What are the pragmatic challenges to the semantics of contract automation?

Theoretical Frameworks 2: coursework

Candidate Number: Suzanne Huldt

Word count: 4000 (including subtitles, example sentences and in-text citations; excluding bibliography and title page)

Introduction

Beginning in the late 1980s the possibility that the execution of legal contracts may be automated has been explored in both academic and professional areas. In this nascent field of research such projects are referred to variously as “Ricardian contracts” (Grigg, 2001), “Smart Contracts” (Clack, 2016 a,b), “contract formalisation” (Hvitved) etc. In spite of varying terminology, each of these contract automation ventures shares certain key mechanical features. In basic terms, contract automation takes the meaning expressed by the prose of a legal contract, and converts or ‘translates’ it into a formalised language drawn from a combination of logic, computer programming, and domain specific languages. If this code is machine-readable, the obligations, agreements and other relations expressed in the contract prose can then be executed by computer. Note the double meaning of the term ‘execute’ here: the contract code is executed in the computational sense, a process that itself constitutes (at least in part) the execution of the contract in the legal sense. Contract automation is motivated primarily by a desire for ambiguity reduction, alongside the labour saving potential of automation. Additionally, an automated contract may be “tamper proof”, in the sense that its execution cannot be halted once it has been initiated (Clack et al, 2016, p4), and less vulnerable to a breakdown in execution, for example in cases of blame assignment (Hvitved, 2011).

In addition to the current lack of consensus as to terminology, there are a number of proposed formalisms for encoding legal prose (see Hvtived, 2011, for a review), and there is no general agreement as to how contract automation should be implemented. Suggestions from current research include execution on shared ledgers (Clack et al, 2016) in the manner of established crypto-finance and crypto-law platforms such as Ethereum, a decentralised platform for smart contract execution[1]. Others, including Grigg (2001) suggest purpose-built systems. The application potential for contract automation is also broad, including tenant-landlord agreements (Hvitved, 2011, p6) and payments for services between private individuals (Lee, 1989). However, in light of the potential advantages of ambiguity reduction and labour saving, primary interest comes from industries requiring highly complex contracting arrangements, in particular banking and financial services. Despite different proposed execution platforms, instantiating contract automation in these industries presents wide ranging challenges, particularly in the interaction between previously quite separate areas of academic enquiry and professional practice. For example, Grigg’s (2001) ‘seven-layer’ financial cryptography system, and the proposed ‘Smart Contracts’ of Clack et al (2016) are both designed to “interface with a wide variety of platforms” (Clack et al, 2016, p2) for the purpose of automating elements of corporate financial services agreements. Domains governed by quite different rules, conventions and bodies of knowledge may struggle to interact. For example, automating contracts may be computationally possible, but will still present problems of legal enforceability (Clack, 2016a, p3).

A full consideration of these issues is beyond the scope and purpose of this paper. Rather, the focus here is on a fundamental problem that arises when the meaning of natural language is formalised: the pragmatic gap between the linguistic stimulus and the meaning that is communicated[2]. The meaning of human language does not arise only from the type of entailments that can be captured in logical formalisms. Rather, the meaning that is communicated by speakers arises both from the lexical content and grammatical rules of the words in an utterance, and from the interpretation of that utterance by interlocutors in both its linguistic and non-linguistic context (Grice, 1975). Contract automation relies on the capturing of the meaning of natural language into a formalised code – one that then supersedes the original prose in the execution of the contract, both in the computational and the legal sense. This presents a significant problem: if linguistic meaning is interpreted in context, but only a portion of that meaning can be expressed in contract code, how can it be certain that the execution of the contract is faithful to the meaning expressed by the human-authored legal text? There are three potential ways for unresolved ambiguity to arise, or for meaning to be lost, dependent on how the contract prose is encoded: Lee (1988) suggests a natural language parser, by which the contract is encoded by machine. This is potentially problematic if the parser does not compute the expected meaning from the legal prose. The contract prose may also be formalised by speakers of the natural language into a contract formalism, in which case the full meaning of the text may not be adequately captured. Thirdly, for example Clack et al (2016a,b) propose the development of domain specific languages in which the legal prose itself may be written by lawyers.

Clack et al (2016) suggest the question "can we be absolutely certain of the meaning of a contract" (p11) as one of several long term research challenges. This paper will address this question in terms of the pragmatic gap between machine readable codes and human language interpretation. A review of the existing body of contract automation literature suggests that this problem is both under-defined and under-addressed. Indeed, Grigg, in discussing the challenges posed by the interaction of previously unfamiliar disciplines and industries required for contract automation, lists more than ten areas whose input is required for such a project. While including "programming", "cryptography", "finance and banking", and "internet" (Grigg, 2001, pp332-333), areas such as 'linguistics', 'pragmatics', or even 'logic' are conspicuously absent.

The first part of this paper will summarise existing semantic options for encoding contract prose. The second will consider these formalisms from pragmatics perspective; in reference to both linguistic pragmatics in general, and the problems it poses for computational approaches to natural language, and legal-specific pragmatics theory. We will highlight the problems that arise in the pragmatic gap between language as code, and language as it is interpreted. Finally, drawing on current pragmatic theory and research, we will sketch some proposals for closing this pragmatic gap to the benefit of contract automation, bearing in mind that it is unlikely that such pragmatic gaps can ever be absolutely closed

Semantics

A variety of possible formalisms of encoding contact code currently exists – although not all are fully developed for implementation. A full review of all these encoding languages is beyond the scope of this paper (see Hvitved, 2011, for a full review). Rather, we will focus on two classes of contract code that are well suited for analysis in a linguistic-pragmatic perspective: logic based programmes and functional programming (note that Hvitved identifies three classes of contract codes: logic based, event-condition-action, and trace based (2011, p8).

The types of logical formalisation required in both types of contract code will illustrated by worked examples from the ISDA agreement[3]. The ISDA agreement and its supplementary documentation is a standardised contract for derivatives trading, elements of which can be modified to the extent necessary for the parties making the agreement. In addition to the interest in contract automation expressed by the financial services sector, this type of document is well suited to contract automation: the obligations, responsibilities, and rights that are expressed between two parties are standardised and therefore need only to be appropriately encoded once, after which the formalisms may be reused. Furthermore, the compositional structure of financial contracts is highly suited to formalisation by compositional codes, including Functional Programming (Hvitved, 2011, p13).

Lee (1988) presents a logic-programming based system for encoding and executing contracts. In this system, Petri nets are used as a model of temporal relations, as parties to a contract move from state to state according to certain conditions. This logical basis is then converted into logic programming notation so that it may be implemented on an execution platform (Lee, 1988, p34). Functional Programming is a type of non-imperative computer programming in which code is stated as a series of expressions (functions), rather than instructions (see Clack, Myers, and Poon, 1995). At its lower levels, Functional programming is based on the logic of the lambda calculus. Within formal semantics, elements of the pure lambda calculus (including lambda abstraction and conversion) (Partee et al, 1990) are adapted and used to express the meaning of words as functions between types within semantic models. It can be argued, therefore, that formalisms based on the lambda calculus are highly suited to the encoding of the semantics of natural language, as well as offering clear examples of the possible mismatches between the meaning that can or cannot be captured by formalisms. Functional programming has been applied to contract automation by Peyton Jones and Eber (2000). In this model a contract is broken down into a series of combinators – such as ‘and’, ‘or’, ‘give’, ‘unit of currency’ that can then be combined to express the operational semantics of the contract prose (Hvitved, 2011, p13).

Deontic

As is common in contract texts, large portions of the ISDA agreement state relationships of obligation. Parties to the contract agree to uphold obligations to one another, or to certain actions on the condition that obligations are fulfilled. For example:

(1) “the Defaulting Party will pay to the Non-Defaulting Party, if a positive number, the Non-Defaulting Party’s loss in respect to this agreement”

ISDA Master Agreement, p9

Such relationships of obligation can be captured with deontic logics. Consider fragment (1) modelled with standard deontic logic, a system that builds on propositional logic (McNamara, 2014) with additional deontic operators, for example the OB obligation operator below.

OB P(d,n)

Where: d is the defaulting party, n is the non-defaulting party, and P is a proposition ‘pay’

Note that this formalisation extracts only the relationships of obligation expressed in (1). The definition of “Defaulting Party”, “Non-Defaulting Party”, and “agreement” etc must be pre-defined in the model. In Lee’s logic programming this is achieved by the use of a DEF operator, that is used to assign the contract in question a number of pre-determined variables (Lee, 1988, p38).

Temporal

A significant proportion of the ISDA agreement expresses relationships governed by time. For example, consider the sentence fragment:

(1) “on or after the date on which the transaction is entered into”

The phrase above contains two temporal expressions – “on the date” and “after the date” in a disjunctive relationship: ‘on the date’ denotes one fixed point, and ‘after the date’ denotes the period (ie. every point in time) after the specified date. The truth conditions for the disjunctive meaning of ‘or’ require that one and or both (here, an impossibility) of ‘on’ and ‘after’ the date is True, ie. that the action is fulfilled at one of those times, but not neither.

Several temporal logics (logical formalisms specialised for the precise expression of events as they occur in time) are available to model a fragment such as this. *Basic temporal logic* is based on an underlying model of time as discrete and linear, a ‘timeline’ analogous to the natural numbers (N) (Fisher, 2011, p11). Within this framework, time is treated as a “sequence of moments”, corresponding to a static propositional state in conventional, non-temporal logic (Fisher, 2011, p11). Movement between these states is described by the use of specific temporal operators. Using basic temporal logic, the phrase in (1) may be expressed as follows:

Where: the square operator indicates that, for the model M, it is the case from date i (an agreement) holds for all future moments.

Although intuitive, basic temporal logic also has limitations in its capacity to formalise the types of time expressions that occur in contracts. Specifically, much contract prose describes situations in which the fulfilment of, or failure to fulfil, some condition by a certain point in time will lead to two distinct outcomes. For example:

(1) "on the date that electronic message is received, unless the date of that delivery (or attempted delivery) or that receipt, as applicable, is not a Local Business Day"

ISDA Master Agreement, p13

For our purposes, the key element in the fragment above is "unless". The text specifies that some action shall be taken on the same day as event occurs (the message is received), if that date does not fall into the category of days that are not "Local Business Days". In that case, the action is not required. This sentence sets up temporal branch in the execution of the contract. In order to accurately capture this crucial feature of contract prose, branching temporal logics are required. Buchi Automata can be used to model the movement from one state to another in this way, as below. However, they may be overly deterministic (Fisher, 2011, p31) to correctly express the choices available to parties to a contract.

Furthermore, the examples above have been dealt with in a relatively simple manner: only the relevant semantic relationship – deontic or temporal has been extracted. However, the reality of encoding contract prose is significantly more complex. In particular, it is not always easy to disentangle, for example, a 'temporal' expression from a 'deontic' one. Consider this passage from the ISDA agreement, with the deontic portions marked in red, and temporal expressions in green:

(1) "a single partial exercise of any right, power or privilege will not be presumed to preclude any subsequent or further exercise of that right"

ISDA Master Agreement, p12

This combination of types of meaning in a single phrase is commonplace in natural language, however significantly increases the complexity of the expression and therefore the opportunity for error to arise in its encoding. In Lee's model both considerations of branching time, and the dependence of those branches on deontic requirements are captured in the Petri net model (Lee, 1988, pp36-38), which is able to model the movement between states depending on conditions being fulfilled. Functional Programming's combinatorial model is well suited to the inclusion of, for example, deontic and temporal expressions within the same expression. However, note that in its current iteration Peyton Jones and Eber's model is significantly restricted in its ability to capture granularities of time and other complexities; for example defining its time operator 'when' as "immediately upon the completion" of some previous task (Hvitved, 2011, p32).

Pragmatics

From a computer science perspective, the 'semantics' of contracts can be divided into the 'operational' – the stating of precise actions to be taken according to the contract execution, and 'denotational' – the non-operational, legal interpretation (Clack et al, 2016, p5). Linguistics, however, divides the analysis of meaning into two differently defined types: the semantic and the pragmatic. Semantic meaning is that which emerges from language only: that is, the meaning of the words that make up an utterance, and the compositional meaning of their combination in the sentence as a whole. A great deal of this meaning may be captured by the type of logical formalisms described above. However, speakers also communicate meaning beyond that which is encoded in the linguistic stimulus (Breheny, p2): in particular, meaning arises from the interaction of an utterance with its linguistic and non-linguistic context. These elements of meaning are the domain of pragmatics.

There is a large theoretical and experimental literature of pragmatics. For the purposes of this paper, we will briefly summarise the core assertions of pragmatic theories of implication. The Gricean model posits a set of maxims that govern speaker communication. These include the Maxim of Quality – that speakers should not say what they believe to be false; the Maxim of Quantity – that conversational contributions should not be over or under-informative; and the maxims of Manner and Relevance – that linguistic expression should not be obscure or irrelevant to the conversation (Dale and Reiter, 1996, pp3-4). Crucially, the Gricean Maxims serve as rules not only for production, but for interpretation. In Grice's view, an exchange of utterances between speakers proceeds according to the "cooperative principle", a recognition by each party to the conversation that speakers share a common purpose in communicating (Grice, 1975, p45)

This commonality is vital. Alongside a shared assumption that conversation creates a common ground of information, humans participate in speaking situations with the expectation that they and their interlocutors share rational expectations as to what good communicators will provide (Breheny, p1). Neo-Gricean theories propose a simplified set of principles based upon the original Gricean maxims. In particular, the Q(uantity)-principle, to 'say as much as you truthfully or relevantly can'; and the I-principle, often characterised as a combination of the maxim to be relevant and informative – 'say no more than you must' (Carston, 2012, pp13-14). Relevance Theory simplifies this framework even further, arguing that new information is judged as relevant to an interaction depending on whether it yields new cognitive effects in interaction with existing knowledge and context (Sperber et al, 1995, p48)

It is this shared set of communicative principles that allows for pragmatic inferences to arise reliably from utterances in context in the case of words like ‘and’, ‘not’, ‘all’, ‘if’ and ‘some’ (Grice, 1975, p41). Consider ‘some’, a widely studied proposition that gives rise the scalar implicatures (Bott and Noveck, 2004, p1). Its meaning can be formally expressed as: , that there is some thing that exists. However, speakers regularly understand ‘some’ not to mean ‘some amount more than one’, but rather “strengthen” (Potts et al, 2015) its meaning to interpret it as ‘some but not all’. This sense, the ‘scalar implicature’, is formally expressed as .

Despite difference approaches, all pragmatic accounts of communication are united in their assertion that human cognition approaches language with certain indelible assumptions that are themselves part of the framework for linguistic comprehension. Machines process language with no such assumptions. Indeed, Grice, in formulating his maxims for conversation explicitly refers to the “divergences in meaning” between natural language and the type of formalisms required for contract automation discussed in the previous section (Grice, 1975, p41). It is this divergence that presents the key implementation challenge for contract automation, that in its current form must rely on highly formal codes.

There are important differences between spoken language, text, and legal prose. Text is interpreted often in a context separated in time and space from its authorship; legal prose is often highly specified as to meaning. Consider for example the pseudo-logical style of the variable definitions in this section of the ISDA agreement: “the Settlement Amount of the party with the higher settlement amount (“X”) and the Settlement Amount of the party with the lower Settlement Amount (“Y”)” (ISDA Master agreement, p10). However, legal prose does present its own pragmatic challenges, and pragmatic analyses drawn from linguistics and philosophy have been applied to considerations of non-automated, traditional legal interpretation.

A central debate in legal scholarship concerns the differences between ‘interperative’ and ‘textualist’ readings of law. In the interpretive view, legal texts are read in reference to the context in which and assumed considerations with which they were originally drafted (Flanagan, 2010). The textualist position, conversely, is defined by the commitment to interpreting legal enactments only according the meaning contained in the text (Carston, 2010, p21).

Vagueness is key consideration in such debates (Soames, 2011). Consider this potentially ambiguous fragment form the ISDA agreement:

(1) “in each case by the date specified in the Schedule or such Confirmation or,

if none is specified, as soon as is reasonable practicable"

In particular, how should the phrase "as soon as in reasonable" be interpreted? Both "soon" and "reasonable" are relative terms. "Soon" may denote a longer period of time in on a scale of years as opposed to days. In the textualist approach, no recourse may be made to the intentions of the author of the contract in this particular choice of vocabulary. However, some scholars contend that it is impossible to "freeze" the meaning of an enactment in its semantics (Flanagan, 2010, p270): for such items as "soon", there must be reference to at least *some* relevant context (Flanagan, 2010, p269). This position is particularly clear in the linguistic view: whereas for textualist legal scholars 'context' may refer to, for example, the historical purpose of an enactment (Flanagan, 2010; Carston, 2012), from the perspective of pragmatics, the assumption that all those who read the legal text will infer similar meanings from such terms is itself an act of pragmatic reasoning. Indeed, the 'Plain Language' in which legal scholars and practitioners such as Scalia wish to enforce the law (Carston, 2012, p21) is itself, as pragmatics research has shown, subject to interpretation and adjustment in context.

This type of adjustment of the meaning of words in context cannot be fully captured by the lexical ambiguity of homonyms (Carston, 2012, p11): there are not different definitions of the word "soon" present in the mental lexicon, rather, its meaning is interpreted in linguistic and non-linguistic context. From a Gricean perspective we might argue that the use of "soon" indicates that there is no fixed deadline set – that the speaker is observing the maxim of Quality by not specifying an arbitrary time point unnecessarily. However, a contract that is executed by machine is to be precisely enforced, without reference to either the general knowledge context (that concerns legal scholars) or Gricean reasoning (that concerns linguists).

Suggested Directions and Conclusions

To conclude, we draw on a debate in pragmatic theory to suggest how the pragmatic gap between code and language may be narrowed. A fully realised proposal is far beyond our scope, rather the following discussion is intended to serve as an opening for future research. Despite a general consensus that some level of inference occurs in all implicature (Potts et al, 2016), some theorists contend that not all scalar implicatures arise only by inference. 'Grammatical' accounts argue that scalar implicature phenomena occur in contexts that cannot be accounted for by Gricean principles, and that their meaning arises from the grammar of the language. This has been modelled with "silent operators" (Chierchia et al, 2012, p40) grammatical-semantic entities that function to derive the appropriate meaning.

Experimental evidence conflicts with this view (see Bott and Noveck, 2004). However, our purpose here is not to argue for the psychological or linguistic reality of these theories, rather, to explore whether the model of meaning that they propose can be emulated in the encoding of contract prose. If the meaning of ‘some’ in certain constructions defaults to an interpretation as ‘some but not all’, it may follow that this meaning is somehow part of the grammar of the language. There is nothing preventing this meaning from being encoded in the syntax and semantics of a domain specific contract code. For example, an extension of the Functional Programming model can define combinators more specifically: ‘soon’ may be restricted in its remit to specific ‘any time that is shorter than x number of days’. Further, the type of ‘silent’ operators suggested in Chirchiea et al’s (2012) Grammatical account may be added to the syntactic rules of a domain specific language.

The implementation of such solutions itself presents huge challenges: it may be trivial to build in to code a definition of ‘some’ as , but the precise definition of for example ‘reasonable’ may be much more difficult to capture. Indeed, Flanagan argues (in a traditional legal context), against the practical possibility that every conceivable alternative may ever be formulated within law (2010, p267). It is also vital to consider that neither the ‘lexicalisation’ of inferred meaning in semantic code, nor the addition of operators in its syntax may be enough to account for how an item will interact in its particular linguistic context. Indeed, in her analysis of legal interpretation in a Relevance Theoretical perspective, Carston suggests that adjustment of meaning in context may be an inherent property of all language systems (2012, p13).

It is prudent therefore to conclude on a note of caution. This paper has shown that otherwise successful contract codes still contain pragmatic gaps that may prevent full interface with natural language. For the contract automation project to go forward it is vital that these are addressed, and the concluding remarks here are presented as prompts for future research.

References and Bibliography

Bott, L., and Noveck, I., 2004, Some utterances are underinformative: the onset and timescale of scalar implicatures, Journal of Memory and Language 51 (2004) 437–457

Breheny, R., Scalar Implicatures in a Gricean Cognitive System, (unpublished draft, to appear Katsos, N., and Cummins, C. (eds), Handbook of Experimental Pragmatics, OUP)

Carston, R., 2012, Legal Texts and Canons; a view from current pragmatic theory, unpublished chapter proof, Research Gate

Chierchia, G., Fox, D., & Spector, B.. 2012. The grammatical view of scalar implicatures and the relationship between semantics and pragmatics

Clack, C., Bakshi, V., Braine, L., 2016, Smart Contract templates: essential requirements and design options, Barclay's Bank, PLC

Clack, C., Bakshi, V., Braine, L., 2016, Smart Contract templates: foundations, design landscape and research directions, Barclay's Bank, PLC

Clack, C., Myers, E., Poon, E., 1995, Programming with Miranda, Prentice Hall

Flanagan, B., 2010, Revisiting the Contribution of Literal Meaning to Legal Meaning, Oxford Journal of Legal Studies, Vol. 30, No. 2, pp. 255–271

Fisher, M., 2011, An Introduction to Practical Formal Methods in Using Temporal Logic, John Wiley and Sons

Dale, R., and Reiter, D., 1996, The Role of Gricean Maxims in the Generation of Referring Expressions, American Association of Artificial Intelligence

Greene, A., 2006, The Missing Step of Textualism, Fordham Law Review

Grice, H. P., 1975, Logic and Conversation

Grigg, I., 2000, Financial Cryptography in 7 Layers, *Proceedings of Financial Cryptography Conference,* Anguiila, British West Indies February 2000 Springer

Hvitved, T., 2012, Contract Formalisation and Modular Implementation of Domain Specific Languages, PhD dissertation

ISDA Master Agreement, 1992, International Swap Dealers Association

ISDA Credit Support Annex, 1992, International Swap Dealers Association

ISDA Schedule to the Master Agreement, 1992, International Swap Dealers Association

Lee, R., 1988, A Model for Electronic contracting, North Holland Decision Support Systems, Vol 4, pp27-

McNamara, Paul, "Deontic Logic", *The Stanford Encyclopedia of Philosophy* (Winter 2014 Edition), Edward N. Zalta (ed.), URL = https://plato.stanford.edu/archives/win2014/entries/logic-deontic/

Partee, B., ter Meulen, A., and Wall, R., 1990, Mathematical Methods in Linguistics, Kluwer Academic Publishers

Peyton Jones, S., Eber, J.M., and Seward, J., 2000, Composing Contracts, and Adventure in Financial Engineering, International Conference on Functional Programming, Montreal

Potts, C., Lasseter, D., Levy, R., and Frank, M., 2016, Journal of Semantics, 33, pp 755–802

Soames, S., 2011, Vagueness and the Law

Sperber, D., Cara, F., anf Gittorio, V., 1995, Relevance theory explains the selection task, Cognition, Vol 57, pp31-95

Equations generated with CodeCogs online Latex editor:
https://www.codecogs.com/latex/eqneditor.php