans and Machines', *First Monday* **19**(6). http://firstmonday.org/ojs/index.php/fm/article/view/5331.

Engle, E. A. (2004), 'An Introduction to Artificial Intelligence and Legal Reasoning: Using XTalk to Model the Alien Tort Claims Act and Torture Victim Protection Act', *Richmond Journal of Law and*

Technology **11**(2).

Exari (2014), 'About Us', Exari, Accessed December 13, 2014. http://www.exari.com/About-Us.html.

Foreign Exchange Committee (1997), 'The 1997 International Foreign Exchange Master Agreement (IFEMA)', Technical report, Federal Reserve Bank of New York, Accessed December 12, 2014. http://www.newyorkfed.org/fmlg/ifema.pdf.

Friedl, J. E. F. (2006), *Mastering Regular Expressions. 3rd ed.*, O'Reilly.

Fudenberg, D. & Tirole, J. (1991), *Game Theory*, MIT Press.

FutureLaw 2013 (2013), 'Computational Law and Contracts', *Stanford CodeX program*, Panel Discussion. http://www.youtube.com/watch?v=KBI8_tv2VDM.

Gintis, H. (2009), *The Bounds of Reason: Game Theory and the Unification of the Behavioral Sciences*, Princeton University Press.

GNU Project (2014), 'Bison', Free Software Foundation, Accessed December 13, 2014. https://www.gnu.org/software/bison/.

Goodenough, O. R. (2013), 'Governance for Cloud Computing: The Role of Public and Private Rulemaking in Promoting the Growth of a New Industry'(34-13), Technical report, Vermont Law School. http://ssrn.com/abstract=2342594.

Goodenough, O. R. (2011), 'Digital Firm Formation', Ch. 14 in: *Rules for Growth, Promoting Innovation and Growth Through Legal Reform*, Kauffman Task Force on Law, Innovation and Growth, . http://www.kauffman.org/~/media/kauffman_org/research%20reports%20and%20covers/2011/02/rulesforgrowth.pdf.

Goodenough, O. R. (2008), Values, Mechanism Design, and Fairness", in Zak, Paul J., ed., *Moral Markets the Critical Role of Values in the Economy*, Princeton University Press, , pp. 228-255.

Greenwood, R. & Scharfstein, D. (2013), 'The Growth of Finance', *Journal of Economic Perspectives* 27(2), 3-28.

Helander, M.G.; Landauer, T. & Prabhu, P. V., eds. (2007), *The Human-computer Interaction Handbook Fundamentals, Evolving Technologies, and Emerging Applications. 2nd ed.*, Lawrence Erlbaum Associates.

Herzig, Kim, S. J. & Zeller., A. (2014), 'It's Not a Bug, It's a Feature: How Misclassification Impacts Bug Prediction', Technical report, Saarland University, Accessed December 12, 2014. https://www.st.cs.uni-saarland.de/softevo//bugclassify/.

Hodson, H. (2013), 'AI Gets Involved with the Law." The New Scientist'. http://www.newscientist.com/article/mg21829175.900-ai-gets-involved-with-the-law.html.

Hopcroft, J., R. M. & Ullman, J. (2001), *Introduction to Automata Theory, Languages, and Computation.*

2nd ed., Pearson/Addison Wesley.

HotDocs (2014), 'HotDocs', Accessed December 13, 2014. http://www.hotdocs.com/.

Hull, J. (2014), *Options, Futures, and Other Derivatives. Ninth ed.*, Pearson Prentice Hall.

Huth, M. & Ryan, M. (2004), *Logic in Computer Science: Modelling and Reasoning about Systems. 2nd ed.*, Cambridge University Press.

ISDA (2013), 'ISDA Master Agreement', ISDA Bookstore, Accessed December 12, 2014. http://www.isda.org/publications/isdamasteragrmnt.aspx.

Kastelle, T. (2012), 'Two Great Innovation Misquotes', The Discipline of Innovation, Accessed December 13, 2014. http://timkastelle.org/blog/2012/01/two-great-innovation-misquotes/.

Katz, D. M. (2014), 'Ethereum Contracts as Legal Contracts', *Computational Legal Studies*, Accessed December 13, 2014. http://computationallegalstudies.com/2014/06/23/ethereum-contracts-legal-contracts/.

Katz, D. M. (2014), 'Using Data to Predict Supreme Court's Decisions', Technical report, Michigan State University, Accessed December 13, 2014. http://msutoday.msu.edu/news/2014/using-data-to-predict-supreme-courts-decisions/.

Katz, D. M. (2012), 'Quantitative Legal Prediction — or — How I Learned to Stop Worrying and Start Preparing for the Data Driven Future of the Legal Services Industry', *Emory Law Journal* 62, 909-966.

Katz, D. M.; Bommarito, M. J. & Blackman, J. (2014), 'Predicting the Behavior of the Supreme Court of the United States: A General Approach', Technical report, Michigan State University. http://arxiv.org/abs/1407.6333.

Kelley, K. (2008), 'The Technium: Turing'd." The Technium: Turing'd', Accessed December 13, 2014. http://kk.org/thetechnium/2008/03/turingd/.

KMStandards (2014), 'Welcome to KMStandards', KMStandards, Accessed December 13, 2014. http://kmstandards.com/.

Kozen, D. C. (1997), *Automata and Computability*, Springer.

Kuhn, H. W. (2007), Introduction, in: J. Neumann and O. Morgenstern. *Theory of Games and Economic Behavior. 60th Anniversary edition*', Princeton University Press.

Law Technology News (2014), 'Law Technology News', LTN, Accessed December 13, 2014. http://www.lawtechnologynews.com/.

Legal XML (2000), 'Transcripts Workgroup Website', Technical report, OASIS, Accessed December 13, 2014. http://www.legalxml.org/workgroups/substantive/transcripts/transcriptwgcharter_2000_03_04.s

html.

Legal XML (2014), 'EContracts Technical Committee', Technical report, OASIS, Accessed December 13, 2014. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=legalxml-econtracts.

LegalTech (2014), 'LegalTech 2014: Exhibitor List', Technical report, LegalTech Conference, Accessed December 13, 2014. http://www.legaltechshow.com/r5/cob_page.asp?category_id=76594&initial_file=cob_page-exhibitors.asp.

LegalTechnology.com (2014), 'LegalIT Insider', Technical report, LegalTechnology. http://www.legaltechnology.com/.

Leith, P. (2010), 'The Rise and Fall of the Legal Expert System', *European Journal of Law and Technology* 1. http://ejlt.org/article/view/14/1

Lessig, L. (2000), *Code: And Other Laws of Cyberspace*, Basic Books.

Lettieri, N. & Faro, S. (2012), 'Computational Social Science and Its Potential Impact upon Law', *European Journal of Law and Technology* **3**(3).

Liu, H. & Lieberman, H. (2005), 'Metafor: Visualizing Stories as Code', Proceedings of IUI'05, January 10–13, San Diego, California, USA. http://web.media.mit.edu/~hugo/publications/papers/IUI2005-metafor.pdf.

Love, N. & Genesereth, M. (2005), 'Computational Law." Stanford Logic Group, Accessed December 13, 2014. http://logic.stanford.edu/people/genesereth/papers/computationallaw.pdf.

Lynn, D. M. (1994), 'Enforceability of Over-the-Counter Financial Derivatives', *The Business Lawyer ABA*, 291-337.

Macey, J. R. (2013), *The Death of Corporate Reputation: How Overregulation Has Destroyed Integrity on Wall Street*, Financial Times Press/Pearson Education.

Lex Machina (2014), 'About Us', *Lex Machina*, Accessed December 13, 2014. https://lexmachina.com/about/.

MacLeod, W. B. (2007), 'Reputations, Relationships, and Contract Enforcement', *Journal of Economic Literature* **45(3)**, 595-628.

Mailath, G. J. & Samuelson, L. (2006), *Repeated Games and Reputations: Long-run Relationships*, Oxford University Press.

Martin, K. (2012), 'Contract Analysis and Contract Standards." Contract Analysis and Contract Standards', Accessed December 13, 2014. http://contractanalysis.blogspot.com/.

McCarty, L. T. (1990), 'Artificial Intelligence and Law: How to Get There from Here', *Ratio Juris* 3(2), 189-200.

McGinnis, J. O. & Pearce, R. G. (2014), 'The Great Disruption: How Machine Intelligence Will Transform the Role of Lawyers in the Delivery of Legal Services', *Fordham Law Review* **82**(6), 3042-3066.

Mealy, G. H. (1955), 'A Method for Synthesizing Sequential Circuits', *Bell System Technical Journal*, 1045-1079.

Mik, E. (2012), 'From Clay Tablets to AJAX: Replicating Writing and Documents in Internet Transactions', *Journal of Internet Law* **15**(8).

Moore, E. F. (1956), 'Gedanken-experiments on Sequential Machines', *Automata Studies*, 34, 129-153. http://books.google.com/books?hl=en&lr=&id=oL57iECEeEwC&oi=fnd&pg=PA129#v=onepage&q&f=false.

Mueller, S. M. & Paul, W. J. (2000), *Computer Architecture Complexity and Correctness*, Springer.

von Neumann, J. & Morgenstern, O. (1944), *Theory of Games and Economic Behavior*, Princeton University Press.

Nevada Secretary of State (2014), 'Silverflume: Digital Operating Agreement', Accessed December 14, 2014. https://nvsilverflume.gov/digitaloa/home.

Nisan, N. (2014), 'Algorithmic Mechanism Design: Through the Lens of Multi-unit Auctions', Accessed December 13, 2014. http://www.cs.huji.ac.il/~noam/amd-mua.pdf.

Nisan, N.; Vazirani, V.; Roughgarden, T. & Tardos, E. (2007), *Algorithmic Game Theory*, Cambridge University Press.

Object Management Group (2014), 'Enterprise Data Management Council (EDMC) Financial Industry Business Ontology (FIBO) Standard, Version 1.0 - Beta 1', Technical report, OMG, Accessed December 13, 2014. http://www.omg.org/spec/EDMC-FIBO/FND/.

Ossowski, S. (2013), *Agreement Technologies*, Springer Science Business Media.

Ostrom, E. (2012), 'Enhancing the Evolution of Institutions for Collective Action', *Social Evolution Forum*.

Ostrom, E. & Gardner, R. (1994), *Rules, Games, and Common-pool Resources*, University of Michigan Press.

Overdahl, J. & Schachter., B. (1995), 'Derivatives Regulation and Financial Management: Lessons from Gibson Greetings', *Financial Management* **24**(1), 68-78.

Patent Vector (2014), *Patent Vector*, Password access required. http://www.patentvector.com/.

Philip, L. (2010), 'The Rise and Fall of the Legal Expert System', *European Journal of Law and Technology* 1(1).

Rosenberg, A. L. (2010), *The Pillars of Computation Theory State, Encoding, Nondeterminism*, Springer.

Rowe, D. M. (2013), 'Risk Management Beyond VaR', *EDPACS: The EDP Audit, Control, and Security Newsletter* 48(1), 1-28.

Sator, G.; Biasiotti, M. & Casellas, N. (2011), 'Legal Ontology Engineering: Methodologies, Modelling Trends, and the Ontology of Professional Judicial Knowledge''Approaches to Legal Ontologies: Theories, Domains, Methodologies'.

Schmedders, K. (2014), *Handbook of Computational Economics*, North Holland.

Shannon, C. E. & McCarthy, J. (1956), 'Automata Studies', **34**, 129-153.

Simonson, S. (2013), 'Theory of Computation', *ADUni.org*. http://www.aduni.org/courses/theory/.

Sipser, M. (2013), *Introduction to the Theory of Computation. 3rd ed.*, Cengage Learning.

Smith, A. (1937), *An Inquiry into the Nature and Causes of the Wealth of Nations*, Modern Library.

Society for Computational Economics (2014), *Computational Economics*, Springer.

Software Engineering Institttute (2009), ''Happy Path' Testing', *Acquistion Archetypes*. http://www.sei.cmu.edu/library/assets/happy.pdf.

Stanford Computational Law (2005), 'Stanford Computational Law', Accessed December 13, 2014. http://complaw.stanford.edu/.

Staudt, R. W. & Medeiros, A. P. (2013), 'Access to Justice and Technology Clinics: A 4% Solution'(2013-41 88), Technical report, Chicago-Kent College of Law. http://ssrn.com/abstract=2335060.

Stout, L. A. (2012), *The Shareholder Value Myth How Putting Shareholders First Harms Investors, Corporations, and the Public*, Berrett-Koehler.

Sun, R. (2008), *The Cambridge Handbook of Computational Psychology*, Cambridge University Press.

Surden, H. (2012), 'Computable Contracts', *U.C. Davis Law Review* **46**, 629-39.

Susskind, R. E. (2010), *The End of Lawyers?: Rethinking the Nature of Legal Services. Rev. ed.*, Oxford University Press.

Taub, J. (2014), *Other People's Houses: How Decades of Bailouts, Captive Regulators, and Toxic Bankers Made Home Mortgages a Thrilling Business*, Yale University Press.

Tesfatsion, L. & Judd, K. (2006), *Handbook of Computational Economics*, Elsevier.

Thie, P. R. & Keough, G. E. (2008), *An Introduction to Linear Programming and Game Theory. 3rd ed.*, Wiley.

Trebilcock, M. J. (1997), *The Limits of Freedom of Contract*, Harvard University Press.

XUnit Patterns (2009), 'Happy Path', *XUnit Patterns* , Technical report, xUnitPatterns.com, Accessed December 12, 2014. http://xunitpatterns.com/happy%20path.html.

Zak, P. J. (2008), *Moral Markets the Critical Role of Values in the Economy*, Princeton University Press.