Generalized Quantifiers in Declarative and Interrogative Sentences

RAFFAELLA BERNARDI, *UiL OTS, Utrecht University, Trans 10, 3512 JK Utrecht, NL, E-mail: Raffaella.Bernardi@let.uu.nl*

RICHARD MOOT, *UiL OTS, Utrecht University, Trans, 10 3512 JK Utrecht, NL, E-mail: Richard.Moot@let.uu.nl*

Abstract

In this paper we present a logical system able to compute the semantics of both declarative and interrogative sentences. Our proposed analysis takes place at both the sentential and at the discourse level. We use syntactic inference on the sentential level for declarative sentences, while the discourse level comes into play for our treatment of questions. Our formalisation uses a type logic sensitive to both the syntactic and semantic properties of natural language. We will show how an account of the linguistic data follows naturally from the logical relations inherent in the type logic.

Keywords: Type Logical Grammar, Type Logical Semantics, Generalised Quantifiers, Question Semantics

1 Introduction

Natural reasoning inferences are derived from structures whose grammaticality is affected by both syntactic and semantic constraints. A system modelling these semantic inferences must be able to take both types of constraint into account. The role of the syntax-semantics interface in the analysis of natural reasoning inferences is illustrated by the example below.

Sentences with multiple quantifier occurrences may receive different interpretations. The different readings of a sentence allow different inferential possibilities. For instance,

(a) [Few> Every]

$$\frac{\text{Few referees read every short abstract}}{\text{Few referees read every excellent short abstract}}$$

(b) [Every > Few]

$$\frac{\text{Few referees read every short abstract}}{\text{Few referees read every abstract}}$$

The inferences (a) (resp. (b)) are correctly derived from *Few referees read every short abstract* when interpreted with the object narrow-scope (resp. wide-scope) reading.

In natural language, generalised quantifiers (GQs) do not always realize the full set of combinatorial possibilities for scope dependencies predicted by their semantic type assignment as sets of properties. As a result, certain inference substitutions that would be logically valid are not available in natural reasoning. For instance, one would expect the following two inferences to be derivable from *Three good referees read few abstracts*, similarly to what we have seen above.

(a) [Three > Few]

$$\frac{\text{Three good referees read few abstracts}}{\text{Three referees read few abstracts}}$$

(b) [Few > Three]

$$\frac{\text{Three good referees read few abstracts}}{\text{Three good Dutch referees read few abstracts}}$$

where (a) would be logically correct in the interpretation [Three > Few] and (b) in the wide scope reading [Few > Three]. However, while (a) is a correct natural reasoning inference, (b) is not. This difference at the reasoning level is a side effect of some different properties proper of the quantifier phrases *few referees* and *every abstract* [3].

The example shows that natural language structures contribute to natural reasoning and illustrates the need to account for this information when aiming to model natural reasoning. Moreover, it sheds light on the importance of some differences holding among items of the same semantic type, e.g. within the class of quantifiers, which are irrelevant for the meaning assembly, but effect the form composition. In this paper, we focus on this preliminary task to be carried out by a formal system employed to account for natural reasoning inferences.

Lambek calculi [15] are well known for being able to properly account for natural language syntactic-semantic interface by means of the Curry-Howard correspondence between proofs and lambda-terms [11, 4]. Thanks to this relation, proofs of the grammaticality of a string correspond to lambda terms. The form/meaning assembly is carried out in parallel by means of function application and abstraction. However, the example above shows that expressions with the same meaning can have different syntactic distribution. This difference between the syntactic and semantic levels cannot be expressed by a system in a one-to-one correspondence with the lambda calculus. Syntactic types must encode some features which are non visible in the semantic types.

In [12], Kurtonina and Moortgat extended the logical language of the Lambek calculi with unary operators, obtaining Multimodal Categorial Logics (MMCL). In this paper, we show how the latter have the right expressivity to encode fine-grained distinctions among expressions of the same semantic type.

The paper is divided into two main parts: In Section 2 and Section 3 we give a brief presentation of the linguistic data concerning scope ambiguity phenomena and we introduce MMCL showing how it can be used to account for these linguistic data. When building the lexicon in this part, we concentrate on the type language of the system, hiding the corresponding semantic representation since the constraints are purely syntactic. Finally, in Section 4 we show how the results at the syntactic level contribute to giving definitions for the semantic representations for polarity and constituent questions.

2 Scope Ambiguity

Quantifiers offer interesting challenges for the treatment of the syntax/semantic interface. First of all, they can take scope wider than where they occur overtly as illustrated by the object wide scope reading assigned to *Few referees read every short abstract.* Moreover, quantifiers differ with respect to the ways of scope taking as shown by the non-validity of the inference derived from the object wide scope reading of *Three good referees read few abstracts.*

For quite a long time, linguists have concentrated only on the first problem exhibited by GQs. In the generative tradition since the pioneer work of May [16], all GQs have been treated as having the same scope possibilities. We can refer to this approach as the *Uniformity of Quantifier Scope Assignment.* Beghelli and Stowell [3] present evidence against this approach and propose a move to a more flexible theory which explains how and why different types of GQs can have different scoping possibilities.

In [3] scope is seen as the by-product of agreement processes, and mismatches in agreement give rise to ungrammatical sentences. Beghelli and Stowell distinguish five classes of GQs. Membership in any of the GQ types is indicated by some syntactic properties which are morphologically encoded in the determiner position. They claim that for certain combinations of quantifier types the grammar simply excludes certain logically possible scope construals. We refer the reader interested in the linguistic details of the theory to [3], we just summarise their data in Table 1 on page 421.

Sentence	Scope
1. (a) **What didn't** *every actor* know?	$?\forall$
(b) ***How didn't** *every actor* behave ?	–
2. (a) Coppola **didn't** direct *any movie.*	$\neg\exists$
(b) Coppola **didn't** direct *a movie.*	$\exists\neg$ and $\neg\exists$
(c) Coppola **didn't** direct *some movie.*	$\exists\neg$
3. (a) **Any actor* **didn't** like Kubrick.	–
(b) *An actor* **didn't** like Kubrick.	$\exists\neg$ and $\neg\exists$
(c) *Some actor* **didn't** like Kubrick.	$\exists\neg$

TABLE 1. Scope data

We will focus on three types of GQ represented by *any*, *a* and *some*, and on the way they interact with negation and wh-questions. To be more precise, *some* and *a* belong to the same class, viz. 'group-denoting' quantifier phrases. However, they differ the way they take scope with respect to negation, which is our main point of interest.

We start analysing sentences 2-3 in Section 3, we then move to discuss the examples in 1, in Section 4.

3 Quantifier Scope in Multimodal Categorial Logic

In the type logical tradition, Montague [22] makes the first step towards the solution of the first problem presented by quantifier scope, namely the ability of GQs to take scope wider than where they occur at surface structure. His solution has been further developed by Hendriks [9] and incorporated into Categorial Type Logic frameworks in [17]. In this paper, we improve on the latter solution showing how MMCL can account for the second problem exhibited by GQs, namely their difference in the way of scope taking. Intuitively, the binary operators of the Lambek calculi will model the merge of the structures, and the unary operators introduced in [12] will be employed to zoom in on the domains of interpretation and distinguish the objects exhibiting different distributional behaviour. A detailed comparison of the categorial logic approach introduced here with the minimalist analysis proposed by Beghelli and Stowell is given in [5].

In our system, different scope possibilities of a sentence correspond to different *proofs* of the parsed string. Syntactic and semantic information is stored in the lexicon and propagated through the proof by means of logical rules. We will use these characteristics of the system to account for scope ambiguity phenomena, and the unary operators of MMCL to account for the different ways of scope taking identified by Beghelli and Stowell.

In this section we first briefly present the system, and then we show how we can infer the linguistic data given in Table 1 from the rules of the system starting from the lexical assignments.

Derivable objects in MMCL are of the form $\Gamma : A$ where Γ is a *structure* (typically a tree representation of a sentence) and A is a *formula* indicating the syntactic type of this structure.

Definition 3.1 (Formulas) Over a finite set of atomic formulas $\mathcal{A}$, we define the set of *formulas* $\mathcal{F}$ as follows:

$$\mathcal{F} ::= \mathcal{A} \mid \Diamond\mathcal{F} \mid \Box^{\downarrow}\mathcal{F} \mid \mathcal{F}/\mathcal{F} \mid \mathcal{F} \bullet \mathcal{F} \mid \mathcal{F}\backslash\mathcal{F}$$

Definition 3.2 (Structures) Over a countably infinite set of structural variables $\mathcal{V}$, we define the set of *structure terms* as follows:

$$\mathcal{S} ::= \mathcal{V} \mid \langle\mathcal{S}\rangle \mid (\mathcal{S} \circ \mathcal{S})$$

To make our proofs a bit more readable, we will typically use lexical word as structural variables. A structure term is then a tree of words.

The natural deduction calculus for MMCL tells us how to combine proofs from an initial set of lexical assignments to produce phrases of different types. See [21] for a more detailed explanation and many linguistic applications of the Fitch-style natural deduction calculus presented below.

In the logical rules below X, Y, Z range over structure terms, A, B, C range over formulas and $\mathbf{x}, \mathbf{y}$ are structural variables not occurring elsewhere in the proof. Finally, $Z[X]$ denotes a structure term Z with a distinguished subterm occurrence X.

Tables 2 on page 423 and 3 on page 424 list the logical rules of type-logical grammar. The $[/E]$ rule tells us that whenever rule n of our proof shows structure X to be of type A/B and line m of this same proof shows structure Y to be of type B, then the combined structure $(X \circ Y)$ is of type A. The $[/I]$ rule indicates that, once we hypothesise a B formula with a fresh structural variable $\mathbf{x}$, we can discharge this formula once we derive a formula A with structure $(X \circ \mathbf{x})$ to produce a formula A/B with structure X. Note that we mark the scope of a hypothesis with a vertical bar and that all hypotheses should be discharged at the end of the proof.

According to the Curry-Howard interpretation, natural deduction proofs correspond to typed lambda terms. For our current applications, we are only interested in the semantics of the implications. When we want to compute the meaning of a syntactic expression, it is often convenient to add semantic labels to the proof steps. Derivable objects are then of the $\Gamma : A - t$, where Γ is a structure, A is a formula and t is the semantics of the expression. Table 4 lists the semantically annotated rules for the implications; the other connectives have their own Curry-Howard interpretations, but

Lexicon

$$\begin{array}{lll} n & \mathbf{x} : A & \mathit{Lex} \end{array}$$

Hypothesis

$$\begin{array}{l|ll} n & \mathbf{x} : A & \mathit{Hyp} \end{array}$$

Binary Rules

$$\begin{array}{lll} n & X : A/B & \\ m & Y : B & \\ & (X \circ Y) : A & /E(n,m) \end{array}$$

$$\begin{array}{l|ll} n & \mathbf{x} : B & \mathit{Hyp} \\ m & (X \circ \mathbf{x}) : A & \\ & \multicolumn{1}{l}{X : A/B} & /I(n,m) \end{array}$$

$$\begin{array}{lll} n & Y : B & \\ m & X : B\backslash A & \\ & (Y \circ X) : A & \backslash E(n,m) \end{array}$$

$$\begin{array}{l|ll} n & \mathbf{x} : B & \mathit{Hyp} \\ m & (\mathbf{x} \circ X) : A & \\ & \multicolumn{1}{l}{X : B\backslash A} & \backslash I(n,m) \end{array}$$

$$\begin{array}{l|ll} n & \multicolumn{1}{l}{X : A \bullet B} & \\ m & \mathbf{x} : A & \mathit{Hyp} \\ m+1 & \mathbf{y} : B & \mathit{Hyp} \\ p & Z[(\mathbf{x} \circ \mathbf{y})] : C & \\ & \multicolumn{1}{l}{Z[X] : C} & \bullet E(n,m,m+1,p) \end{array}$$

$$\begin{array}{lll} n & X : A & \\ m & Y : B & \\ & (X \circ Y) : A \bullet B & \bullet I(n,m) \end{array}$$

TABLE 2. The logical rules of type-logical grammar

they are not relevant for our current applications.

The semantically annotated [$/E$] rule now tells us that whenever we combine an A/B formula with semantics t with a B formula with semantics u the resulting A formula has the term $(t\ u)$ as its semantics. For the [$/I$] rule, the hypothesis B is initially assigned a fresh variable x as its semantics, then we continue our proof until we derive a formula A with some semantics t. We can now withdraw our B hypothesis by abstracting over its variable, producing $\lambda x.t$ as the semantics of the expression A/B. Note that without the structural labelling these are just the logical rules of implication in intuitionistic logic. Also note that on the semantics level, we are not interested in the difference between the two implications: both have the same semantic content.

The reader may wonder about the complexity of this system and about how proof search would proceed. Natural deduction proofs have the pleasant property that for

Unary Rules

$$\begin{array}{lll}
n & X : \Diamond A & \\
m & \vert\, \mathbf{x} : A & \mathit{Hyp} \\
p & \vert\, Z[\langle \mathbf{x} \rangle] : C & \\
 & Z[X] : C & \Diamond E(n, m, p)
\end{array}$$

$$\begin{array}{lll}
n & X : A & \\
 & \langle X \rangle : \Diamond A & \Diamond I(n)
\end{array}$$

$$\begin{array}{lll}
n & X : \Box^{\downarrow} A & \\
 & \langle X \rangle : A & \Box^{\downarrow} E(n)
\end{array}$$

$$\begin{array}{lll}
n & \langle X \rangle : A & \\
 & X : \Box^{\downarrow} A & \Box^{\downarrow} I(n)
\end{array}$$

TABLE 3. The logical rules for the unary connectives

Implications

$$\begin{array}{lll}
n & X : A/B - t & \\
m & Y : B - u & \\
 & (X \circ Y) : A - (t\ u) & /E(n, m)
\end{array}$$

$$\begin{array}{lll}
n & \vert\, \mathbf{x} : B - x & \mathit{Hyp} \\
m & \vert\, (X \circ \mathbf{x}) : A - t & \\
 & X : A/B - \lambda x.t & /I(n, m)
\end{array}$$

$$\begin{array}{lll}
n & Y : B - u & \\
m & X : B\backslash A - t & \\
 & (Y \circ X) : A - (t\ u) & \backslash E(n, m)
\end{array}$$

$$\begin{array}{lll}
n & \vert\, \mathbf{x} : B - x & \mathit{Hyp} \\
m & \vert\, (\mathbf{x} \circ X) : A - t & \\
 & X : B\backslash A - \lambda x.t & \backslash I(n, m)
\end{array}$$

TABLE 4. The logical rules for the implications with semantics

finding proofs of a given logical statement we only need to consider the subformulas of this logical statement, thereby bounding the search space for proof search. With respect to the complexity, de Groote [8] presents a polynomial algorithm for the system as described above, but adding more complex structural possibilities, like we need for our treatment of generalised quantifiers will increase the complexity and can make the system PSPACE complete in the worst case [20].

We illustrate how the logical system can be used to reason with linguistic signs by showing a simple proof consisting only of elimination rules.

Example 3.3 Given the lexicon below we have specified that *tarantino* and *pulp fiction* are lexical expressions of type *np*, that *oscar* and *movie* are of type *n* and that *directed* and *won* are of type $(np\backslash s)/np$. The latter complex type means that it combines first with an *np* to the immediate right then with an *np* to the immediate left to produce an *s*, which is the atomic type assigned to sentences. We will discuss the correct type assignments to determiners like *a* and *some* after this example.

Lexicon

pulp fiction : np – ***pulp***	*tarantino* : np – ***tarantino***
oscar : n – ***oscar***	*won* : $(np\backslash s)/np - \lambda xy.((\textbf{\textit{win}}\ x)\ y)$
movie : n – ***movie***	*directed* : $(np\backslash s)/np - \lambda xy.((\textbf{\textit{direct}}\ x)\ y)$

Using this lexicon, together with the logical rules above, we can prove that *tarantino directed pulp fiction* is a well-formed expression of type s and its meaning representation is **(direct pulp) tarantino**. For the sake of simplicity, we replace the structural formulas with their corresponding linguistic expressions.

1.	*tarantino* : np – ***tarantino***	*Lex*
2.	*directed* : $(np\backslash s)/np - \lambda xu.((\textbf{\textit{direct}}\ x)\ y)$	*Lex*
3.	*pulp fiction* : np – ***pulp***	*Lex*
4.	$($*directed* $\circ$ *pulp fiction*$)$: $np\backslash s - \lambda x.(\textbf{\textit{direct pulp}})\ x$	$/E(2,3)$
5.	$($*tarantino* $\circ$ $($*directed* $\circ$ *pulp fiction*$))$: s – ***(direct pulp) tarantino***	$\backslash E(1,4)$

The proof starts from the lexical assumptions. The elimination of the main connective of the complex type assigned to the transitive verb, namely $/E$ is applied to the premises 2 and 3, yielding a structure of type $np\backslash s$. Similarly, the connective $\backslash$ is eliminated, composing the final structure which is proved to be of type s. Furthermore, the example shows that the elimination rules of $\backslash, /$ correspond to functional applications. Note that steps 4 and 5 hide the application of β-reduction.

According to the Montegovian tradition GQs are denoted by functions which take scope at the sentence level [2]. A type suitable for a subject generalised quantifier would be $s/(np\backslash s)$, that is a type which produces a sentence when it finds to its right a sentence missing an *np* to its left. Similarly, an object generalised quantifier would be assigned the type $(s/np)\backslash s$. The reader might complain that there are no good motivations for such a duplicate type assignment. A solution to this problem is given in [17, 18], where it is shown how MMCL can be extended in such a way that a single type assignment suffices for each quantifier, regardless of its position in the sentence. However, for the sake of simplicity, we will abstract over this issue and adopt the general notation $(np \to s) \to s$. Doing so we can focus on the problem we are interested in, namely the different scope possibilities of quantifiers.

The use of a uniform logical type assignment $(np \to s) \to s$ for all GQs could be seen as a deductive version of May's [16] Scope Uniformity thesis, and would fail in accounting for the different scope possibilities of GQs discussed in the previous section. In order to diversify the way GQs scope on sentences, we refine this type assignment further, distinguishing three different sentential levels, s_1, s_2 and s_3 to which three logically related types are assigned. We consider the standard sentential type s to be the type of the medium sentential level s_2 and we derive the other two types as shown below.

Suppose that we have a proof that a structure X is of type s, then we can prove it is of type $\Box^{\downarrow}\Diamond s$ as follows.

1.	$X : s$	
2.	$\langle X \rangle : \Diamond s$	$\Diamond I(1)$
3.	$X : \Box^{\downarrow}\Diamond s$	$\Box^{\downarrow} I(2)$

similarly, we can prove that $\Diamond\Box^{\downarrow}s$ derives ($\Rightarrow$) s, as follows.

1.	$X : \Diamond\Box^{\downarrow}s$	
2.	$\vert\ \mathbf{x} : \Box^{\downarrow}s$	*Hyp*
3.	$\vert\ \langle \mathbf{x} \rangle : s$	$\Box^{\downarrow}E(2)$
4.	$X : s$	$\Diamond E(1,2,3)$

The converse derivability relations do not hold. The reader can verify this by trying all possible proofs using only subformulas of the logical types.

Summing up, the logical derivability relation connecting the three types is:

sentential levels	s_1		s_2		s_3
logical types	$\Diamond\Box^{\downarrow}s$	$\Rightarrow$	s	$\Rightarrow$	$\Box^{\downarrow}\Diamond s$

The logical derivability relation among these types will play a crucial role in our analysis, as we will show when presenting the examples.

Let us now look at the way GQs differ on the level of the sentence they are allowed to take scope over. The sentences in Section 2 show that *any* cannot have scope over a negative sentence (see examples 2.-3.(a)), whereas *a* can (see examples 2.-3.(b)), and *some* must (see examples 2.-3.(c)). We consider a negative sentence as the borderline that can, cannot or must be reached or overcome by a GQ. Its type is s and it is on the sentential level s_2. As a consequence of the logical relations above, the level below s is of category $\Diamond\Box^{\downarrow}s$, and the level above it is of category $\Box^{\downarrow}\Diamond s$. GQs like *any* can have scope only on the lowest level s_1, and are blocked when trying to reach the higher ones. *Some*, instead, must operate on a higher level than negative sentences s: it takes scope on the level s_3 of type $\Box^{\downarrow}\Diamond s$. Finally, *a* is on the same level as a negative sentence, therefore we correctly predict scope ambiguity in this case.

With this in place, we can represent the three GQ types as:

GQ$_1$:	$(np \to s_1) \to s_1$	(e.g. *any movie*)
GQ$_2$:	$(np \to s_2) \to s_2$	(e.g. *a movie*)
GQ$_3$:	$(np \to s_3) \to s_3$	(e.g. *some movie*)

which mean that GQ$_1$ cannot operate on the level s_2 we obtain when negating a sentence, whereas GQ$_2$ can and GQ$_3$ lifts the level up to s_3. Another important consequence of these different type assignments is on the way generalised quantifiers can combine with each-other when more than one GQ occurs in the same sentence. The three types we have been describing so far give few combinatory possibilities. We can summarise the permitted scopings as: GQ$_i$ will always have wide scope over GQ$_j$ where $i > j$. Looking at linguistic data, however, we can easily find generalised quantifiers which behave differently. We will comment on this after showing the GQ types above at work.

Before introducing the type for *didn't* two comments must be made. First of all, negation-like expressions, e.g. *didn't*, play a crucial role in our approach; they move the sentential level up from the initial s_1 level to the 'negated' s_2 level. We have to take this into account for the lexical type assignment.

Moreover, to get the desired interaction between negation and generalised quantifiers, we follow Carpenter's raising strategy [6]. Instead of considering *didn't* as a standard verb phrase modifier vp/vp – where $vp = np\backslash s$, we lift its goal formula to $((s/vp)\backslash s)/vp$, i.e. a type which takes an s incomplete for an np to produce an s incomplete for a GQ. We enrich this type with the information on the sentential levels while abstracting over the directionality of the implication operators, resulting in the following lexical entry.

$$\text{didn't} : (np \to s_2) \to (((np \to s_2) \to s_2) \to s_2)$$

Example 3.4 Given the lexicon below

Lexicon

$coppola : np$
$the\ godfather : np$
$oscar : n$
$movie : n$
$directed : (np\backslash s_1)/np$
$won : (np\backslash s_1)/np$
$any : ((np \to s_1) \to s_1)/n$
$a : ((np \to s_2) \to s_2)/n$
$some : ((np \to s_3) \to s_3)/n$
$didn't : (np \to s_2) \to (((np \to s_2) \to s_2) \to s_2)$

sentences 2-3 of Table 1 can be correctly parsed. We simplify the logical types using the sentential levels instead of their corresponding types, while we abbreviate subproofs of the relations between the sentence level by (derived) rules we name $s_{i,j}$.

We can see in Figure 1 on page 428 that *direct* first combines with *didn't* and then the result combines with *a movie*. Besides the application of the elimination rules which as seen before correspond to functional application, the proof contains applications of the introduction rules of the functional connectives. As marked by the label of the steps 2 and 6, these rules correspond to hypothetical reasoning. For instance, the hypothesis **v** of type $np \to s_2$ assumed in 2, is discharged at the step 4 by means of $\to I$ lifting the noun phrase type of *Coppola* to the GQ type. This order composition produces the reading where the existential quantifier has wide scope ($\exists\neg$): it takes the built structure (Coppola $\circ$ (didn't $\circ$ direct)) in its scope.

In order to understand the way the different lexical type assignments of the GQs properly account for their different scope possibilities, attention has to be drawn on the step 14 in the proof, where we have a structure of type s_2. For the case at hand, with *a movie* in object position, the proof proceeds simply by means of the logical rules of the binary operators. On the other hand, if we consider the sentence 2c where *a* is replaced by *some*, we first have to lift s_2 to s_3 and then proceed as before. Finally, since we cannot derive the type s_1 from s_2, the reading ($\exists\neg$) is disallowed in case *a* is replaced by *any* as in 2a.

An alternative proof exists for sentence 2b, as shown in Figure 2 on page 429, giving the second reading.

Here, instead, *direct* combines first with *a movie* and then with *didn't*. In other words, in this reading the negation has wide scope ($\neg\exists$).

Now, in order to get the required argument $np \to s_2$ at step 16 it is essential we

1.	Coppola : np	*Lex*
2.	$\mid$ $\mathbf{v} : np \to s_2$	*Hyp*
3.	$\mid$ $(\text{Coppola} \circ \mathbf{v}) : s_2$	$\to E\ (1,2)$
4.	Coppola : $(np \to s_2) \to s_2$	$\to I\ (2,3)$
5.	direct : $(np\backslash s_1)/np$	*Lex*
6.	$\mid$ $\mathbf{y} : np$	*Hyp*
7.	$\mid$ $(\text{direct} \circ \mathbf{y}) : np\backslash s_1$	$/E\ (5,6)$
8.	$\mid$ $\mid$ $\mathbf{x} : np$	*Hyp*
9.	$\mid$ $\mid$ $(\mathbf{x} \circ (\text{direct} \circ \mathbf{y})) : s_1$	$\backslash E\ (7,8)$
10.	$\mid$ $\mid$ $(\mathbf{x} \circ (\text{direct} \circ \mathbf{y})) : s_2$	$s_{1,2}\ (9)$
11.	$\mid$ $(\text{direct} \circ \mathbf{y}) : np \to s_2$	$\to I\ (8,10)$
12.	$\mid$ didn't : $(np \to s_2) \to (((np \to s_2) \to s_2) \to s_2)$	*Lex*
13.	$\mid$ $(\text{didn't} \circ (\text{direct} \circ \mathbf{y})) : ((np \to s_2) \to s_2) \to s_2$	$\to E\ (11,12)$
14.	$\mid$ $(\text{Coppola} \circ (\text{didn't} \circ (\text{direct} \circ \mathbf{y}))) : s_2$	$\to E(4,13)$
15.	$(\text{Coppola} \circ (\text{didn't} \circ \text{direct})) : np \to s_2$	$\to I\ (6,14)$
16.	a : $((np \to s_2) \to s_2)/n$	*Lex*
17.	movie : n	*Lex*
18.	$(\text{a} \circ \text{movie}) : (np \to s_2) \to s_2$	$/E\ (16,17)$
19.	$((\text{Coppola} \circ (\text{didn't} \circ \text{direct})) \circ (\text{a} \circ \text{movie})) : s_2$	$\to E\ (15,18)$
20.	$((\text{Coppola} \circ (\text{didn't} \circ \text{direct})) \circ (\text{a} \circ \text{movie})) : s_3$	$s_{2,3}\ (19)$

FIG. 1. *Coppola didn't direct a movie*, $\exists\neg$ reading.

have an s_1 or s_2 result at step 15, which we can only have when the quantifier is *any* or *a*. The failure to derive $s_3 \Rightarrow s_2$ blocks this derivation for the quantifier *some*.

Before considering the interrogatives, we give some comments on how the small fragment we have given can be extended.

For example, when a generalised quantifier like *every actor* combines with other GQs, we have to account for multiple readings. In other words, we need to assign to *every* a type which will allow for more scope possibilities than any of the assignments we have seen so far. A proper type for this kind of GQ is $(np \to s_3) \to s_1$, which will allow *every* to have both wide and narrow scope with respect of a second GQ. So using heterogeneous combinations of sentential types, we can account for GQs with more complex behaviour than *some* and *a*.

Another interesting example is given by the negative polarity items (NPIs), as *any*: items which require to be in the scope of a negative operator [14]. The type assigned to *any* above, will satisfy the request that *when* the negation occurs the only possible reading will be the one where the negation has wide scope. However, using this type *any* can still occur in positive contexts, contrary to linguistic reality. We refer the reader to [5] for a solution to this problem and for the discussion of a larger English fragment with GQs.

4 Interrogatives

Now that the three types of GQs have been introduced and the criteria for differentiating them have been explained, we can discuss the last type of GQ we are interested

1.	Coppola : np	*Lex*
2.	$\mid$ $\mathbf{v} : np \rightarrow s_2$	*Hyp*
3.	$\mid$ $(\text{Coppola} \circ \mathbf{v}) : s_2$	$\rightarrow E\ (1,2)$
4.	Coppola : $(np \rightarrow s_2) \rightarrow s_2$	$\rightarrow I\ (2,3)$
5.	direct : $(np\backslash s_1)/np$	*Lex*
6.	$\mid$ $\mathbf{x} : np$	*Hyp*
7.	$\mid\mid$ $\mathbf{y} : np$	*Hyp*
8.	$\mid\mid$ $(\text{direct} \circ \mathbf{y}) : np\backslash s_1$	$/E\ (5,7)$
9.	$\mid\mid$ $(\mathbf{x} \circ (\text{direct} \circ \mathbf{y}) : s_1$	$\backslash E\ (6,8)$
10.	$\mid\mid$ $(\mathbf{x} \circ (\text{direct} \circ \mathbf{y}) : s_2$	$s_{1,2}\ (9)$
11.	$\mid$ $(\mathbf{x} \circ \text{direct}) : np \rightarrow s_2$	$\rightarrow I\ (7,10)$
12.	$\mid$ a : $((np \rightarrow s_2) \rightarrow s_2)/n$	*Lex*
13.	$\mid$ movie : n	*Lex*
14.	$\mid$ $(\text{a} \circ \text{movie}) : (np \rightarrow s_2) \rightarrow s_2$	$/E\ (12,13)$
15.	$\mid$ $((\mathbf{x} \circ \text{direct}) \circ (\text{a} \circ \text{movie})) : s_2$	$\rightarrow E (11,14)$
16.	$(\text{direct} \circ (\text{a} \circ \text{movie})) : np \rightarrow s_2$	$\rightarrow I\ (6,15)$
17.	didn't : $(np \rightarrow s_2) \rightarrow (((np \rightarrow s_2) \rightarrow s_2) \rightarrow s_2)$	*Lex*
18.	$(\text{didn't} \circ (\text{direct} \circ (\text{a} \circ \text{movie}))) : ((np \rightarrow s_2) \rightarrow s_2) \rightarrow s_2$	$\rightarrow E\ (16,17)$
19.	$(\text{Coppola} \circ (\text{didn't} \circ (\text{direct} \circ (\text{a} \circ \text{movie})))) : s_2$	$\rightarrow E\ (4,18)$
20.	$(\text{Coppola} \circ (\text{didn't} \circ (\text{direct} \circ (\text{a} \circ \text{movie})))) : s_3$	$s_{2,3}(19)$

FIG. 2. *Coppola didn't direct a movie*, $\neg\exists$ reading.

in, namely wh-phrases. This brings us to move from a syntactical approach dealing with sentential inference, to a more semantical one, which involves the discourse level. Therefore, instead of considering only the syntactic types of our lexical items, we will discuss their semantic representation as well.

In natural language we can distinguish two basic categories of questions: *polarity questions* (also known as yes/no questions) and *constituent questions* (also known as wh-questions). We will first discuss the first type, then we will present the second one.

In our framework we will consider a string of words which form a question to be of a different category and to have a different meaning than a declarative sentence. However, their type and denotation will be a logical consequence of the one attributed to sentences. The distinction of levels described above will therefore reappear. In particular, we will distinguish two levels of questions, q_1 and q_2, deriving their types from the one assigned to sentences in s_1 and s_2. For reasons which will become clear later, we also need a third question level q_3 derivable from the types of both q_1 and q_2. The logical relations among the types in these three question levels and in the sentential ones are as shown below. For the sake of simplicity we abbreviate the logical types using their corresponding sentential/question level:

$$\begin{tikzcd}
s_1 \arrow[rr, Rightarrow] \arrow[d, Rightarrow] & & s_2 \arrow[rr, Rightarrow] \arrow[d, Rightarrow] & & s_3 \\
q_1 \arrow[dr, Rightarrow] & & q_2 \arrow[dl, Rightarrow] & & \\
& q_3 & & &
\end{tikzcd}$$

where the relation between types of different levels is the logical derivability relation discussed above; and the one between sentences and questions is the lifting theorem [19]. Hence, q_1 stands for $s_1/(s_1\backslash s_1)$ which in turn abbreviates $\Diamond\Box^{\downarrow}s/(\Diamond\Box^{\downarrow}s\backslash\Diamond\Box^{\downarrow}s)$ and q_2 stands for $s_2/(s_2\backslash s_2)$, viz. $s/(s\backslash s)$. Reading out this logical types, a yes/no question is seen as a function which takes a sentential modifier and yields a sentence. As might be clear, the two categories q_1, q_2 are at the level of positive and negative yes/no questions, respectively. Their type are

Positive: $\Diamond\Box^{\downarrow}s/(\Diamond\Box^{\downarrow}s\backslash\Diamond\Box^{\downarrow}s)$

Negative: $s/(s\backslash s)$

A question to investigate further is whether the type obtained from lifting the type in s_3, i.e. $\Box\Diamond s$, can also play a role in a type logical approach to questions.

In the logical, philosophical and linguistic literature several frameworks have been proposed for the meaning of questions [7]. The logical type we have assigned to the yes/no questions finds its semantic motivation in the structured meaning approach (also called "categorial"), which traces back to Ajdukiewicz [1], as noticed in [10], and has been developed in [23, 13]. The basic idea which characterizes this approach is that:

> Question meanings are functions that, when applied to the meaning of the answer, yield a proposition.

Yes/no questions expect answers like *yes*, *no*. In [13] it is suggested that *no* can be considered as a propositional operator that reverses the truth value, $\lambda p.\neg p$, and *yes* as a propositional operator that retains the truth value, i.e. the identity function: $\lambda p.p$.

Before going to discuss some examples, it is worth to notice the contribution the proof theoretical approach here assumed gives to the semantic investigation on questions. The results we describe bring evidence to the correctness of the categorial approach and complete the framework with the syntatical conterpart. As pointed out in the beginning of the paper, at the heart of any categorial grammar analysis there is the Curry-Howard isomorphism between lambda terms and types. The former are used as semantic representation of the natural language expressions, and the latter as their sytactic type. Since the studies on questions are mostely related with their interpretation in this section we discuss the lambda term representation of the lexical items as well as their type assignments. We start from the final semantic representation of yes-no questions and we then build the lexicon behind it. Let us first present the theory intuitively by means of an example with a polarity question.

Example 4.1

Q	*Did Tarantino direct Titanic?*	$\lambda Y.(Y\ ((\boldsymbol{direct\ titanic})\ \boldsymbol{tarantino}))$
A	*No.*	$\lambda p.\neg p$
Q(A)	*By twice beta-reduction.*	$\neg((\boldsymbol{direct\ titanic})\ \boldsymbol{tarantino})$

Translating into the type language what we have treated so far, an auxiliary as *did* or *didn't* will be a function which takes a sentence and yields a question. More specifically, *did* yields a question of the first level, whereas *didn't* of the second one.

Now that the theory is clear, we can present the lambda terms formally. We first show the desired lambda terms representing a wh-question, and then we give the lexicon displaying both types and lambda terms for the items involved. Additionally, we introduce wh-phrases which give rise to the second type of questions when combined with the first one.

As is explained in [24] wh-phrases differ in the way they behave with respect to negation. This fact can be easily accounted for in our framework, thanks to the distinction between the two levels q_1, q_2 for positive and negative questions. Following the criterion given in [13] and quoted above, we consider wh-questions to be functions taking an answer to yield a sentence. The type of the question therefore depends on the type of its possible answer.

Let us consider *what* as an example.

Example 4.2

Q	*What did Cameron direct?*	$\lambda Y.(Y\ \lambda x.((\textbf{\textit{direct}}\ x)\ \textbf{\textit{cameron}}))$
A	*Titanic*	$\lambda P.P(\textbf{\textit{titanic}})$
Q(A)	*By twice beta-reduction.*	$((\textbf{\textit{direct titanic}})\ \textbf{\textit{cameron}})$

Translating this into the type language, we have that a wh-question is a function which takes a GQ and yields a sentence. We abbreviate this category with *wh*, knowing that $wh = s/GQ$, and add the following lexical entries to our lexicon:

Lexicon

Tarantino : np – **tarantino**

Cameron : np – **cameron**

Titanic : np – **titanic**

no : $s_2 \rightarrow s_2 - \lambda p.\neg p$

direct : $(np\backslash s)/np - \lambda x.\lambda y.((\textbf{direct}\ x)\ y)$

did : $GQ \rightarrow ((np \rightarrow s_1) \rightarrow q_1) - \lambda P.\lambda Q.\lambda R.(R\ (P\ Q))$

didn't : $GQ \rightarrow ((np \rightarrow s_1) \rightarrow q_2) - \lambda P.\lambda Q.\lambda R.(R\ \neg(P\ Q))$

what : $wh/(np \rightarrow q_?) - \lambda Z.\lambda P.(P\ \lambda x.((Z\ x)\ (\lambda U.U)))$

Notice that the lexical type of the auxiliary selects for a generalized quantifier and a *vp* to produce a question. The *GQ* can be any of the three different types we treated before. Therefore, the type selected by the auxiliary has to be general enough to satisfy this request. In other words, *GQ* is such that $GQ_1 \Rightarrow GQ$, $GQ_2 \Rightarrow GQ$ and $GQ_3 \Rightarrow GQ$ are all derivable. These logical properties are assured by $GQ = (np \rightarrow s_1) \rightarrow s_3$. The type assigned to *what* simply means that wh-phrases combined with a yes/no question missing an *np* result into a wh-question. In our language we have two different types of yes/no questions. Before discussing which of them is requested by a wh-phrase we show a derivation of a wh-question. Notice that the type assigned to wh-phrases can account for cases where the answer is a simple proper name (e.g. Titanic), a set of proper names (e.g. Terminator, Aliens, Titanic) or a GQ (e.g. Several famous movies).

Example 4.3 We give a proof of 'What did Cameron direct?' in Figure 3 on page 432

1.	Cameron : np – **cameron**	*Lex*
2.	**v** : $np \rightarrow s_1 - V$	*Hyp*
3.	(Cameron ∘ **v**) : s_1 – (V **cameron**)	$\rightarrow E(1, 2)$
4.	(Cameron ∘ **v**) : s_3 – (V **cameron**)	$s_{1,3}(3)$
5.	Cameron : $(np \rightarrow s_1) \rightarrow s_3 - \lambda V.$($V$ **cameron**)	$\rightarrow I(2, 4)$
6.	did : $GQ \rightarrow ((np \rightarrow s_1) \rightarrow q_1) - \lambda P.\lambda Q.\lambda R.(R\ (P\ Q))$	*Lex*
7.	(did ∘ Cameron) : $(np \rightarrow s_1) \rightarrow q_1$ $-(\lambda P.\lambda Q.\lambda R.(R\ (P\ Q))\ \lambda V.$($V$ **cameron**))	$\rightarrow E(5, 6)$
8.	(did ∘ Cameron) : $(np \rightarrow s_1) \rightarrow q_1$ $-\lambda Q.\lambda R.(R\ (Q$ **cameron**))	$\beta(7)$
9.	direct : $(np\backslash s_1)/np - \lambda x.\lambda y.$((**direct** $x)y)$	*Lex*
10.	**z** : $np - z$	*Hyp*
11.	(direct ∘ **z**) : $np\backslash s_1 - (\lambda x.\lambda y.$((**direct** $x)\ y)\ z)$	$/E(9, 10)$
12.	(direct ∘ **z**) : $np\backslash s_1 - \lambda y.$((**direct** $z)\ y)$	$\beta(11)$
13.	((did ∘ Cameron) ∘ (direct ∘ **z**)) : q_1 – $(\lambda Q.\lambda R.(R\ (Q$ **cameron**)) $\lambda y.$((**direct** $z)\ y))$	$\rightarrow E(8, 12)$
14.	((did ∘ Cameron) ∘ (direct ∘ **z**)) : q_1 – $\lambda R.(R$ ((**direct** z) **cameron**))	$\beta(13)$
15.	((did ∘ Cameron) ∘ (direct ∘ **z**)) : q_3 – $\lambda R.(R$ ((**direct** z) **cameron**))	$q_{1,3}(14)$
16.	((did ∘ Cameron) ∘ direct) : $np \rightarrow q_3$ $-\lambda z.\lambda R.(R$ ((**direct** z) **cameron**))	$\rightarrow I(10, 15)$
17.	what : $wh/(np \rightarrow q_?) - \lambda Z.\lambda P.(P\ \lambda x.((Zx)\ (\lambda U.U)))$	*Lex*
18.	(what ∘ ((did ∘ Cameron) ∘ direct)) : wh $-(\lambda Z.\lambda P.(P\ \lambda x.((Z\ x)\ (\lambda U.U)))$ $\lambda z.\lambda R.(R$ ((**direct** z) **cameron**)))	$/E(16, 17)$
19.	(what ∘ ((did ∘ Cameron) ∘ direct)) : wh $-\lambda P.(P\ \lambda x.((\lambda z.\lambda R.(R$ ((**direct** z) **cameron**))) $x)\ (\lambda U.U))$	$\beta(18)$
20.	(what ∘ ((did ∘ Cameron) ∘ direct)) : wh $-\lambda P.(P\ \lambda x.((\lambda R.(R$ ((**direct** x) **cameron**))) $(\lambda U.U))$	$\beta(19)$
21.	(what ∘ ((did ∘ Cameron) ∘ direct)) : wh $-\lambda P.(P\ \lambda x.((\lambda U.U)$ ((**direct** x) **cameron**)))	$\beta(20)$
22.	(what ∘ ((did ∘ Cameron) ∘ direct)) : wh $-\lambda P.(P\ \lambda x.$((**direct** x) **cameron**))	$\beta(21)$

FIG. 3. Derivation of *what did Cameron direct*

As shown in the proof the type assigned to *Cameron*, viz. np has to be lifted to the one of GQ. Having chosen the type GQ derivable from $GQ_{1,2,3}$, allows 'did' to be combined with either an arbitrary generalized quantifier or a simple noun phrase.

We are now ready to answer the open question of the previous paragraph: what is the type of the question taken as an argument by a wh-phrase? In [24] it is shown that wh-phrases differ from each-other in the way they behave with respect to negation. Szabolcsi and Zwart give algebraic motivations for this linguistic phenomenon which fits naturally into our framework. Having distinguished positive and negative questions enables us to deal with contrasting pairs like *what* and *how* presented in Section 2 and repeated here.

1.(a) What didn't every actor know?

1.(b) *How didn't every actor behave?

As these examples show the wh-phrase *what* can have scope on a question with a negation in it, i.e. a question of category q_2, whereas *how* cannot. Due to this diverse behavior of the two wh-phrases, we differently instantiate the type $q_?$ selected by them, as shown below:

what : $wh/(np \to q_3)$
how : $wh/(adj \to q_1)$

which mean that 'how' can combine only with a positive question, whereas 'what' can select both a positive and a negative one. This fact motivates the choice of q_3 which is the type derivable from both q_1 and q_2, i.e. $s_3/(s_3\backslash s_1)$, viz. $\Box^{\downarrow}\Diamond s/(\Box^{\downarrow}\Diamond s\backslash\Diamond\Box^{\downarrow}s)$. The type *adj* is the one assigned to adjectives.

Concluding this section on interrogatives in our framework, we want to point out a nice consequence of the categorial approach to yes/no questions. As discussed in [13], this approach predicts the different behavior of polarity questions with respect to alternative questions, i.e. questions as: 'Didn't Tarantino direct Titanic, or did he?'. Although the two types of questions may look similar, they are semantically different as shown by the answers they expect. This difference is generated by the disjunctive phrase. This phenomenon fits naturally into our framework by simply extending the lexicon with the type for the disjunctive phrase:

or : $q_i \to ((q_j \to vp) \to (s_? \to s_?))$

where $i, j \in \{1, 2\}$ and $i \neq j$. The type means that 'or' first combines with a positive (resp. negative) polarity question then with a negative (resp. positive) ones missing a verb phrase, to yield a type $s_? \to s_?$. An alternative question will therefore received a type which selects a sentence as answer. This is what is claimed in the categorial approach, and this is what we expect looking at a possible answer to the question above, e.g. 'No, he didn't'. Finally, due to the fact that the answer to alternative questions may be of any of the type in the three sentential levels, we instantiate it with $s_3 \to s_3$.

5 Conclusion

We have presented a proof theoretical analysis of scope ambiguity phenomena involving both syntactic and semantic constraints. We have shown how to profit of the Curry-Howard isomorphism between types and lambda terms, and of the logically derivability relation between types. We have used a type logical grammar to provide evidence to the semantic theory on questions proposed in [13]. A question for further investigation is whether the distinction on three sentential levels, is purely syntactic or whether there are some semantic motivations behind it.

Acknowledgements

We like to thank Maria Aloni for having pointed out the connection of our work with the categorial approach to questions, and the anonymous referees for their detailed

comments and suggestions.

References

[1] K. Ajdukiewicz. Die syntaktische konnexität. *Studia Philosophica*, 1:1–27, 1935.

[2] J. Barwise and R. Cooper. Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4, 1980.

[3] F. Beghelli and T. Stowell. Distributivity and negation: The syntax of each and every. In A. Szabolcsi, editor, *Ways of Scope Taking*, pages 72–107. Kluwer, 1997.

[4] J. van Benthem. The Lambek calculus. In E. Bach R. Oehrle and D. Wheeler, editors, *Categorial Grammars and Natural Language Structures*, pages 35–68. Dordrecht: Reidel Publishing Company, 1988.

[5] R. Bernardi. *Reasoning with Polarity in Categorial Type Logic.* PhD thesis, UiL OTS, Utrecht University, 2002.

[6] B. Carpenter. *Type-Logical Semantics.* MIT Press, Cambridge MA, 1996.

[7] J. Groenendijk and M. Stokhof. Questions. In J. van Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*, pages 1055–1124. Elsevier, MIT Press, 1997.

[8] P. de Groote. The Non-associative Lambek calculus With Product in Polynomial Time In N. V. Murray, editor, *Automated Reasoning With Analytic Tableaux and Related Methods*, pages 128–139. Springer Lecture Notes in Artificial Intelligence 1617, 1999.

[9] H. Hendriks. *Studied Flexibility. Categories and Types in Syntax and Semantics.* PhD thesis, ILLC, University of Amsterdam, 1993.

[10] Hiz, editor. *Questions.* Dordrecht: Reidel, 1978.

[11] W. A. Howard. The formulae-as-types notion of construction. In *To H. B. Curry: essays on combinatory logic, lambda calculus and formalism*, pages 480–490. Academic Press, London, 1980.

[12] N. Kurtonina and M. Moortgat. Structural control. In P. Blackburn and M. de Rijke, editors, *Logic, Structures and Syntax.* Kluwer, Dordrecht, 1995.

[13] M. Krifka. For a structured account of questions and answers. In C. Smith, editor, *Proceedings to Workshop on Spoken and Written Text.* University of Texas at Austin, 1999.

[14] W. Ladusaw. *Polarity Sensitivity as Inherent Scope Relations.* PhD thesis, University of Texas at Austin, 1979.

[15] J. Lambek. The mathematics of sentence structure. *American Mathematical Monthly*, 65:154–170, 1958.

[16] R. May. *The Grammar of Quantification.* PhD thesis, MIT, 1977.

[17] M. Moortgat. Generalized quantification and discontinuous type constructors. In W. Sijtsma and A. van Horck, editors, *Discontinuous Constituency.* De Gruyter, 1991.

[18] M. Moortgat. In situ binding: A modal analysis. In P. Dekker and M. Stokhof, editors, *Proceedings of the Tenth Amsterdam Colloqium*, pages 539–549. ILLC, 1995.

[19] M. Moortgat. Categorial Type Logics. In J. van Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*, pages 93–178. The MIT Press, Cambridge, Massachusetts, Cambridge, 1997.

[20] R. Moot. *Proof Nets for Linguistics Analysis.* PhD thesis, UiL OTS, Utrecht University, 2002.

[21] G. Morrill. *Type Logical Grammar.* Dordrecht:Kluwer, 1994.

[22] R. Thomason, editor. *Formal Philosophy: Selected papers of Richard Montague.* Yale University Press, New Haven, 1974.

[23] A. Stechow. Topic, Focus and Local Relevance. In W. Klein and W. Levelt, editors, *Crossing the Boundries in Linguistics.* Dordrecht: Reidel, 1981.

[24] A. Szabolcsi and F. Zwarts. Weak islands and a algebraic semantics for scope taking. In A. Szabolcsi, editor, *Ways of Scope Taking*, pages 217–262. Kluwer, 1997.

Received 16 January 2001