Implementations of Natural Logics

Lawrence S. Moss

Indiana University, Bloomington, IN 47405, USA lmoss@indiana.edu

Abstract

We discuss what is known about implementations of logical systems whose syntax is either a small fragment of natural language, or alternatively is a formal language which looks more like natural language than standard logical systems. Much of this work in this area is now carried out under the name of *natural logic*. Just as in modal logic or description logic, there are many systems of natural logic; indeed, some of those systems have features reminiscent of modal logic and description logic. For the most part, quantification in natural logics looks more like description logic or term logic (i.e., syllogistic logic) than first-order logic. The main design criteria for natural logics are that (1) one can be able to use them to carry out significant parts of reasoning; and (2) they should be decidable, and indeed algorithmically manageable. All of the questions that we ask about the implementation of any logics can be asked about natural logics. This paper surveys what is known in the area and mentions open questions and areas.

1 Introduction

This workshop is concerned with Automated Reasoning in Quantified Non-Classical Logics. The specific contribution in this paper might be called *automated reasoning pertaining to human reasoning, but done in a setting that deviates from the normative frameworks in logic*. The specific area of application is natural language semantics, especially done with an eye towards automated reasoning in formal languages which approximate natural language. Admittedly, these days formal logic in computer science is primarily a tool in areas like verification; connections of logic to AI are less prominent. Nevertheless, the original and primary motivation for logic is (arguably) the study of *inference in language*, from Aristotelian syllogisms onward. This is the target area of this paper.

The natural logic program This is a summary of the program of natural logic, taken from papers and talks:

1. Show the aspects of natural language inference that can be modeled at all can be modeled using logical systems which are decidable.

2. To make connections to proof-theoretic semantics, and to psycholinguistic studies about human reasoning.

3. Whenever possible, to obtain *complete axiomatizations*, because the resulting logical systems are likely to be interesting, and also *complexity results*.

4. To implement the logics, and thus to have running systems that directly work in natural language, or at least in formal languages that are closer to natural language than to traditional logical calculi.

5. To re-think aspects of natural language semantics, putting inference at the center of the study rather than at the periphery.

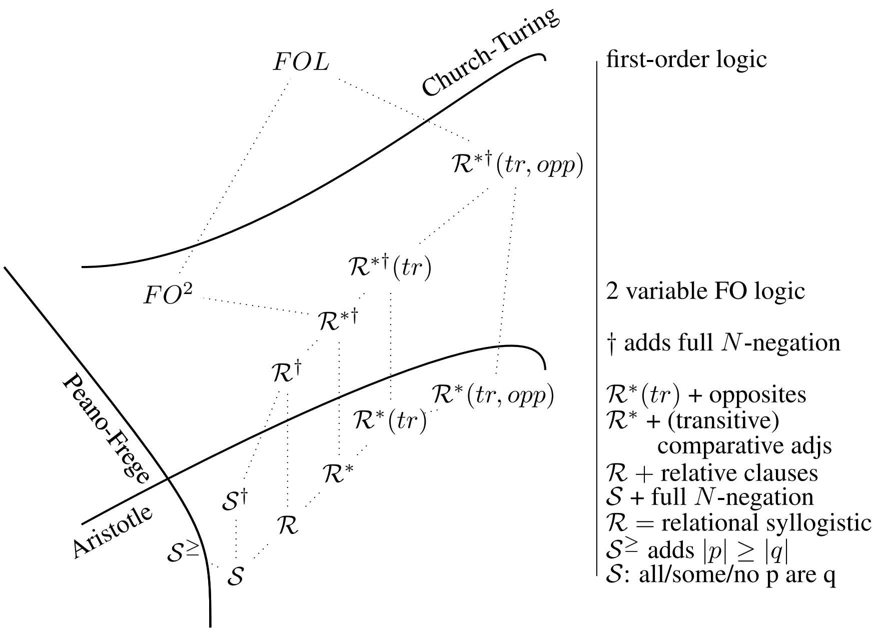
FOL
Church-Turing
first-order logic
$\mathcal{R}^{*\dagger}(tr, opp)$
$\mathcal{R}^{*\dagger}(tr)$
FO^2
2 variable FO logic
$\mathcal{R}^{*\dagger}$
† adds full N-negation
$\mathcal{R}^{\dagger}$
Peano-Frege
$\mathcal{R}^*(tr, opp)$
$\mathcal{R}^*(tr)$ + opposites
$\mathcal{R}^*(tr)$
$\mathcal{R}^*$ + (transitive) comparative adjs
$\mathcal{R}^*$
$\mathcal{R}$ + relative clauses
$\mathcal{S}^{\dagger}$
$\mathcal{S}$ + full N-negation
$\mathcal{R}$
$\mathcal{R}$ = relational syllogistic
Aristotle
$\mathcal{S}^{\geq}$
$\mathcal{S}^{\geq}$ adds $|p| \geq |q|$
$\mathcal{S}$
$\mathcal{S}$: all/some/no p are q

Figure 1: Some logical systems in the natural logic area, along with FOL and others.

Goal (4) is the topic of this paper, but goal (3) is also relevant. Most of the work on natural logic has pursued goals (1) and (3). Concerning (3), there are now a myriad of logical systems which are complete and decidable and which have something to do with natural language. Some of them are listed in a chart in Figure 1. Here is a guide to those systems, starting with the three boundary lines.

The line called "Church-Turing" at the top separates the logics which are undecidable (above the line) from those which are not. FOL is first-order logic. FO^2 is two-variable first-order logic; this is well-known as a decidable logic. This logic is not directly relevant to our study, but since many logics can be expressed in FO^2, they inherit the decidability.

The line called Peano-Frege separates the sub-logics of FOL (to the right) from the one logic on this chart which is not sub-logic of FOL. These logics are "non-classical", being unrelated to FOL. And they deal with "quantity" (but not "quantification" in the usual sense). I would hope that people in the ARQNL community will find them interesting. I discuss them in Section 3. It should be mentioned that there are other logics to the left of the Peano-Frege boundary. Space didn't permit a larger display.

The line called Aristotle separates the logics with a syllogistic formulation from those which lack such a formulation. This line is much less "firm" than the other lines, for two reasons. First, there is no accepted definition of what a syllogistic proof system even is. For us, it means a logic with a finite set of rules, each of which is essentially a Horn clause, and also allowing *reductio ad absurdum*. The second noteworthy feature of the Aristotle line is that sometimes a logic L which is provably above the line has a super-logic L' which is *below* the line. That is, sometimes, adding vocabulary to a logical system enables one to find a syllogistic presentation. This feature has made some doubt whether there is much of a point to the Aristotle boundary. Be that as it may, there certainly is work to do in clarifying the line which we name for Aristotle.

Specific logics We go into details on the syntax and semantics of some of the logics in the chart, starting with $\mathcal{S}$ at the bottom. This is the classical syllogistic. The syntax starts with a collection of *nouns*; we use lower-case letters like p, q, x, and y for nouns. The syntax just has sentences of the form *all x are y*, *some x are y*, and *no x are y*. These sentences are not analyzed as usual in first-order logic. The sentences of $\mathcal{S}$ do not involve recursion. Also, there are no propositional connectives in this tiny logic.

For the semantics, we consider *models* $\mathcal{M}$ consisting of a set M and subsets $[\![x]\!] \subseteq M$ for all nouns x. We then declare

$$\begin{array}{ll} \mathcal{M} \models \textit{all } x \textit{ are } y & \text{iff} \quad [\![x]\!] \subseteq [\![y]\!] \\ \mathcal{M} \models \textit{some } x \textit{ are } y & \text{iff} \quad [\![x]\!] \cap [\![y]\!] \neq \emptyset \\ \mathcal{M} \models \textit{some } x \textit{ are } y & \text{iff} \quad [\![x]\!] \cap [\![y]\!] \neq \emptyset \end{array}$$

Please note that we are not rendering sentences like *all x are y* into first-order logic. Instead, we are giving a semantics to *syllogistic logic*.

Then we ask some very traditional questions. For a set Γ of sentences in this language, and for another sentence φ, we say that $\Gamma \models \varphi$ if every model $\mathcal{M}$ which satisfies all sentences in Γ also satisfies φ. This is the notion of *semantic consequence* from logic. There is a matching proof system, defining a relation $\Gamma \vdash \varphi$. For example, here are two rules

$$\frac{\textit{all } x \textit{ are } y \quad \textit{all } y \textit{ are } z}{\textit{all } x \textit{ are } z}\ \textsc{barbara} \qquad \frac{\textit{all } x \textit{ are } y \quad \textit{some } x \textit{ are } z}{\textit{some } y \textit{ are } z}\ \textsc{darii}$$

One feature of the logics in this area is that frequently there are many rules.

As expected, there is a Completeness Theorem. For the purposes of this paper, the important point is that the relation $\Gamma \vdash \varphi$ (for Γ a finite set) is in NLOGSPACE (see [20]), and it is straightforward to implement.

We return to the chart to briefly discuss the other logics. The logic $\mathcal{R}$ above $\mathcal{S}$ adds transitive verbs (such as *see*, *love*, and *hate*), interpreted as arbitrary binary relations on the universe. The syntax of $\mathcal{R}$ is bit complicated, so we won't go into full details. But the following are examples of sentences of it: *all x see some y* and *no x see all y*. In English, sentences like the first of these are ambiguous, and to make a proper logical study we take just the standard reading, where *all* has wide scope. Moving up, $\mathcal{R}^*$ adds *terms* in a recursive way to $\mathcal{R}$. We present the syntax of two related systems formally in Section 2. So *all (see all (love all dogs)) (hate some cat)* is a sentence of $\mathcal{R}^*$; its usual rendering in English would be *all who see all who love all dogs also hate some cat*. $\mathcal{R}^*(tr)$ adds comparative adjectives such as *taller than*, interpreted again as binary relations, but insisting that those interpretations be transitive (hence the *tr*) and irreflexive. Adding converse relations such as *shorter than* takes us up to $\mathcal{R}(tr, opp)$. For example, a logical rule in this logic would be to assume *all x are taller than all y* and conclude *all y are shorter than all x*.

The logics with the dagger † add full negation on all nouns. So in $\mathcal{S}^\dagger$, one can say *some non-x are non-y*. This sentence is not part of the classical syllogistic. We interpret "non" in a classical way, so $[\![\textit{non-}x]\!] = M \setminus [\![x]\!]$. If one wanted to be "non-classical" here, one certainly could.

More on natural logic may be found in the Handbook article [15].

Complexity A great deal is known about the complexity of the consequence relation $\Gamma \vdash \varphi$. The logics at the bottom of the chart, $\mathcal{S}$, $\mathcal{S}^\dagger$, $\mathcal{R}$, and $\mathcal{S}^\leq$ are in NLOGSPACE (see [20]). $\mathcal{R}^*$ is NP-COMPLETE (see [13]). The logics $\mathcal{R}^\dagger$, $\mathcal{R}^{\dagger *}$, and $\mathcal{R}^{\dagger *}(tr)$ are complete for EXPTIME. It is open to investigate approximation algorithms for these kinds of logics.

Of special interest for implementations Most of these logics have not been implemented, but of course it would be interesting to do so. Further, we do not even have approximation algorithms for these logics. All of this would be good to do. The smallest logics in the chart have been implemented, and there are some interesting features. First, once a logic has negation, contradictions are possible. Perhaps the most natural way to handle these is by adding *reductio ad absurdum* to the proof system. However, this complicates the proof search in a big way. And in general, allowing *reductio ad absurdum* in syllogistic logics raises the complexity beyond PTIME. To circumvent *reductio ad absurdum*, one can use *ex falso quodlibet*. This principle allows us to derive an arbitrary conclusion from a contradiction.

Another interesting thing about the implementations of logics "low in the chart" is that *completeness and model-building* are closely related. Here is how this works. Suppose we are given Γ and φ and want to know whether or not $\Gamma \vdash \varphi$; if this hold, we want a proof, and if not, we want a counter-model. One proves a kind of conservativity fact about these logics: if there a derivation of $\Gamma \vdash \varphi$, then there is one with the property that all terms in it are subterms of the premises or of the conclusion. This is a distant relative of the *subformula property*, and it make proof search efficient. One simply generates *all* of the consequences of Γ and looks for φ. If φ is not found, then the sentences which are consequences of it usually gives us a model in a canonical way. The upshot is that one does not need a separate model finder.

Predecessors There are two main predecessor areas to the work that I am discussing, one from Artificial Intelligence, the other from Philosophy. In AI, there is a long tradition of thinking about inference from language. Sometimes this is even taken as a primary problem of natural language processing. But most of that work does not propose small fragments the way we are doing it here. Still, some early papers in the area do work (in effect) with fragments, such as [18, 13]. Those papers have observations that are useful even now. The other background area is Philosophy, especially to those interested in reconstruction of ancient logical systems such as the syllogistic. For this line of work, see [11, 6, 4, 12]. The main difference between it and what we have mentioned so far is that the syllogistic logics in the natural logic area are *extended* syllogistic logics; we are willing to go beyond the ancient systems in every way.

2 Tableau for Beyond Syllogistic Logic

The most prominent topic for the ARQNL workshop is *quantified non-classical logics*. I take it that "quantified" here means "having the standard quantifiers." In Section 3 below, we consider a different sense of "quantified": dealing with quantities (but doing so in a non-classical way). But first, in this section, I want to suggest another direction. Let us return to the chart in Figure 1. The logics above the line marked Aristotle can be shown to have *no syllogistic proof system, not even one that uses (*RAA*)*. The language that I want to mention in this section is $\mathcal{R}^{*\dagger}$. It is a term logic; the variables range over subsets of a model, not individuals. Its syntax comes from [20] and is found in Figure 2. (Actually, I have changed the original syntax in small ways, and my syntax is closer to that of [14].)

For the semantics, we interpret nouns by subsets of the universe M of a model (as in the previous section), verbs by relations on M, noun and verb complements classically, and then inductively interpret terms by

$$\begin{array}{lcl} [\![r \textit{ all } x]\!] & = & \{n \in M : \text{for all } m \in [\![x]\!], (m,n) \in [\![r]\!]\} \\ [\![r \textit{ some } x]\!] & = & \{n \in M : \text{for some } m \in [\![x]\!], (m,n) \in [\![r]\!]\} \end{array}$$

In terms of the complexity, the consequence relation for $\mathcal{R}^*$ is CO-NP COMPLETE (see [13]). For $\mathcal{R}^{*\dagger}$, the complexity jumps to EXPTIME-complete (see [20], following Pratt-Hartmann [19]). In general, the addition of full negation is responsible for a number of complexity jumps in the area.

It is possible to construct natural deduction style proof systems for this logic and related ones [14]. People interested in proof calculi for fragments of first-order logic might find this work interesting. But

Expression	Variables	Syntax
nouns	p, q, x, y	
verbs	s	
unary literal	l	$p \mid \bar{p}$
binary literal	r	$s \mid \bar{s}$
term	c, d	$l \mid r$ *some* $c \mid r$ *all* c
$\mathcal{R}^*$ sentence	φ	*some* b^+ $d \mid$ *some* d $b^+ \mid$ *all* b^+ $d \mid$ *all* d $\overline{b^+}$

Figure 2: Syntax of $\mathcal{R}^{*\dagger}$.

nobody has investigated or implemented proof search for the logics in [14]. So from the point of view of this paper, those systems are less interesting.

Wennstrom [22] provided an *analytic tableau* system for $\mathcal{R}^{*\dagger}$. His system is complete. He showed that for every finite set s of sentences in the language which is unsatisfiable, there is a (finite) closed tableau, proving the unsatisfiability of s. And if s is satisfiable, then the proof search algorithm essentially provides us with a model. In addition, Wennstrom, provided three different tableau search strategies for the language. (We should mention that the tableau system is not strongly complete.) These all have to do with witnesses to existential sentences. In order to have finiteness results of the kind we mentioned, some care is needed.

Wennstrom also implemented all three strategies. He worked in miniKanren [5], a relational programming language embedded in Scheme. He evaluated these on a number of test sets. Evaluation was done using Petite Chez Scheme on a Windows 7 (64 bit) laptop. To give an idea, here is one of the tests:

$$\{\forall(k,m), \exists(\forall(\forall(\exists(k,h),s),r), \exists(\exists(\exists(m,h),s),\bar{r}))\}$$

(The syntax here employs evident abbreviations from what we mentioned in Figure 2. For example $\exists(t,r)$ abbreviates r *some* t.) The fourfold nesting of the quantifiers means that this example was too complicated to actually run.

Other tableau systems and implementations Wennstrom [22] was influenced by Pratt-Hartmann and Moss [20], the paper that put forward most of the systems in Figure 1. But the idea of using tableau comes from Muskens [17], the primary source on tableau methods in connection with natural language reasoning. This work was taken up by Abzianidze in his papers [1, 2, 3]. These papers build on [17], but in addition they implement the ideas. To date, this is the most sophisticated and most successful implementation effort of the kind reported in this paper. The main reason not to go into more detail on it in this paper is that I cannot point to so many problems that could interest people at the ARQNL workshop.

3 Sizes of sets

This section discusses a topic that I think could be of interest at ARQNL, the logic $\mathcal{S}^{\leq}$. This logic adds to syllogistic logic some decidedly non-first order semantics, but with a simple syntax. We add to the classical syllogistic two extra kinds of sentences: *there are at least as many x as y*, and *there are more x than y*. The semantics is just what one would expect: use set theoretic models, and use the standard notion of cardinality. In this discussion, we are going to restrict attention to finite models, since this is most in the spirit of the subject. (But one can work on infinite sets; see [16].)

Again, we should emphasize that these logical systems do not have quantifiers; for that matter, they do not have propositional connectives $\wedge$, $\vee$, or $\neg$. The only sentences are the ones mentioned. In a sense, they are the cognitive module for reasoning about size comparison of sets, in the simplest possible setting.

It might come as a surprise, but reasoning about the sizes of sets does not by itself require a “big” logical system.

There is a complete proof system. It has 22 rules and so we won’t present them all. But let us show several of them:

$$\frac{\textit{all } p \textit{ are } q \qquad \textit{there are at least as many } p \textit{ as } q}{\textit{all } q \textit{ are } p}\ (\textsc{card-mix})$$

$$\frac{\textit{there are more } q \textit{ than } p}{\textit{there are more } \overline{p} \textit{ than } \overline{q}}\ (\textsc{more-anti}) \qquad \frac{\exists^{\geq}(p,\overline{p}) \quad \exists^{\geq}(\overline{q},q)}{\exists^{\geq}(p,q)}\ (\textsc{half})$$

(CARD-MIX) requires that the universe be finite. In the last two rules, we use $\overline{x}$ as an abbreviation of *non-x*. In (CARD-MIX), we use $\exists^{\geq}(x,y)$ as an abbreviation of *there are at least as many x as y*. So, here is an explanation of the (HALF) rule. If there are at least as many p as non-p, then the p’s are at least half of the universe. So if there are at most as many q as non-q, then q’s have at most half of the elements in the universe.

As we mentioned before, the logic does not have *reductio ad absurdum*, but it has the related principle of *ex falso quodlibet*. In more detail, *reductio ad absurdum* is an admissible rule, but it is not needed. One can use the following weaker forms

$$\frac{\exists^{\geq}(p,q) \quad \exists^{>}(q,p)}{\varphi}\ (\textsc{x-card}) \qquad \frac{\textit{some } p \textit{ are } q \quad \textit{no } p \textit{ are } q}{\varphi}\ (\textsc{x})$$

These basically say that anything follows from a contradiction. It is open to construct and implement meaningful natural logics which are paraconsistent.

3.1 Implementation

The logic has been implemented in Sage[1]. This implementation is on the cloud, and the author shares it. See `https://cocalc.com/`. Other syllogistic logics have been implemented in Haskell.

For example, one may enter in the CoCalc implementation the following.

```
assumptions= [’All non-a are b’,
’There are more c than non-b’,
’There are more non-c than non-b’,
’There are at least as many non-d as d’,
’There are at least as many c as non-c’,
’There are at least as many non-d as non-a’]
conclusion = ’All a are non-c’
follows(assumptions,conclusion)
```

[1] As a programming language, Sage is close to Python. It is mainly associated with CoCalc (`https://cocalc.com/`); CoCalc incorporates a lot of mathematical software, all of it open source. It provides Jupyter notebooks and course management tools.

We are asking whether the conclusion follows from the assumptions. This particular set of assumptions is more complicated than what most people deal with in everyday life, even when they consider the sizes of sets. This is largely due to the set complements.

As we mentioned, the proof search algorithm basically gives counter-models to non-theorems. In the case of this logic, this fact takes special work. Returning to our example, the result appears instantly. (That is, for a set of assumptions of this type, the algorithm takes less than one second, running over the web.) We get back

```
The conclusion does not follow
Here is a counter-model.
We take the universe of the model to be {0,1,2,3,4,5}
noun semantics
a     {2,3}
b     {0,1,4,5}
c     {0,2,3}
d     {}
```

So the system gives the semantics of `a`, `b`, `c`, and `d` as subsets of $\{0,\ldots,5\}$. Notice that the assumptions are true in the model which was found, but the conclusion is false.

Here is an example of a derivation found by our implementation. We ask whether the putative conclusion below really follows:

$$\begin{array}{c} \textit{All non-x are x} \\ \underline{\textit{Some non-y are z}} \\ \textit{There are more x than y} \end{array}$$

In this case, the conclusion does follow, and the system returns a proof.

```
Here is a formal proof in our system:

1 All non-x are x     Assumption
2 All y are x     One 1
3 All non-x are x     Assumption
4 All non-y are x     One 3
5 Some non-y are z     Assumption
6 Some non-y are non-y     Some 5
7 Some non-y are x     Darii 4 6
8 Some x are non-y     Conversion 7
9 There are more x than y   More 2 8
```

What about propositional logic? Propositional connectives make complex sentences from sentences. We did not add the propositional connectives to this particular logic to keep the complexity low, and also to illustrate the idea that one should find special-purpose logics for topics like sizes of sets. However, one certainly could add in the connectives. The resulting logic has been explored, but we lack the space to expand on this. It is open to implement such logics, or to connect the work to SAT solvers.

4 Reasoning without grammar, and forgetting about completeness

The ultimate goal of the work that I am reporting on is to have working systems that do inference in natural language, or something close. It is clear that the program of natural logic that I have discussed, where one works with ever more complicated fragments, has a long way to go. I hope that some of the languages which we have found so far will be of interest even to people outside of the area. It is equally clear that for natural language inference itself, the method of fragments has its limitation: to get started with it, one must have a language with a clear syntax and semantics.

Readers of this paper will be surprised by the title of this section. The standard assumptions in all areas of formal logic is that logical language come with a precisely defined semantics and especially a precisely defined syntax. This syntax is usually trivial in the sense that parsing a logical language is a non-issue. For natural language, the opposite is true. Parsing is a complex matter. (The semantics is even worse, of course.)

This section loosen this assumption in connection with sizes of sets. It is based on [9, 8]. It calls on some knowledge of formal and computational linguistics.

We are interested in *polarity marking*, as shown below:

More dogs$^{\downarrow}$ *than cats*$^{\uparrow}$ *walk*$^{\downarrow}$
Most$^{\uparrow}$ *dogs*$^{=}$ *who*$^{=}$ *every*$^{=}$ *cat*$^{=}$ *chased*$^{=}$ *cried*$^{\uparrow}$
Every dog$^{\downarrow}$ *scares* $^{\uparrow}$ *at least two*$^{\downarrow}$ *cats*$^{\uparrow}$

The $^{\uparrow}$ notation means that whenever we use the given sentence truthfully, if we replace the marked word w with another word which is "$\geq w$" in an appropriate sense (see below for an example), then the resulting sentence will still be true. So we have a *semantic inference*. The $^{\downarrow}$ notation means the same thing, except that when we substitute using a word $\leq w$, we again preserve truth. Finally, the $^{=}$ notation means that we have neither property in general; in a valid semantic inference statement, we can only replace the word with itself rather than with something larger or smaller. We call $\uparrow$ and $\downarrow$ *polarity indicators*.

For example, suppose that we had a collection of background facts like *cats* $\leq$ *animals*, *beagles* $\leq$ *dogs*, *scares* $\leq$ *startles*, and one $\leq$ two. This kind of background fact could be read off from WordNet, thought of as a proxy for word learning by a child from her mother. In any case, our $^{\uparrow}$ and $^{\downarrow}$ notations on *Every dog*$^{\downarrow}$ *scares* $^{\uparrow}$ *at least two*$^{\downarrow}$ *cats*$^{\uparrow}$ would allow us to conclude *Every beagle startles at least one animal*. In general, $\uparrow$ notations permit the replacement of a "larger" word and $\downarrow$ notations permit the replacement of a smaller one.

The goal of work on the *monotonicity calculus* such as [21, 10, 9] is to provide a computational system to determine the notations $\uparrow, \downarrow, =$ on input text "in the wild". That means that the goal would be to take text from a published source or from the internet, or wherever, and to then accurately and automatically determine the polarity indicators. Then using a stock of background facts, we get a very simple "inference engine," suitable for carrying out a reasonable fraction of the humanly interesting inferences. The system would handle *monotonicity inferences* [7, 21]. Such a system would not be complete at all, because many forms of inference are not monotonicity inferences.

We must emphasize that [9] is also not a complete success. The work there depends on having a correctly parsed representation of whatever sentence is under consideration, either as a premise of an argument or as the conclusion. And here is the rub: it is rather difficult to obtain semantically useable parses. We have wide-coverage parsers; that is, programs capable of training on data (probabilistically) and then giving structure to input sentences. And the parses of interest to us are those in a grammatical framework called *combinatory categorial grammar* (CCG). CCG is a descendant of categorial grammar (CG), and it is lexicalized; that is, the grammatical principles are encoded in complex lexical types rather than in top-down phrase structure rules. From our point of view, this is a double-edged sword.

On the one hand, CG and CCG connect syntax and semantics because string concatenation in the syntax is matched by function application in the semantics, or to combinators of various sorts. On the other hand, there is no real hope of writing a complete set of rules for logical systems whose syntax is so complicated as to require a grammar of this form. In effect, we give up on completeness in order to study a syntax that is better than a toy. For that matter, working with a wide-coverage parser for a framework like CCG means that some of the parses will be erroneous in the first place. And so the entire notion of deduction will have to be reconsidered, allowing for mistakes along the way.

The connection to automated reasoning There are several reasons why this topic of automated monotonicity reasoning deserves mention in a paper on natural logic for the ARQNL community. Given a set of assumptions in natural language (or even in formal logic), if one has ↑ and ↓ information on the individual words and the constituent phrases, then a "good deal" of logical inference follows "easily", by substitution. Admittedly, we are vague here; see [8]. It is a nice open question to quantify in a meaningful way the amount of logical inference that is due to very simple operations, such as monotonicity substitution. Beyond this, there are some interesting technical questions about how exactly one would compute ↑ and ↓ information from parse trees.

5 Conclusion

My feeling is that people interested in the topics like "proof calculi, automated theorem proving systems and model finders for all sorts of quantified non-classical logics" will find much to like in the topic of natural logic. Although the systems in the area will be new, and so on a technical level one would be tempted to see differences, there is a close ideological kinship. Both are about automated reasoning, both are willing to consider new systems of various types.

We close with a review of the main points in this survey paper for people in the area.

First, first-order logic (FOL) is not the only approach to the topic of quantification: extended syllogistic logics in the natural logic family offer an alternative. That alternative is not nearly as well-studied as FOL. There are numerous computational problems related to extended syllogistic logics, and there are also a few working systems.

Even if one prefers FOL for the things that it can do, FOL cannot handle some manifestly second-order phenomena such as reasoning about the sizes of sets. We mentioned logics which can cover this reasoning and showed a simple implementation.

An old effort in AI is to study inference from text using automated logic. This topic is currently under revision, due to the connection with the monotonicity calculus. From the point of view of this paper, automating monotonicity inference is an important next step in doing automated reasoning concerning quantification.

References

[1] Lasha Abzianidze. A tableau prover for natural logic and language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 2492–2502, 2015.

[2] Lasha Abzianidze. A tableau prover for natural logic and language. In *EMNLP*, pages 2492–2502, 2015.

[3] Lasha Abzianidze. Natural solution to fracas entailment problems. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics, *SEM 2016, Berlin, Germany, 11-12 August 2016*, 2016.

[4] John Corcoran. Completeness of an ancient logic. *Journal of Symbolic Logic*, 37(4):696–702, 1972.

[5] William E. Byrd Daniel P. Friedman and Oleg Kiselyov. *The Reasoned Schemer*. MIT Press, 2005.

[6] George Englebretsen. *Three Logicians*. Van Gorcum, Assen, 1981.

[7] Bart Geurts. Monotonicity and processing load. *Journal of Semantics*, 22(1):97–117, 2005.

[8] Hai Hu, Thomas F. Icard III, and Lawrence S. Moss. Automated reasoning from polarized parse trees. In *Proc. Natural Language in Computer Science (NLCS'18)*, 2018.

[9] Hai Hu and Lawrence S. Moss. Polarity computations in flexible categorial grammar. In M. Nissim (et al), editor, *Proceedings of The Seventh Joint Conference on Lexical and Computational Semantics, *SEM 2018, New Orleans, Louisiana*, 2018.

[10] Thomas F. Icard and Lawrence S. Moss. Recent progress on monotonicity. *Linguistic Issues in Language Technology*, 9(7):167–194, 2014.

[11] Jan Łukasiewicz. *Aristotle's Syllogistic*. Clarendon Press, Oxford, 2nd edition, 1957.

[12] John N. Martin. Aristotle's natural deduction revisited. *History and Philosophy of Logic*, 18(1):1–15, 1997.

[13] David A. McAllester and Robert Givan. Natural language syntax and first-order inference. *Artificial Intelligence*, 56:1–20, 1992.

[14] Lawrence S. Moss. Logics for two fragments beyond the syllogistic boundary. In *Fields of Logic and Computation: Essays Dedicated to Yuri Gurevich on the Occasion of His 70th Birthday*, volume 6300 of *LNCS*, pages 538–563. Springer-Verlag, 2010.

[15] Lawrence S. Moss. Natural logic. In *Handbook of Contemporary Semantic Theory, Second Edition*, chapter 18. John Wiley & Sons, 2015.

[16] Lawrence S. Moss and Selçuk Topal. Syllogistic logic with cardinality comparisons on infinite sets. *Review of Symbolic Logic*, 2018.

[17] Reinhard Muskens. An analytic tableau system for natural logic. In *Logic, Language and Meaning - 17th Amsterdam Colloquium, Amsterdam, The Netherlands, December 16-18, 2009, Revised Selected Papers*, pages 104–113, 2009.

[18] Noritaka Nishihara, Kenichi Morita, and Shigenori Iwata. An extended syllogistic system with verbs and proper nouns, and its completeness proof. *Systems and Computers in Japan*, 21(1):760–771, 1990.

[19] Ian Pratt-Hartmann. Fragments of language. *Journal of Logic, Language and Information*, 13:207–223, 2004.

[20] Ian Pratt-Hartmann and Lawrence S. Moss. Logics for the relational syllogistic. *Review of Symbolic Logic*, 2(4):647–683, 2009.

[21] Johan van Benthem. *Essays in Logical Semantics*, volume 29 of *Studies in Linguistics and Philosophy*. D. Reidel Publishing Co., Dordrecht, 1986.

[22] Erik Wennstrom. Tableau-based model generation for relational syllogistic logics. In *International Symposium on Artificial Intelligence and Mathematics, ISAIM 2014, Fort Lauderdale, FL, USA, January 6-8, 2014*, 2014.