EasyChair Preprint

№ 225

Automated Reasoning from Polarized Parse Trees

Hai Hu, Thomas Icard and Larry Moss

June 2, 2018

Automated Reasoning from Polarized Parse Trees

Hai Hu* Thomas F. Icard† Lawrence S. Moss‡

Abstract

This paper contributes to symbolic inference from text, including naturally occurring text. The idea is to take sentences in a framework like CCG, and then run a *polarizing algorithm* like the one in [2] to determine inferential polarity markings of all the constituents. From this, it is just a small step to obtain an inference engine which is both simple to describe and implement and at the same time is suprisingly powerful. We have implemented the basic inference step. This paper is work in progress, also going into detail on our projected next steps. The overall goal is to have a working symbolic inference system which covers "in-practice" inference and also is correct and efficient.

1 Introduction

This paper contributes to the development of automated tools for natural language inference, building on a good deal of work in the area. We take further steps by starting with parsed input, as opposed to raw text on its own.

The leading idea is to start with a syntactic parse in a syntactic formalism which admits a good syntax-semantic interface. We are mainly working with combinatory categorial grammar (CCG) in this paper. Given a parse tree in CCG, one could hope to automatically find the polarities of all constituents. This leads to technical problems which were solved in the predecessor paper to this one, [2]. Indeed, that paper describes an algorithm and an implementation[1]. It is important to mention that this step of polarizing a CCG parse is provably faithful to the input. However, a parser frequently does not give a semantically correct parse; so as an end-to-end system, what we have is imperfect.

Be that as it may, this paper assumes that the input to the polarizing algorithm is a parse whose corresponding semantics is intuitively correct. At this point, one has correct polarities for all constituents, and one can ask about logical inference. The theme of the paper is that the logical inferences of this form cover "a good deal of the in-practice logical inference" that one would want to do. This paper explores exactly how much "a good deal" comes to here, and also discusses our

*Department of Linguistics, Indiana University, Bloomington IN 47401, USA, huhai@indiana.edu
†Department of Philosophy, Stanford University, Stanford, CA 94305, USA, icard@stanford.edu
‡Department of Mathematics, Indiana University, Bloomington IN 47401, USA, lsm@cs.indiana.edu
[1]https://github.com/huhailinguist/ccg2mono

initial experience with building computational systems to do automated symbolic reasoning using text as it comes, with the help of parses rich in constituency information.

2 Polarizing a derivation tree in CCG

Suppose that we enter the sentence *Every cat that Fido chased ran* into the C&C parser [1] of CCG. parser, and then obtain the tree below.

Input to the algorithm in [2] A parse tree $\mathcal{T}$ in CCG as shown below:

$$\dfrac{\dfrac{\mathit{Every} : NP/N \quad \mathit{cat} : N}{\mathit{Every\ cat} : NP}\,{>} \quad \dfrac{\mathit{that} : (NP\backslash NP)/(S/NP) \quad \dfrac{\dfrac{\dfrac{\mathit{Fido} : N}{\mathit{Fido} : NP}\,\text{LEX}}{\mathit{Fido} : S/(S\backslash NP)}\,\text{T} \quad \mathit{chased} : (S\backslash NP)/NP}{\mathit{Fido\ chased} : S/NP}\,\text{B}}{\mathit{that\ Fido\ chased} : NP\backslash NP}\,{>}}{\mathit{every\ cat\ that\ Fido\ chased} : NP}\,{<} \quad \mathit{ran} : S\backslash NP}{\mathit{every\ cat\ that\ Fido\ chased\ ran} : S}\,{>}$$

This is not exactly the tree returned by the parser, but the essential details match. [2] presents an algorithm which takes $\mathcal{T}$ as input. Of special interest is the fact $\mathcal{T}$ uses a rule LEX which is not part of the "official version" of CCG (or any grammar) but which used by the parser to make proper names into *NP*s. Further (and a source of more trouble), the parse itself doesn't conform to what we would expect in semantics, because the relative clause *that Fido chased* is combined after the determiner *every* combines with *cat*. We might have expected *(Every (cat that Fido chased)) ran*, and indeed we begin by "correcting" $\mathcal{T}$ on this and other points.

Then the polarity marking algorithm of [2] takes $\mathcal{T}$ and *polarizes* it, obtaining[2]:

$$\dfrac{\mathit{every}^{\uparrow} : N \xrightarrow{-} NP^{+} \quad \dfrac{\mathit{cat}^{\downarrow} : N \quad \dfrac{\mathit{that}^{\downarrow} : (NP^{+} \xrightarrow{+} S) \xrightarrow{+} (N \xrightarrow{+} N) \quad \dfrac{\dfrac{\dfrac{\mathit{Fido}^{\downarrow} : e}{\mathit{Fido}^{\downarrow} : (et) \xrightarrow{+} t}\,\text{T} \quad \mathit{chased}^{\downarrow} : e \xrightarrow{+} et}{\mathit{Fido\ chased}^{\downarrow} : et}\,\text{B}}{\mathit{Fido\ chased}^{\downarrow} : NP^{+} \xrightarrow{+} S}\,\text{J}}{\mathit{that\ Fido\ chased}^{\downarrow} : N \xrightarrow{+} N}\,{>}}{\mathit{cat\ that\ Fido\ chased}^{\downarrow} : N}\,{<}}{\mathit{every\ cat\ that\ Fido\ chased}^{\uparrow} : NP^{+}}\,{>} \quad \dfrac{\mathit{ran}^{\uparrow} : e \xrightarrow{+} t}{\mathit{ran}^{\uparrow} : NP^{+} \xrightarrow{+} t}\,\text{J}}{\mathit{every\ cat\ that\ Fido\ chased\ ran}^{\uparrow} : S}\,{<}$$

The signs $+$ and $-$ on the arrows are *markings*; markings apply to arrows only. We have a third marking, $\cdot$, but this does not figure into the tree above. Markings are used to tell if a function is interpreted (in every model) by a function which is always monotone ($+$), always antitone ($-$), or neither in general ($\cdot$). The arrows $\uparrow$ and $\downarrow$ are *polarities*. We also have a third polarity, $=$.

[2]The pdf of this submission uses colors for the markings and polarities, and so it may be easier to read than a black-and-white printout of this paper

$$\frac{(x \overset{m}{\to} y)^{d} \quad x^{md}}{y^{d}}\ {>} \qquad \frac{(x \overset{m}{\to} y)^{d} \quad (y \overset{n}{\to} z)^{md}}{(x \overset{mn}{\longrightarrow} z)^{d}}\ \mathrm{B} \qquad \frac{x^{md}}{((x \overset{m}{\to} y) \overset{+}{\to} y)^{d}}\ \mathrm{T}$$

$$\frac{(x \overset{m}{\to} (y \overset{n}{\to} z))^{d} \quad (x \overset{mn}{\longrightarrow} y)^{nd}}{(x \overset{m}{\to} z)^{d}}\ \mathrm{S} \qquad \frac{(e \to x)^{=}}{(NP \overset{+}{\to} x)^{=}}\ \mathrm{I} \qquad \frac{(e \to x)^{d}}{(NP^{+} \overset{+}{\to} x)^{d}}\ \mathrm{J} \qquad \frac{(e \to x)^{\mathit{flip}\, d}}{(NP^{-} \overset{+}{\to} x)^{d}}\ \mathrm{K}$$

Figure 1: Rules of marking and polarity for CCG. The letters m and n stand for one of the markings $+$, $-$, or $\cdot$; d stands for $\uparrow$ or $\downarrow$ (but not $=$). In (I), (J), and (K), x must be a boolean category. See charts in the text for the operations $m, d \mapsto md$ and $m, n \mapsto mn$.

Polarities are for specific occurrences. (Please note also that we are using the word "polarity" in a way different from what one finds, for example, in the linguistic literature on negative polarity items.) We shall see the value of the polarities in Section 4; inference using them is the centerpiece of this work.

Note that the input and output deviate in a number of respects, and so it is rather an involved story to say what the exact relationship holds.

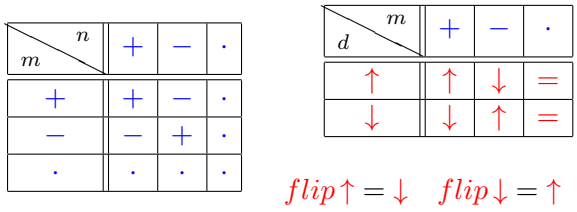
The algorithm concludes by reading off the $\uparrow$, $\downarrow$, $=$ notations for all the constituents. In particular, it reads off the notations for the leaves of the tree which we showed, obtaining

$$\textit{Every}^{\uparrow}\ \textit{cat}^{\downarrow}\ \textit{that}^{\uparrow}\ \textit{Fido}^{\downarrow}\ \textit{chased}^{\downarrow}\ \textit{ran}^{\uparrow}$$

Output specification The goal of the algorithm is to convert $\mathcal{T}$ to a different tree $\mathcal{T}^*$ satisfying the following properties: (1) the semantic terms in $\mathcal{T}$ and $\mathcal{T}^*$ should denote the same function in each model; (2) the lexical items in $\mathcal{T}^*$ must receive their types from a *typed lexicon* (ignored in this abstract, but a very important part of the system overall); (3) the polarity of the root of $\mathcal{T}^*$ must be $\uparrow$; (4) at each node in $\mathcal{T}^*$, one of the *polarity/monotonicity rules* in the system must be matched. Most of those rules are listed in Figure 1.

Explanation of the operations on markings and polarities The rules in Figure 1 are schematic in the sense that m, n, and d are variables. The rules refer to two operations which are defined in charts below. The chart on the left is for combining two markings m and n to obtain a marking mn. The chart on the right is for combining a marking m and a polarity d, obtaining a new polarity

dm. (Thus, in both cases we have an operation symbol which we elide from the notation.)



Comments on the rules In Figure 1, x, y and z are variables ranging over marked types. (Marked types are like types in the simply-typed lambda calculus, but the function types $x \to y$ give rise to three versions: $x \overset{+}{\to} y$, $x \overset{-}{\to} y$, and $x \overset{\cdot}{\to} y$.)

The application rule (>) is essentially taken from [14] (see also [6] for a survey of related algorithms); the work in [2] will also give rise to several algorithms.

To illustrate (>), let us take $m = -$ and $d = \uparrow$. By looking at the charts, we have a specific (>) rule

$$\frac{(x \overset{-}{\to} y)^{\uparrow} \quad x^{\downarrow}}{y^{\uparrow}} \; > \tag{1}$$

The rules (I), (J), and (K) are not standard in the CG literature, and indeed they are introduced in [2] in order to make the system work. In them, the semantic type x must be *Boolean*. That is, x must belong to the smallest collection B containing t and with the property that if $z \in B$, then $(y \overset{\cdot}{\to} z) \in B$ for all y. In all our work, $N = (et) = e \overset{+}{\to} t = e \overset{-}{\to} t$. And $NP = (et) \overset{\cdot}{\to} t$, $NP^{+} = (et) \overset{+}{\to} t$, and $NP^{-} = (et) \overset{-}{\to} t$.

Figure 2 shows two applications of the (K) rules. First, the lexical entry for *chased* is $e \to et$. The first application of (K) promotes this to $NP^{-} \overset{+}{\to} et$. The *NP* receives a $-$ because its argument is of type NP^{-}. Note that the polarity flips when we do this. If we had used (J), the promotion would be to $NP^{+} \overset{+}{\to} et$, and there would be no polarity flipping. This would be used in sentence where the object VP was *some cat* or *every cat*. The second application promoted *chased no cat* from the type et to $NP^{-} \overset{+}{\to} S$, again with a polarity flip. If we had used (I), we would have obtained $NP \overset{+}{\to} S$. However, this would have trivialized the polarity to $=$, and this effect would have been propagated up the tree. Rule (I) would be needed for the sentence *most dogs chased no cat*.

Several rules are not shown including "backwards" versions of (>), (B), and (T), and also versions where all polarizations are $=$. This is a technical point that is not pertinent to this short version. We should mention that due to these rules, every tree may be polarized in a trivial way, by using $=$ at all nodes. So we are really interested in the *maximally informative* polarizations, the ones that make the most predictions.

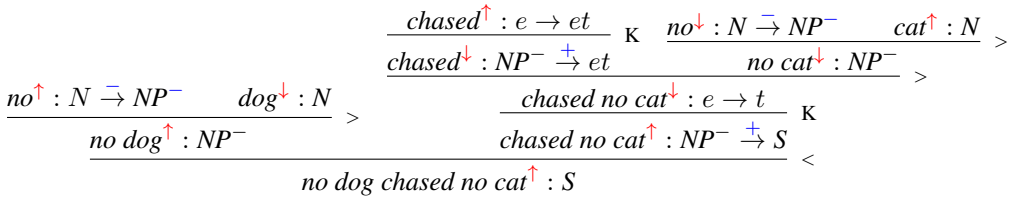


Figure 2: Two applications of the (κ) rules.

Boolean connectives, etc. We take *and* and *or* to be polymorphic of the types $B \xrightarrow{m} (B \xrightarrow{m} B)$, when B is a Boolean category and $m = +, -$, or $\cdot$. Negation flips polarities. Relative pronouns and relative clauses also can be handled. Adjectives are taken to be $N \xrightarrow{+} N$.

3 Capabilities of the Algorithm

The algorithm may be run on input sentences which are tokenized using a script from the ccg2lambda system [8] and then parsed using the C&C parser [1]. We are able to take simple sentences all the way through. For example, our system correctly determines the polarities in

No↑ *man*↓ *walks*↓
Every↑ *man*↓ *and*↑ *some*↑ *woman*↑ *sleeps*↑
Every↑ *man*↓ *and*↑ *no*↑ *woman*↓ *sleeps*=
If↑ *some*↓ *man*↓ *walks*↓, *then*↑ *no*↑ *woman*↓ *runs*↓
Every↑ *man*↓ *does*↓ *n't*↑ *hit*↓ *every*↓ *dog*↑
No↑ *man*↓ *that*↓ *likes*↓ *every*↓ *dog*↑ *sleeps*↓
Most↑ *men*= *that*= *every*= *woman*= *hits*= *cried*↑
Every↑ *young*↓ *man*↓ *that*↑ *no*↑ *young*↓ *woman*↓ *hits*↑ *cried*↑

As shown, the algorithm polarizes all words in the input. For determiners, this actually is useful. It is (arguably) background knowledge, for example that *every* $\leq$ *some*; *at least two* $\leq$ *at least one* $\equiv$ *some*, *no* $\leq$ *at most one* $\leq$ *at most two*, etc. These facts about determiners are *background facts* that figure into the inference engine which we discuss in Section 4 below.

An overall caveat This system, and the work in this paper, is only as good as the output tree obtained from the parser. As we mentioned, in many cases we must modify that tree.

4 Substitution as Inference: An Example

The new work reported in this paper begins here.

Now that we obtain the parse tree and all the polarities for each sentence, we can make simple inferences by substituting constituents in the parse tree with tree fragments that have appropriate $\leq$ relation with them. We will call a set of such $\leq$ relations a *knowledge base*, K. Such a knowledge base can be extracted from either English sentences or common NLP resources such as WordNet or DBpedia. At this stage, we are manually coding these relations to build up our knowledge base. In the future, we will automatically add English sentences to K and also use extant resources.

Predication as a source of ordering statements A sentence like *Whiskers is a cat* gives us inequalities to put into a knowledge base: *every cat* $\leq$ *Whiskers*, and also *Whiskers* $\leq$ *some cat*. Note that we want to permit $\leq$ statements between different ordered types, but those types must be in an appropriate relation called $\preceq$; see [4, 5].

Example Let K be the knowledge base K with the following six pairs in the relation $\leq$:

cat $\leq$ *animal*	*young man* $\leq$ *man*	*chased some cat* $\leq$ *liked every dog*
old dog $\leq$ *dog*	*every man* $\leq$ *John* $\leq$ *some man*	*every* $\leq$ *most*

(Most of these would be "general purpose knowledge", valid in pretty much any situation. But the assertion that chased some cat $\leq$ liked every dog is very special, and of course this is not generally valid. We include both kinds of assumptions just to show what the system contains.) And suppose that we begin with another sentence,

$$\textit{every}^{\uparrow}\ \textit{man}^{\downarrow}\ \textit{chased}^{\uparrow}\ \textit{some}^{\uparrow}\ \textit{cat}^{\uparrow}\ .^{\uparrow} \tag{2}$$

The main point for us is that this sentence (2) is already parsed and polarized. That is, the point of this paper is to carry out inference by substitution from input sentences which are parsed and polarized. We construct an algorithm called `infer(K)`.

A single call of `infer(K)` to a given polarized sentence will examine each of the constituents, and see if the constituent is contained in any pair in `K` with the matching polarity. If one such pair is found, then we do a simple substitution. For example, here is what happens when we run `infer(K)` on (2). The constituent *chased some cat* can be found in the last pair in K, and the $\leq$ relation matches the polarity $\uparrow$ on the constituent (which is not shown above). Thus `infer(K)` will substitute *liked every dog* for it, resulting in the sentence *every*$^{\uparrow}$ *man*$^{\downarrow}$ *liked*$^{\uparrow}$ *every*$^{\uparrow}$ *dog*$^{\downarrow}$.$^{\uparrow}$ Crucially, all the inferences will also be parsed and polarized, ready for further inferences by `infer(K)`.

Table 1 shows some inferences after the first few rounds of `infer(K)`. Once we have a tool for polarity computation, we can do a considerable amount of inference easily. Note that since *most* is not expressible in first-order logic, even step 1 goes beyond what could be expressed in a conventional theorem prover.

5 Natural Logic Rules and Monotonicity: What is Missing?

One theme of this paper is to ask about how much of inference could be attributed to monotonicity reasoning in some form. We have seen positive examples of where monotonicity reasoning, or something close, really can help. At this point, we want to go in the other direction. The topic of natural logic has uncovered dozens of logical systems, and in most cases those systems are logically complete. This means that any natural language argument which (in a semantic sense

After 1 substitution	every$^{\uparrow}$ young$^{\downarrow}$ man$^{\downarrow}$ chased$^{\uparrow}$ some$^{\uparrow}$ cat$^{\uparrow}$ most$^{\uparrow}$ man$^{=}$ chased$^{\uparrow}$ some$^{\uparrow}$ cat$^{\uparrow}$ John$^{\uparrow}$ chased$^{\uparrow}$ some$^{\uparrow}$ cat$^{\uparrow}$ every$^{\uparrow}$ man$^{\downarrow}$ liked$^{\uparrow}$ every$^{\uparrow}$ dog$^{\downarrow}$
After 2 substitutions	every$^{\uparrow}$ young$^{\downarrow}$ man$^{\downarrow}$ chased$^{\uparrow}$ some$^{\uparrow}$ animal$^{\uparrow}$ John$^{\uparrow}$ liked$^{\uparrow}$ every$^{\uparrow}$ dog$^{\downarrow}$ every$^{\uparrow}$ man$^{\downarrow}$ liked$^{\uparrow}$ every$^{\uparrow}$ old$^{\downarrow}$ dog$^{\downarrow}$
After 3 substitutions	some$^{\uparrow}$ man$^{\uparrow}$ chased$^{\uparrow}$ some$^{\uparrow}$ animal$^{\uparrow}$ John$^{\uparrow}$ liked$^{\uparrow}$ every$^{\uparrow}$ old$^{\downarrow}$ dog$^{\downarrow}$ every$^{\uparrow}$ young$^{\downarrow}$ man$^{\downarrow}$ liked$^{\uparrow}$ every$^{\uparrow}$ old$^{\downarrow}$ dog$^{\downarrow}$

Table 1: Examples of inferences based on the knowledge base K starting with just sentence (2). Our algorithm polarizes each constituent, but this is not shown.

given by *models*) ought to follow from some given premises really does follow. We want to ask now which features of existing natural logic systems go beyond what we have done so far in this paper. The reason for doing this is to ask what would be necessary to add to `infer(K)` in order to obtain a fuller automated system for natural language inference.

We begin with ancient traditions in logic, the classical syllogisms. Of the most common syllogistic figures, the following are obtainable from monotonicity: Barbara, Celarent, Darii, Cesare, Camestres, Baroco, Bocardo. The following syllogistic forms are not mere instances of monotonicity: Ferio (see also Section 6 below), Festino, Ferison, Fresison. The following use existential import and so are classically invalid: Darapti, Disamis.

In the simplest modern syllogistic systems (found in [12], for example), the first two inferences are *not* derivable from $\uparrow$ and $\downarrow$ polarity indications:

$$\frac{\textit{Some } y \textit{ are } x}{\textit{Some } x \textit{ are } y} \qquad \frac{\textit{Some } y \textit{ are } x}{\textit{Some } x \textit{ are } x} \qquad \frac{\textit{Some } y \textit{ (}v \textit{ some } x\textit{)}}{\textit{Some } x \textit{ are } x}$$

(and similarly with *no* replacing *some*). In logics with verbs and relative clauses, we also lack principles such as the third inference above.

It goes without saying that we cannot represent full propositional reasoning: we can represent the simplest aspects of it, since they are consequences of monotonicity, but we lack *reductio* proofs and also de Morgan's laws.

There are several "applied" topics that have been studied from a logical angle, and we wish to mention reasoning about the sizes of sets [9, 10].

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*$^{\uparrow}$

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*. 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*. But some inferential patterns would *not* be obtainable.

6 Next Steps: Combining Natural Logics

One way to augment `infer(K)` is to include some basic patterns involving expressions within a single grammatical category. We have been assuming that our knowledge base K includes inequalities "$x \leq y$", where $\leq$ has an intended interpretation for each syntactic category. For predicates in particular, we think of $A \leq B$ as meaning, *Everything that is an A is also a B*. As we mentioned, this information, especially at the basic lexical level, is readily available in large-scale lexical resources such as WordNet. We can also glean such information from any (parsed) text that tells us explicitly that *All A are B*, or by using more sophisticated techniques involving word embeddings (see [13]).

In addition to such "inclusion" information, we can also encode facts about "exclusion" into K. So this would be a worthwhile next step. For example, *dog* is excluded from *cat* in the sense that nothing is both a dog and a cat. This type of information could be extracted directly from lexical resources or by matching raw texts of the form *No A are B*. Once again more sophisticated techniques are conceivable (see [11]).

Perhaps the most natural way to incorporate this information into our system is to allow for a restricted kind of *negation*, whereby we can reason about the complement $\overline{A}$ of A, for any (Boolean) expression A. The complete logic of "$\leq$" together with complementation is known and simple. But it already allows going beyond what we have seen so far, even appealing to nothing more than monotonicity as we have presented it. Indeed, suppose we encode *No x are y* as $x \leq \overline{y}$ (i.e., *All x are non-y*). Then the following instance of a monotonicity inference (using the fact that $Some$ is monotone in its second argument) in fact gives us the syllogism Ferio, which was otherwise missing:

$$\frac{z \leq \overline{y} \quad \textit{Some } x \textit{ are } z}{\textit{Some } x \textit{ are } \overline{y}}$$

Festino, Ferison, Fresison are all obtainable in this way as well (though, strictly speaking, the latter two require knowing that *Some* is symmetric). This also allows a rich mixture of basic syllogistic reasoning with more complex expressions. For example, the following is analogous to the syllogism Fresison, but follows only from the monotonicity of *Some* (first argument) and the assumption that being partial precludes being fair (i.e., *parital* implies being *unfair*):

$$\frac{\textit{partial} \leq \overline{\textit{fair}} \quad \textit{Some partial judges cannot combat their own biases}}{\textit{Some unfair judges cannot combat their own biases}}$$

Such exclusion relations extend to the quantifiers themselves (as well as other functional expressions), in a way that mirrors the classical square of opposition. For instance, both $no \leq \overline{some}$ and $\overline{some} \leq no$, while $all \leq \overline{\textit{not-all}}$ and (assuming existential import) $all \leq \overline{no}$, and so on. With only

a bit more work we can then derive consequences like this:

$$\frac{\textit{few} \leq \overline{\textit{many}} \quad \textit{Most judges know few lawyers}}{\textit{Most judges do not know many lawyers}}$$

Adding New Function Types Adding exclusion reasoning opens up further possibilities for expanding our range of shallow inferences. As observed first by [7], functional expressions also *project* exclusion relations in predictable ways. To use an example from earlier, on the assumption that *no* $\leq \overline{\textit{some}}$, it follows that the two sentences *No dog chased no cat* and *No dog chased some cat* cannot both be true. This is because *no* is not just antitone in its second argument, but also anti-additive (see[3]). Thus, the sentences themselves stand in an exclusion relation. Using the fact that *no* $\leq \overline{\textit{some}}$ and also $\overline{\textit{some}} \leq$ *no*, we can derive that the sentences *No dog chased some cat* and *Some dog chased some cat* are not only exclusive, but also exhaustive (one or the other must be true—they are outright negations of each other). Interestingly, by stringing together these two facts, we are permitted to make the following inference (again assuming existential import, that there are dogs and cats):

$$\frac{\textit{No dog chased no cat}}{\textit{Some dog chased some cat}}$$

Joining together relations of inclusion and exclusion in this way can be done systematically to derive further patterns [7]. Some of these evidently cannot be derived by monotonicity reasoning alone (the above being one such example, even if we assume *all* $\leq$ *some* and *no* $\leq$ *some-not*). The system presented in this paper can be seamlessly expanded to include a richer array of polarity markings, including those corresponding to additivity, anti-additivity, multiplicativity, and anti-multiplicativity ([3]). The polarity marking algorithm would work in a similar way to what we presented, except that it would need to keep track of this richer function information at each step. We leave the implementation of this extension for near-future work.

7 Conclusion

Probably everyone who worked on polarity inference has noticed that if one had a good source of polarity markings, a certain amount of inference would come merely by substitution. However this observation was of limited value, because only a limited set of sentences could be marked for polarity algorithmically. With stronger polarity marking algorithms, we can re-open the matter of inference. This paper is a first attempt to build a working system. We are only starting on this project. We have taken the first step by formulating and implementing `infer(K)` (Section 4). We know many interesting inferences that are missing, even ones that could/should be added to `infer(K)` (Section 5), and also a plausible next step for this line of work (Section 6).

References

[1] Stephen Clark and James R Curran. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Computational Linguistics*, 33(4):493–552, 2007.

[2] 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.

[3] Thomas F. Icard. Inclusion and exclusion in natural language. *Studia Logica*, 100(4):705–725, 2012.

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

[5] Thomas F. Icard, Lawrence S. Moss, and William Tune. A monotonicity calculus and its completeness. In M. Kanazawa et al, editor, *Proceedings of the 15th Meeting on the Mathematics of Language*, pages 75–87. 2017.

[6] José-de-Jesús Lavalle-Martínez, Manuel Montes-y Gómez, Luis Villaseñor-Pineda, Héctor Jiménez-Salazar, and Ismael-Everardo Bárcenas-Patiño. Equivalences among polarity algorithms. *Studia Logica*, Sep 2017.

[7] Bill MacCartney and Christopher D. Manning. An extended model of natural logic. In *IWCS-8, Proceedings of the Eighth International Conference on Computational Semantics*, pages 140–156, 2009.

[8] Pascual Martínez-Gómez, Koji Mineshima, Yusuke Miyao, and Daisuke Bekki. ccg2lambda: A compositional semantics system. In *Proceedings of ACL 2016 System Demonstrations*, pages 85–90, Berlin, Germany, August 2016. Association for Computational Linguistics.

[9] Lawrence S. Moss. Syllogistic logic with cardinality comparisons. In *J. Michael Dunn on Information Based Logics*, pages 391–415. Springer Outstanding Contributions to Logic, 2016.

[10] Lawrence S. Moss and Charlotte Raty. Reasoning about the sizes of sets: Progress, problems and prospects. In *Proceedings of the Fourth Workshop on Bridging the Gap between Human and Automated Reasoning*, 2018.

[11] Ellie Pavlick, Johan Bos, Malvina Nissim, Charley Beller, Benjamin Van Durme, and Chris Callison-Burch. Adding semantics to data-driven paraphrasing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2015.

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

[13] Peter D. Turney and Saif M. Mohammad. Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering*, 21(3):437–376, 2015.

[14] Johan van Benthem. *Essays in Logical Semantics*. Reidel, Dordrecht, 1986.