Computational Law
The Cop in the Backseat

Michael Genesereth
CodeX: The Center for Legal Informatics
Stanford University

Abstract: Computational Law is that branch of legal informatics concerned with the mechanization of legal analysis (whether done by humans or machines). It emphasizes explicit behavioral constraints and eschews implicit rules of conduct. Importantly, there is a commitment to a level of rigor in specifying laws that is sufficient to support entirely mechanical processing. While the idea of mechanized legal analysis is not new, its prospects are better than ever due to recent technological developments - including progress in Computational Logic, the growth of the Internet, and the proliferation of autonomous systems (such as self-driving cars and robots). Legal technology based on Computational Law has the potential to dramatically change the legal profession, improving the quality and efficiency of legal services and possibly disrupting the way law firms do business. More broadly, the technology has the potential to bring legal understanding and legal tools to everyone in society, not just legal professionals, thus enhancing access to justice and improving the legal system as a whole.

1. Introduction

> "It is one of the greatest anomalies of modern times that the law, which exists as a public guide to conduct, has become such a recondite mystery that is incomprehensible to the public and scarcely intelligible to its own votaries." - Lee Loevinger 1949

You are in another state on business. You get up in your hotel, have breakfast, and head off to visit your client. Driving on an unfamiliar road, you are unsure of the speed limit, and you cannot seem to find a sign. Is the car pool lane in effect at this time? Can you use your cell phone while driving in this state? You approach your destination. Can you make a U-turn on this street? Can you make a right turn on red? Can you park at this time? Back in your hotel later in the day, you decide to take care of some personal chores. Can you order your medications from that online Canadian pharmacy? Can you send that wine as a birthday gift to your aunt in Virginia? Checking on your medical bills. Under what conditions does your mother's health insurance cover her in-home nursing expenses? The law has answers to all of these questions. But the answers are not available when you need them, at least not without a lot of work on your part or the expense of hiring an expert.

We live in a complex regulatory environment. As citizens, we are subject to governmental regulations from multiple jurisdictions - federal, state, and local. As members of organizations, we are subject to organizational policies and rules. As social beings, we are bound by contracts we make with others. As individuals, we are bound by personal rules of conduct.

The sheer number and size of regulations can be daunting. We may all agree on a few general principles; but, at the same time, we may disagree on how those principles apply in specific situations. In order to minimize such disagreements, regulators are often forced to create numerous

regulations or very large regulations, to deal with special cases.

A recent article in the National Review made this case forcefully. "The Lord's Prayer is 66 words, the Gettysburg Address is 286 words, there are 1,322 words in the Declaration of Independence, but government regulations on the sale of cabbage total 26,911 words."

Complicating the situation is the complexity of these regulations. Even small regulations can be very complex. While this complexity can sometimes be mitigated by careful drafting, such care is not always possible due to time constraints; moreover, once regulations are created, complexity often increases as the regulations are changed and then changed again.

A simple example of the problem of complexity is the Michigan Lease Termination Clause shown here. This case was first highlighted in a paper by Bob Kowalski to illustrate this very point.

> The University may terminate this lease when the Lessee, having made application and executed this lease in advance of enrollment, is not eligible to enroll or fails to enroll in the University or leaves the University at any time prior to the expiration of this lease, or for violation of any provisions of this lease, or for violation of any University regulation relative to resident Halls, or for health reasons, by providing the student with written notice of this termination 30 days prior to the effective date of termination; unless life, limb, or property would be jeopardized, the Lessee engages in the sales of purchase of controlled substances in violation of federal, state or local law, or the Lessee is no longer enrolled as a student, or the Lessee engages in the use or possession of firearms, explosives, inflammable liquids, fireworks, or other dangerous weapons within the building, or turns in a false alarm, in which cases a maximum of 24 hours notice would be sufficient.

The rule itself is actually fairly simple. However, there are many conditions; there are conditions that modify other conditions; and so forth. Moreover, there are cases where the exceptions occur in other clauses. Typical in insurance contracts. In my homeowner's insurance contract, there is a statement on page 112 stating that the coverage on page 12 does not apply when various conditions exist. The upshot is a regulation that is difficult for most people to understand without specialized legal knowledge and a substantial amount of study.

These problems make it difficult for affected individuals to find and comply with applicable regulations. The result is lack of compliance, widespread inefficiency, and frequent disenchantment with the regulatory system. Fortunately, these problems are not insurmountable. To the extent that they are information problems, they can be mitigated by information technology. What is needed is appropriate legal technology - information technology applied to laws. And the most interesting possibility here is an extreme form of legal technology known as Computational Law.

Computational Law is that branch of legal informatics concerned with the automation of legal analysis. The goal of work in the field is the development of computer systems capable of doing legal calculations of various sorts, such as compliance checking, legal planning, and regulatory analysis.

Intuit's Turbotax is a simple example of a rudimentary Computational Law system. Millions use it each year to prepare their tax forms. Based on values supplied by its user, it automatically computes the user's tax obligations and fills in the appropriate tax forms. If asked, it can supply explanations for its results in the form of references to the relevant portions of the tax code.

On the other hand, document management systems (such as Westlaw, LexisNexis, LegalZoom, and RocketLawyer) are *not* examples of Computational Law. They provide value by helping their users to create and/or find documents. However, they do not themselves understand the content of those documents. As a result, human specialists are needed to extract and utilize that content. By contrast, Computational Law systems dispense with traditional documents in favor of data structures that represent legal content in computable form; and, using this data, they are capable of conducting legal analysis entirely on their own, i.e. without the intervention of human experts.

Computational Law is a technological innovation with disruptive potential. It has obvious consequences for the legal profession by improving the quality and efficiency of legal services and possibly changing the way law firms do business. More broadly, it has the potential to bring legal tools to everyone in society, not just legal professionals, thus mitigating the problems of legal size and complexity mentioned above and increasing access to justice.

This paper is a brief introduction to Computational Law. In the next section, we look at basic idea in greater detail; and, in the section after that, we discuss some its limitations. We then describe some ideas for the deployment of the technology. And, in the concluding section, we discuss the implications of Computational Law for the legal system as a whole.

2. Basic Idea

The most popular approach to building Computational Law systems today is based on Computational Logic. There are two components to this approach - (1) the representation of facts and regulations as sentences in formal logic and (2) the use of mechanical reasoning techniques to derive consequences of the facts and laws so represented.

As a simple example, let's consider a small enterprise and look at how Computational Law might play a role in analyzing compliance with the enterprise's business rules. Applications to governmental regulations are analogous.

The sentences below encode some basic facts about the enterprise. John manages Kat and Ken, and Jill manages Mary and Mike. John and Ken are in room 22 and Jill and Mike are in room 24. John, Ken, and Mike are male; Jill, Kat, and Mary are female. (Note that we are using a simplified form of English here for ease of understanding; in practice, these sentences would normally be written using a more mathematical notation.)

John manages Ken.	*John is in office 22.*	*John is male.*
John manages Kat.	*Jill is in office 24.*	*Jill is female.*
Jill manages Mark.	*Ken is in office 22.*	*Ken is male.*
Jill manages Mike.	*Kat is in office 24.*	*Kat is female.*
		Mary is female.
		Mike is male.

The language of logic extends the language of simple data in two ways. First of all, there are variables, which allow us to refer to arbitrary entities. Secondly, there are logical operators, which allow us to express relationships between facts. In what follows, we use individual capital letters as variables, e.g. X, Y, Z. Our logical operators include not, and, or, and if-then.

One use of these representational extensions is to define new relations in terms of existing relations. The sentence below is an example. Here, we define the officemate relation in terms of the office relation. If a person X is in office Z and and a person Y is in office Z and X and Y are not the same

person, then X is an officemate of Y.

If X is in office Z and Y is in office Z and X and Y are distinct, then X is an officemate of Y.

We can encode rules and regulations in similar fashion by writing rules that define the concept of illegality. As an example, consider how we might express the organizational regulation that no manager may have a direct subordinate as officemate. The Computational Logic sentence shown below expresses this fact using the vocabulary used above. If there is a person X who manages person Y and X is an officemate of Y, then that is illegal.

If X manages Y and X is an officemate of Y, then that is illegal.

Reasoning with facts and logical sentences encoded in this way is quite simple. Given some facts and rules, we can derive logical conclusions by matching facts and conclusions of rules to the conditions of other rules and asserting their conclusions. Such patterns of reasoning are called rules of inference. For example, if we have a sentence P and a sentence Q, we can conclude the sentence "P and Q.". If we have a sentence "If R, then S" and we also have the sentence R, we can conclude S. By applying such reasoning steps repeatedly, we can eventually produce all logical conclusions from our set of premises.

Consider, the data we saw earlier and the logical definitions and rules. from these, we can mechanically derive several conclusions. We can conclude that John is an officemate of Ken. Since John is the manager of Ken, we can also conclude that this is illegal (according to the rules of the corporation).

John is in office 22 and Ken is in office 22.
John is an officemate of Ken.
John manages Ken and John is an officemate of Ken.
That is illegal.

By inverting this sort of reasoning, by working backwards, computers can also be used to arrange matters to avoid such illegalities. For example, in this case, it is possible to satisfy our constraints by assigning offices as shown here.

John is in office 22.
Jill is in office 24.
Ken is in office 22.
Kat is in office 24.

Finally, we can use a well-known extension to this compliance checking technique to analyze regulations for unresolvable inconsistencies. For example, if we require that every project have a manager and subordinates and we required that no manager share an office with a subordinate, we could use mechanical techniques to show the inconsistency of conflicting rules, e.g. a rule requiring all Skunkworks personnel to be housed in a common work room.

All projects have managers and subordinates.
No manager may share an office with a subordinate.
All skunkworks personnel must be housed in a common room.

Legal planning and regulatory analysis are more complex than compliance checking since multiple

hypothetical possibilities must be considered. However, both can be automated using well-known extensions to the compliance checking technique illustrated above.

While the language just described is sufficient to express many types of regulations, there are others sorts of regulations that are more complicated. In a seminal article on the use of logic in representing law [Sergot et al], Bob Kowalski and his colleagues identified a number of different shortcomings of this approach.

The examples in Kowalski's paper are all centered on the British Nationality act. He found that he could represent certain aspects of the act with ease. One of his positive examples is the following clause.

> *A person born in the United Kingdom after commencement shall be a British citizen if at the time of birth his father or mother is (a) a British citizen or (2) settled in the United Kingdom.*

Unfortunately, in examining the Act, Kowalski also came upon a number of conditions that were not readily representable in this form. For example, some laws depend on people's beliefs about the facts (... *the Secretary of State is satisfied that* ...). Some laws depend on "defaults" (... *unless the contrary is shown* ...). Some laws require the representation of metalevel information, i.e. references within the law to other parts of the law (... *as defined in subsection (1)* ...).

To make matters worse, regulations are not always well coordinated, arising, as they do, in different settings for different purposes. Sometimes, there are gaps, leaving important cases uncovered. More often, regulations overlap other regulations and in some instances are inconsistent with each other.

The good news is that these problems have largely been addressed in the years since that article was written and can be handled by well-understood extensions to the language and reasoning techniques mentioned above.

3. Limitations

Philosophically, Computational Law is closely aligned with the Legal Formalism school of Jurisprudence. As such, it treats written laws as definitive, on the assumption that all normative considerations have already been incorporated into those laws by their authors.

This contrasts with the tradition of Legal Realism, which permits arbitrary discretion in legal judgment to balance the interests of affected parties on a case-by-case basis. There is no allowance for this sort of normative innovation in Computational Law.

Given its philosophical stance, Computational Law is most relevant in Civil Law settings, where laws are taken more or less literally. It is less relevant to legal systems based on Common law, where judicial innovation is the norm.

The good news is that, even in Common Law countries, like the United States, there are numerous categorical statutes in which judicial discretion is limited. There are possibilities in dealing with privacy and security matters, in intellectual property rights management, in enterprise management (e.g. constraints on travel, expenditure, reporting), in assessing compliance of plans with building codes (affected by local, county, state, and federal safety requirements), in electronic commerce (e.g. import/export restrictions on technology, drugs, and so forth), in labor law (e.g. occupational safety regulations and health care benefits), and so forth.

Moreover, as the technology becomes established, it is conceivable that regulators may find advantage in creating more and more categorical regulations, thus enlarging the applicability of the technology.

One technical problem with Computational Law, familiar to many individual with legal training, is due to the open texture of laws. Consider a municipal regulation stating "No vehicles in the park". On first blush this is fine, but it is really quite problematic. Just what constitutes a vehicle? Is a bicycle a vehicle? What about a skateboard? How about roller skates? What about a baby stroller? A horse? A repair vehicle? For that matter, what is the park? At what altitude does it end? If a helicopter hovers at 10 feet, is that a violation? What if it flies over at 100 feet?

The resolution of this problem is to limit the application of Computational Law to those cases where such issues can be externalized or marginalized. We allow human users to make judgments about such open texture concepts in entering data or we avoid regulatory applications where such concepts abound.

A different sort of challenge to Computational Law stems from the fact that not all legal reasoning is deductive. Edwina Rissland [Rissland et al.] notes that, "Law is not a matter of simply applying rules to facts via modus ponens"; and, when regarding the broad application of AI techniques to law, this is certainly true. The rules that apply to a real-world situation, as well as even the facts themselves, may be open to interpretation, and many legal decisions are made through case-based reasoning, bypassing explicit reasoning about laws and statutes. The general problem of open texture when interpreting rules, along with the parallel problem of running out of rules to apply when resolving terms, presents significant obstacles to implementable automated rule-based reasoning.

This is a serious challenge. Computational Law derives its power from its emphasis on deductive reasoning. As such, it simply cannot be applied in cases requiring analogical or inductive reasoning. Fortunately, it is sometimes the case that there are enough judicial rulings that the net result is, in effect, a set of categorical constraints even where the original wording of the regulations is not definitive. And, in such cases, Computational Law can be applied to the combination of regulatory and judicial law.

4. Deployment

One approach to deploying Computational Law is to create legal applications to which users turn when they are in need of legal analysis. An approach to deployment with even greater potential is to embed legal technology in computer applications that people use for other purposes.

What makes Computational Law especially interesting today is the dramatic increase in digitally mediated activity. Increasingly, our activities take place online. We routinely use the Internet to buy products, ship them, book travel, interact with government agencies, and so forth. This development is important because it means that the data necessary to do legal analysis are already in digital form. In many cases, these activities are managed by web services with the data and computational resources necessary to apply appropriate regulations. The opportunity here is that we can embed computational law in our computer systems - so that we can be aware of the legal status of our actions as we are performing them.

Let's look at an example. Existing building construction projects are covered by numerous laws and regulations, including local building codes, federal environmental rules, and accessibility laws such as the Americans With Disabilities Act. Standard practice today id for architects to prepare their

plans and then submit to municipalities for approval. This process can take weeks. Moreover, since there are so many different regulations in different municipalities, architects cannot know them all, and there are usually problems. Once the architect is informed of problems, the cycle repeats. It is a frustrating and inefficient process.

Project Calc was one of the first projects done in CodeX, under the direction of Harry Surden. The idea of the project was to embed compliance checking within the CAD systems used by architects and thus avoid such problems. CALC examined the degree to which existing laws governing the domain of building design can be modeled within computer systems and made to interact with systems currently used in the field. It examined whether computer systems could assist design professionals in knowing and complying with these rules. CALC also explored legal theoretical problems related to the representation of laws in computer systems and proposed principles for selecting and creating such laws. The project was never completed due to funding limitations, but it is a great example of embedded law.

Of course, the idea is not restricted to deployment on traditional computers; we can also embed the technology in the our everyday environment via our cell phones and other devices - and in so doing we bring law to the *point of decision* so that we are informed of our legal responsibilities and our legal rights before we act and get ourselves into trouble and so that we know our legal rights.

This makes lots of things possible. You are walking through the woods of Maine and see an attractive flower. You take a photo with your iPhone. Your plant app identifies it as a type of orchid and lets you know. At the same time, your legal app tells that, no, you may not pick it.

In thinking about this sort of Embedded Law, consider the metaphor of the Cop in the Backseat. Suppose that we had the benefit of a friendly policeman in the backseat of our car whenever we drove around (or perhaps an equivalent computer built into the dash panel of our car). The cop, real or computerized, could offer regulatory advice as we drive around - telling us speed limits, which roads are one-way, where U-turns are legal and illegal, where and when we can park, and so forth.

This already exists to limited extent in aviation, where where displays like this one provide feedback on restricted areas and areas with special requirements.

5. Conclusion

Technological progress has led to the development of web-based computer systems for managing the affairs of enterprises. Most large corporations today utilize enterprise management software applications to run the operations of their businesses, such as accounting/finance, human capital management, supply chain & manufacturing, etc. Most large companies run dedicated enterprise management systems internally, while many small and medium-sized businesses use services that reside in the cloud. The development of such software and services has led to sizable businesses for companies like SAP, Oracle and IBM. It seems clear that these same technologies can be applied to the public sector, except with governmental rules and regulations in place of business rules.

Moreover, given the problems described in the Introduction, we may end up needing this technology for the proper functioning of the law as a mechanism for achieving social good. One of the functions of the law is to help individuals predict the consequences of their actions. If we do not know what the law is, the law does not serve this function. The Constitution of the United States, in both the fifth and the fourteenth amendments, mandates "due process" for its citizens. Part of due process is the concept of notice. Citizens must receive notice of applicable regulations before they can be charged with violations. Some legal scholars have argued that, when the law becomes so

recondite that citizens are unable to understand it, then they have not received adequate notice and cannot be charged.

In a sense, Computational Law is the natural next step in a progression that began millenia ago. Around 1750 BC, Hammurabi had the laws of the land encoded in written form (literally cast in stone) so that citizens could know what was expected of them and what would happen if they violated those expectations. Since then, it has been the norm to encode rules in written form and disseminate first via books and more recently via the Internet. However, with the proliferation of rules and regulations, just writing things down is not enough when the laws are voluminous and difficult to understand. In a way, Computational Law is the next step in the evolution of the legal system.

References

Rissland, E. L., Ashley, K. D., and Loui, R. P.: "AI and law: a fruitful synergy", *Artificial Intelligence*, 150(1-2):1-15, 2003.

Sergot, M. J., F. Sadri, R.A. Kowalski, F. Kriwaczek, P. Hammond and H.T. Cory: "The British Nationality Act as a logic program, *Communications of the ACM*, vol. 29, no. 5, pp. 370-386, May 1986.