A ‘Natural Logic’ inference system using the Lambek calculus

Anna Zamansky, Nissim Francez, Yoad Winter

Abstract

This paper develops an inference system for natural language within the ‘Natural Logic’ paradigm as advocated by Van Benthem [11], Sánchez [10] and others. The system that we propose is based on the Lambek calculus and works directly on the Curry-Howard counterparts for syntactic representations of natural language, with no intermediate translation to logical formulae. The Lambek-based system we propose extends the system by Fyodorov et al. [3], which is based on the Ajdukiewicz/Bar-Hillel (AB) calculus ([6]). This enables the system to deal with new kinds of inferences, involving relative clauses, non-constituent coordination, and meaning postulates that involve complex expressions. Basing the system on the Lambek calculus leads to problems with non-normalized proof terms, which are treated by using normalization axioms.

Keywords: natural logic, inference, lambek calculus, normalization

1 Introduction

Model-theoretic semantics of natural language involves partially ordered domains, so that meanings of expressions of the same semantic type are naturally comparable. Formal semantics treats order relations between denotations of complex expressions as compositionally derived from order relations between denotations of their subexpressions, described using a given grammar and semantic properties of lexical items. For instance, under standard assumptions about the meaning of the adjective tall, the meaning of nominal expressions like tall student is “smaller” than the meaning of the noun student. This ordering, together with the “order reversing” meaning of the determiner no, is responsible for the fact that the meaning of the noun phrase no tall student is “greater” than the meaning of the noun phrase no student. Such order relations between constituents often result in an ordering of meanings of natural language sentences. In an adequate semantic theory, this ordering between sentence meanings corresponds to intuitively valid entailment relations. For instance, the above mentioned order relations, together with the other elements in the sentence, are responsible for the valid entailment John saw no student $\Rightarrow$ John saw no tall student.

In modeltheoretic semantics, appealing to models makes it hard to derive inferences in a computationally feasible way. On the other hand, working with proof systems for first order logic (FOL) for natural language, as proposed in many works (see, e.g. [9]) also has its weaknesses. First of all, not all NL constructs are expressible in FOL. For instance, in the valid entailment: John is very tall $\Rightarrow$ John is tall, the restrictive modifier 'very' is not expressible in FOL. Furthermore, using FOL proof systems for computing natural language entailments requires complex mappings from syntactic structures to FOL formulae. These mappings are motivated mainly by the particular choice of syntax, and not by independent linguistic considerations. This paper follows previous work in aiming to develop an inference system that is based on insights from model-theoretic semantics, but using only syntactic representations of natural language, with no direct appeal to models. The close relationship between syntactic structure and meaning in model-theoretic semantics eliminates the need for translating the syntactic representations into intermediate logical levels of representation, such as first order logic.

This initial conception of *Natural Logic* was introduced in [11]. Different versions of Natural Logic were proposed by [10], [1], [3] and others. Sánchez [10] proposes a mechanism that decorates categorial grammar proofs of natural language expressions using signs that indicate the *monotonicity* properties of the denotations of these expressions. Bernardi [1] follows Sánchez and introduces a system for monotonicity reasoning that is based on a more complex categorial type logic than Sánchez' work. Bernardi concentrates on monotonicity reasoning as capturing the *syntactic distribution* of negative polarity items. Neither Sánchez nor Bernardi provide a formal calculus for computing inferences. The situation was partially amended in [3]. Fyodorov et al. define an *Order Calculus* based on similar annotations decorating syntactic derivations of the Ajdukiewicz/Bar-Hillel (AB) calculus, a simple version of categorial grammar that only contains slash elimination rules. Fyodorov et al.'s calculus allows a rather straightforward derivation of inferences with monotone and some non-monotone quantifiers and cross-categorial conjunctions and disjunctions. However, despite the value of Fyodorov et al.'s proposal for demonstrating a novel technique of inference in natural language, it fails to derive many inferences, even ones that are strictly based on simple semantic order relations between expressions (see some examples below). One of the reasons for this weakness comes from the limitations of the AB calculus as a categorial grammar.

In this paper we show that Fyodorov et al.'s system can be improved by basing the inference mechanism on the Lambek calculus ($\mathcal{L}$, see [7, 8]), which also contains slash introduction rules in addition to the elimination rules of the AB calculus. We propose an $\mathcal{L}$-based Order Calculus ($\mathcal{L}$-OC) as an intermediate step towards a more general system that would support various kinds of inferences in natural language in a more expressive syntactic framework. As in the previous works that were mentioned, the items on which the inferential system works are syntactic terms, representing structural derivations of natural language expressions. These derivations now also include deductions using hypothetical reasoning produced by the introduction rule of the Lambek calculus, and not only function-argument constructions as in the AB-based order calculus of Fyodorov et al. Despite the more general syntactic formalism we employ, the manipulation of semantically-motivated annotations is still done at the level of the syntactic representation. In this sense, we believe that our proposal is within the realm of what previous works called *Natural Logic*.

We extend the system of Fyodorov et al. in a number of aspects:

1. The inferences are computed using proof terms representing syntactic derivations via the Curry-Howard correspondence, as opposed to the manipulation in [3] of the syntactic derivations directly.

2. We add an inference rule called *Abstraction*, which works on proof terms in $\mathcal{L}$ with free variables, corresponding to undischarged assumptions. This rule enables the Order Calculus to deal with inferences involving sentences with extraction. Consider for example the following entailments, which are now derivable in our system (using additional axioms, as will be shown in the sequel):

   (a) Every student whom Mary touched smiled $\Rightarrow$ Every student whom Mary kissed smiled

   (b) Some boy, the brother of whom Mary loves, walked $\Rightarrow$ Some boy walked

3. We add β/η-normalization axioms, based on β/η-reduction of proof terms, which resolve complications caused by proof terms in $\mathcal{L}$ that are not in

normal form. The normalization axioms enable the system to compute more entailments, like, for instance, entailments that involve "non-constituent" conjunctions as in the following example:

John does and Mary doesn't move $\Rightarrow$ Mary doesn't walk

4. Extending the system of [3] to $\mathcal{L}$ also enables us to formulate non-logical axioms about complex expressions. For example, it is possible to express the fact that the relation denoted by passionately love (though not necessarily the love relation itself) contains in every model the relation denoted by adore. This is made possible due to the derivability of function composition in $\mathcal{L}$.

As in [3], we concentrate on entailments between natural language sentences that are syntactically disambiguated. For the sake of simplicity, we do not assume any ambiguity at a semantic level. A proof search procedure for $\mathcal{L}$-OC, which is an extension of the proposal by [3] including treatment of abstraction, is formulated in [15].

The structure of this paper is as follows. Section 2 provides definitions of some basic notions from model-theoretic semantics, and introduces *decorated semantic types* to be used in $\mathcal{L}$-OC. Section 3 describes $\mathcal{L}$ derivations and their corresponding proof terms with decorated semantic types. Section 4 defines the Order Calculus $\mathcal{L}$-OC and its semantics. Section 5 demonstrates how $\mathcal{L}$-OC can be applied for deriving natural language inferences. Section 6 focuses on the problem of normalization, explaining how the non-normalized proof terms are created in $\mathcal{L}$-OC, why they pose a problem and how the problem is solved using normalization. Section 8 presents conclusions and directions for further research.

2 Semantic types and order relations

The main objective of the Natural Logic systems, as introduced by Van Benthem, Sánchez, and Fyodorov et al., is to use the boolean regularities in natural language (cf. [2]) as a key for an inference system that works directly on syntactic representations. In this section we review the basic boolean semantic notions that will be employed in this work, and introduce the way they are used for decorating types by semantic features.

2.1 Basic semantic notions

Model-theoretic semantic theories associate natural language expressions with *syntactic categories*, and their denotations with (closely related) *semantic types*. Furthermore, most expressions denote objects in partially ordered (PO) domains, so that meanings of equi-typed expressions are naturally comparable. Thus in the finite set of *primitive types* (denoted by T^0), we distinguish the subset of *partially ordered primitive types*, denoted by T^0_{po}, which are interpreted over partially ordered domains.

Formally, the set of *types* is defined as the smallest set T so that $T^0 \subseteq T$ and if $\tau \in T$ and $\sigma \in T$ then also $(\tau\sigma) \in T$. The set of *PO types* is the smallest set $T_{po} \subseteq T$ s.t. ${T^0}_{po} \subseteq T_{po}$ and if $\tau \in T$ and $\sigma \in T_{po}$ then also $(\tau\sigma) \in T_{po}$. Standardly, types *e* (for entities) and t (for truth values) are among the primitive types, where t is among the PO primitive types.

For each primitive type $\tau \in T^0$, let D_τ be a non-empty domain, assuming that the domains for primitive types are mutually disjoint. We also assume that the domain D_σ of any primitive PO type σ is endowed with a given partial order relation $\leq_\sigma$. For each non-primitive type $\tau\sigma$, the domain $D_{\tau\sigma}$ is the set of all functions from D_τ to D_σ. The partial order $\leq_{\tau\sigma}$ for complex PO types is defined pointwise: if σ is a PO type with partial order $\leq_\sigma$ over the domain D_σ, then for any $d_1, d_2 \in D_{\tau\sigma}$: $d_1 \leq_{\tau\sigma} d_2$ iff for every $d' \in D_\tau$ $d_1(d') \leq_\sigma d_2(d')$.

Next, we review some semantic properties of functions over these typed domains, which will be useful in the rest of this paper. First, we refer to types of the form $\tau\tau$ as *modifier types*. When τ is a PO type, a function $f \in D_{\tau\tau}$ of the modifier type $\tau\tau$ is called *restrictive* iff for every $d \in D_\tau$: $f(d) \leq_\tau d$. For example, the denotations of adjectives like tall, pretty and adverbs like slowly, happily are commonly analyzed as restrictive functions of type $(et)(et)$. Thus, it is assumed that the denotation of an expression like tall boy is "smaller" or equal to the denotation of the expression boy, and that the denotation of slowly move is "smaller" or equal to the denotation of move. Order relations produced by restrictive modifiers are one of the simplest ways for generating order relations between natural language expressions in the order calculus.

Another important source for order relations are expressions of the *coordination types* – types of the form $\tau(\tau\tau)$. Functions of this type in natural language are often interpreted as *greatest lower bound* or *least upper bound* operators. A function $f \in D_{(\tau(\tau\tau))}$, where τ is a PO type, is called a *greatest lower bound* (g.l.b.) function iff for all $d_1, d_2, d_3 \in D_\tau$ the following two conditions hold:

1. $(f(d_1))(d_2) \leq_\tau d_1$ and $(f(d_1))(d_2) \leq_\tau d_2$;

2. if $d_3 \leq_\tau d_1$ and $d_3 \leq_\tau d_2$ then $d_3 \leq_\tau (f(d_1))(d_2)$.

The first requirement requires that f be restrictive, or returns a lower bound, on both of its arguments; the second requirement ensures that f returns a *greatest* lower bound on both of its arguments.

A dual notion is the notion of *least upper bound* (l.u.b.) functions: a function $f \in D_{\tau(\tau\tau)}$ of the coordination type $\tau(\tau\tau)$, where τ is a PO type, is called a *l.u.b.* function iff for all $d_1, d_2, d_3 \in D_\tau$ the following two conditions hold:

1. $d_1 \leq_\tau (f(d_1))(d_2)$ and $d_2 \leq_\tau (f(d_1))(d_2)$;

2. if $d_1 \leq_\tau d_3$ and $d_2 \leq_\tau d_3$ then $(f(d_1))(d_2) \leq_\tau d_3$.

In natural language there are at least three kinds of g.l.b. functions:

1. Conjunctions: the standardly assumed meaning of conjunctions such as dance and smile, Mary danced and John smiled, and every teacher and some student is the g.l.b. of the meanings of the conjuncts.

2. Relative clauses: a 'subject oriented' relative clause such as child who sneezed is treated as a g.l.b. of the noun (child) denotation and the verb phrase (sneezed) denotation. Similarly, an 'object oriented' relative clause such as child whom Mary saw is treated as a g.l.b. of the noun (child) denotation and the denotation of the "gapped" verb phrase (Mary saw), which is interpreted as the set of objects seen by Mary.

3. Intersective adjectives: adjectives such as blue and pregnant when viewed as modifiers are often assumed to denote 'intersective functions': functions of type $((et)(et))$ that intersect their argument with an implicit argument of type (et). For instance, the nominal blue car is synonymous with the nominal car that is blue, which is formed using a g.l.b. relative.

One l.u.b. function in natural language is the disjunction or: the standardly assumed meaning of disjunctions such as dance or smile, Mary danced or John smiled, and every teacher or some student is the l.u.b. of the meanings of the disjuncts.

Another useful property of functions in natural language is *monotonicity*, namely, order preservation/reversal. Let σ_1 and σ_2 be PO types. A function $f \in D_{(\sigma_1\sigma_2)}$ is:

- *upward monotone* iff for all $d_1, d_2 \in D_{\sigma_1}$: $d_1 \leq_{\sigma_1} d_2 \Rightarrow f(d_1) \leq_{\sigma_2} f(d_2)$;

- *downward monotone* iff for all $d_1, d_2 \in D_{\sigma_1}$: $d_1 \leq_{\sigma_1} d_2 \Rightarrow f(d_1) \geq_{\sigma_2} f(d_2)$.

For example, the denotation of the determiner every is analyzed as a function of type $((et)((et)t))$ that is downward monotone w.r.t. its first argument and upward monotone w.r.t. its second argument. In this way we capture the following entailments:

- Every student ran $\Rightarrow$ Every tall student ran (assuming tall student $\leq$ student)
- Every student ran $\Rightarrow$ Every student moved (assuming ran $\leq$ moved)

2.2 Decoration of types

In order to use the semantic properties that were reviewed above in a calculus, we follow [3] and use semantic decorations of types of linguistic expressions as an *abstraction* of their full denotations. In this way the decorated type of an expression E can be used to derive order relations between more complex expressions containing E, and ultimately entailment relations with sentences containing E.

We first define the set of *semantic features* that decorate types according to the semantic properties discussed above.

Definition 2.2.1 *(Semantic features) The set of semantic features* $Feat = \{+, -, R, C, D\}$.

Henceforth we use the meta-variables F, F' to range over subsets of $Feat$.
The intended use of these marks is as follows:

- '+'/'-' marks upward/downward monotonicity of functional types $\tau\sigma$, where both τ and σ are PO types.
- 'R' marks restrictivity of modifier types $\tau\tau$, where τ is a PO type.
- 'C'/'D' marks g.l.b./l.u.b. behavior of coordination types $\tau(\tau\tau)$, where τ is a PO type.

Definition 2.2.2 *(Decorated types and decorated PO types) Let T^0 be a set of primitive types and T^0_{PO} a set of primitive PO types, such that $T^0_{PO} \subseteq T^0$. The sets of decorated types and PO decorated types are the smallest sets T_{dec}, T^{PO}_{dec} so that:*

- *$T^0 \subseteq T_{dec}$, $T^0_{PO} \subseteq T^{PO}_{dec}$ (null decoration)*
- *if $\tau \in T_{dec}$, $\sigma \in T_{dec}$ and $\rho \in T^{PO}_{dec}$ then $(\tau^F\sigma) \in T_{dec}, (\tau^F\rho) \in T^{PO}_{dec}$ for any $F \subseteq Feat$ satisfying the following conditions:*

    1. If $F \neq \emptyset$, then $\tau, \sigma \in T^{PO}_{dec}$.

2. *If $R \in F$ then $\tau = \sigma$.*
3. *If C or $D \in F$ then (i) If $\tau = (\tau_1{}^{F'}\tau_2)$ then $F' = \emptyset$ and (ii) $\sigma = (\tau^{\emptyset}\tau)$.*

Condition 1 guarantees that only functional types $(\tau^F\sigma)$, where both τ and σ are PO decorated types can be marked with $F \neq \emptyset$. Condition 2 guarantees that only modifier types are marked with 'R'. Condition 3 guarantees that an expression of a type marked with 'C' or 'D' is treated as denoting a binary function and all its markings are specified on the functor type.

Definition 2.2.3 *Let τ be a decorated type in T_{dec}. The (non-decorated) type $\tau^{\circ} \in T_{dec}$ corresponding to τ is defined by:*

1. *If τ is primitive then $\tau^{\circ} = \tau$.*
2. *If $\tau = (\alpha^F\beta)$ then $\tau^{\circ} = (\alpha^{\circ}\beta^{\circ})$.*

After defining the decorated types, the corresponding domains are naturally defined as follows.

Definition 2.2.4 *(Domains of decorated types) For each non-primitive decorated type $\tau^F\sigma \in T_{dec}\backslash T^0$, the domain $D_{\tau^F\sigma} \subseteq D_{\tau\sigma}$ is the set of functions in $D_{\tau\sigma}$ that have the semantic properties denoted by the semantic features in F.*

For example, $D_{(\sigma^+\tau)}$ is the set of upward monotone functions from D_σ to D_τ.

3 The calculus $\mathcal{L}$

In the proposed system, entailments between natural language sentences are computed based on lambda terms with decorated types, representing the syntactic derivations of these sentences. The Lambek calculus is an appealing formalism to be used in such framework because of the built-in interface between the syntactic structure of natural language expressions and their compositional semantics due to the Curry-Howard correspondence between proofs[1] and lambda terms. We use the product-free associative Lambek calculus (in its Natural Deduction formulation, see [8]) and extend it to the calculus $\mathcal{L}$ defined below enriched with decorations of semantic types. The set of syntactic categories **CAT** is the smallest set, such that a finite set of primitive categories $\mathbf{CAT}^0$ (standardly containing s) is included in it, and for every $A, B \in \mathbf{CAT}$: $(A/B), (A\backslash B) \in \mathbf{CAT}$. Let

[1]By 'proofs' we mean here derivation from assumptions.

$\mathbf{type^0} : \mathbf{CAT}^0 \rightarrow T$ be a typing function for primitive categories, such that $\mathbf{type}^0(s) = t$. This function is extended to the function $\mathbf{type} : \mathbf{CAT} \rightarrow T$ as follows[2]: $\mathbf{type}(A/B) = \mathbf{type}(A\backslash B) = (\mathbf{type}(A)\mathbf{type}(B))$.

In our original presentation of $\mathcal{L}$-OC ([14, 15]) we presented the system using *directed* linear lambda terms, as only such terms materialize the Curry-Howard correspondence with L as 1-1. However, directionality is not essential for conveying the main ideas of L-OC, and here we simplify the presentation by using ordinary linear lambda terms (see [12]). $Free(\psi)$, the set of free variables of a linear term ψ, is defined standardly. For any term γ such that no free variables of γ occur bound in α, the term $\alpha[x/\gamma]$ is obtained from α by substituting all free occurrences of x by γ.
Our term language contains also a set $\mathbf{Const}$ of constants, that are in a 1-1 correspondence with the set of natural language words. The NL words are displayed in san-serif font, and the constants in italic font. Thus, the constant *girl* corresponds to the word girl. Most importantly, the constants are typed, carrying the decorated types driving the inferences.
Similarly to [3], we say that two decorated types τ, σ are *formally equivalent*, and denote it by $\tau \equiv_f \sigma$, if τ and σ are identical up to their decoration.

Definition 3.1 *(The calculus $\mathcal{L}$) Let $\Gamma, \Gamma_1, \Gamma_2$ range over finite non-empty sequences of pairs $A : \psi_\tau$, where A is a syntactic category and ψ_τ a term of a (decorated) type τ. The notation $\Gamma \triangleright A : \psi_\tau$ means that the sequence Γ is reducible to $A : \psi_\tau$. The rules of $\mathcal{L}$ are as follows :*

$$(axiom) A : x_\tau \triangleright A : x_\tau,\ where\ x_\tau \in \mathbf{VAR} \cup \mathbf{Const}\ and\ \mathbf{type}(A) = \tau^\circ$$

$$(/E)\frac{\Gamma_1 \triangleright (A/B) : \psi_{(\tau_1{}^F\tau_2)} \quad \Gamma_2 \triangleright B : \varphi_{\tau_1'}}{\Gamma_1\Gamma_2 \triangleright A : (\psi_{(\tau_1{}^F\tau_2)}(\varphi_{\tau_1'}))_{\tau_2}},\quad (\backslash E)\frac{\Gamma_2 \triangleright B : \varphi_{\tau_1'} \quad \Gamma_1 \triangleright (A\backslash B) : \psi_{(\tau_1{}^F\tau_2)}}{\Gamma_2\Gamma_1 \triangleright A : (\psi_{(\tau_1{}^F\tau_2)}(\varphi_{\tau_1'}))_{\tau_2}}$$

$$where\ \tau \equiv_f \tau_1', \mathbf{type}(A) = \tau_2^\circ,\ \mathbf{type}(B) = \tau_1^\circ$$

$$(/I)\frac{\Gamma_1,\ B : x_{\tau_1} \triangleright A : \psi_{\tau_2}}{\Gamma_1 \triangleright (A/B) : (\lambda x_{\tau_1}.\psi_{\tau_2})_{(\tau_1\tau_2)}}\quad (\backslash I)\frac{B : x_{\tau_1},\ \Gamma_1 \triangleright A : \psi_{\tau_2}}{\Gamma_1 \triangleright (A\backslash B) : (\lambda x_{\tau_1}.\psi_{\tau_2})_{(\tau_1\tau_2)}}$$

$$for\ \Gamma_1\ not\ empty, \mathbf{type}(A) = \tau_2^\circ,\ \mathbf{type}(B) = \tau_1^\circ$$

[2]Note that the function $\mathbf{type}$ returns a non-decorated type.

If a sequent $\Gamma \triangleright A : \psi$ has a proof in $\mathcal{L}$, we denote it by $\vdash_{\mathcal{L}} \Gamma \triangleright A : \psi$.

Definition 3.2 *(Type-Logical Categorial Grammar) A type-logical categorial grammar is a tuple $G = \langle \Sigma, \mathbf{CAT}^0, A^0, \alpha \rangle$[3], where:*

- *Σ is the alphabet.*
- *$\mathbf{CAT}^0$ is the set of basic syntactic categories.*
- *A^0 is the target category.*
- *$\alpha : \Sigma \to 2^{\mathbf{CAT} \times \mathbf{Const}}$ is an assignment of finite sets of (abstracted) signs, pairs of categories and constants, to lexical items, such that for every $\langle A, w_\tau \rangle \in \alpha(\mathsf{w})$: $\tau^\circ = \mathbf{type}(A)$. We will refer to an assignment α as a* lexicon.
- *The* abstracted language $L[G]$ *is defined as:*

$$L[G] = \{\langle \mathsf{w}, M_t \rangle \mid \exists \Gamma \in \alpha(\mathsf{w}), s.t. \vdash_{\mathcal{L}} \Gamma \triangleright s : M_t\}$$

Note that M, which usually specifies the semantic denotation *of* w*, is used here to carry the abstracted type of the denotation. In a similar way, we define $L[G, A]$, the expressions of category A, so that $L[G] = L[G, s]$.*

4 The $\mathcal{L}$-based Order Calculus

In this section we introduce the main part of the proposed system – the $\mathcal{L}$-based Order Calculus ($\mathcal{L}$-OC). $\mathcal{L}$-OC manipulates ordered pairs of proof terms that represent $\mathcal{L}$ derivations of natural language expressions. These pairs, which are referred to as *order statements*, are so defined to specify semantic order relations between denotations of proof terms. Similarly to [3], however, order statements are purely syntactic objects with no direct appeal to models (as opposed to the works of [10, 1]). The soundness proof in [13] implies that the denotations of terms in a $\mathcal{L}$-OC-provable order statement indeed satisfy the ordering in every model.

4.1 $\mathcal{L}$-OC

Order statements – the items that are manipulated by $\mathcal{L}$-OC– are defined to be of the form $\varphi_\tau \leq_{\tau^\circ} \psi_{\tau'}$, where φ and ψ are directed lambda terms of formally equivalent types τ and τ', and τ° is the non-decorated type derived by recursively erasing the decorations from τ (or equivalently, from τ').

[3]Standardly, A_0 is taken to be s, the category designated for sentences in natural language.

The definition of $\mathcal{L}$-OC contains, similarly to the system of [3], rules of the following three kinds:

1. Structural rules of reflexivity (REFL) and transitivity (TRANS) for the order relation $\leq_\tau$.
2. Rules that describe the order behavior of monotonic expressions (MON+ and MON-), restrictive modifiers (RMOD), conjunctions ($\text{C}_{1\text{-}2}$) and disjunctions ($\text{D}_{1\text{-}2}$).
3. A rule of "function replacement" (FR), which captures the pointwise behavior of the order relation.

In addition to the rules of the Order Calculus of [3], $\mathcal{L}$-OC also includes an abstraction (Ab) rule and standard β and η normalization axioms. The abstraction rule of $\mathcal{L}$-OC is used for deriving order statements between terms that are obtained using the introduction rule of the $\mathcal{L}$ calculus. The normalization axioms solve some problems that appear due to possible loss of semantic features when derivations of order relations contain non-normalized terms.

Definition 4.1.1 *($\mathcal{L}$-OC:)*

For $\tau \equiv_f \tau' \equiv_f \hat{\tau} \equiv_f \tilde{\tau},\ \rho \equiv_f \rho' \equiv_f \hat{\rho}$[4]:

$$\textbf{(REFL)}\frac{\emptyset}{\alpha_\tau \leq_{\tau^\circ} \alpha_{\tau'}} \quad \textbf{(TRANS)}\frac{\alpha_\tau \leq_{\tau^\circ} \delta_{\tau'} \quad \delta_{\tau'} \leq_{\tau^\circ} \gamma_{\hat{\tau}}}{\alpha_\tau \leq_{\tau^\circ} \gamma_{\hat{\tau}}}$$

$$\textbf{(MON+)}\frac{\alpha_\tau \ \leq_{\tau^\circ} \ \delta_{\tau'}}{\gamma_{(\hat{\tau}^+\rho)}(\alpha_\tau) \ \leq_{\rho^\circ} \ \gamma_{(\hat{\tau}^+\rho)}\,(\delta_{\tau'})} \quad \textbf{(MON-)}\frac{\delta_{\tau'} \ \leq_{\tau^\circ} \ \alpha_\tau}{\gamma_{(\hat{\tau}^-\rho)}(\alpha_\tau) \ \leq_{\rho^\circ} \ \gamma_{(\hat{\tau}^-\rho)}\,(\delta_{\tau'})}$$

$$\textbf{(FR)}\frac{\alpha_{(\tau^F\rho)} \ \leq_{(\tau\rho)^\circ} \ \psi_{(\tau'^{F'}\rho')} \quad \gamma_{\hat{\tau}} \ \equiv_{\tau^\circ} \ \delta_{\tilde{\tau}}}{\alpha_{(\tau^F\rho)}\,(\gamma_{\hat{\tau}}) \ \leq_{\rho^\circ} \ \psi_{(\tau'^{F'}\rho')}\,(\delta_{\tilde{\tau}})} \quad \textbf{(RMOD)}\frac{\emptyset}{\alpha_{(\tau^R\tau')}\,(\gamma_{\hat{\tau}}) \ \leq_{\tau^\circ} \ \gamma_{\hat{\tau}}}$$

$$\textbf{(Ab)}\frac{\alpha_\rho \ \leq_{\rho^\circ} \ \gamma_{\rho'}}{\lambda x_\tau.\alpha_\rho \ \leq_{(\tau\rho)^\circ} \ \lambda x_{\tau'}.\gamma_\rho}$$

$$\lambda x.\alpha, \lambda x.\gamma \ are\ linear\ terms$$

[4]Note that τ° is equal to τ without any semantic decorations, thus $\tau^\circ \equiv_f \tau \equiv_f \tau' \equiv_f \hat{\tau} \equiv_f \tilde{\tau}$ and $\leq_{\tau^\circ}$ is compatible with $\leq_\tau,\ \leq_{\tau'},\ \leq_{\hat{\tau}}$ and $\leq_{\tilde{\tau}}$. The case for ρ° is similar.

$$\textbf{(C}_1\textbf{)}\frac{\emptyset}{(\delta_{(\tau^C(\tau\tau))}(\gamma_{\tau'}))(\psi_{\hat{\tau}}) \ \leq_{\tau^\circ} \ \Omega} \qquad \textbf{(C}_2\textbf{)}\frac{\alpha_{\tilde{\tau}} \ \leq_{\tau^\circ} \ \psi_{\tau'} \quad \alpha_{\tilde{\tau}} \ \leq_{\tau^\circ} \ \gamma_{\hat{\tau}}}{\alpha_{\tilde{\tau}} \ \leq_{\tau^\circ} \ (\delta_{(\tau^C(\tau\tau))}(\gamma_{\hat{\tau}}))(\psi_{\tau'})}$$

$$\Omega = \psi_{\hat{\tau}} \ or \ \Omega = \gamma_{\tau'}$$

$$\textbf{(D}_1\textbf{)}\frac{\emptyset}{\Omega \ \leq_{\tau^\circ} \ (\delta_{(\tau^D(\tau\tau))}(\gamma_{\tau'}))(\psi_{\hat{\tau}})} \qquad \textbf{(D}_2\textbf{)}\frac{\psi_{\tau'} \ \leq_{\tau^\circ} \ \alpha_{\hat{\tau}} \quad \gamma_{\tilde{\tau}} \ \leq_{\tau^\circ} \ \alpha_{\hat{\tau}}}{(\delta_{(\tau^D(\tau\tau))}(\gamma_{\tilde{\tau}}))(\psi_{\tau}) \ \leq_{\tau^\circ} \ \alpha_{\hat{\tau}}}$$

$$\Omega = \psi_{\tau'} \ or \ \Omega = \gamma_{\tilde{\tau}}$$

Normalization axioms:

$$\textbf{(}\beta\textbf{)}\frac{\emptyset}{(\phi_\tau[y_\rho/\gamma_{\rho'}])_\tau \equiv_{\tau^\circ} (\lambda y_\rho.\phi_\tau)_{(\rho\tau)}(\gamma_{\rho'})} \quad \textbf{(}\eta\textbf{)}\frac{\emptyset}{\psi_{(\tau^F\rho)} \equiv_{(\tau\rho)^\circ} (\lambda x_\tau.\psi_{(\tau^F\rho)}(x_\tau))_{(\tau\rho)}}$$

$$x_\tau \notin Free(\psi)$$

The Abstraction (Ab) rule captures the discharge of an assumption in a $\mathcal{L}$-derivation. Given a premise $\varphi_1 \leq \varphi_2$, where both φ_1 and φ_1 represent derivation trees with a free variable x occurring exactly once, the order statement $\lambda x.\varphi_1 \leq \lambda x.\varphi_2$ is derived. The normalization axioms (β) and (η) capture β and η reductions of proof terms. The application of these axioms is discussed in detail in section 6. For explanation of the rest of the rules, the reader is referred to [3].

4.2 The semantics of $\mathcal{L}$-OC

The *semantics* of $\mathcal{L}$-OC is naturally defined using standard models (i.e., full Henkin models, see [5]) for the extensional fragment of Montague's IL [4] and the point-wise definition of semantic order relations. A model M is a set of (non-empty) domains D_τ for every primitive type $\tau \in T^0_{dec}$. For each non-primitive type $\sigma = (\tau^F\rho)$, $D_{(\sigma)}$ is the domain of all functions from D_τ to D_ρ, satisfying the semantic conditions specified by F. Every proof term φ_ρ is associated with a *denotation* $[\![\varphi_\rho]\!]_{M,g}$ relative to a model M and an assignment function g, which assigns to any variable of decorated type ρ some element of D_ρ.

Definition 4.2.1 *(Denotations of proof terms) Let M be a model and g an assignment function. For a given proof term ψ_τ, the denotation $[\![\psi_\tau]\!]_{M,g}$ is defined as follows:*

- *If $\psi_\tau \in$ VAR, then $[\![\psi_\tau]\!]_{M,g} = g(\psi_\tau)$.*

- *If $\psi_\tau = \varphi_{(\sigma^F\tau)}(\phi_\sigma)$, then $[\![\psi_\tau]\!]_{M,g} = [\![\varphi_{(\sigma^F\tau)}]\!]_{M,g}([\![\phi_\sigma]\!]_{M,g})$.*
- *If $\psi_\tau = \lambda x_\sigma.\varphi_\rho$, then $[\![\psi_\tau]\!]_{M,g}$ is that function $h \in D_\tau$ s.t. for all $d \in D_\sigma$: $h(d) = [\![\varphi_\rho]\!]_{M,g[x:=d]}$, where $g[x := d]$ is an assignment function similar to g, except that it assigns d_σ to x_σ.*

Definition 4.2.2 *(Semantics of order statements) Let φ_1, φ_2 be terms of (decorated) type τ and g an assignment function.*

1. *$M, g \models \varphi_1 \leq_\tau \varphi_2$ iff $[\![\varphi_1]\!]_{M,g} \leq_\tau [\![\varphi_2]\!]_{M,g}$*
2. *$M \models \varphi_1 \leq_\tau \varphi_2$ iff $\forall g:\ M, g \models \varphi_1 \leq_\tau \varphi_2$.*

In [13] it is shown that $\mathcal{L}$-OC is strongly sound relative to this semantics, that is:

$$\vdash_{\mathcal{L}-OC} \alpha \leq \gamma \ \Rightarrow\ \forall M, g : [\![\alpha]\!]_{M,g} \leq [\![\gamma]\!]_{M,g}$$

5 $\mathcal{L}$-OC-based inference system for natural language

This section illustrates how $\mathcal{L}$-OC can be used for deriving inferences in natural language. We first introduce a toy lexicon which is used for defining a small fragment of English. Then we define a way to represent natural language assertions as $\mathcal{L}$-OC order statements. In addition, we extend the postulate introduced by [3] for universal quantification in order to expand the range of inferences derived by the system. We also introduce some non-logical axioms for complex expressions. Finally, we present examples of deriving inferences with sentences involving relative clauses and pied piping, as well as inferences using the extended postulate for universal quantification.

To keep the relation between terms (derivations) and natural language expressions clear, we sometimes denote a term $\varphi(\psi)$ by $[\psi]\varphi$, thus restoring the distinction between the rightward and leftward slash elimination rules. For instance, the (normalized) $\mathcal{L}$-derivation of adores and loves is represented by the directed term $([adores]and)(loves)$, rather than the non-directed term $(and(adores))(loves)$.

5.1 Lexicon

A lexicon in a type-logical categorial grammar is a function $\alpha : \Sigma \rightarrow 2^{\mathbf{CAT}\times\mathbf{Const}}$, from words to finite sets of pairs of categories and constants with decorated types. These sets are of course non-empty, and contain more than one pair for any word that is lexically ambiguous. In Table 1 we introduce a toy lexicon for a fragment of

Word	**Category**	**Type**
W^T	s	t
every	$((s/(s\backslash np))/n), ((s\backslash(s/np))/n)$	$((et)^-((et)^+t))$
no	$((s/(s\backslash np))/n), ((s\backslash(s/np))/n)$	$((et)^-((et)^-t))$
some	$((s/(s\backslash np))/n), ((s\backslash(s/np))/n)$	$((et)^+((et)^+t))$
student,boy	n	(et)
walk,walked,smile, smiled, move, moved	$(s\backslash np)$	(et)
touched, loved	$((s\backslash np)/np)$	$(e(et))$
tall, nice, smart, intelligent, creative	(n/n)	$(et)^R(et)$
Mary, John	$(s/(s\backslash np)), (s\backslash(s/np))$	$((et)^+t)$
does	$((s\backslash np)/(s\backslash np))$	$(et)^+(et)$
doesn't	$((s\backslash np)/(s\backslash np))$	$(et)^-(et)$
whom	$((n\backslash n)/(s/np))$	$(et)^C((et)(et))$
the-brother-of-whom	$((n\backslash n)/(s/np))$	$(et)^C((et)(et))$
and	$((s\backslash s)/s), (((s\backslash np)\backslash(s\backslash np))/(s\backslash np))$	$(t^C(tt)), ((et)^C((et)(et))),$

Table 1: Lexicon

English, including the decorated types that are assigned to the decorated syntactic categories.

Some remarks on this lexicon are in place:

1. Following [3], we use a fictitious sentence W^T that is assigned the constant proof term w_t^T. This proof term is used in the representation of a natural language assertion S as the order statement $w_t^T \leq_t \psi_t^S$, where ψ_t^S is a proof term representing an $\mathcal{L}$-derivation of S. This representation of assertions makes it easy to treat natural language sentences using order relations, where w_t^T is understood as a sentential "top-element" term, with a denotation that is constantly *true*.

2. Determiners and proper names are assigned two categories, which allow them to appear in both subject and object positions. The semantic markings of their types, as in [3], captures their monotonicity properties. For instance, the determiner *every* is marked as downward monotone on its noun argument, and upward monotone on its verb argument.

5.2 Natural Logic inferences

In general, we represent Natural Logic inferences in $\mathcal{L}$-OC as follows.

Definition 5.2.1 *($\vdash_{NatLog}$) Let G be some type-logical grammar. Let $S, S_1, ..., S_n$ be non-ambiguous sentences in $L[G]$ (i.e., having only one reading) and let $\alpha_t^S, \alpha_t^{S_1}, ..., \alpha_t^{S_n}$ be any proof terms representing $\mathcal{L}$-derivation trees of $S, S_1, ..., S_n$*

$$\boxed{\textbf{Input: Sentences } S, S_1, ..., S_n} \Rightarrow \boxed{\textbf{Find } \mathcal{L} \textbf{ proof terms } \psi^S, \psi_t^{S_1}, ..., \psi_t^{S_n} \textbf{ for } S, S_1, ..., S_n} \Rightarrow$$

$$\boxed{\textbf{Prove } \vdash_{\mathcal{L}-OC} w^T \leq \psi^S \textbf{ from } \vdash_{\mathcal{L}-OC} w^T \leq \psi^{S_1}, ..., \vdash_{\mathcal{L}-OC} w^T \leq \psi^{S_n}}$$

Figure 1: Deriving $S_1, ..., S_n \vdash_{NatLog} S$ in the system

respectively. We say that $S_1, ..., S_n \vdash_{NatLog} S$ *if* $\vdash_{\mathcal{L}-OC} w_t^T \leq \alpha_t^{S_1}, ..., \vdash_{\mathcal{L}-OC} w_t^T \leq \alpha_t^{S_n}$ *implies* $\vdash_{\mathcal{L}-OC} w_t^T \leq \alpha_t^S$.

In order to prove $S_1 \vdash_{NatLog} S_2$, it is enough to show $\vdash_{\mathcal{L}-OC} \alpha_1 \leq \alpha_2$, where α_1, α_2 are proof terms representing derivations of S_1, S_2 resp., and the rest follows from transitivity. We do so in all the following examples to shorten up the presentation.
The general process of deriving $S_1, ..., S_n \vdash_{NatLog} S$ is summarized in Figure 1. In this paper we only handle the case of $n = 1$.

5.3 *Non-logical axioms and the* ‘every’ *postulate*

Non-logical axioms are order statements that reflect possible meaning assumptions on the denotations of natural language expressions. For example, in the models that we would like to consider, a student is also a person, and a walking object is a moving object. It is also natural to assume that in the relevant models a *creative intelligent X* is a *smart X*, for any nominal *X*. Similarly, we may assume that *passionately loving* some *x* entails *adoring x*, for any entity *x*. The following non-logical axioms of $\mathcal{L}$-OC, which correspond to these intuitions, will be useful in the examples of $\mathcal{L}$-OC proofs that are introduced below.

$$\frac{\emptyset}{walked_{(et)} \leq moved_{(et)}}\ a_1 \qquad \frac{\emptyset}{walk_{(et)} \leq move_{(et)}}\ a_2$$

$$\frac{\emptyset}{kissed_{(et)} \leq touched_{(et)}}\ a_3 \qquad \frac{\emptyset}{student_{(et)} \leq person_{(et)}}\ a_4$$

$$\frac{\emptyset}{\lambda x_{(et)}.creative_{((et)^R(et))}(intelligent_{((et)^R(et))}(x_{(et)})) \leq smart_{((et)^R(et))}}\ a_5$$

$$\frac{\emptyset}{\lambda x_e.passionately_{((et)^R(et))}(loves_{(e(et))}(x_e)) \leq adores_{(e(et))}}\ a_6$$

Our $\mathcal{L}$-based system, as opposed to the AB-based system of [3], makes it possible to define non-logical axioms involving complex proof terms. For example, one of the terms involved in a_4 is a composition of two functional terms $creative_{((et)^R(et))}$ and $intelligent_{((et)^R(et))}$, which can not be derived in the less powerful AB calculus.

Another advantage of $\mathcal{L}$ is that it allows us to use the ad hoc postulate that [3] defines for the determiner 'every' also for other positions beside subject position. The determiner 'every' is treated by inducing an order statement between its two arguments. For example, from the order statement

$$w_t^T \leq (\textit{every}_{(et)^-((et)^+t)}\ \textbf{(}\textit{student}\textbf{)})\ \textbf{(}\textit{smiled}\textbf{)}$$

[3] induce the order statement $student_{(et)} \leq smiled_{(et)}$. However, [3] cannot handle a similar case when 'every' is in an object position. For example, the fact that the order statement

$$(*)\ \textit{student}_{(et)} \ \leq\ \lambda x_e.(\textit{Mary}_{((et)^+t)}\textit{(kissed}_{(e(et))}\textbf{(}x_e\textbf{))})$$

should be induced from the order statement

$$w_t^T \leq \textbf{[}\lambda x_e.\textit{Mary}_{((et)^+t)}\textbf{(}\ \textit{kissed}_{(e(et))}\textbf{(}x_e\textbf{))]}(\textit{every}_{(et)^-((et)^+t)}\textbf{(}\textit{student}_{(et)}\textbf{)})$$

cannot be accounted for by [3]. We define a generalized postulate for 'every' as follows:

$$\boxed{\frac{w_t^T\ \leq\ (every_{(et)^-((et)^+t)}\ (\alpha_{(et)}))\ (\gamma_{(et)})}{\alpha_{(et)}\ \leq\ \gamma_{(et)}}\ (\textbf{(ev)})}$$

Some examples for the use of the postulates above for deriving inferences are given in the following subsection.

5.4 *Examples of $\mathcal{L}$-OC derivations*

In this subsection we show examples of $\mathcal{L}$-OC derivations of inferences involving sentences with relative clauses and pied piping, as well as inferences using the extended 'every' postulate, which cannot be derived by the AB-based system in [3]. Instances of the Reflexivity axiom are omitted. Also, when the rule FR is used with its second premise an identity (not just formal equivalence), the second premise is omitted for brevity.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\emptyset}{kissed_{(e(et))} \;\leq\; touched_{(e(et))}}\,(a_3)}{(kissed\,(x_e))_{(et)} \;\leq\; (touched\,(x_e))_{(et)}}\,\text{FR}}{(Mary_{((et)^+t)}\,(kissed\,(x)))_t \;\leq\; (Mary_{((et)^+t)}\,(touched\,(x)))_t}\,\text{MON+}}{(\lambda x.Mary(kissed\,(x)))_{(et)} \;\leq\; (\lambda x.Mary\,(touched\,(x)))_{(et)}}\,\text{Ab}}{\begin{array}{c}(whom_{(et)^C((et)(et))}\,(\lambda x.Mary\,(kissed\,(x))))_{(et)(et)} \leq \\ (whom_{(et)^C((et)(et))}\,(\lambda x.Mary(touched\,(x))))_{(et)(et)}\end{array}}\,\text{MON+}}{\begin{array}{c}([student_{(et)}]\,(whom\,(\lambda x.Mary\,(kissed\,(x)))))_{(et)} \leq \\ ([student_{(et)}]\,(whom\,(\lambda x.Mary\,(touched\,(x)))))_{(et)}\end{array}}\,\text{FR}}{\begin{array}{c}(every_{(et^-)((et)^+t)}\,([student]\,(whom\,(\lambda x.Mary\,(touched\,(x))))))_{((et)^+t)} \;\leq \\ ((every_{(et^-)((et)^+t)}\,([student]\,(whom\,(\lambda x.Mary\,(kissed\,(x))))))_{((et)^+t)}\end{array}}\,\text{MON-}}{\begin{array}{c}((every\,([student]\,(whom\,(\lambda x.Mary\,(touched\,(x))))))(smiled_{(et)}))_t \leq \\ ((every\,([student]\,(whom\,(\lambda x.Mary\,(kissed\,(x))))))\,(smiled_{(et)}))_t\end{array}}\,\text{FR}
$$

Figure 2: Relative clauses:
Every student whom Mary touched smiled $\vdash_{NatLog}$Every student whom Mary kissed smiled

$$\cfrac{\cfrac{\cfrac{\emptyset}{[boy_{(et)}](the-brother-of-whom_{((et)^C((et)(et)}(\lambda x.Mary(loves(x))_{(et)})) \le boy_{(et)}}\ \text{C1}}{some_{(et)^+((et)^+t)}([boy_{(et)}](the-brother-of-whom_{((et)^C((et)(et)}(\lambda x.Mary(loves(x))_{(et)}))) \le some_{(et)^+((et)^+t)}(boy)}\ \text{MON+}}{some_{(et)^+((et)^+t)}([boy_{(et)}](the-brother-of-whom_{((et)^C((et)(et)}((\lambda x.Mary(loves(x)))_{(et)})))(walked_{(et)}) \le (some_{(et)^+((et)^+t)}(boy))(walked_{et})}\ \text{FR}$$

Figure 3: Pied piping: Some boy, the brother of whom Mary loves, walked $\vdash_{NatLog}$Some boy walked

$$\cfrac{\cfrac{\cfrac{\cfrac{w_t^T \le ([\lambda x_e.Mary_{((et)^+t)}\ (kissed_{(e(et))}\ (x_e))]\ (every_{(et)^-((et)^+t)}\ (student_{(et)})))_t}{student_{(et)}\ \le\ (\lambda x.Mary\ (kissed\ (x)))_{(et)}}\ (ev)}{student_{(et)}\ \le\ ([student_{(et)}]\ whom_{(et)^C(et)(et)}\ (\lambda x.Mary\ (kissed\ (x))))_{(et)}}\ \text{C2}}{(No_{(et)^-((et)^-t)}\ ([student]\ (whom\ (\lambda x.Mary\ (kissed\ (x))))))_{((et)^-t)} \le (No_{(et)^-((et)^-t)}\ (student))_{((et)^-t)}}\ \text{MON-}}{\cfrac{w_t^T\ \le\ ((No\ ([student]\ (whom\ (\lambda x.Mary\ (kissed\ (x))))))\ (walked_{(et)}))_t\ \le ((No\ (student))(walked_{(et)}))_t}{w_t^T\ \le\ ((No\ (student))(walked))_t}\ \text{TRANS} \times 2}\ \text{FR}$$

Figure 4: Using the extended non-logical postulate for ‘every’:Mary kissed every student, No student whom Mary kissed walked $\vdash_{NatLog}$ No student walked

Figures 2 and 3 illustrate the derivation of inferences with simple relative clauses including an object gap. In Figure 2 we see an $\mathcal{L}$-OC derivation, from which it follows that Every student whom Mary touched smiled $\vdash_{NatLog}$ Every student whom Mary kissed smiled. In this derivation the Abstraction rule of $\mathcal{L}$-OC is used to discharge the assumption x_e. In Figure 3 we show a $\mathcal{L}$-OC derivation, from which we conclude Some boy, the brother of whom Mary loves, walked $\vdash_{NatLog}$ Some boy walked. For simplicity we assume that the category $((n\backslash n)/(s/np))_{(et)^C((et)(et))}$ can be derived for the expression the brother of whom. In Figure 4 the extended 'every' postulate is used. Figures 5 and 6 illustrate the use of $\mathcal{L}$-OC for term composition as in the non-logical axioms *a4* and *a5*.

6 Normalization in $\mathcal{L}$-OC

In this section we focus on normalization[5] in $\mathcal{L}$-OC. First of all, we demonstrate how non-NF proof terms emerge in $\mathcal{L}$-OC. Consider the following example.

$$\dfrac{\dfrac{\emptyset}{\lambda x_{(et)}.creative_{((et)^R(et))}(intelligent_{((et)^R(et))}(x_{(et)})) \leq smart_{((et)^R(et))}}\, a_5}{\lambda x.creative(intelligent(x))(boy_{(et)}) \leq smart(boy_{(et)})}\,\text{FR}$$

In this example the term $\lambda x.creative(intelligent(x))(boy)$ is not in NF and it β-reduces to $creative(intelligent(boy))$.
Another example for the creation of non-NF terms is as follows:

$$\dfrac{\dfrac{\emptyset}{happy_{(et)^R(et)}(tall_{(et)^R(et)}(x_{(et)})) \leq tall_{(et)^R(et)}(x_{(et)})}\,\text{RMOD}}{\lambda x.happy_{(et)^R(et)}(tall_{(et)^R(et)}(x_{(et)})) \leq \lambda x.tall_{(et)^R(et)}(x_{(et)})}\,\text{Ab}$$

Here, the term $\lambda x.tall(x)$ is not in NF and it η-reduces to $tall$.

In these examples we see two main problematic aspects in the emergence of non-normalized terms in $\mathcal{L}$-OC. The first problem is *abstraction terms* with unmarked semantic types. Basing the system on $\mathcal{L}$ allows us to derive order statements that involve non-lexical expressions, and apply composition of terms in the representation of their derivation. In AB, in contrast, the creation of functional terms that do not originate from the lexicon is impossible due to the lack

[5]Normalization in $\mathcal{L}$-OC was initially proposed in [15].

of introduction rules. In $\mathcal{L}$-OC new functional terms that are created via abstraction can apply as functions to other terms, creating non-NF terms. Some of the abstraction terms may denote monotone (restrictive, etc.) functions, but their types are not respectively marked. For instance, consider the abstraction term $\mu = \lambda x_\tau.\psi_{(\sigma^+\rho)}(\phi_{(\tau^+\sigma)}(x_\tau))$, which is a composition of the terms ϕ and ψ. Since their types are marked for upward monotonicity, the denotation of their composition is an upward monotone function. Thus, given $\vdash_{\mathcal{L}-OC} \gamma_\tau \leq_\tau \delta_\tau$, we expect $\mathcal{L}$-OC to derive $(\lambda x.\psi(\phi(x)))(\gamma) \leq_\rho (\lambda x.\psi(\phi(x)))(\delta)$. But the type of $\lambda x.\psi(\phi(x))$ is not marked for monotonicity, thus we cannot directly use the MON+ rule (or any other $\mathcal{L}$-OC rules).

Let us show a more concrete example. Using (a_1), MON and FR we can derive:

$$\vdash_{\mathcal{L}-OC} Mary(doesn't(move)) \leq Mary(doesn't(walk))$$

Consider, however a similar order statement:

$$\vdash_{\mathcal{L}-OC} [\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move) \leq$$

$$\lambda y.Mary(doesn't(y))(walk)$$

Since the type of $\lambda y.Mary(doesn't(y))$ is not marked for downward monotonicity, without normalizing it we cannot use the non-logical axiom (a_1) in any way. In Figure 7 we show a derivation of this example using β-normalization. Alternatively, instead of normalization we can add to $\mathcal{L}$-OC a mechanism for dynamically marking monotonicity of types, so that markings of lexical expressions are correctly inherited by complex expressions, also with non-normalized derivations. Such a method is described in the next section.

Another problematic aspect of non-NF terms in $\mathcal{L}$-OC is *effectiveness considerations*. In the general architecture of our system (Figure 1), one of the integral parts is finding $\mathcal{L}$-derivations for the goal sentences. However, finding a non-normalized derivation of a natural language expression is problematic due to the lack of the sub-formula property in non-normalized derivations, which in turn creates an infinite proof search space. Therefore, any realistic $\mathcal{L}$ parser would search for normal form derivations only. Thus, for the purpose of implementation, we need to express the relation between non-NF terms representing $\mathcal{L}$ derivations of the goal sentences and their normal form equivalents. This technical development is carried out in [13].

$$\dfrac{\dfrac{\emptyset}{\begin{gathered}([adores]and(\lambda x.passionately(loves(x))))\\ \le \lambda x.passionately(loves(x))\end{gathered}}\,{\scriptstyle \mathrm{C1}} \qquad \dfrac{\dfrac{\emptyset}{passionately_{(et)^R(et)}(loves(x)) \le loves(x)}\,{\scriptstyle \mathrm{RMOD}}}{\lambda x.passionately(loves(x)) \le \lambda x.loves(x)}\,{\scriptstyle \mathrm{Ab}} \qquad \dfrac{\emptyset}{\lambda x.loves(x) \equiv loves}\,\eta}{\dfrac{\dfrac{\dfrac{\dfrac{([adores]and(\lambda x.passionately(loves(x)))) \le loves}{([adores]and(\lambda x.passionately(loves(x)))(y)) \le loves(y)}\,{\scriptstyle \mathrm{FR}}}{Mary([adores]and(\lambda x.passionately(loves(x)))(y)) \le Mary(loves(y))}\,{\scriptstyle \mathrm{MON}+}}{\lambda y.(Mary([adores]and(\lambda x.passionately(loves(x)))(y))) \le \lambda y.Mary(loves(y))}\,{\scriptstyle \mathrm{Ab}}}{\begin{gathered}[\lambda y.([Mary([adores]and(\lambda x.passionately(loves(x)))(y))])]John\\ \le [\lambda y.Mary(loves(y))]John_{(et^+t)}\end{gathered}}\,{\scriptstyle \mathrm{MON}+}}\,{\scriptstyle \mathrm{TRANS}}$$

Figure 5: Mary adores and passionately loves John $\vdash_{NatLog}$ Mary loves John

$$\dfrac{\dfrac{\emptyset}{\begin{gathered}creative(intelligent(boy)) \equiv\\ \lambda x.creative(intelligent(x))(boy)\end{gathered}}\,\beta \qquad \dfrac{\dfrac{\emptyset}{\lambda x_{(et)}.creative_{((et)^R(et))}(intelligent_{((et)^R(et))}(x_{(et)})) \le smart_{(et)}}\,(a_5)}{\lambda x.creative(intelligent(x))(boy_{(et)}) \le smart(boy_{(et)})}\,{\scriptstyle \mathrm{FR}}}{\dfrac{\dfrac{creative(intelligent(boy)) \le smart(boy)}{Some_{((et)^+((et)^+t))}(creative(intelligent(boy))) \le Some_{((et)^+((et)^+t))}(smart(boy))}\,{\scriptstyle \mathrm{MON}}}{Some(creative(intelligent(boy)))(smiled) \le Some(smart(boy))(smiled)}\,{\scriptstyle \mathrm{FR}}}\,{\scriptstyle \mathrm{TRANS}}$$

Figure 6: Some creative intelligent boy smiled $\vdash_{NatLog}$ Some smart boy smiled, using the non-logical axiom (a5).

$$\boxed{\dfrac{\boxed{\dfrac{\dfrac{\emptyset}{\begin{array}{c}[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))\\ \leq\ \lambda y.Mary(doesn't(y))\end{array}}\,\text{C1}}{\begin{array}{c}[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move)\\ \leq (\lambda y.Mary(doesn't(y)))(move)\end{array}}\,\text{FR}} \quad \begin{array}{c}(\lambda y.Mary(doesn't(y)))(move)\\ \equiv Mary(doesn't(move))\end{array}\ (\beta) \quad \boxed{\dfrac{\dfrac{\dfrac{\emptyset}{walk_{(et)} \leq move_{(et)}}\ (a_2)}{\begin{array}{c}doesn't_{(et)^-(et)}(move)\\ \leq doesn't(walk)\end{array}}\,\text{MON-}}{\begin{array}{c}Mary_{((et)^+t)}(doesn't(move))\\ \leq Mary_{((et)^+t)}(doesn't(walk))\end{array}}\,\text{MON+}}}{[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move) \leq Mary(doesn't(walk))}\,\text{TRANS}}$$

Figure 7: The problematic inference is derivable using β-normalization

7 Dynamic marking

In this section we focus on the problem of marking the types of *abstraction* terms created in $\mathcal{L}$-OC derivations. We describe a method of dynamic marking (first proposed in [14]), which marks the types of abstraction terms for monotonicity. We show that the proposed method does not in fact increase the expressive power of $\mathcal{L}$-OC , in the sense that any derivation using dynamic marking can be simulated by $\mathcal{L}$-OC.

To implement dynamic marking we use the notion of *polarity* introduced by [11] and used by [10] and [1].

Definition 7.1 *(Polarity of occurrences) Given a term ψ and a subterm ϕ of ψ, a specified occurrence of ϕ in ψ is called* positive *(*negative*) according to the following clauses:*

1. *ϕ is positive in ϕ.*
2. *If $\psi = \alpha(\gamma)$ then:*
   - *ϕ is positive (negative) in ψ if ϕ is positive (negative) in α.*
   - *ϕ is positive (negative) in ψ if ϕ is positive (negative) in γ and α denotes an upward monotone function.*
   - *ϕ is negative (positive) in ψ if ϕ is positive (negative) in γ and α denotes a downward monotone function.*
3. *If $\psi = \lambda x.\mu$ then ϕ is positive (negative) in ψ if ϕ is positive (negative) in μ.*

Fact 7.2 *[11] If x is positive (negative) in ϕ then $\lambda x.\phi$ denotes an upward (downward) monotone function.*

Dynamic marking is performed by $\mathcal{DDL}$ – an extended version of $\mathcal{L}$. Instead of linear lambda terms, $\mathcal{DDL}$ uses *extended* linear terms, where variables are assigned a polarity marking $\Pi \in Pol = \{\oplus, \ominus, \otimes\}$, which is an *abstraction* of its actual polarity: $\oplus, \ominus, \otimes$ mark positive, negative and unspecified polarity respectively.

Definition 7.3 *(Extended linear terms) Let $\mathbf{V^E} = \{y^\Pi \mid y \in \mathbf{VAR}, \Pi \in Pol\}$. The set ELTerms is the smallest set s.t.:*

- $\mathbf{V^E} \cup \mathbf{Const} \subseteq$ *ELTerms*

- *If* $\Phi_{(\sigma^F\tau)}, \Psi_\sigma \in$ *ELTerms , then* $(\Phi(\Psi))_\tau \in$ *ELTerms*
- *If* $x_\rho \in \mathbf{VAR}$*,* $\Phi_\tau \in$ *ELTerms and for some* $\Pi \in Pol$*:* x_ρ^Π *occurs in* Φ_τ *exactly once, then* $(\lambda x.\Phi)_{(\rho\tau)} \in$ *ELTerms*

The set of free variables (with an assigned polarity marking) $Free(\Psi)$ for $\Psi \in$ **ELTerms** is standardly defined.

Definition 7.4 *(*Π*-strip) For* $\Psi \in$ *ELTerms , its* Π*-strip linear term* $\Pi strip(\Psi)$ *is defined as follows:*

$$\Pi strip(\alpha^\Pi) = \alpha,\ for\ \alpha \in \mathbf{VAR}, \Pi \in Pol$$

$$\Pi strip(\Phi(\Delta)) = \Pi strip(\Phi)(\Pi strip(\Delta))$$

$$\Pi strip(\lambda x.\Phi) = \lambda x.\Pi strip(\Phi)$$

In words, $\Pi strip(\Psi)$ is the term obtained from Ψ by deleting the polarity marking of all of its variables. It is easy to show that given a linear extended term Ψ, representing some $\mathcal{DDL}$ derivation, the term $\Pi strip(\Psi)$, obtained by deleting all polarity markings from Ψ, is a linear term representing some $\mathcal{L}$ derivation.
Next, we define the functions $Flip$: **ELTerms** $\rightarrow$ **ELTerms** and $Anull$: **ELTerms** $\rightarrow$ **ELTerms**. $Flip(\Psi)$ is the (extended) term obtained from Ψ by swapping the polarity marking of the free variables in Ψ as follows: '$\ominus$' to '$\oplus$', '$\oplus$' to '$\ominus$', '$\otimes$' to '$\otimes$'. $Anull(\Psi)$ is the (extended) term obtained from Ψ by setting the polarity marking of all the free variables in Ψ to '$\otimes$'. We also define the function $Pol2Feat : Pol \rightarrow 2^{Feat}$ that decorates the type of abstraction terms according to the polarity marking of the discharged assumption:

$$Pol2Feat(\oplus) = \{+\},\ Pol2Feat(\ominus) = \{-\},\ Pol2Feat(\otimes) = \emptyset$$

Definition 7.5 *(*$\mathcal{DDL}$*) Let* $\Gamma, \Gamma_1, \Gamma_2$ *range over finite non-empty sequences of pairs* $A : \Psi_\tau$*, where* A *is a syntactic category and* $\Psi \in ELTerms$*. The notation* $\Gamma \triangleright A : \Psi_\tau$ *means that the sequence* Γ *is* $\mathcal{DDL}$*-reducible to* $A : \Psi_\tau$*. The rules of* $\mathcal{DDL}$ *are as follows :*

$$(axiom_1)\ A : x_\tau^\oplus \triangleright A : x_\tau^\oplus\ \ for\ x_\tau \in \mathbf{V^E}\ and\ \mathbf{type}(A) = \tau^\circ$$

$$(axiom_2)\ A : w_\tau \triangleright A : w_\tau\ \ for\ w_\tau \in \mathbf{Const}\ and\ \mathbf{type}(A) = \tau^\circ$$

$$for\ \tau_1 \equiv_f \tau_1',\ \mathbf{type}(A) = \tau_2^\circ,\ \mathbf{type}(B) = \tau_1^\circ :$$

Elimination rules:

$$(/E_\ominus)\frac{\Gamma_1 \rhd (A/B) : \Psi_{(\tau_1 - \tau_2)} \quad \Gamma_2 \rhd B : \Phi_{\tau_1'}}{\Gamma_1\Gamma_2 \rhd A : (\Psi_{(\tau_1 - \tau_2)}(Flip(\Phi_{\tau_1'})))_{\tau_2}} \ (\backslash E_\ominus)\frac{\Gamma_2 \rhd B : \varphi_{\tau_1'} \quad \Gamma_1 \rhd (A\backslash B) : \Psi_{(\tau_1 - \tau_2)}}{\Gamma_2\Gamma_1 \rhd A : (\Psi_{(\tau_1 - \tau_2)}(Flip(\Phi_{\tau_1'})))_{\tau_2}}$$

$$(/E_\oplus)\frac{\Gamma_1 \rhd (A/B) : \Psi_{(\tau_1 + \tau_2)} \quad \Gamma_2 \rhd B : \Phi_{\tau_1'}}{\Gamma_1\Gamma_2 \rhd A : (\Psi_{(\tau_1 + \tau_2)}(\Phi_{\tau_1'}))_{\tau_2}}, \ (\backslash E_\oplus)\frac{\Gamma_2 \rhd B : \Phi_{\tau_1'} \quad \Gamma_1 \rhd (A\backslash B) : \Psi_{(\tau_1 + \tau_2)}}{\Gamma_2\Gamma_1 \rhd A : (\Psi_{(\tau_1 + \tau_2)}(\Phi_{\tau_1'}))_{\tau_2}}$$

$$For F \notin \{\{+\}, \{-\}\} :$$

$$(/E)\frac{\Gamma_1 \rhd (A/B) : \Psi_{(\tau_1{}^F \tau_2)} \quad \Gamma_2 \rhd B : \Phi_{\tau_1'}}{\Gamma_1\Gamma_2 \rhd A : (\Psi_{(\tau_1{}^F \tau_2)}(Anull(\Phi_{\tau_1'})))_{\tau_2}}, \ (\backslash E)\frac{\Gamma_2 \rhd B : \Phi_{\tau_1'} \quad \Gamma_1 \rhd (A\backslash B) : \Psi_{(\tau_1{}^F \tau_2)}}{\Gamma_2\Gamma_1 \rhd A : (\Psi_{(\tau_1{}^F \tau_2)}(Anull(\Phi_{\tau_1'})))_{\tau_2}}$$

Introduction rules:

$$(/I)\frac{\Gamma_1, \ B : x_{\tau_1}^{\Pi} \rhd A : \Psi_{\tau_2}}{\Gamma_1 \rhd (A/B) : (\lambda x_{\tau_1}.\Psi_{\tau_2})_{(\tau_1^{Pol2Feat(\Pi)}\tau_2)}} \ (\backslash I)\frac{B : x_{\tau_1}^{\Pi}, \ \Gamma_1 \rhd A : \Psi_{\tau_2}}{\Gamma_1 \rhd (A\backslash B) : (\lambda x_{\tau_1}.\Psi_{\tau_2})_{(\tau_1^{Pol2Feat(\Pi)}\tau_2)}}$$

$$for \ \Gamma_1 \ not \ empty, \ x_{\tau_1} \in \mathbf{VAR}$$

The Elimination rules update the polarity marking of the variables of the argument term: a downward monotone function triggers a "flipping" of polarity marking, an upward monotone function leaves polarity intact, and a function unmarked for monotonicity nullifies polarity marking. The Introduction rules mark the type of the dynamically created functional term according to the polarity marking of the abstracted variable corresponding to the discharged assumption. (The polarity marking of the bound variables becomes irrelevant.)

The rules of the Order Calculus that is based on $\mathcal{DDL}$ ($\mathcal{DDL}$-OC) are very similar to the rules of $\mathcal{L}$-OC, except that they manipulate order-statements between extended linear terms, representing $\mathcal{DDL}$ derivations. We do not include normalization axioms in $\mathcal{DDL}$-OC .

Definition 7.6 *($\mathcal{DDL}$-OC :)*

$$\text{For } \tau \equiv_f \tau' \equiv_f \hat{\tau} \equiv_f \tilde{\tau}, \ \rho \equiv_f \rho' \equiv_f \hat{\rho}$$

$$\textbf{(REFL)}\frac{\emptyset}{\Psi_\tau \leq_{\tau^\circ} \Psi_{\tau'}} \ \textbf{(TRANS)}\frac{\Psi_\tau \leq_{\tau^\circ} \Sigma_{\tau'} \ \ \Sigma_{\tau'} \leq_{\tau^\circ} \Phi_{\hat{\tau}}}{\Psi_\tau \leq_{\tau^\circ} \Phi_{\hat{\tau}}}$$

$$\textbf{(MON+)}\frac{\Psi_\tau \ \leq_{\tau^\circ} \ \Sigma_{\tau'}}{\Phi_{(\hat{\tau}+\rho)}(\Psi_\tau) \ \leq_{\rho^\circ} \ \Phi_{(\hat{\tau}+\rho)} \ (\Sigma_{\tau'})} \ \ \textbf{(MON-)}\frac{\Sigma_{\tau'} \ \leq_{\tau^\circ} \ \Psi_\tau}{\Phi_{(\hat{\tau}-\rho)}(Flip(\Psi_\tau)) \ \leq_{\rho^\circ} \ \Phi_{(\hat{\tau}-\rho)} \ (Flip(\Sigma_{\tau'}))}$$

$$\textbf{(FR)}\frac{\Psi_{(\tau^F\rho)} \ \leq_{(\tau\rho)^\circ} \ \Gamma_{(\tau'^{F'}\rho')} \ \ \Phi_{\hat{\tau}} \ \equiv_{\tau^\circ} \ \Sigma_{\tilde{\tau}}}{\Psi_{(\tau^{F'}\rho)} \ (\mathbf{\Lambda}_F[\Phi_{\hat{\tau}}]) \ \leq_{\rho^\circ} \ \Gamma_{(\tau'^{F'}\rho')} \ (\mathbf{\Lambda}_{F'}[\Sigma_{\tilde{\tau}}])}$$

$$where\ \mathbf{\Lambda}_{\{+\}} = id_{\mathbf{ELTerms}}{}^{6}, \mathbf{\Lambda}_{\{-\}} = Flip, \mathbf{\Lambda}_F = Anull\ for\ F \notin \{\{+\},\{-\}\}$$

$$\textbf{(RMOD)}\frac{\emptyset}{\Sigma_{(\tau^R\tau')}\ (\Pi_{\hat{\tau}})\ \leq_{\tau^\circ}\ \Pi_{\hat{\tau}}} \quad \textbf{(Ab)}\frac{\Psi_\rho(x_\tau^{\Pi})\ \leq_{\rho^\circ}\ \Phi_{\rho'}(x_{\tau'}^{\Pi'})}{(\lambda x_\tau.\Psi_\rho)_{(\tau^{Pol2Feat(\Pi)}\rho)}\ \leq_{(\tau\rho)^\circ}\ (\lambda x_{\tau'}.\Phi_\rho)_{(\tau'^{Pol2Feat(\Pi')}\rho)}}$$

$$\lambda x.\Psi, \lambda x.\Phi \in \mathbf{ELTerms}$$

$$\textbf{(C}_1\textbf{)}\frac{\emptyset}{(\Sigma_{(\tau^C(\tau\tau))}(Anull(\Phi_{\tau'})))(Anull(\Gamma_{\hat{\tau}}))\ \leq_{\tau^\circ}\ \Omega} \quad \textbf{(C}_2\textbf{)}\frac{\Psi_{\tilde{\tau}}\ \leq_{\tau^\circ}\ \Gamma_{\tau'}\quad \Psi_{\tilde{\tau}}\ \leq_{\tau^\circ}\ \Phi_{\hat{\tau}}}{\Psi_{\tilde{\tau}}\ \leq_{\tau^\circ}\ (\Sigma_{(\tau^C(\tau\tau))}(Anull(\Phi_{\hat{\tau}})))(Anull(\Gamma_{\tau'}))}$$

$$\Omega = \Gamma_{\hat{\tau}}\ \text{or}\ \Omega = \Phi_{\tau'}$$

$$\textbf{(D}_1\textbf{)}\frac{\emptyset}{\Omega\ \leq_{\tau^\circ}\ (\Sigma_{(\tau^D(\tau\tau))}(Anull(\Phi_{\tau'})))(Anull(\Gamma_{\hat{\tau}}))} \quad \textbf{(D}_2\textbf{)}\frac{\psi_{\tau'}\ \leq_{\tau^\circ}\ \Psi_{\hat{\tau}}\quad \Phi_{\tilde{\tau}}\ \leq_{\tau^\circ}\ \Psi_{\hat{\tau}}}{(\Sigma_{(\tau^D(\tau\tau))}(Anull(\Phi_{\tilde{\tau}})))(Anull(\Gamma_\tau))\ \leq_{\tau^\circ}\ \Psi_{\hat{\tau}}}$$

$$\Omega = \Gamma_{\tau'}\ \text{or}\ \Omega = \Phi_{\tilde{\tau}}$$

Let us now demonstrate the way $\mathcal{DDL}$-OC works. Consider the following order-statement between extended terms[7]:

$$(1)\ [\lambda x.John(does(x^{\oplus}))]and(\lambda y.Mary(doesn't(y^{\ominus})))(move)$$

$$\leq \lambda y.Mary(doesn't(y^{\ominus}))(walk)$$

First let us note that the extended term $\lambda y.Mary(doesn't(y^{\ominus}))$ represents a valid $\mathcal{DDL}$ derivation, where its type is dynamically marked for downward monotonicity:

$$\dfrac{\dfrac{(s/((s\backslash np))) : Mary_{(et)^{+}t} \qquad \dfrac{((s\backslash np)/(s\backslash np)) : doesn't_{(et)^{-}(et)} \quad (s\backslash np) : y^{\oplus}}{(s\backslash np) : doesn't(y^{\ominus})}\,(/E_{\ominus})}{s : Mary(doesn't(y^{\ominus}))}\,(/E_{\oplus})}{s/(s\backslash np) : (\lambda y.Mary(doesn't(y^{\ominus})))_{(et)^{-}(et)}}\,(/I)$$

In Figure 8 we present the $\mathcal{DDL}$-OC derivation of (1). Note that by discarding all polarity markings from it, we do not necessarily get a valid $\mathcal{L}$-OC derivation. This is because the type of $\lambda y.Mary(doesn't(y))$ cannot be marked for downward monotonicity in $\mathcal{L}$, and so the MON rule cannot be applied. Nevertheless, we can still show that we can simulate in $\mathcal{L}$-OC any derivation of $\mathcal{DDL}$-OC . More precisely, for any order-statement $\Psi \leq \Phi$ derivable in $\mathcal{DDL}$-OC (where Ψ, Φ are extended linear terms), $\Pi strip(\Psi)\ \leq\ \Pi strip(\Phi)$ is derivable in $\mathcal{L}$-OC (where $\Pi strip(\Psi), \Pi strip(\Phi)$ are the linear terms obtained from Ψ, Φ respectively by deleting their polarity markings). From this we conclude that basing the Order

[6]By $id_{\mathbf{ELTerms}}$ we mean the identity function $id\ :\ \mathbf{ELTerms}\ \rightarrow\ \mathbf{ELTerms}$, such that $id(\Psi) = \Psi$ for every ELTerm Ψ.

[7]Note that we specify here the polarity marking of the bound variables.

Calculus on $\mathcal{DDL}$ instead of $\mathcal{L}$ does not increase its expressive power. Note, however, that this negative result only holds for the current semantic features. In general, there need not exist such a bypass, and therefore dynamic marking is a general solution which should be further investigated.
The following lemma formalizes the relation between $\mathcal{DDL}$-OC and $\mathcal{L}$-OC discussed above:

Lemma 7.7 $\vdash_{\mathcal{DDL}-OC} \Psi \leq \Phi \ \Rightarrow \vdash_{\mathcal{L}-OC} \Pi strip(\Psi) \leq \Pi strip(\Phi)$.

The full proof of the lemma is deferred to appendix A. Now we exemplify the lemma by returning to the $\mathcal{DDL}$-OC derivation in Figure 8. According to the above lemma, we should be able to show the following:

$$\vdash_{\mathcal{L}-OC} [\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move)$$

$$\leq \lambda y.Mary(doesn't(y))(walk)$$

It was already shown in Figure 7 that:

$$\vdash_{\mathcal{L}-OC} [\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move)$$

$$\leq \lambda y.Mary(doesn't(walk))$$

Using the (β) axiom:

$$\lambda y.Mary(doesn't(walk)) \equiv \lambda y.Mary(doesn't(y))(walk)$$

and the TRANS rule, we can indeed construct the desired $\mathcal{L}$-OC derivation.

8 Conclusions

In this paper we have proposed a Natural Logic inference system that is based on $\mathcal{L}$ and transcends the AB-based system of [3]. Basing the system on $\mathcal{L}$ brought about a complication – non-normalized proof terms, with which we dealt by augmenting the system with normalization, or, alternatively, using dynamic monotonicity marking. We have shown that this allows to derive new kinds of inferences involving sentences with extraction, pied piping and non-logical axioms with complex expressions. Further work is needed to allow inferences with more than one sentential premise. It is clear, however, that $\mathcal{L}$ is also not the optimal categorial formalism to underly a Natural Logic inference system, due to its own limitations, mainly overgeneration and incapability of dealing with non-peripheral extraction. Therefore, we view the proposed inference system only as an intermediate step towards a more complex one, to be finally based on some decidable fragment of type-logical grammar. Much work still has to be done in this direction. Further research should also enlarge the variety of semantic properties used for natural language inference, beyond the ones employed in current work on Natural Logic. These reservations notwithstanding, we believe that the present work has shown some advances in extending the Natural Logic paradigm to a more substantial system of reasoning in natural language.

$$\dfrac{\boxed{\dfrac{\dfrac{\emptyset}{\begin{array}{c}[\lambda x.John(does(x^{\oplus}))]and(\lambda y.Mary(doesn't(y^{\ominus})))\\ \leq\ \lambda y.Mary(doesn't(y^{\ominus}))\end{array}}\,\text{C1}}{\begin{array}{c}[\lambda x.John(does(x^{\oplus}))]and(\lambda y.Mary(doesn't(y^{\ominus})))(move)\\ \leq \lambda y.Mary(doesn't(y^{\ominus}))_{(et)^{-}(et)}(move)\end{array}}\,\text{FR}} \quad \dfrac{(a_2)\ walk \leq move}{\begin{array}{c}\lambda y.Mary(doesn't(y^{\ominus}))_{(et)^{-}(et)}(move)\\ \leq \lambda y.Mary(doesn't(y^{\ominus}))_{(et)^{-}(et)}(walk)\end{array}}\,\text{MON}}{\begin{array}{c}[\lambda x.John(does(x^{\oplus}))]and(\lambda y.Mary(doesn't(y^{\ominus})))(move)\\ \leq \lambda y.Mary(doesn't^{(}y^{\ominus}))(walk)\end{array}}\,\text{TRANS}$$

Figure 8: A $\mathcal{DDL}$-OC derivation

A Proof of Lemma 7.7

First we prove the following lemma that we will use in the sequel:

Lemma A.1 *Let* $\Psi \in$ *ELTerms s.t.* $x \in Free(\Pi strip(\Psi))$ *and* x *is marked*[8] *for positive (negative) polarity in* Ψ*. Then* $\vdash_{\mathcal{L}-OC} \gamma \leq_\sigma \delta$ *(*$\vdash_{\mathcal{L}-OC} \delta \leq_\sigma \gamma$*) implies* $\vdash_{\mathcal{L}-OC} \Pi strip(\Psi)[x/\gamma] \leq_\tau \Pi strip(\Psi)[x/\delta]$.

Proof: by induction on the complexity of Ψ. We prove only for the positive polarity case; the proof for negative polarity is symmetric.

- $\Psi = x^{\oplus}$. Then $\Pi strip(\Psi) = x$ and $\vdash_{\mathcal{L}-OC} \gamma \leq_\sigma \delta$ implies $\vdash_{\mathcal{L}-OC} x[x/\gamma] \leq_\sigma x[x/\delta]$.
- $\Psi = \Theta_{(\zeta^F \rho)}(\Lambda_\zeta)$. Then (since the terms are linear) only one of the following holds: $x \in Free(\Pi strip(\Theta))$ or $x \in Free(\Pi strip(\Lambda))$.
    - Suppose $x \in Free(\Pi strip(\Theta))$. If x is marked for positive polarity in Ψ, then x is marked for positive polarity also in Θ. By the induction hypothesis, $\vdash_{\mathcal{L}-OC} \gamma \leq_\sigma \delta$ implies $\vdash_{\mathcal{L}-OC} \Pi strip(\Theta)[x/\gamma] \leq_{(\zeta\rho)} \Pi strip(\Theta)[x/\delta]$. By applying the FR rule, we can prove

    $$\vdash_{\mathcal{L}-OC} \underbrace{(\Pi strip(\Theta)[x/\gamma])(\Pi strip(\Lambda))}_{\Pi strip(\Theta(\Lambda))[x/\gamma]} \leq_\rho \underbrace{(\Pi strip(\Theta)[x/\delta])(\Pi strip(\Lambda))}_{\Pi strip(\Theta(\Lambda))[x/\delta]}$$

    - Suppose $x \in Free(\Pi strip(\Lambda))$. If x is marked for positive polarity in Ψ, then (i) either the type of Θ is marked for upward monotonicity and x is marked for positive polarity in Λ or (ii) the type of Θ is marked for downward monotonicity and x is marked for negative polarity in Λ. Suppose that (i) holds. By the induction hypothesis, $\vdash_{\mathcal{L}-OC} \gamma \leq_\sigma \delta$ implies $\vdash_{\mathcal{L}-OC} \Pi strip(\Lambda)[x/\gamma] \leq_\zeta \Pi strip(\Lambda)[x/\delta]$. By applying MON+ rule, we can prove

    $$\vdash_{\mathcal{L}-OC} \underbrace{(\Pi strip(\Theta)_{(\zeta^+\rho)})(\Pi strip(\Lambda)_\zeta[x/\gamma])}_{\Pi strip(\Theta(\Lambda))[x/\gamma]} \leq_\rho \underbrace{(\Pi strip(\Theta))_{(\zeta^+\rho)}(\Pi strip(\Lambda)[x/\delta])}_{\Pi strip(\Theta(\Lambda))[x/\delta]}$$

      The proof for (ii) is symmetric.
- $\Psi = \lambda y_\zeta.\Phi$, where $y \neq x$. Since x is marked for positive polarity in Ψ, then it is marked for positive polarity in Φ. By the induction hypothesis, $\vdash_{\mathcal{L}-OC} \gamma \leq_\sigma \delta$ implies $\vdash_{\mathcal{L}-OC} \Pi strip(\Phi)[x/\gamma] \leq_\rho \Pi strip(\Phi)[x/\delta]$.

[8] Recall that since we use extended linear terms, every variable occurs at most once, and so we can speak of the marking of x and not of its specific occurrence.

By applying the Ab rule, we can prove $\vdash_{\mathcal{L}-OC}\ \lambda y.(\Pi strip(\Phi)[x/\gamma]) = \Pi strip(\lambda y.\Phi)[x/\gamma] \leq_\rho \lambda y.(\Pi strip(\Phi)[x/\delta]) = \Pi strip(\lambda y.\Phi)[x/\delta]$, and so we have:

$$\vdash_{\mathcal{L}-OC} \Pi strip(\lambda y.\Phi)[x/\gamma] \leq_{(\zeta\rho)} \Pi strip(\lambda y.\Phi)[x/\delta]$$

□

Now we prove lemma 7.7, which states the following:

$$\vdash_{\mathcal{DDL}-OC} \Psi \leq \Phi\ \Rightarrow \vdash_{\mathcal{L}-OC} \Pi strip(\Psi) \leq \Pi strip(\Phi)$$

Suppose that there is a proof Ω_1 of $\Psi \leq \Phi$ in $\mathcal{DDL}$-OC. We construct a $\mathcal{L}$-OC proof Ω_2 of $\Pi strip(\Psi) \leq_\tau \Pi strip(\Phi)$. First we delete all polarity markings from the variables of the extended terms in Ω_1. Then we delete all monotonicity markings of abstraction terms in Ω_1.
If the proof contains no MON rule applications based on dynamic marking of types of abstraction terms, then it is easy to see that the obtained derivation is a valid $\mathcal{L}$-OC proof of $\Pi strip(\Psi) \leq_\tau \Pi strip(\Phi)$.
Otherwise, the resulting invalid applications of the MON rule of the form:

$$\dfrac{\Omega\ \boxed{\dfrac{\bigtriangledown}{\Pi strip(\Theta) \leq_\tau \Pi strip(\Delta)}}}{(\lambda x_\tau.\Pi strip(\Psi))_{(\tau\rho)}(\Pi strip(\Theta_\tau)) \leq_\rho (\lambda x_\tau.\Pi strip(\Psi))_{(\tau\rho)}(\Pi strip(\Delta_\tau))}\ {\scriptstyle\text{MON}} \qquad (1)$$

where the type of $\lambda x_\tau.\Pi strip(\Psi)$ is not marked for monotonicity. We choose the innermost invalid MON instance, that is such that does not have invalid MON instances in Ω. First of all, due to its being the innermost non-valid instance of MON, Ω is a valid $\mathcal{L}$-OC proof of the order statement $\Pi strip(\Theta) \leq \Pi strip(\Delta)$. Secondly, since the type of $\lambda x.\Psi$ is marked for upward monotonicity, x must be marked for positive polarity in Ψ. By lemma A.1:

$$\vdash_{\mathcal{L}-OC} \Pi strip(\Psi)[x/\Pi strip(\Theta)] \leq_{(\tau\rho)} \Pi strip(\Psi)[x/\Pi strip(\Delta)]$$

Thus we can replace (1) by the following valid $\mathcal{L}$-OC proof:

In this way we can systematically remove all invalid instances of MON.

□

$$\dfrac{\dfrac{\emptyset}{\begin{array}{c}\lambda x.\Pi strip(\Psi)(\Pi strip(\Theta))\\ \equiv_{(\tau\rho)} \Pi strip(\Psi)[x/\Pi strip(\Theta)]\end{array}}\,\beta \qquad \boxed{\dfrac{\bigtriangledown}{\begin{array}{c}\Pi strip(\Psi)[x/\Pi strip(\Theta)]\\ \leq_{(\tau\rho)} \Pi strip(\Psi)[x/\Pi strip(\Delta)]\end{array}}} \qquad \dfrac{\emptyset}{\begin{array}{c}\Pi strip(\Psi)[x/\Pi strip(\Delta)]\\ \equiv_{(\tau\rho)} \lambda x.\Pi strip(\Psi)_{(\tau\rho)}(\Pi strip(\Delta))\end{array}}\,\beta}{(\lambda x.\Pi strip(\Psi))(\Pi strip(\Theta)) \leq (\lambda x.\Pi strip(\Psi))(\Pi strip(\Delta))}\,\text{TRANS}$$

Example: Let us demonstrate the method presented above using the $\mathcal{DDL}$-OC derivation from Figure 8. First we delete all the polarity markings and monotonicity markings of types of abstraction terms and get the following “derivation”:

$$\boxed{\dfrac{\boxed{\dfrac{\dfrac{\emptyset}{\begin{array}{c}[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))\\ \leq\ \lambda y.Mary(doesn't(y))\end{array}}\,\text{C1}}{\begin{array}{c}[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move)\\ \leq (\lambda y.Mary(doesn't(y)))(move)\end{array}}\,\text{FR}} \quad \dfrac{walk \leq move}{\begin{array}{c}\lambda y.Mary(doesn't(y))_{(et)(et)}(move)\\ \leq \lambda y.Mary(doesn't(y))_{(et)(et)}(walk)\end{array}}\,\underline{\textbf{MON}}}{[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move) \leq \lambda y.Mary(doesn't(y))(walk)}\,\text{TRANS}}$$

Of course, the underlined application of MON is not valid, since the type of $\lambda y.Mary(doesn't(y))$ is no longer marked for downward monotonicity.
Since the type of abstraction term $\lambda y.Mary(doesn't(y))$ was dynamically marked for ‘-’, it means that y was marked for ‘$\ominus$’ in $Mary(doesn't(y^{\ominus}))$. By lemma A.1 and using the non-logical axiom $walk \leq move$:

$$\vdash_{\mathcal{L}-OC} Mary(doesn't(move)) \leq Mary(doesn't(walk))$$

(see the right side of Figure 7), and using β-normalization as described above, we obtain the $\mathcal{L}$-OC derivation in Figure 9.

$$\dfrac{\boxed{\dfrac{\dfrac{\emptyset}{\begin{array}{c}[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))\\ \leq\ \lambda y.Mary(doesn't(y))\end{array}}\ \text{C1}}{\begin{array}{c}[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move)\\ \leq (\lambda y.Mary(doesn't(y)))(move)\end{array}}\ \text{FR}} \quad \begin{array}{c}(\lambda y.Mary(doesn't(y)))(move)\\ \equiv Mary(doesn't(move))\end{array}\ (\beta) \quad \boxed{\dfrac{\dfrac{\dfrac{\emptyset}{walk_{(et)} \leq move_{(et)}}\ (a_2)}{\begin{array}{c}doesn't_{(et)^-(et)}(move)\\ \leq doesn't(walk)\end{array}}\ \text{MON-}}{\begin{array}{c}Mary_{((et)^+t)}(doesn't(move))\\ \leq Mary_{((et)^+t)}(doesn't(walk))\end{array}}\ \text{MON+}} \quad \begin{array}{c}Mary(doesn't(walk)) \equiv\\ (\lambda y.Mary(doesn't(y)))(walk)\end{array}\ (\beta)}{[\lambda x.John(does(x))]and(\lambda y.Mary(doesn't(y)))(move) \leq \lambda y.Mary(doesn't(y))(walk)}\ \text{TRANS}$$

Figure 9: Example of applying lemma 7.7: $\mathcal{L}$-OC derivation

References

[1] R. Bernardi, *Reasoning with Polarity in Categorial Type Grammar*, Ph.D. thesis, Utrecht University, 2002.

[2] L. Faltz and E. L. Keenan, *Boolean Semantics for Natural Language*, Reidel, 1985.

[3] Y. Fyodorov, Y. Winter, and N. Francez, *Order-Based Inference in Natural Logic*, Logic Journal of the IGPL (2003).

[4] L.T.F Gamut, *Logic, Language and Meaning*, vol. 2, The University of Chicago Press, Chicago, 1991.

[5] L. Henkin., *Completeness in the theory of types*, Journal of Symbolic Logic (1950), 81–91.

[6] Y. Bar Hillel, *Language and Information*, Addison Wesley, Reading MA, 1964.

[7] J. Lambek, *The mathematics of sentence structure*, vol. 64, Amer. Math. monthly, Chicago, 1991.

[8] M. Moortgat, *Categorial Type Logics, Handbook for Logic and Language*, Johan van Benthem and Alice ter Muelen eds., Elsevier/MIT Press, 1997.

[9] I. Pratt-Hartman, *Fragments of Language Quantification*, Journal of Logic, Language and Information **13** (2004), 207–223.

[10] V. Sánchez, *Studies on Natural Logic and Categorial Grammar*, Ph.D. thesis, University of Amsterdam, 1991.

[11] J. van Benthem, *Meaning: interpretation and inference*, Synthese **73** (1987), 451–470.

[12] H. Wansing, *The logic of information structures*, Springer Lecture Notes, Springer-Verlag, Berlin, 1993.

[13] A. Zamansky, *A 'Natural Logic' inference system based on the Lambek calculus*, Master's thesis, Technion, Haifa, 2004.

[14] A. Zamansky, Y. Winter, and N. Francez, *Order-Based Inference using the Lambek calculus*, Proceedings of the 7-th conference on Formal Grammar, University of Trento (2002).

[15] ______, *A 'Natural Logic' inference system using normalization*, Proceedings of the 4-th workshop on Inference in Computational Semantics, LORIA, Nancy, P. Blackburn and J. Bos eds. (2003).