Order-Based Inference using the Lambek Calculus

1 Abstract

This paper develops a 'Natural Logic' inference system which is based on the Lambek calculus and works directly on natural language syntactic representations, with no intermediate translation to logical formulae. We extend the system of Fyodorov et al., which is based on Ajdukiewicz-Bar Hillel calculus and overcome various syntactic limitations of their system. The proposed system is able to derive new kinds of inferences, including derivations of statements with "gapped" sentences.

2 Introduction

The goal of this paper is to develop an inference system based on the (associative) Lambek calculus ($\mathcal{L}$) using the 'Natural Logic' paradigm of [Van Benthem(1987)] and [Sánchez(1991)]. We define an Order Calculus (OC) for deriving inferences in natural language. The OC manipulates *order statements* between $\mathcal{L}$ derivation trees of natural language expressions. The order statements reflect semantic order relations between elements of partially ordered domains. Hence we can derive inferences directly from $\mathcal{L}$ derivation trees, without intermediate logical levels of representation such as first order logic or the lambda calculus.
We hereby extend an existing system of Fyodorov et al. (henceforth FWF) which is based on the Ajdukiewicz-Bar Hillel (AB) calculus. The system of FWF allows a rather straightforward derivation of inferences with monotone and non-monotone quantifiers and cross-categorial conjunctions and disjunctions. However, despite its value for demonstrating a novel technique of inference in natural language, FWF's system fails to derive many kinds of inferences due to the syntactic limitations of the AB calculus. It is clear that in order to define a more realistic system that can cope with a whole variety of inferences derivable in natural language, a more complex grammatical formalism is needed.

We develop an OC that is based on $\mathcal{L}$ as an intermediate step towards a general system supporting various kinds of inferences in natural language. By extending the AB-based FWF system, we overcome several of its limitations and enable it to derive new kinds of inferences, including ones that involve "gapped" sentences.
In a recent work, [Bernardi(2002)] follows [Sánchez(1991)] and introduces a system for monotonicity reasoning that is based on a more complex categorial type logic than Sanchez' work. However, Bernardi concentrates on monotonicity reasoning in order to capture the *syntactic* distribution of negative polarity items, without fully accounting for monotonicity inferences in natural language. By contrast, the $\mathcal{L}$-based OC that we develop in this paper follows FWF in concentrating on semantic inference relations, without any attempt to treat negative polarity items. Interestingly, within the $\mathcal{L}$-based OC (unlike the AB-based OC of FWF), we will show that it is useful to incorporate Sanchez'/Bernardi's technique of polarity marking into the inference engine, independently of its use for treating negative polarity items, for computing the monotonicity marking of functions that are dynamically created via application of Abstraction rules.

3 Semantic order relations

We assume that natural language expressions are associated with *types*. The *primitive types* are a finite set in which we identify a subset of *primitive partial order (PO) types*. The sets of types and PO types are formally defined as follows:

Definition 3.1 ***(Types)*** *Let T^0 be some finite set of primitive types. The set of types is the smallest set T so that $T^0 \subseteq T$ and if $\tau \in T$ and $\sigma \in T$ then also $(\tau\sigma) \in T$.*

Standardly, types e (for entities) and t (for 'truth values') are among the primitive types.

Definition 3.2 ***(PO types)*** *Let ${T^0}_{po} \in T$ be some finite set of primitive types. The set of PO types is the smallest set $T_{po} \subseteq T$ so that ${T^0}_{po} \subseteq T_{po}$ and if $\tau \in T_{po}$ and $\sigma \in T_{po}$ then also $(\tau\sigma) \in T_{po}$.*

Standardly, type t is among the PO types.

Definition 3.3 ***(Domains)*** *For each primitive type $\tau \in T^0$, let D_τ be a non-empty domain. Let these primitive domains be mutually disjoint. For* *each non-primitive type $(\tau\sigma) \in T \backslash T^0$, let $D_{(\tau\sigma)}$ be the set of functions from D_τ to D_σ.*

The domain D_σ of any primitive PO type σ is endowed with a given partial order relation $\leq_\sigma$. The partial order for complex PO types is defined as follows:

Definition 3.4 ***(Pointwise partial order)*** *If σ is a PO type with partial order $\leq_\sigma$ over the domain D_σ, then the partial order $\leq_{(\tau\sigma)}$ over the domain $D_{(\tau\sigma)}$ is defined pointwise: $d_1 \leq_{(\tau\sigma)} d_2$ iff for every $d' \in D_\tau$: $d_1(d') \leq_\sigma d_2(d')$.*

4 Syntactic categories

The set of (undecorated) categories in $\mathcal{L}$ is standardly defined as in applicative categorial grammar (see [Moortgat(1997)]) in the slash format 'result on left' of [Steedman(2000)]. For abbreviation, we use the pattern $A|B$ to match the patterns $A\backslash B$ and A/B. We assume that the set of primitive categories includes at least two categories, s (for sentences) and $\overline{s}$ (a category of a special word w_T the use of which will be explained later on.) Each category is assigned a semantic type by a mapping **type** from categories to types in the following way: for a primitive category A, $\textbf{type}(A)$ is given; for complex categories $A\backslash B$ and A/B: $\textbf{type}(A\backslash B) = \textbf{type}(A/B) = (\textbf{type}(A)\ \textbf{type}(B))$: the functional type with domain $\textbf{type}(A)$ and range $\textbf{type}(B)$.
In section 6 we define the Dynamically Decorated $\mathcal{L}$ calculus ($\mathcal{DDL}$) which is a modification of $\mathcal{L}$ that is defined over *semantically decorated* categories. A decorated category encodes semantic information about the natural language expressions to which it is assigned. It is an expression of form $A/^F B$ or $A\backslash^F B$, where A and B are either primitive categories or decorated categories, and F is either null or a semantic feature from a given set of features including: $+/-$ (for upward/downward monotonicity), R (restrictive modification) and C/D (conjunction/disjunction). We will use the patterns $A/^* B$, $A\backslash^* B$ and $A|^* B$, which match $A/^F B$, $A\backslash^F B$ and $A|^F B$ (resp.), for any value of the feature F. Each decorated category is assigned a semantic type like the corresponding undecorated category: $\textbf{type}(A|^* B) = (\textbf{type}(A)\ \textbf{type}(B))$.
One difference of $\mathcal{DDL}$ from the standard $\mathcal{L}$ calculus is that the argument category B in a functor category (A/B) or $(A\backslash B)$ may be decorated differently from the category *B'* that combines with the functor (using the Application rules). Therefore, in Application rules of $\mathcal{DDL}$ we require B

and B' to have the same syntactic category, but not necessarily the same semantic marking. We call such categories *formally equivalent.*

Definition 4.1 ***(Formally Equivalent Categories):*** *For any two decorated categories A,B we say that A is formally equivalent to B and denote it by* $A \equiv_f B$ *iff (i)A and B are primitive and* $A = B$ *or* $\{A, B\} = \{s, \overline{s}\}$*, or (ii)*$A = A_1|^* A_2,\ B = B_1|^* B_2,\ A_1 \equiv_f B_1,\ A_2 \equiv_f B_2$

Another difference between $\mathcal{DDL}$ and $\mathcal{L}$ is that $\mathcal{DDL}$ marks the polarity of its assumption categories, as will be explained in section 6.

5 Definition of $\mathcal{DDL}$

The $\mathcal{DDL}$ system that we define below uses the *polarity* of assumptions to decorate the monotonicity properties of categories that are created by abstracting over these assumptions. The polarity of an assumption of category A in the derivation tree T is a string $P \in \{+, -, null\}$ that is associated with it ($'+'$ $/'-'$ for positive/negative polarity, $'null'$ for undefined polarity). The polarity is updated in the Application rules of $\mathcal{DDL}$. *Flipping* the polarity of A means that if $P = +(-)$, we set $P = -(+)$. We will denote an assumption of category A in a derivation tree T with polarity P associated with it by A^P. Polarity of an assumption of category A in a derivation tree T is formally defined in section 6.
Each syntactic category in $\mathcal{DDL}$ is assigned a directed lambda term that encodes the derivation of this category using the Curry-Howard isomorphism (see [Wansing(1993)] for similar use of directed lambda terms).

Definition 5.1 *($\mathcal{DDL}$) Let* $\Gamma, \Gamma_1, \Gamma_2$ *range over finite non-empty sequences of pairs* $A_i : \psi_i$*, where* A_i *is a decorated category and* ψ_i *a directed* λ*-term. The notation* $\Gamma \triangleright A : \psi$ *means that the sequence* Γ *is reducible to* $A : \psi$*. The rules of the* $\mathcal{DDL}$ *calculus are as follows (the boxes after each rule contain the required update of polarity):*

$$(axiom) A^+ : x \triangleright A^+ : x$$

$\boxed{\textit{Positive polarity introduction.}}$

Application rules with polarity updating:

$$(/E)\frac{\Gamma_1 \triangleright (A/^F B) : \psi_1 \quad \Gamma_2 \triangleright B' : \psi_2}{\Gamma_1\Gamma_2 \triangleright A : \psi_1(\psi_2)}, \quad (\backslash E)\frac{\Gamma_2 \triangleright B' : \psi_2 \quad \Gamma_1 \triangleright (A\backslash^F B) : \psi_1}{\Gamma_2\Gamma_1 \triangleright A : [\psi_2]\psi_1}$$

$$for\ B \equiv_f B'$$

$\boxed{\textit{If } F{=}- \textit{ then } \forall(A^P : x) \in \Gamma_2 \textit{ flip } P\ .}$

$\boxed{\textit{If } F{=}null \textit{ or } F{=}R \textit{ then } \forall(A^P : x) \in \Gamma_2 \textit{ set } P = null}$

$\boxed{\textit{Otherwise no update.}}$

Abstraction rules with dynamic decoration:

$$(/I)\frac{\Gamma_1,\ B^P : x \rhd A : \psi_1}{\Gamma_1 \rhd (A/^P B) : \overrightarrow{\lambda} x.\psi_1} \quad (\backslash I)\frac{B^P : x,\ \Gamma_1 \rhd A : \psi_1}{\Gamma_1 \rhd (A\backslash^P B) : \overleftarrow{\lambda} x.\psi_1} \quad \textit{for } \Gamma_1 \textit{ not empty}$$

$\boxed{\textit{No update.}}$

In the Application rule, applying a function that is marked as downward monotone entails flipping the polarity in all the assumptions of its argument. When the function is unmarked or marked as restrictive, we eliminate all the polarity markings, since it is assumed that the function is non-monotone. When the function is marked as upward monotone, no change of polarity marking is required. Functions that are marked as conjunctive or disjunctive are special cases of upward monotone functions, hence they do not require any change of marking either.
We will abbreviate the derivation trees as terms as follows: (i) A lexical item **w** of category A is denoted by $\mathbf{w}_{type(A)}$.(ii) An assumption of category A^P is denoted by $x_{type(A)}$, where x is the free variable associated with A.(iii) The semantic types of lambda terms inherit the semantic marking of the categories. For example: $type((s/^+(np\backslash s)) = ((et)^+t)$. (iv) /-application (\-application) between terms $\varphi_{type(A/^*B)}$, $\psi_{type(B')}$ is denoted by $(\varphi(\psi))_{type(A)}$ $(([\psi]\varphi)_{type(A)})$.(v) /-abstraction (\-abstraction) in a term $\varphi_{type(B)}$ with a free variable $\mathbf{x}_{type(A)}$ is denoted by $(\overrightarrow{\lambda} x.\varphi\ (\overleftarrow{\lambda} x.\varphi)$.
We also use the following abbreviations: (i) The pattern $\overline{\lambda} x.\varphi$ for $\overrightarrow{\lambda} x.\varphi$ or $\overleftarrow{\lambda} x.\varphi$.(ii) The pattern $\psi\langle\phi\rangle$ for $\psi(\phi)$ or $[\phi]\psi$.(iii) A term φ of type $type(A)$ is denoted $\varphi_{type(A)}$.(iv) A term $\varphi_{type(A)}$ a rightmost (leftmost) subterm x of type $type(B)$ is denoted by $\varphi_{type(A)}{}^{\overrightarrow{x}_{type(B)}}$ $(\varphi_{type(A)}{}^{\overleftarrow{x}_{type(B)}})$. We denote $\varphi_{type(A)}{}^{\overline{x}_{type(B)}}$ when the peripherality direction is immaterial.

6 Dynamic decoration

In FWF's system the semantic decoration of categories originates from the lexicon only. In $\mathcal{DDL}$, however, complex categories created by Abstraction rules may be dynamically decorated (by $'+'$ $/'-'$ monotonicity marking).
Following [Sánchez(1991)] and [Bernardi(2002)], we define the *polarity* of occurrence of lambda term ϕ in lambda term ψ:

Definition 6.1 ***Polarity of occurrences:***[1] *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 $\psi = \alpha\langle\beta\rangle$ 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 β and α denotes an upward monotone function.*
   - *ϕ is negative (positive) in ψ if ϕ is positive (negative) in β and α denotes a downward monotone function.*
3. *If $\psi = \overline{\lambda}x.\mu$ then ϕ is positive (negative) in ψ if ϕ is positive (negative) in μ and $x \notin FV(\mu)$.*

The $\mathcal{DDL}$ that is presented in section 5 updates the polarity of all assumptions in a derivation tree according to the above definition. When an Abstraction rule is applied to an assumption x with polarity P in derivation tree represented by ϕ, the $('+'/'-')$ decoration of the dynamically created category of $\lambda x.\phi$ is determined by the polarity of x in ϕ, according to the following fact due to van Benthem:

Fact 6.1 *If x is positive (negative) in ϕ then $\lambda x.\phi$ denotes an upward (downward) monotone function.*

7 $\mathcal{L}$-based OC

The OC is a mechanism for proving order statements $\varphi_{type(A)} \leq \psi_{type(B)}$ between terms that represent derivation trees, whose denotations are semantically comparable. In such case we say that the categories A and B are *semantically equivalent*.

Definition 7.1 *For any two decorated categories A,B we say that A is semantically equivalent to B and denote it by $A \equiv_s B$ iff $type(A) = type(B)$.*

Note that as in FWF's system, lexical markings are deduced from standard denotations. However, there is no appeal to denotations when computing

[1] We assume that all lambda terms are linear, that is, every abstraction binds exactly one (peripheral) occurrence of a free variable in a given term.

entailments (via OC order-statements). This is in contrast with Natural Logic of [Sánchez(1991)].
When the order statements $\varphi \leq \psi$ and $\psi \leq \varphi$ are both provable, we denote this by $\varphi \equiv \psi$ (note the difference from formal and semantic equivalence between *categories*). The inference rules in the OC derive such order statements between proof terms.

Definition 7.2 ***(OC:)***

$$\textbf{(Refl)}\frac{\emptyset}{\alpha \leq \alpha} \quad \textbf{(Trans)}\frac{\alpha \leq \beta \quad \beta \leq \gamma}{\alpha \leq \gamma}$$

$$\textbf{(Mon+)}\frac{\alpha_{type(B')} \ \leq \ \beta_{type(B'')}}{\gamma_{type(A|^{+}B)}\langle\alpha_{type(B')}\rangle \ \leq \ \gamma_{type(A|^{+}B)} \ \langle\beta_{type(B'')}\rangle} \ ; \ B \equiv_f B' \equiv_f B''$$

$$\textbf{(FR)}\frac{\alpha_{type(A|^{*}B)} \ \leq \ \beta_{type(C|^{*}D)} \quad \gamma_{type(B')} \ \equiv \ \delta_{type(D')}}{\alpha_{type(A|^{*}B)} \ \langle\gamma_{type(B')}\rangle \ \leq \ \beta_{type(C|^{*}D)} \ \langle\delta_{type(D')}\rangle}$$
$$B \equiv_f B', \ D \equiv_f D', \ A \equiv_s C, \ B \equiv_s D$$

$$\textbf{(Rmod)}\frac{\emptyset}{\alpha_{type(A'|^{R}A)} \ \langle\beta_{type(A'')}\rangle \ \leq \ \beta_{type(A'')}} \ ; \quad A \equiv_s A' \equiv_s A''$$

$$\textbf{(C}_1\textbf{)}\frac{\emptyset}{(\alpha_{type((A|B)|^{C}C)} \ \langle\gamma_{type(C)}\rangle) \ \langle\beta_{type(B)}\rangle \ \leq \ \Psi}$$
$$\Psi = \beta_{type(B)} \ or \ \Psi = \gamma_{type(C)}$$

$$\textbf{(C}_2\textbf{)}\frac{\alpha'_{type(A')} \ \leq \ \beta_{type(B)} \quad \alpha'_{type(A')} \ \leq \ \gamma_{type(C)}}{\alpha'_{type(A')} \ \leq \ (\alpha_{type((A|B)|^{C}C)} \ \langle\gamma_{type(C)}\rangle) \ \langle\beta_{type(B)}\rangle}$$

$$\textbf{(D}_1\textbf{)}\frac{\emptyset}{\Psi \ \leq \ (\alpha_{type((A|B)|^{D}C)} \ \langle\gamma_{type(C)}\rangle) \ \langle\beta_{type(B)}\rangle}$$
$$\Psi = \beta_{type(B)} \ or \ \Psi = \gamma_{type(C)}$$

$$\textbf{(D}_2\textbf{)}\frac{\beta_{type(B)} \ \leq \ \alpha'_{type(A')} \quad \gamma_{type(C)} \ \leq \ \alpha'_{type(A')}}{(\alpha_{type((A|B)|^{D}C)} \ \langle\gamma_{type(C)}\rangle) \ \langle\beta_{type(B)}\rangle \ \leq \ \alpha'_{type(A')}}$$

$$\textbf{(Ab)}\frac{\alpha_{type(A)}{}^{\overline{x}_{type(B)}} \ \leq \ \beta_{type(A')}{}^{\overline{x}_{type(B')}}}{\overline{\lambda}x_{type(B)}.\alpha_{type(A)}{}^{\overline{x}_{type(B)}} \ \leq \ \overline{\lambda}x_{type(B')}.\beta_{type(A')}{}^{\overline{x}_{type(B')}}}$$
$$A \equiv_f A', \ B \equiv_f \ B''$$

$$\textbf{(PWC)}\frac{\emptyset}{\begin{array}{c}((\alpha_{type(((A|B)|(A|B)|^{C/D}(A|B))} \ \langle\beta_{type(A|B)}\rangle) \ \langle\gamma_{type(A|B)}\rangle) \ \langle\delta_{type(B)}\rangle \equiv \\ (\alpha_{type((A|^{C/D}A)|A)} \ \langle\beta_{type(A|B)} \ \langle\delta_{type(B)}\rangle\rangle) \ \langle\gamma_{type(A|B)} \ \langle\delta_{type(B)}\rangle\rangle\end{array}}$$

The Reflexivity and Transitivity rules are general properties of the partial order relation. The RMOD rule (restrictive modification) is for deriving order statements such as between *tall student* and *student*, where the adjective *tall* is *restrictive.* [2] The Upward Monotonicity rule (+) derives an order statement between proof terms $\gamma\langle\alpha\rangle$ and $\gamma\langle\beta\rangle$ where γ is of type $type(A|B)$ that is marked for upward monotonicity (by +), given an order statement $\alpha \leq \beta$ between proof terms to which γ is applied. The Downward Monotonicity rule (**Mon-**) is similar. The FR rule handles a situation, where we have an order statement between the function terms α and β, and their arguments γ and γ' are equivalent (both $\gamma \leq \gamma'$ and $\gamma' \leq \gamma$ hold). The C_1 and C_2 rules specify Conjunctive items (e.g. *and*) as *greatest lower bounds* with respect to the partial order relation. Similarly, the D_1 and D_2 rules specify Disjunctive items (e.g. *or*) as *least upper bounds.* The Pointwise Coordination rule presupposes that any $(A|B)|(A|B)|^{C/D}(A|B)$ can also be analyzed as category $(A|^{C/D}A)|A$ respectively, and conversely: any expression of category $(A|^{C/D}A)|A$ is also of category $(A|B)|(A|B)|^{C/D}(A|B)$, where B is a category derivable in $\mathcal{L}$ from a given lexicon. To handle proof terms in $\mathcal{L}$ that involve the Abstraction rules, we add a new rule *(Ab)* to the OC. Given a premise $\varphi_1 \leq \varphi_2$, where both φ_1 and φ_1 represent derivation trees with rightmost (leftmost) free variable x, the order statement $\overline{\lambda}x.\varphi_1 \leq \overline{\lambda}x.\varphi_2$ is derived.
The *semantics* of the OC is naturally defined using standard models for the extensional fragment of Montague's IL (see [Gamut(1991)]) and a pointwise definition for a semantic order relation $\leq$. Any proof term $\varphi_{type(A)}$ is associated with a denotation $[\![\varphi]\!]_{M,g}$ relative to a model M and an assignment g to free variables. This denotation is of type $type(A)$. For constants, standing for lexical entities, we give denotations that satisfy natural requirements according to their semantic decorations. For instance, we require that any model maps the constant **tall**, of category $n/^R n$, to a *restrictive* function f of the appropriate PO type $(\tau\tau)$. That is: for every $d \in D_\tau$: $f(d) \leq_\tau d$.

Definition 7.3 ***(Semantics of Order Statements)*** *Let φ_1, φ_2 be terms of type τ and g an assignment. Then: (i)$M, g \models \varphi_1 \leq_\tau \varphi_2 \quad \Leftrightarrow \quad [\![\varphi_1]\!]_{M,g} \leq_\tau [\![\varphi_2]\!]_{M,g}$*
(ii) $M \models \varphi_1 \leq_\tau \varphi_2 \quad \Leftrightarrow \forall g: \ M, g \models \varphi_1 \leq_\tau \varphi_2$ (In this case, we say that M satisfies $\varphi_1 \leq_\tau \varphi_2$).

[2] In this paper we assume that all modifiers are restrictive. We do not treat intersective modifiers such as blue (e.g. a blue car denotes the intersection of cars with blue things) and non-restrictive modifiers such as fake (e.g. fake diamond is not a diamond).

Relative to these semantic notions, it can be shown that the (Ab) rule of the $\mathcal{L}$-based OC defined above is strongly sound, that is for every model M: if M satisfies the order statement $\alpha \leq \beta$ then it also satisfies $\overline{\lambda}x.\alpha \leq \overline{\lambda}x.\beta$. For the other rules, the soundness proof of FWF can be easilly adapted to their soundness in our formulation.

8 The OC as an inference system for natural language

In this section we illustrate how the OC as defined above can be used for deriving inferences in natural language. We introduce a lexicon (for a toy fragment of English) as well as lexical and extra-logical order statements that will be used as additional assumptions.

8.1 The lexicon

In the lexicon we use a set T^0 of primitive types that includes the types t (truth values) and e (entities). The set of primitive categories with their corresponding types is: $type(s) = t$; $type(\overline{s}) = t$; $type(np) = e$; $type(n) =$ *(et)*.
The categories s and $\overline{s}$ are for sentences, np for noun phrases and n for nouns. The lexicon itself is given below:

Words	**Category**	**Type**
w_T	$\overline{s}$	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,girl	n	(et)
walked, smiled	$s\backslash np$	(et)
touched, kissed, hugged, loved	$(s\backslash np)/np$	$(e(et))$
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	$(n\backslash n)/^{C}(s/np)$	$(et)^{C}((et)(et))$
and	$((s\backslash s)/^{C}s), (s/np)\backslash(s/np)/^{C}(s/np)$	$(t^{C}(tt))$, $(et)^{C}((et)(et))$

The word w_T is assumed to be in the lexicon. We regard natural language assertions as order statements of the form $\mathsf{T} \leq \alpha_t$, where α_t is the derivation of the assertion and T ('top') is a special derivation tree $w_{T\,type(\overline{s})}$, the denotation of which is the value *true*. Thus we use w_T to treat assertions as a special case of order statements, between a derivation tree of the assertion and T.

8.2 Lexical/extra-logical order statements

We postulate the following order statements between natural language expressions: $\mathsf{kissed} \leq \mathsf{touched}$, $\mathsf{walked} \leq \mathsf{moved}$, $\mathsf{student} \leq \mathsf{person}$.
We would also like to extend the ad hoc postulate that FWF define for the determiner 'every'. According to it, the determiner 'every' induces an order statement between its two arguments. For example, in the derivation tree

$$(\mathsf{every}_{(et)^-((et)^+t)}\ (\mathsf{student}))\ (\mathsf{smiled})$$

the order statement $\mathsf{student}_{(et)} \leq \mathsf{smiled}_{(et)}$ is induced. However, FWF are not able to handle a similar case with transitive verbs. For example, the order statement $(*)\mathsf{student}_{(et)} \ \leq\ \overrightarrow{\lambda} x_e.(\mathsf{Mary}_{((et)^+t)}(\mathsf{kissed}_{(e(et))}(x_e)))$ is naturally derived from the derivation tree $[\overrightarrow{\lambda} x_e.\mathsf{Mary}_{((et)^+t)}(\mathsf{kissed}_{(e(et))}(x_e))](\mathsf{every}_{(et)^-((et)^+t)}(\mathsf{student}_{(et)}))$. Note however, that the sentence *Mary kissed every student* has another derivation tree: $\mathsf{Mary}_{((et)^+t)}(\overleftarrow{\lambda} y_e.([\overrightarrow{\lambda} x_e.([y_e](\mathsf{kissed}_{(e(et))}(x_e)))]\ \mathsf{every}_{(et)^-((et)^+t)}(\mathsf{student}_{(et)})))$, from which we can not derive (*). We address the issue of multiple derivations in section 10. The generalized postulate for 'every' is defined as follows:[3]

$$\boxed{\dfrac{\mathsf{T}\ \leq\ (\mathsf{every}_{(et)^-((et)^+t)})\ (\alpha_{(et)}))\ \langle\beta_{(et)}\rangle}{\alpha_{(et)}\ \leq\ \beta_{(et)}}\ \textbf{(ev)}}$$

9 Examples of derivations in the $\mathcal{L}$-style OC

Below we give some examples of inferences that are derivable in the $\mathcal{L}$-based OC. The derivation of the first inference involves a rather direct application of the Abstraction rule, and a standard treatment of extraction using hypothetical reasoning. The second inference involves the usage of the generalized 'every' postulate as defined above. The third inference uses dynamic decoration of categories created via Abstraction rules.
Extraction. We will now show that the sentence *Every student whom Mary touched smiled* entails the sentence *Every student whom Mary kissed smiled* assuming that whenever an object is kissed, it is always touched (the assumption is represented by the premise $\mathsf{kissed}{\leq}\mathsf{touched}$ in the OC). Hence, we show the derivation of the following inference[4]:

$$((\mathsf{every}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{touched}\ (x))))))(\mathsf{smiled}))_{\mathbf{t}} \leq$$

[3]We assume that the first, nominal argument of 'every' always appears on the right.

[4]It is shown in FWF that the upward monotonic behavior of conjunctive elements can be derived by their system. Hence, any conjunctive category $(A|B)|^C C$ can be replaced by $(A|^+B)|^+C$ with no loss of generality. Thus we treat the conjunctive category of whom as upward monotone.

$$\leq ((\mathsf{every}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))))))\ (\mathsf{smiled}))_{\mathbf{t}}$$

Since the sentences are non-ambiguous, the sentence *Every student whom Mary touched smiled* entails the sentence *Every student whom Mary kissed smiled.*
The derivation is as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathsf{kissed}_{e(et)}\ \leq\ \mathsf{touched}_{e(et)}\qquad x_e\ \equiv\ x_e}{(\mathsf{kissed}\ (x_e))_{\mathbf{(et)}}\ \leq\ (\mathsf{touched}\ (x_e))_{\mathbf{(et)}}}\ \mathbf{(FR)}}{(\mathsf{Mary}_{((et)+t)}\ (\mathsf{kissed}\ (x)))_{\mathbf{t}}\ \leq\ (\mathsf{Mary}_{((et)+t)}\ (\mathsf{touched}\ (x)))_{\mathbf{t}}}\ \mathbf{(Mon+)}}{(\overrightarrow{\lambda} x.\mathsf{Mary}(\mathsf{kissed}\ (x)))_{\mathbf{(et)}}\ \leq\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{touched}\ (x)))_{\mathbf{(et)}}}\ \mathbf{(Ab)}}{\begin{array}{c}(\mathsf{whom}_{(et)C((et)(et))}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))))_{\mathbf{(et)(et)}} \leq \\ \leq (\mathsf{whom}_{(et)C((et)(et))}\ (\overrightarrow{\lambda} x.\mathsf{Mary}(\mathsf{touched}\ (x))))_{\mathbf{(et)(et)}};\end{array}}\ \mathbf{(Mon+)}$$

$$\dfrac{\dfrac{\dfrac{\mathsf{student}_{(et)}\ \equiv\ \mathsf{student}_{(et)}}{([\mathsf{student}_{(et)}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x)))))_{\mathbf{(et)}}\ \leq\ ([\mathsf{student}_{(et)}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{touched}\ (x)))))_{\mathbf{(et)}}}\ \mathbf{(FR)}}{\begin{array}{c}(\mathsf{every}_{(et-)((et)+t)}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{touched}\ (x))))))_{((\mathbf{et})^{+}\mathbf{t})}\ \leq \\ ((\mathsf{every}_{(et-)((et)+t)}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))))))_{((\mathbf{et})^{+}\mathbf{t})}\end{array}}\ \mathbf{(Mon\text{-})}}{\begin{array}{c}((\mathsf{every}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{touched}\ (x))))))(\mathsf{smiled}_{(et)}))_{\mathbf{t}} \leq \\ ((\mathsf{every}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))))))\ (\mathsf{smiled}_{(et)}))_{\mathbf{t}}\end{array}}\ \mathbf{(Mon+)}$$

Use of more than one premise and the generalized "every"-postulate. We will derive the inference

$$\dfrac{\begin{array}{c}\mathsf{T}\ \leq\ (\mathsf{No}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))))))\ (\mathsf{walked}), \\ \mathsf{T} \leq [\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))]\ (\mathsf{every}\ (\mathsf{student}))\end{array}}{\mathsf{T}\ \leq\ (\mathsf{No}\ (\mathsf{student}))(\mathsf{walked})}$$

The derivation is as follows:

$$\dfrac{\dfrac{\begin{array}{c}\dfrac{\dfrac{\dfrac{\begin{array}{c}\mathsf{student}_{(et)}\ \leq\ \mathsf{student}_{(et)}, \\ \mathsf{T} \leq ([\overrightarrow{\lambda} x_e.\mathsf{Mary}_{((et)+t)}\ (\mathsf{kissed}_{(e(et))}\ (x_e))]\ (\mathsf{every}_{(et)-((et)+t)}\ (\mathsf{student}_{(et)})))_{\mathbf{t}}\end{array}}{\mathsf{student}_{(et)}\ \leq\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x)))_{\mathbf{(et)}}}\ \mathbf{(ev)}}{\mathsf{student}_{(et)}\ \leq\ ([\mathsf{student}_{(et)}]\ \mathsf{whom}_{(et)C(et)(et)}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))))_{\mathbf{(et)}}}\ \mathbf{C_2}}{\begin{array}{c}(\mathsf{No}_{(et)-((et)-t)}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))))))_{((\mathbf{et})^{-}\mathbf{t})} \leq \\ (\mathsf{No}_{(et)-((et)-t)}\ (\mathsf{student}))_{((\mathbf{et})^{-}\mathbf{t})},\end{array}}\ \mathbf{(Mon\text{-})} \\ \mathsf{walked}_{(et)}\ \equiv\ \mathsf{walked}_{(et)}\end{array}}{\mathsf{T}\ \leq\ ((\mathsf{No}\ ([\mathsf{student}]\ (\mathsf{whom}\ (\overrightarrow{\lambda} x.\mathsf{Mary}\ (\mathsf{kissed}\ (x))))))\ (\mathsf{walked}_{(et)}))_{\mathbf{t}}\ \leq ((\mathsf{No}\ (\mathsf{student}))(\mathsf{walked}_{(et)}))_{\mathbf{t}}}\ \mathbf{(FR)}}{\mathsf{T}\ \leq\ ((\mathsf{No}\ (\mathsf{student}))(\mathsf{walked}))_{\mathbf{t}}}\ \mathbf{(Trans)}$$

As in the first example, since the sentences are non-ambiguous, we can say that the sentences *No student whom Mary kissed walked* and *Mary kissed every student* entail the sentence *No student walked.*
Using dynamic decoration. We will now show a derivation which makes

use of the dynamically decorated category $(s/^{-}(s\backslash np))$ of the "gapped" expression 'John doesn't'. The '-' marking of the category is determined by the polarity of x in the derivation tree

$$(\overrightarrow{\lambda} x_{(et)}.\mathsf{John}_{((et)^{+}t)}(\mathsf{doesn't}_{(et)^{-}(et)}(x_{(et)})))_{((et)^{-}t)}$$

We will now derive the following order statement:

$$([\overrightarrow{\lambda} y.\mathsf{Mary}(\mathsf{does}(y))]\ \mathsf{and})\ (\overrightarrow{\lambda} x.\mathsf{John}(\mathsf{doesn't}(x))))\ (\mathsf{move}) \ \leq$$

$$\leq (\overrightarrow{\lambda} x.\mathsf{John}(\mathsf{doesn't}(x)))\ (\mathsf{walk})$$

given the order statement walk$\leq$move. The (abbreviated) derivation is as follows:

$(([\overrightarrow{\lambda} y_{(et)}.\mathsf{Mary}(\mathsf{does}_{(et)^{+}(et)}(y))]\ \mathsf{and}_{(et)^{C}((et)(et))})\ (\overrightarrow{\lambda} x_{(et)}.(\mathsf{John}(\mathsf{doesn't}_{(et)^{-}(et)}(x_{(et)}))))\ )(\mathsf{move})$
$\equiv$ ((**PWC**)) $([((\overrightarrow{\lambda} y.\mathsf{Mary}\ (\mathsf{does}(y)))(\mathsf{move}))_{\mathbf{t}}]\ \mathsf{and}_{t^{C}(tt)})\ ((\overrightarrow{\lambda} x.(\mathsf{John}\ (\mathsf{doesn't}(x))))_{\mathbf{t}}(\mathsf{move})) \leq$
$(\mathbf{C}_1)$
$(\overrightarrow{\lambda} x.\mathsf{John}\ (\mathsf{doesn't}\ (x)))_{((\mathbf{et})^{-}\mathbf{t})}(\mathsf{move}_{(et)}) \leq$ (**Mon-**) $(\overrightarrow{\lambda} x.\mathsf{John}\ (\mathsf{doesn't}\ (x))))_{((\mathbf{et})^{-}\mathbf{t})}(\mathsf{walk}_{(et)})$

Then, using **(Trans)** rule we derive the above order statement.
Again, as in the first example, since the sentences are non-ambiguous we can say that the sentence *Mary does and John doesn't walk* entails the sentence *John doesn't move* under the assumption that all walking objects are moving objects.

10 Multiple derivations

Multiple derivations of the same sequent are a well known property of $\mathcal{L}$. Assigning several derivation trees to a natural language sentence reflects several semantic readings that the sentence may have due to reasons such as scope ambiguity etc. For example, the sentence *No man loved some woman* has two possible readings. (i)There exists such woman that no man loved. Under this kind of reading, the sentence *No man loved some tall woman* entails the sentence *No man loved some woman.* (ii)There is no such man that loved some woman. Under this kind of reading, the sentence *No man loved some woman* entails the sentence *No man loved some tall woman.*
The ambiguity in this example is reflected by the fact that there are two possible derivation trees of the sentence:
(i)$[\overleftarrow{\lambda} y_{np}.(\mathsf{No}_{(s/^{-}(s\backslash np))/^{-}n}\ (\mathsf{man}_n))\ (\mathsf{loved}_{(s\backslash np)/np}\ (y))](\mathsf{some}_{(s/^{+}(s\backslash np))/^{+}n}\ (\mathsf{woman}_n))$
(ii)$(\mathsf{No}_{(s/^{-}(s\backslash np))/^{-}n}\ (\mathsf{man}_n))\ (\overrightarrow{\lambda} x_{np}.[\overleftarrow{\lambda} y_{np}.[x](\mathsf{loved}_{(s\backslash np)/np}\ (y))]\mathsf{some}_{(s/^{+}(s\backslash np))/^{+}n}\ (\mathsf{woman}_n)$
The derivations match the readings of the sentence respectively. Thus we

can derive both of the following order statements:

(i) $[\overleftarrow{\lambda} y.(\mathsf{No}\ (\mathsf{man}))\ (\mathsf{loved}\ (y))](\mathsf{some}\ (\mathsf{woman})) \leq [\overleftarrow{\lambda} y.(\mathsf{No}\ (\mathsf{man}))\ (\mathsf{loved}\ (y))](\mathsf{some}\ (\mathsf{tall}\ (\mathsf{woman})))$

(ii) $(\mathsf{No}\ (\mathsf{man}))\ (\overrightarrow{\lambda} x.([\overleftarrow{\lambda} y.[x](\mathsf{loved}\ (y))](\mathsf{some}\ ((\mathsf{tall}\ (\mathsf{woman})))) \leq$

$(\mathsf{No}\ (\mathsf{man}))\ (\overrightarrow{\lambda} x.([\overleftarrow{\lambda} y.[x](\mathsf{loved}\ \ (y))](\mathsf{some}\ (\mathsf{woman}))).$

Note, however, that this does not mean that the sentences *No man loved some woman* and *No man loved some tall woman* are equivalent. We derive two order statements using completely different derivation trees. Recall, that an order statement is defined for derivation trees, not sentences.

11 Conclusions

In this paper we have modified FWF's system of natural reasoning to fit the Abstraction rule of the Lambek calculus. This is a necessary step towards extending the paradigm of natural logic into an inference system with a large coverage of natural language structures. Especially interesting from this perspective is the necessity we found to use dynamic marking rules, in the style of Sánchez and Bernardi. While this marking was not motivated in the simple AB calculus that FWF use, the Abstraction rule of the Lambek calculus makes it necessary to reason about monotonicity properties of function in a way that is not covered by the rules of the OC alone. The implications of this point still need to be further studied, as well as other questions about the proposed system: its completeness properties, decidability and computational implementations. These questions should be tackled if the Natural Logic paradigm is to be extended into a substantial system of reasoning in natural language. We believe that the present work has shown some advances in this direction.

12 Acknowledgements

The parts of the second and third authors were partly supported by the fund for the promotion of research at the Technion, research no. 120-042, and by a BSF grant "Extensions and Implementations of Natural Logic".

References

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

[Bernardi(2002)] Bernardi, R. (2002). Reasoning with Polarity in Categorial Type Grammar. Ph.D. dissertation. Utrecht university.

[FWF(2002)] Fyodorov, Y., Winter, Y., Francez, N. (2002). Order-Based Inference in Natural Logic. To appear in *Journal of Language and Computation 2002*, available at http://www.cs.technion.ac.il/ yaroslav.

[Gamut(1991)] Gamut, L.T.F. (1991). Logic, Language, and Meaning, Vol.2. Intensional Logic and Logical Grammar. The University of Chicago Press, Chicago.

[Moortgat(1997)] Moortgat, M. (1997). Categorial type logics. *Handbook for Logic and Language* , chapter 2, pp. 93-177. Johan van Benthem and Alice ter Muelen, eds., Elsevier/MIT press.

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

[Steedman(2000)] Steedman, M. (2000). The syntactic process. MIT press.

[Wansing(1993)] Wansing, H. (1993). The logic of information structures. *Springer Lecture Notes*, Springer-Verlag, Berlin. AI 681.