FLEXIBLE MONTAGUE GRAMMAR

Herman Hendriks

ITLI Prepublication Series
for Logic, Semantics and Philosophy of Language  LP-90-08

University of Amsterdam
FLEXIBLE MONTAGUE GRAMMAR

Herman Hendriks
Department of Philosophy
University of Amsterdam

Received July 1990

ITLI Prepublications
for Logic, Semantics and Philosophy of Language
ISSN 0924-2082

Supported by NWO (Netherlands Organization for Scientific Research)
Supported by Dyana, Project EBRA-3715
HERMAN HENDRIKS

FLEXIBLE MONTAGUE GRAMMAR*

0. INTRODUCTION

In this paper we shall argue that adoption of flexible type assignment in Montague grammar leads to a more adequate division of labour between the syntactic and semantic component. In syntax, things can be simplified to a great extent: it becomes possible to account for quantifier scope ambiguities without Montagovian rules of quantification or Cooper stores. Moreover, flexible type assignment enables us to generate semantic interpretations that cannot be straightforwardly represented within those approaches. And finally, the flexible approach is more successful in cases where quantification rules and stores lead to wrong results.

In section 1, ‘The Semantics of PTQ’, we shall first review several relevant properties of orthodox Montague grammar. Next, section 2, ‘Arguments for flexibility’, lists arguments that have been put forward in favour of a less rigid category-to-type assignment in model-theoretic semantics. The consequences of the apparently needed flexibility for the design of the grammar are treated in section 3, ‘Flexibility in syntax and semantics’. We shall first consider the Lambek calculus, an influential theory of syntactic/semantic flexibility, and then present our flexible Montague grammar. Section 4, ‘Applications’, shows that the flexible grammar defined in section 3 not only has the wished-for advantages, but allows one to dispense with special syntactic devices for the representation of quantifier scope ambiguities. In the last section, ‘Discussion: compositionality and contextuality’, we pay attention to the position of the flexible grammar relative to the principles of compositionality and contextuality. The semantic properties of the proposed fragment are treated in an Appendix.

1. PTQ

In his paper ‘The Proper Treatment of Quantification in Ordinary English’ (i.e., Montague 1974b, PTQ henceforth), Richard Montague presented a grammar for a fragment of English which distinguishes itself from its generative predecessors by the presence of a semantic component where the expressions defined by the syntactic component are translated into expressions of the logical language IL (= Intensional

* I would like to thank Renate Bartsch, Johan van Benthem, Paul Dekker, Jeroen Groenendijk, Theo Janssen, Reinhard Muskens and Martin Stokhof for their help, encouragement and/or criticism. The research for this paper was supported by the Netherlands Organization for Scientific Research (NWO) and ESPRIT Basic Research Action 3175, Dynamic Interpretation of Natural Language (DYANA).
Logic). These IL-expressions receive a model-theoretic interpretation in the usual way. Thus the English expressions are indirectly assigned a semantic interpretation, viz., via the interpretation of the logical expressions they are translated into.\(^1\) The ‘PTQ fragment’ has since acquired a paradigmatic status within model-theoretic semantics. In this section we shall go into a number of aspects of the fragment, which will be considered with its original semantics (plus individual concepts – cf. Dowty, Wall and Peters (1981), Appendix III).

1.1 Categories

The set of syntactic categories in the PTQ fragment is defined inductively. There are two basic categories, S and E. S is the category of ‘sentences’, and E is an empty category in the sense that no expression belongs to it. Furthermore, there are compound categories: if A and B are categories, then A/B and A\(\backslash\)B are also categories. Intuitively, A/B and A\(\backslash\)B represent a category of expressions that take an expression of category B to form an expression of category A. As a consequence of this definition, we get an infinitely large set of categories at our disposal. In practice we shall only have to deal with a finite subset of the defined categories: the eight categories indicated below.

<table>
<thead>
<tr>
<th>CATEGORY</th>
<th>DESCRIPTION</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>sentence</td>
<td>–</td>
</tr>
<tr>
<td>E</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>CN</td>
<td>common noun</td>
<td>S/E</td>
</tr>
<tr>
<td>IV</td>
<td>intransitive verb phrase</td>
<td>S/E</td>
</tr>
<tr>
<td>T</td>
<td>term (noun) phrase</td>
<td>S/IV = S/(S/E)</td>
</tr>
<tr>
<td>Det</td>
<td>determiner</td>
<td>T/CN = (S/(S/E))/(S/E)</td>
</tr>
<tr>
<td>TV</td>
<td>transitive verb phrase</td>
<td>IV/T = (S/E)/(S/(S/E))</td>
</tr>
<tr>
<td>PV</td>
<td>propositional verb phrase</td>
<td>IV/S = (S/E)/S</td>
</tr>
</tbody>
</table>

With the exception of S and E, all categories contain lexical elements. (The term phrases \([\text{the}_0], [\text{the}_1], [\text{the}_2], \ldots\) are called syntactic variables.)

\(^{1}\) Cf. Janssen (1983) for a clear exposition of this framework; less clear, more concise: Montague (1974a).
1.2 Types

There is a rigid correspondence between syntactic categories and semantic types in PTQ. I.e., if an expression, \( \alpha \), belongs to a certain category, \( C \), then its IL-translation, \( \alpha' \), is invariably of the unique type assigned to \( C \): \( \text{TYPE}(C) \). Put differently, the category-to-type assignment in PTQ, \( \text{TYPE} \), is a function. This function is defined as follows: \( \text{TYPE}(S) = t \), \( \text{TYPE}(E) = e \), and \( \text{TYPE}(A/B) = \text{TYPE}(A/B) = ((s,\text{TYPE}(B)),\text{TYPE}(A)) \). For our eight categories this amounts to:

<table>
<thead>
<tr>
<th>CATEGORY</th>
<th>TYPE(CATEGORY)</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>t</td>
</tr>
<tr>
<td>E</td>
<td>e</td>
</tr>
<tr>
<td>CN</td>
<td>((s,e),t)</td>
</tr>
<tr>
<td>IV</td>
<td>((s,e),t)</td>
</tr>
<tr>
<td>T</td>
<td>((s,((s,e),t)),t)</td>
</tr>
<tr>
<td>Det</td>
<td>((s,((s,e),t)),((s,((s,e),t)),t))</td>
</tr>
<tr>
<td>TV</td>
<td>((s,((s,((s,e),t)),t)),((s,e),t))</td>
</tr>
<tr>
<td>PV</td>
<td>((s,t),((s,e),t))</td>
</tr>
</tbody>
</table>

The above category-to-type assignment is of semantic importance, since it determines for each expression what kind of object in the model is assigned to that expression as its denotation: type \( t \) represents the set of truth values, \( e \) the set of entities, and \( s \) (which is not really a basic type in IL) the set of possible worlds. A type \((c,d)\) stands for the set of functions from objects of type \( c \) to objects of type \( d \). Whenever \( b \) equals \( t \), we can identify such a function of type \((c,b) = (c,t)\) with a set of objects of type \( c \). And when the type is of form \((c,(d,t))\), we can identify the objects in its denotation with sets of ordered pairs \( <\gamma,\delta> \), where \( \gamma \) is an object of type \( c \), and \( \delta \) is an object of type \( d \); hence with binary relations between objects of type \( a \) and objects of type \( b \).

By stipulating that \( \text{TYPE}(S) \) equals \( t \), we establish that the denotation of a sentence is a truth value. If there were expressions of type \( E \), their denotation would be an entity. The denotation of common nouns and intransitive verb phrases is a set of individual concepts. (An individual concept is a function from possible worlds to entities.) Just imagine: an object of type \((s,e),t\) is a function from objects of type \( s,e \) to objects of type \( t \) – a function that can be identified with a set of objects of type \( s,e \), i.e., a set of functions from possible worlds to entities, or a set of individual concepts. The denotation of a noun phrase is a set of properties of individual concepts. (A property of objects of type \( c \) is an object of type \((s,(c,t))\); a function from possible worlds to sets of objects of type \( c \).) A determiner denotes a binary relation between properties of individual concepts, a transitive verb phrase denotes a binary relation between properties of properties of individual concepts and individual concepts, and a propositional verb phrase denotes a
binary relation between propositions and individual concepts. (A proposition is a function from possible worlds to truth values.)

The rigid correspondence between categories and types has some consequences. Note, for instance, that a straightforward option for the denotation of a proper name like John, namely the entity John (to be denoted by an individual constant j of type e in IL), is excluded. For the category John belongs to (the category of noun phrases) is associated with type ((s,((s,e),t)),t) by the TYPE function, and not with e. Therefore, we are obliged to translate John into an expression of the former type. Montague chooses \( \lambda P.\langle P(j) \rangle \), an expression referring to the set of properties of individual concepts such that its extension contains the intension of the denotation of j as its element, and this solution turns out to work. Montague's solution is practically inevitable under the assumptions of the PTQ framework. One could, e.g., think of a different category assignment to proper names (E comes to mind), but this fails to do justice to the reality of syntax: proper names and compound noun phrases like a woman have a very similar distribution; they can be coordinated (John and a woman); etc. Such an assignment would, moreover, complicate the grammar to a great extent: every rule which applies to both proper names and compound noun phrases would have to be doubled.

1.3 Translation rules

The rules of the PTQ grammar consist of a syntactic and a semantic half. The syntactic rules give a recursive definition of the set of expressions that constitute the fragment of English. We shall ignore the details of syntactic analysis here.

The semantic translation rules determine (i) what is the interpretation (strictly speaking: the IL-translation) of the lexical expressions; and (ii) what is the interpretation of compound expressions, given the interpretations of the immediate constituent expressions. Let us look at some examples.

1.3.1 Lexical expressions.

PTQ translates common nouns into non-logical constants of type ((s,e),t). The translation of temperature, for instance, is TEMPERATURE. This IL expression denotes some set of individual concepts. Translations like these appear to be adequate for common nouns like temperature, percentage, etc. (cf. Janssen 1984 and Gamut 1982, section 7.4.1, for arguments), but not for 'normal' common nouns like man, woman, unicorn which denote sets of individuals. That is why Montague supplements his translations of these common nouns as non-logical constants of type ((s,e),t) with a meaning postulate (MP 2 in PTQ). We opt here for a slightly different approach, and incorporate the meaning postulates in the lexical translations (ignoring Montague's \*-notation):
(4) [\text{\text{C}N\text{man}}] \quad \lambda x. [\text{MAN}(\gamma x) \land \exists u (x = \gamma u)]

[\text{\text{C}N\text{woman}}] \quad \lambda x. [\text{WOMAN}(\gamma x) \land \exists u (x = \gamma u)]

[\text{\text{C}N\text{unicorn}}] \quad \lambda x. [\text{UNICORN}(\gamma x) \land \exists u (x = \gamma u)]

[\text{\text{C}N\text{temperature}}] \quad \text{TEMPERATURE}

In doing so, we account for the fact that these ‘normal’ common nouns denote sets of constant individual concepts (that is, ‘in fact’ sets of individuals). The remaining lexical expressions get the following translations:

(5) [\text{TV\text{\text{w}alk}}] \quad \lambda x. \text{WALK}(\gamma x)

[\text{TV\text{\text{r}ise}}] \quad \text{RISE}

[\text{TV\text{\text{J}ohn}}] \quad \lambda P. \forall P(\gamma j)

[\text{TV\text{\text{M}ary}}] \quad \lambda P. \forall P(\gamma m)

[\text{TV\text{\text{t}he\text{i}}}] \quad \lambda P. \forall P(\gamma i)

[\text{Det\text{\text{\a}}}] \quad \lambda P \alpha Q. \exists x [\forall P(x) \land \forall Q(x)]

[\text{Det\text{\text{\e}very}}} \quad \lambda P \alpha Q. \forall x [\forall P(x) \rightarrow \forall Q(x)]

[\text{Det\text{\text{\t}he}}} \quad \lambda P \alpha Q. \exists x [\forall y [\forall P(y) \leftrightarrow x = y] \land \forall Q(x)]

[\text{Det\text{\text{\on\e}}} \quad \lambda P \alpha Q. \exists x \forall y [\forall P(y) \land \forall Q(y)] \leftrightarrow x = y]

[\text{TV\text{\text{\se}ek}}} \quad \lambda T \alpha x. \text{SEEK}(T)(\gamma x)

[\text{TV\text{\text{\f}ind}}] \quad \lambda T \alpha x. \forall \gamma y. \text{FIND}(\gamma y)(\gamma x)

[\text{TV\text{\text{\c}laim\text{\text{\t}hat}}} \quad \lambda P \alpha x. \text{CLAIM}(p)(\gamma x)

[\text{TV\text{\text{\b}elieve\text{\text{\t}hat}}} \quad \lambda P \alpha x. \text{BELIEVE}(p)(\gamma x)

Montague’s meaning postulate 3 is hidden in the translation of \text{\text{w}alk}. This intransitive verb holds of the \text{\text{v}alue} of the individual concepts that constitute its denotation: the subject position of verbs like these is extensional. The meaning postulate does not hold for \text{\text{r}ise}; and so this verb is simply translated as a non-logical constant of type \text{((s,e),t)}. In the translations of \text{\text{\se}ek}, \text{\text{f}ind}, \text{\text{\c}laim\text{\text{\t}hat} and \text{\text{b}elieve\text{\text{\t}hat}} we have incorporated the PTQ meaning postulates 5 and 6, respectively, accounting for the extensionality of the subject position of TVs and PVs. In the case of \text{\text{f}ind} the object position is extensional as well, which explains why the translation of this verb also shows traces of Montague’s meaning postulate 4.

1.3.2 Compound expressions

PTQ contains a number of rules for the translation of compound expressions that belong to the categories mentioned in section 1.1 above. (Because we do not consider relative clauses, the rule for the formation of compound CNs is omitted here. The CN rule of quantification is absent for the same reason.) \alpha' and \beta' denote the translations of \alpha and \beta, respectively.
(A) Rules of application

(I) \( \alpha_T + \beta_{IV} \rightarrow [s \alpha_T \beta_{IV}] \) Translation: \( \alpha'(\wedge \beta') \)

(II) \( \alpha_{TV} + \beta_T \rightarrow [IV \alpha_{TV} \beta_T] \) Translation: \( \alpha'(\wedge \beta') \)

(III) \( \alpha_{Det} + \beta_{CN} \rightarrow [T \alpha_{Det} \beta_{CN}] \) Translation: \( \alpha'(\wedge \beta') \)

(IV) \( \alpha_{PV} + \beta_S \rightarrow [IV \alpha_{PV} \beta_S] \) Translation: \( \alpha'(\wedge \beta') \)

(B) Rules of conjunction and disjunction.

(V) \( \alpha_S + \beta_S \rightarrow [s \alpha_S \text{ and } \beta_S] \) Translation: \( [\alpha' \wedge \beta'] \)

(VI) \( \alpha_S + \beta_S \rightarrow [s \alpha_S \text{ or } \beta_S] \) Translation: \( [\alpha' \lor \beta'] \)

(VII) \( \alpha_{IV} + \beta_{IV} \rightarrow [IV \alpha_{IV} \text{ and } \beta_{IV}] \) Translation: \( \lambda x. [\alpha'(x) \wedge \beta'(x)] \)

(VIII) \( \alpha_{IV} + \beta_{IV} \rightarrow [IV \alpha_{IV} \text{ or } \beta_{IV}] \) Translation: \( \lambda x. [\alpha'(x) \lor \beta'(x)] \)

(IX) \( \alpha_T + \beta_T \rightarrow [T \alpha_T \text{ and } \beta_T] \) Translation: \( \lambda P. [\alpha'(P) \wedge \beta'(P)] \)

(X) \( \alpha_T + \beta_T \rightarrow [T \alpha_T \text{ or } \beta_T] \) Translation: \( \lambda P. [\alpha'(P) \lor \beta'(P)] \)

(C) Rules of quantification (for \( n \in \{0,1,2,\ldots\} \)).

(XI), \( n \) \( \alpha_T + \beta_S \rightarrow \beta_S^* \) Translation: \( \alpha' (\lambda x_n \beta') \)

(XII), \( n \) \( \alpha_T + \beta_{IV} \rightarrow \beta_{IV}^* \) Translation: \( \lambda y. \alpha' (\lambda x_n [\beta'(y)]) \)

In the rule schemes (XI), \( n \) and (XII), \( n \), \( \beta_S^* \) and \( \beta_{IV}^* \) stand for \( \beta_S \) and \( \beta_{IV} \), respectively, with (i) if \( \alpha_T \) is a syntactic variable, all occurrences of \( he_n \) replaced by \( \alpha_T \); and (ii) if \( \alpha_T \) is not a syntactic variable, the first occurrence of \( he_n \) replaced by \( \alpha_T \), and the remaining occurrences of \( he_n \) replaced by anaphoric pronouns of appropriate case and gender.

1.3.3 Scope ambiguities

The rules (XI), \( n \) and (XII), \( n \) are used to represent quantifier scope and de dicto/de re ambiguities. E.g., in constructing the sentence \( \text{Every man finds a woman} \) we can only apply the rules of application (I), (II) and (III), which yields the wide scope-every translation (6). But we can also construct \( \text{Every man finds him}_n \) first, which translates as (7), and then apply (XI),8 (an instantiation of the rule scheme (XII), \( n \)) to \( a \text{ woman} \) and \( \text{Every man finds him}_n \). In this way we get the wide scope-a translation (8).

(6) \( \forall u [\text{MAN}(u) \rightarrow \exists v [\text{WOMAN}(v) \wedge \text{FIND}(v)(u)]] \)

(7) \( \lambda x_n. \forall u [\text{MAN}(u) \rightarrow \text{FIND}('x_n')(u)] \)

(8) \( \exists v [\text{WOMAN}(v) \wedge \forall u [\text{MAN}(u) \rightarrow \text{FIND}(v)(u)]] \)

Similarly, sentence (9) is assigned three readings by the above semantics.

(9) \( [s \text{ Mary}_T \ [IV \text{ walks}_{IV} \text{ and } [IV \text{ seeks}_{IV} [T \text{ every}_{Det} \text{ unicorn}_{CN} \text{ and } IV \text{ is }])}] \)

The \( \text{de dicto} \) reading (10) can be built up without rules of quantification. Apart from conjunction rule (VII), only rules of applications are used in its construction.
(10) \text{WALK}(m) \land \text{SEEK}(\forall v[\text{UNICORN}(v) \rightarrow \forall Q(\Delta v)](m))

Within the PTQ framework, sentence (9) is assigned two more translations, viz., (11) and (12), in which the quantified noun phrase \textit{every unicorn} is read \textit{de re}. In order to get these translations, we have to make use of the rule schemes of quantification (XII),n and (XI),n, respectively.

(11) \text{WALK}(m) \land \forall v[\text{UNICORN}(v) \rightarrow \text{SEEK}(\forall \lambda P. \forall P(v)(m))]
(12) \forall v[\text{UNICORN}(v) \rightarrow \text{WALK}(m) \land \text{SEEK}(\forall \lambda P. \forall P(v)(m))]

Both translations require that we first build up the verb phrase \textit{seek him}, with some syntactic variable occupying the position of the quantified direct object.

To get (11), we first apply (XII),i, quantifying \textit{every unicorn} into the verb phrase \textit{seek him}, with the result \textit{seek every unicorn}. Then (VII) is used to form \textit{walk and seek every unicorn}, which is finally combined with \textit{Mary}.

To get (12), we first conjoin \textit{seek him} with \textit{walk}; then we use (I), which gets us the sentence \textit{Mary walks and seeks him}; and finally we apply (XI),i, quantifying-in \textit{every unicorn} into this sentence.

Note, however, that whereas (11) is okay, (12) is an incorrect translation of sentence (9), witness the equivalence of the following formulas:

(13) \forall v[\text{UNICORN}(v) \rightarrow \text{WALK}(m) \land \text{SEEK}(\forall \lambda P. \forall P(v)(m))] \iff
    \forall v[[\text{UNICORN}(v) \rightarrow \text{WALK}(m)] \land [\text{UNICORN}(v) \rightarrow \text{SEEK}(\forall \lambda P. \forall P(v)(m))]] \iff
    \forall v[\text{UNICORN}(v) \rightarrow \text{WALK}(m)] \land \forall v[\text{UNICORN}(v) \rightarrow \text{SEEK}(\forall \lambda P. \forall P(v)(m))] \iff
    [\exists v \text{UNICORN}(v) \rightarrow \text{WALK}(m)] \land \forall v[\text{UNICORN}(v) \rightarrow \text{SEEK}(\forall \lambda P. \forall P(v)(m))]

(12) apparently means: Mary walks if there are unicorns, and for every unicorn it holds that Mary seeks it. A truth in every possible world without unicorns, whether or not Mary walks. This is wrong: sentence (9) does not express that Mary’s walking is dependent on the existence of unicorns.

An analogous problem arises with (14).\textsuperscript{2} This sentence is assigned two readings. The correct reading (15) is obtained without rules of quantification. But using (XI),n, we arrive at the obvious non-reading (16).

(14) John runs and no unicorn walks
(15) \text{RUN}(j) \land \neg \exists v[\text{UNICORN}(v) \land \text{WALK}(v)]
(16) \neg \exists v[\text{UNICORN}(v) \land \text{WALK}(v)] \iff
    \neg \text{RUN}(j) \lor \neg \exists v[\text{UNICORN}(v) \land \text{WALK}(v)]

\textsuperscript{2} Paul Dekker (p.c.)
In the sequel we shall see that the flexible Montague grammar defined in section 3 assigns only the correct readings in these cases.

2. ARGUMENTS FOR FLEXIBILITY

We saw in section 1.1 above that, due to the rigid category-to-type assignment in PTQ, a proper name like John is not translated as a non-logical constant of type e, but as a complex IL-expression of type ((s,((s,e),t)),t): λP.∀P(^j). This, in its turn, has further consequences: an extensional transitive verb like find, for example, is translated as λTL×.∀T(∀λy.FIND(^y)(^x)), of type TYPE(TV) = ((s,((s,((s,e),t)),t)),t), and not as λyλx.FIND(^y)(^x)) of type ((s,e),((s,e),t)), which would be the most direct way to obtain the translation FIND(m)(j) for John finds Mary under the assumptions that (i) proper names have denotations of type e, and that (ii) the rules of application (I) and (II) translate as α'(α'β'). The rigid category-to-type assignment entails a considerable distortion of the semantics, a distortion which, for that matter, has not yet reached its full extent with PTQ, as Bach (1980) argues. For instance, there are verbs whose subject position is intensional, cf. (17) and (18).

(17) A man is missing
(18) A unicorn seems to be approaching

In BE-MISSING(∀λQ.∃v[MAN(v)∧Q(v)]), the non-referential reading of sentence (17), the translation of the intransitive verb phrase, BE-MISSING, is not of TYPE(IV) = ((s,e),t), but of type ((s,TYPE(NP)),t) = ((s,((s,((s,e),t)),t),t). Mutatis mutandis, the same holds for (18). The existence of intransitive verb phrases like being missing entails within the PTQ framework a raising of the type assignment (plus a change in the definition) of a number of categories: for all intransitive verbs (hence also for extensional ones like walk) TYPE(IV) should be changed into ((s,((s,((s,e),t)),t),t); without exception, all transitive verbs must be assigned a new TYPE(TV): ((s,((s,((s,e),t)),t)),((s,((s,((s,e),t)),t)),t)) (while no transitive verb with this amount of intensionality is known, cf. Partee and Rooth: ‘there are [apparently] no basic lexical verbs that are intensional with respect to both subject and object’ (1983, p. 379)); and the new TYPE(PV) for all PVs becomes ((s,t),((s,((s,e),t)),t),t)). The resulting category-to-type assignment is the one found in Montague’s ‘Universal grammar’ (= UG, Montague (1974a)), plus individual concepts.

In itself, such a complication is not a disaster, provided that it serves the empirical adequacy of the theory. But there are reasons to assume that Montague’s strategy of ‘generalizing to the worst case’ – uniformly assign all members of a certain syntactic category the ‘highest’ type that is needed for some expression in that category – is not the one to be pursued. This has been claimed by Van Benthem (1986), Groenendijk and

(19) John seeks a fish or a bike
(20) \text{SEEK}(\lambda P. [\exists v FISH(v) \land \neg P(\neg v)] \lor \exists v [BIKE(v) \land \neg P(\neg v)])(j)
(21) \exists v [FISH(v) \land \text{SEEK}(\lambda P. \neg P(\neg v)](j)] \lor \exists v [BIKE(v) \land \text{SEEK}(\lambda P. \neg P(\neg v)](j)]

This sentence has a \textit{de dicto} and a \textit{de re} reading to start with. The \textit{de dicto} reading, (20), can be accounted for in PTQ by applying rule (X) to \textit{a fish and a bike: a fish or a bike}, with the reduced translation (22):

(22) \lambda P. [\exists v FISH(v) \land \neg P(\neg v)] \lor \exists v [BIKE(v) \land \neg P(\neg v)]

If we subsequently combine (22) with the translations of \textit{seek} (rule (II)) and \textit{John} (rule (I)), the result reduces to (20). (21) is also among the possibilities of PTQ. For this reading we first build up \textit{seek him}; (with rule (II)). Rule (I) combines this intransitive verb phrase with the term \textit{John}. The resulting ‘sentence’ is \textit{John seeks him}. With rule (X), i we then quantify the disjunctive noun phrase \textit{a fish or a bike} into \textit{John seeks him}:

(23) \lambda P. [\exists v FISH(v) \land \neg P(\neg v)] \lor \exists v [BIKE(v) \land \neg P(\neg v)] (\lambda \xi. \text{SEEK}(\lambda P. \neg P(\neg v)](j))

(23) \iff (21). But (20) and (21) do not exhaust the possibilities of (19). As Partee and Rooth show, there is yet a third reading of the sentence: the so-called \textit{de dicto} wide scope-or reading, which is suggested by the continuation ‘... but I don’t know which’, and which can be formulated in IL as:

(24) \text{SEEK}(\lambda P. \exists v FISH(v) \land \neg P(\neg v)])(j) \lor \text{SEEK}(\lambda P. \exists v [BIKE(v) \land \neg P(\neg v)])(j)

PTQ does not account for (24). Partee and Rooth indicate how this reading could be constructed (we slightly adapt their types): suppose that every expression \gamma with translation \gamma' of type c also has a translation of type ((s, ((s, (s, (s, e), t)), t)), ((s, e), t)), ((s, e), t), (s, e), t)):

(25) \lambda Q((s, ((s, ((s, (s, (s, e), t)), t)), ((s, e), t)), ((s, e), t)), ((s, e), t)), ((s, e), t)) \text{SEEK}(\lambda P. \exists v [FISH(v) \land \neg P(\neg v)]) and
\lambda Q((s, ((s, ((s, (s, (s, e), t)), t)), ((s, e), t)), ((s, e), t)), ((s, e), t)), ((s, e), t)) \text{SEEK}(\lambda P. \exists v [BIKE(v) \land \neg P(\neg v)])

and suppose, moreover, that the combination of these expressions into \textit{a fish or a bike} follows the general syntactic/semantic scheme of \textit{generalized disjunction}³, then \textit{a fish or a bike} is assigned translation (26) of type ((s, ((s, (s, (s, (s, e), t)), t)), ((s, e), t)), ((s, e), t), (s, e), t)), ((s, e), t)):

³ Defined in Gazdar (1980), Keenan and Faltz (1985), Partee and Rooth (1983); cf. below, section 3.2.1.2.
(26) $\lambda Q \lambda y. [\neg Q(\lambda P. \exists v [FISH(v) \land P(\neg v)])(y) \lor \neg Q(\lambda P. \exists v [BIKE(v) \land P(\neg v)])(y)]$

The application of (26) to $\lambda T \lambda x. SEEK(T)(\forall x)$ reduces to (27). And the application of $\lambda P. \forall P(\forall y)$ to the intension of (27) yields (24).

(27) $\lambda y. [SEEK(\lambda P. \exists v [FISH(v) \land P(\neg v)])(y) \lor SEEK(\lambda P. \exists v [BIKE(v) \land P(\neg v)])(y)]$

In consideration of such examples, Partee and Rooth conclude that we should give up Montague's strategy — uniformly assign all members of a certain syntactic category the 'highest' type that is needed for some expression in that category. Apparently, there is not always a 'highest' type available, or a worst case to generalize to: in the case of the de dicto wide scope-or reading of (19) for example, we need to have the type $((s, ((a, ((s, ((s, ((s, (e, t), t)), ((s, (e, t))), ((s, (e, t)))), ((s, (e, t), t)))))), ((s, (e, t), t))))$ for noun phrases; and for the de dicto wide scope-or readings of the sentences in (28) that express an uncertainty of the speaker (cf. the continuations ‘… but I don’t know exactly what she claims’)4, we need to assign ever higher types to the quantified noun phrase a fish or a bike.

(28) Wanda claims that John seeks a fish or a bike
    Wanda claims that Jim knows that John seeks a fish or a bike
    Wanda claims that Jim knows that Jules knows that Jim knows that John seeks a fish or a bike
    Wanda claims that Jules knows that Jim knows that Jules knows that Jim knows that John seeks a fish or a bike
    ...

Partee and Rooth propose a 'reverse' strategy instead: abandon the uniform and rigid category-to-type assignment; give every expression in the lexicon a basic translation of the minimal type available for that expression (extensional transitive verbs like find and intensional ones like seek thus get different minimal types; the lexical type of the term John is 'lower' than the lexical type of a woman); and derive the translations of higher types from the lexical translation by general rules. (A different argumentation for this 'reverse' strategy is given by Groenendijk and Stokhof (1984, 1987). They show that certain intuitive entailment relations exist in lower types, but are absent from the level of the 'highest type', which is also needed.) This 'reverse' strategy opens up interesting perspectives: cf. sentence (29) with its two readings (30) and (31).

(29) John caught and ate a fish
(30) $\exists v [FISH(v) \land CATCH(v)(j)] \land \exists v [FISH(v) \land EAT(v)(j)]$
(31) $\exists v [FISH(v) \land CATCH(v)(j) \land EAT(v)(j)]$

---
4 I.e., the readings in which a disjunction of claims is ascribed to Wanda.
When PTQ is enriched with a rule of generalized conjunction for transitive verbs,

\[(32) \quad \alpha_{TV} + \beta_{TV} \rightarrow [\alpha_{TV} \land \beta_{TV}] \quad \lambda y \lambda x. [\alpha'(y)(x) \land \beta'(y)(x)],\]

we can reach reading (30) by using this rule to make \textit{caught} and \textit{ate}, and combining this further, via rule (II) and (I), with \textit{a fish} and \textit{John}, respectively. Reading (31) comes about by quantifying the noun phrase \textit{a fish} into the sentence \textit{John caught and ate him}, using rule (XI)i.

However, suppose that the transitive verbs \textit{catch} and \textit{eat} have lexical translations

\[(33) \quad \lambda y \lambda x. \text{CATCH}(\langle y \rangle \langle x \rangle) \quad \text{and} \quad \lambda y \lambda x. \text{EAT}(\langle y \rangle \langle x \rangle),\]

of the minimal type \((s,e),((s,e),t))\); and that the translations of the higher type \((s,((s,((s,e),t)),t)),(s,e),t))\) which PTQ assigns to these expressions,

\[(34) \quad \lambda T \lambda x. \langle T)(\langle \lambda y. \text{CATCH}(\langle y \rangle \langle x \rangle)) \quad \text{and} \quad \lambda T \lambda x. \langle T)(\langle \lambda y. \text{EAT}(\langle y \rangle \langle x \rangle)),\]

are derivable from the lexical translations. In that case we can generate both readings without using rules of quantification or syntactic variables. (30) is the result of conjoining the derived translations in (34), (and applying the conjunction to direct object and subject):

\[(35) \quad \lambda T \lambda x. [\langle T)(\langle \lambda y. \text{CATCH}(\langle y \rangle \langle x \rangle) \land \langle T)(\langle \lambda y. \text{EAT}(\langle y \rangle \langle x \rangle))$]

And (31) is reached by conjoining the basic translations in (33):

\[(36) \quad \lambda y \lambda x. [\text{CATCH}(\langle y \rangle \langle x \rangle) \land \text{EAT}(\langle y \rangle \langle x \rangle)],\]

followed by an application of the same rule that also derives (34) from (33), with the following result (which is then applied to direct object and subject):

\[(37) \quad \lambda T \lambda x. \langle T)(\langle \lambda y. [\text{CATCH}(\langle y \rangle \langle x \rangle) \land \text{EAT}(\langle y \rangle \langle x \rangle)]),\]

This suggests that we can put the ‘reverse’ strategy to use in an account of scope ambiguities without syntactic variables or rules of quantification.

Rules of quantification are syntactic rules which have a primarily semantic motivation. They also serve a more syntactic function (the representation of anaphoric pronouns), but the main reason for their introduction is semantic: the representation of quantifier scope ambiguities. From a syntactic point of view this can be seen as an exceptional interference in syntax by the semantic principle of compositionality\(^5\). Another

---

\(^5\) Rules of quantification are the only rules in PTQ that disturb the ‘intuitive’ syntactic constituent structure.
reason for seeking alternatives is the fact that the syntactic operation of these rules (‘if $\alpha_T$ is a syntactic variable, replace all occurrences of $he_n$ by $\alpha_T$; and if $\alpha_T$ is not a syntactic variable, replace the first occurrence of $he_n$ by $\alpha_T$, and the remaining occurrences of $he_n$ by anaphoric pronouns of appropriate case and gender’) is somewhat strange, to put it mildly. In most theories of syntax (including the ones which are attractive from the point of view of compositional semantics, like GPSG\(^6\) or flexible categorial grammar\(^7\)), such syntactic operations are simply not allowed. But also for those who do not feel bound to restrictions on possible syntactic operations, there are reasons to look for alternatives: we saw that they provide a sentence like (9) with reading (12), and Janssen (1983) amply discusses the unwanted consequences of using syntactic variables and rules of quantification (the production of ‘sentences’ like $he_{38}$ walks is one of the more innocent among them), and introduces a metagrammatical ‘variable principle’ in order to control things.

Therefore, in the more or less recent past various attempts have been made to avoid these syntactic devices, while preserving the results of PTQ as regards the representation of quantifier scope ambiguities. However, the main alternative of ‘Cooper storage’ (see Cooper (1983)) met with objections from a semantic viewpoint (Landman and Moerdijk (1983), Janssen (1983)). The present paper can be considered the umpteenth attempt. It must be kept in mind that we shall not deal with the second function of rules of quantification, the representation of anaphoric pronouns. The theory of pronouns in PTQ is syntactically hardly explicit (for instance, it does not distinguish between reflexive and personal pronouns). A successful attempt to make it less implicit has been made by Landman and Moerdijk (1983). But this theory treats anaphoric pronouns without expectation as bound variables in the semantics, just as PTQ does. Reinhart (1983), Kamp (1981), and many others since, have shown that not every occurrence of an anaphoric pronoun can be considered as such: the true theory of anaphoric pronouns will at least have to include a ‘discourse representation theory’. Nonetheless, part of the intrasentential behaviour of terms (including all occurrences of reflexive pronouns) does have to be analyzed in terms of bound variables. Here, for instance, Pollard and Sag’s (1983) theory of reflexives and reciprocals constitutes an alternative which is consistent with the proposals made in the present paper.

3. FLEXIBILITY IN SYNTAX AND SEMANTICS

In this section we shall present a formal elaboration of the ‘reverse’ strategy sketched above. The traditional correspondence between categories and types (which uniformly assigns every expression in a certain category a translation of the ‘highest’ type that is needed for some expression in that category) is given up. Instead, we give every

---

6 See Gazdar, Klein, Pullum and Sag (1985); Klein and Sag (1985).
7 See Lambek (1958); Van Benthem (1986, chapter 7); Moortgat (1988).
expression a translation of its minimal type, and derive the translations of higher (rather: other) types from this lexical translation by general rules, thus obtaining the flexible Montague grammar of section 3.2. But before we have reached that point, we shall, in section 3.1, first explore the Lambek calculus, a flexible theory of syntax which has been provided with a semantics by Van Benthem (1986, chapter 7). It will turn out that this theory is either too weak (in its directional version) or too strong (in its non-directional version) for our purposes. The flexible Montague grammar of section 3.2 is essentially an intensional version of a subsystem of the non-directional Lambek calculus, together with a semantics along the lines of Van Benthem.

3.1 Lambek calculus

In its original (directional) version (Lambek (1958)), the Lambek calculus \( L \) is a theory of syntactic categories. In keeping with the lexicalist tradition of categorial grammar, the combinatorial properties of expressions are determined by the categories they belong to. Restricting ourselves to the product-free part of \( L \), we distinguish atomic categories and two kinds of compound categories: left- \((y/x)\) and right-searching \((x/y)\) functors:

(38) \hspace{1cm} \text{Let } \text{ATOM be a finite set of atomic categories. Then CAT, the set of categories, is the smallest set such that (i) ATOM } \subseteq \text{ CAT, and (ii) if } x \in \text{ CAT and } y \in \text{ CAT, then } x/y \in \text{ CAT and } y/x \in \text{ CAT.}

L consists of a set of axioms and five inference rules. It characterizes the derivability of so-called sequents of categories, cf. (39). In (40) and (41), \( x, y, z \) denote categories and \( T, U, V \) finite sequences of categories (T non-empty).

(39) A sequent is a pair \( <T,x> \), where \( T \) is a finite non-empty sequence of categories \( (T = <x_1, \ldots, x_n>, \text{ with } n \geq 1) \) and \( x \) is a category.
T is called the left-hand side, and \( x \) the right-hand side of \( <T,x> \).
For the sequent \( <<x_1, \ldots, x_n>,x> \) we write \( x_1, \ldots, x_n \Rightarrow x \).

(40) The axioms of \( L \) are all the sequents of the form \( x \Rightarrow x \).

(41) The inference rules of \( L \) are \( /L, \\neg L, /R, \neg R \) and Cut:

\[
\begin{align*}
/L & \quad T \Rightarrow y \quad U, x, V \Rightarrow z \\
\neg L & \quad T \Rightarrow y \quad U, x, V \Rightarrow z \\
/R & \quad T, y \Rightarrow x \quad T / R \quad y, T \Rightarrow x \\
& \quad T \Rightarrow x/y \\
\text{Cut} & \quad T \Rightarrow x \quad U, x, V \Rightarrow y \\
& \quad U, T, V \Rightarrow y
\end{align*}
\]
Lambek proved that Cut is a derived inference rule — the set of theorems is not increased by adding it to the other four inference rules. Note that this entails the decidability of L: for arbitrary sequents \( T \Rightarrow x \), the proof procedure is guaranteed to terminate after finitely many steps with an answer to the question whether the sequent is derivable. For in each of the inference rules /L, \( \\land \), /R and \( \\lor \), the number of occurrences of / and \( \\land \) in the premises is strictly smaller than that in the conclusion (the conclusion contains a / or \( \\land \) that is absent from the premises). Thus, establishing the derivability of the premise sequents is a more simple goal than establishing the derivability of the conclusion, and it follows that every sequent has only finitely many Cut-free derivations. One of the most important features of L as a syntactic theory is that it defines a flexible notion of constituent structure which can account for the grammaticality of sentences containing so-called non-constituent coordination:

(42) John loves and Bill hates Mary

We shall not dwell on this topic (cf., a.o., Moortgat (1988)), but focus on the semantic interpretation of L instead. That there is a strong link between syntax and semantics in L has repeatedly been stressed by Van Benthem (e.g., (1986)), who showed that the semantic interpretation of an L-derivable sequent is directly determined by the proof of its validity in the syntactic calculus: the rules /L and \( \\land \) correspond to functional application, /R and \( \lor \) to lambda abstraction, and Cut to substitution. The best-known explicit elaboration of Van Benthem's ideas can be found in Moortgat (1988). Within this semantics, every category \( x \) is associated with a type: \( \text{TYPE}(x) \). For atomic categories, this type is stipulated, and \( \text{TYPE}(x/y) = \text{TYPE}(y \backslash x) = (\text{TYPE}(y),\text{TYPE}(x)) \). Moreover, all categories in the left- and right-hand side of a derivable sequent \( x_1,\ldots,x_n \Rightarrow x \) are assigned a semantic value, consisting of a lambda term of type \( \text{TYPE}(x_1) \) in a type-theoretic language: \( x_1 : \alpha_1, \ldots, x_n : \alpha_n \Rightarrow x : \alpha \).

In (43), \( w \) represents a variable of type \( \text{TYPE}(y) \); \( \alpha, \beta \) and \( \delta \) are arbitrary lambda terms of type \( \text{TYPE}(y) \), \( (\text{TYPE}(y),\text{TYPE}(x)) \) and \( \text{TYPE}(x) \), respectively. Only the terms of the 'active' categories are indicated. The (invisible) terms of other categories are assumed to be identical in premise and conclusion.8

(43) \[
\begin{array}{c}
\text{AX} & x : \delta \Rightarrow x : \delta \\
\text{L} & T \Rightarrow y : \alpha & U,x : \beta(\alpha),V \Rightarrow z \\
\end{array}
\]

\[
U,x/y : \beta,T,V \Rightarrow z
\]

8 The semantics is kept extensional for expository purposes. But it can be intensionalized easily. Define \( \text{TYPE}(x/y) = \text{TYPE}(y \backslash x) = ((x,\text{TYPE}(y)),\text{TYPE}(x)) \) instead of \( (\text{TYPE}(y),\text{TYPE}(x)) \); the terms assigned to the axioms and the Cut rule remain the same; and the semantic interpretations of /L and /R are changed in the following way (/L and /R are completely analogous):

/L: if \( T \Rightarrow y : \alpha \) and \( U,x : \beta(\alpha),V \Rightarrow z : \gamma \), then \( U,x/y : \beta,T,V \Rightarrow z \)

/R: if \( T,y : \gamma w \Rightarrow x : \alpha \), then \( T \Rightarrow x/y : \lambda w.\alpha \)
\[ \begin{align*}
\forall L & \quad T \Rightarrow y : \alpha & \quad U,x : \beta(\alpha),V \Rightarrow z \\
\quad & \quad U,T,y;x : \beta,V \Rightarrow z \\
/R & \quad T,y : w \Rightarrow x : \delta & \quad \forall R & \quad y,w,T \Rightarrow x : \delta \\
\quad & \quad T \Rightarrow x/y : \lambda w.\delta & \quad T \Rightarrow y/x : \lambda w.\delta \\
\text{Cut} & \quad T \Rightarrow x : \delta & \quad U,x : \delta,V \Rightarrow y \\
\quad & \quad U,T,V \Rightarrow y
\end{align*} \]

It can be shown that if the sequent \( x_1 : \alpha_1, \ldots, x_n : \alpha_n \Rightarrow x : \alpha \) is derivable in L, then there is a Cut-free L-derivation of this sequent \( x_1 : \alpha_1, \ldots, x_n : \alpha_n \Rightarrow x : \beta \) with \( \alpha \Leftrightarrow \beta \) (cf. Hendriks (1989)). Hence we only have to consider the (finitely many) Cut-free derivations to get all semantic interpretations assigned to a sequent.

This immediately entails that under the usual analysis, in which verbs take noun phrases (being of the atomic category np) as arguments, a sentence like (44) is unambiguous within L.

(44) Every man finds a woman

For if every man and a woman belong to the (atomic) category np, and finds is of category (np/s)/np, then there is only one Cut-free derivation of the sequent np, (np/s)/np, np \( \Rightarrow \) s,

(45)

\[ \begin{align*}
np \Rightarrow np & \quad np \Rightarrow np \\
np,s \Rightarrow s & \quad np,s \Rightarrow s \\
np, (np/s)/np, np \Rightarrow s
\end{align*} \]

and (hence) only one interpretation for it.

However, in some cases scope ambiguities can be accounted for in L: when the following (extensional) assignment of categories, types and translations is assumed,

(46) EXPRESSION | CATEGORY | TYPE | TRANSLATION
---|---|---|---
every man | s/(n's) | ((e,t),t) | \( \lambda P. \forall x[MAN(x) \rightarrow P(x)] = \alpha \)
finds | (n's)/n | (e,e(t)) | \( \lambda x \lambda y.\text{FIND}(x)(y) = \beta \)
a woman | (s/n)s | ((e,t),t) | \( \lambda P. \exists y[\text{WOMAN}(y) \land P(y)] = \gamma \)

then sentence (44) is assigned two readings:

(47) Every man finds a woman

(i) \( \forall x[\text{MAN}(x) \rightarrow \exists y[\text{WOMAN}(y) \land \text{FIND}(y)(x)]] \)

(ii) \( \exists y[\text{WOMAN}(y) \land \forall x[\text{MAN}(x) \rightarrow \text{FIND}(y)(x)]] \)

There are six Cut-free derivations of the sequent s/(n's), (n's)/n, (s/n)s \( \Rightarrow \) s. The first one is assigned interpretation (i); the interpretation of all other derivations is equivalent to
(ii). (It is obvious that the set of Cut-free derivations still embodies a great deal of ‘spurious ambiguity’.)

\[
\begin{align*}
(48) & \quad n \Rightarrow n \\
& \quad s \Rightarrow s \\
& \quad n, n s \Rightarrow s \\
& \quad n, (n s)/n, n \Rightarrow s \\
& \quad n, (n s)/n \Rightarrow s/n \\
& \quad (n s)/n, (s/n)s \Rightarrow s/n s \\
& \quad s/(n s), (n s)/n, (s/n)s \Rightarrow s \\
\end{align*}
\]

\[
\begin{align*}
(49) & \quad n \Rightarrow n \\
& \quad n s \Rightarrow n s \\
& \quad n, n s \Rightarrow s \\
& \quad n s \Rightarrow n s \\
& \quad (n s)/n, n \Rightarrow n s \\
& \quad s/(n s), (n s)/n, n \Rightarrow s \\
& \quad s/(n s), (n s)/n \Rightarrow s/n \\
& \quad s/(n s), (n s)/n, (s/n)s \Rightarrow s \\
\end{align*}
\]

\[
\begin{align*}
(50) & \quad n \Rightarrow n \\
& \quad s \Rightarrow s \\
& \quad n, n s \Rightarrow s \\
& \quad n s \Rightarrow n s \\
& \quad (n s)/n, n \Rightarrow n s \\
& \quad s/(n s), (n s)/n, n \Rightarrow s \\
& \quad s/(n s), (n s)/n \Rightarrow s/n \\
& \quad s/(n s), (n s)/n, (s/n)s \Rightarrow s \\
\end{align*}
\]

\[
\begin{align*}
(51) & \quad n \Rightarrow n \\
& \quad s \Rightarrow s \\
& \quad n, n s \Rightarrow s \\
& \quad (n s)/n, n \Rightarrow n s \\
& \quad s/(n s), (n s)/n, n \Rightarrow s \\
& \quad s/(n s), (n s)/n \Rightarrow s/n \\
& \quad s/(n s), (n s)/n, (s/n)s \Rightarrow s \\
\end{align*}
\]

\[
\begin{align*}
(52) & \quad n s \Rightarrow n s \\
& \quad s \Rightarrow s \\
& \quad s/(n s), (n s)/n \Rightarrow s \\
& \quad s/(n s), (n s)/n \Rightarrow s/n \\
& \quad s/(n s), (n s)/n, (s/n)s \Rightarrow s \\
\end{align*}
\]

\[
\begin{align*}
(53) & \quad n \Rightarrow n \\
& \quad s \Rightarrow s \\
& \quad n, n s \Rightarrow s \\
& \quad n s \Rightarrow n s \\
& \quad n s \Rightarrow n s \\
& \quad (n s)/n, n \Rightarrow s \\
& \quad s/(n s), (n s)/n \Rightarrow s \\
& \quad s/(n s), (n s)/n \Rightarrow s/n \\
& \quad s/(n s), (n s)/n, (s/n)s \Rightarrow s \\
\end{align*}
\]

By way of illustration, we show the interpretation of derivations (48) and (49).
\[(54)\]
\[
\begin{array}{c}
  n \Rightarrow n \\
  v \Rightarrow v \\
  w \Rightarrow w \\
  \hline \\
  n, (n/s)/n, \quad \beta \Rightarrow s \\
  v, \beta, w \Rightarrow \beta(w)(v) \\
  \hline \\
  n, (n/s)/n, w \Rightarrow s \\
  v, \beta, w \Rightarrow \beta(w)(v) \\
  \hline \\
  n, (n/s)/n, \beta \Rightarrow s \\
  v, \beta, w \Rightarrow \beta(w)(v) \\
  \hline \\
  \beta \Rightarrow \lambda w. \beta(w)(v) \\
  \hline \\
  \gamma(\lambda w. \beta(w)(v)) \Rightarrow \gamma(\lambda w. \beta(w)(v)) \\
  \hline \\
  \delta(\lambda(\lambda v. \gamma(\lambda w. \beta(w)(v)))) \Rightarrow \delta \\
  \hline \\
  \alpha, \beta, \gamma \Rightarrow \alpha(\lambda v. \gamma(\lambda w. \beta(w)(v))) \\
\end{array}
\]

Note that \(\alpha(\lambda v. \gamma(\lambda w. \beta(w)(v))) \equiv (i)\), and that \(\gamma(\lambda v. \alpha(\beta(v))) \equiv (ii)\).

However, the analysis given in \((46)\) is a bit tricky. It is crucial that the two noun phrases belong to two different categories, \(s/(n/s)\) and \((s/n)s\), and that these categories appear on the left- and the right-handside of the transitive verb, respectively. For none of the following sequents is L-derivable:9

\[(55)\]
\[
\begin{array}{c}
  n \Rightarrow n \\
  v \Rightarrow v \\
  (n/s)/n, \quad (n/s)/n \Rightarrow s \\
  \hline \\
  \beta \Rightarrow \beta(v) \\
  \hline \\
  \alpha(\beta(v)) \Rightarrow \alpha(\beta(v)) \\
  \hline \\
  \alpha, \beta, v \Rightarrow \alpha(\beta(v)) \\
  \hline \\
  \gamma(\lambda v. \alpha(\beta(v))) \Rightarrow \gamma(\lambda v. \alpha(\beta(v))) \\
  \hline \\
  \delta(\lambda(\lambda v. \gamma(\lambda w. \beta(w)(v)))) \Rightarrow \delta \\
  \hline \\
  \alpha, \beta, \gamma \Rightarrow \alpha(\lambda v. \gamma(\lambda w. \beta(w)(v))) \\
\end{array}
\]

An alternative account of the scope ambiguity of \((44)\) within \(L\) assumes that verbs take noun phrases as arguments, and, moreover, that noun phrases are assigned the complex categories \((s/n)s\) and \(s/(n/s)\). The number of interpretations now depends on which of \(s/(n/s)\) or \((s/n)s\) is chosen as the category for the subject noun phrase \((np_1)\) and the object noun phrase \((np_2)\). If \(np_1 = (s/n)s\) and \(np_2 = s/(n/s)\), only one interpretation is predicted. There are two interpretations when \(np_1 = np_2\) (it does not matter whether we take \(s/(n/s)\) or \((s/n)s)\). The maximum is reached with \(np_1\) and \(np_2\) set equal to \(s/(n/s)\) and \((s/n)s\), respectively, since the sequent

\[(57)\]
\[
\begin{array}{c}
  s/(n/s), \quad ((s/(n/s))/s)/(s/n)s, \quad (s/n)s \Rightarrow s \\
\end{array}
\]

9 Precisely this prompted Lambek (1958, § 6) to propose the category \(s/(n/s)\) for subject pronouns and \((s/n)s\) for direct object pronouns, in view of *him likes he.
has twenty five (!) Cut-free derivations, and five potentially different interpretations (we list only the reduced terms assigned to their conclusions):

\[(58) \quad \alpha(\lambda v.\beta(\gamma(\lambda w.w(v))))\]

\[(59) \quad \alpha(\lambda v.\gamma(\lambda x.\beta(\lambda y.y(x)))(\lambda w.w(v)))\]

\[(60) \quad (\beta(\gamma))(\alpha)\]

\[(61) \quad \gamma(\lambda x.\alpha(\lambda v.\beta(\lambda y.y(x)))(\lambda w.w(v)))\]

\[(62) \quad \gamma(\lambda x.\beta(\lambda y.y(x)))(\alpha)\]

When (58) through (62) are applied to the lexical translations in (63),

<table>
<thead>
<tr>
<th>EXPRESSION</th>
<th>CATEGORY</th>
<th>TYPE</th>
<th>TRANSLATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>every man</td>
<td>np_1</td>
<td>((e,t),t)</td>
<td>(\lambda P.\forall x[\text{MAN}(x)\to P(x)] = \alpha)</td>
</tr>
<tr>
<td>finds</td>
<td>(np_1_s)/np_2</td>
<td>(T,(T,t))</td>
<td>(\beta)</td>
</tr>
<tr>
<td>a woman</td>
<td>np_2</td>
<td>((e,t),t)</td>
<td>(\lambda P.\exists y[\text{WOMAN}(y)\land P(y)] = \gamma)</td>
</tr>
</tbody>
</table>

where \(T = ((e,t),t)\) and \(\beta = \lambda T\lambda U.U(\lambda y.T(\lambda x.\text{FIND}(x)(y)))\), interpretation (58), (59) and (60) collapse into reading (47)(i) (subject wide scope), and the remaining (61) and (62) represent object wide scope (47)(ii). \(\beta\), the lexical translation of finds in (63) encodes wide scope for the subject noun phrase, but this is not essential. With lexical wide scope for the direct object noun phrase, (47)(i) would have been expressed by (58) and (59), and (47)(ii) by (60), (61) and (62). Interestingly, this analysis predicts three readings for every man seeks a woman, provided that the ‘intensional’ verb seek is assigned the UG-like lexical translation \(\beta = \lambda T\lambda U.U(\lambda y.\text{SEEK}(T)(y)))\):

\[(64) \quad (i) \quad \forall x[\text{MAN}(x)\to \exists y[\text{WOMAN}(y)\land \text{SEEK}(\lambda P.P(y))(x)]]\]

\[(ii) \quad \exists y[\text{WOMAN}(y)\land \forall x[\text{MAN}(x)\to \text{SEEK}(\lambda P.P(y))(x)]]\]

\[(iii) \quad \forall x[\text{MAN}(x)\to \text{SEEK}(\lambda P.\exists y[\text{WOMAN}(y)\land P(y)])(x)]\]

(58) and (60) lead to (iii), (59) results in (i), and (61) and (62) arrive at (ii).

It seems, then, that an adequate treatment of quantifier scope in the Lambek calculus is within reach. Still, a curious feature of our account is that the ‘internal directionality’ of noun phrases (s/n\_s) versus (s/n)\_s seems to be a non-negligible factor. Word order has an essential influence on the semantic possibilities of the calculus. When we have SVO (or OVS) order, as in (44), things turn out right. But in SOV cases like the embedded

---

10 Since our semantics is extensional, this is not really an intensional translation. But cf. footnote 8.

11 Note that the difference between (58) and (60) (or (61) and (62)) has no effect, because seek is only intensional in object position. In the case of doubly intensional transitive verbs the difference would matter – but in section 2 above we saw that such transitive verbs do not appear to exist. Something like seem-to-seek would be an example. For these hypothetical verbs, all five theoretical scope possibilities would be realized.
clause in the Dutch sentence (65) (assuming that iedere man = np₁, een vrouw = np₂, and zoekt = np₁ \ (np₂ \ s), with respective translations α, γ, and β).

(65) Het is waar dat iedere man een vrouw zoekt

It is true that every man a woman seeks

'It is true that every man seeks a woman'

we get maximally four derivations and two interpretations, (58) and (60). (That is, under optimal conditions: when iedere man (np₁) and een vrouw (np₂) have categories s/(n\s) and (s/n)s₁², respectively.) (59), (61) and (62) are missing.

(66) α(λ.v. (β(γ))(λ.w. w(v))) = (58) [(68), (69) and (70)], and

(67) (β(γ))(α) = (60) [(71)].

(68)

\[ \frac{n \Rightarrow n}{s \Rightarrow s} \]
\[ \frac{n, n\s \Rightarrow s}{n \Rightarrow s/(n\s)} \]
\[ \frac{n \Rightarrow s/(n\s)}{s \Rightarrow s} \]
\[ \frac{n, (s/(n\s))\s \Rightarrow s}{n \Rightarrow (s/(n\s))\s} \]
\[ \frac{(s/(n\s))\s \Rightarrow n\s}{(s/n)\s, (s/(n\s))\s \Rightarrow n\s} \]
\[ \frac{(s/n)\s, ((s/n)\s)\s \Rightarrow n\s}{s/(n\s), (s/n)\s, ((s/n)\s)\s \Rightarrow s} \]

(69)

\[ \frac{n \Rightarrow n}{s \Rightarrow s} \]
\[ \frac{n, n\s \Rightarrow s}{n \Rightarrow s/(n\s)} \]
\[ \frac{n \Rightarrow s/(n\s)}{s \Rightarrow s} \]
\[ \frac{n, (s/(n\s))\s \Rightarrow s}{n \Rightarrow (s/(n\s))\s} \]
\[ \frac{(s/n)\s, ((s/n)\s)\s \Rightarrow s}{s/(n\s), (s/n)\s, ((s/n)\s)\s \Rightarrow s} \]

(70)

\[ \frac{(s/n)\s \Rightarrow (s/n)\s}{s/(n\s) \Rightarrow s/(n\s)} \]
\[ \frac{s/(n\s), (s/n)\s, ((s/n)\s)\s \Rightarrow s}{s/(n\s), (s/n)\s, ((s/n)\s)\s \Rightarrow s} \]

(71)

\[ \frac{n \Rightarrow n}{s \Rightarrow s} \]
\[ \frac{n, n\s \Rightarrow s}{n \Rightarrow s/(n\s)} \]
\[ \frac{n \Rightarrow s/(n\s)}{s \Rightarrow s} \]
\[ \frac{n, (s/(n\s))\s \Rightarrow s}{n \Rightarrow (s/(n\s))\s} \]
\[ \frac{(s/(n\s))\s \Rightarrow n\s}{(s/n)\s \Rightarrow s} \]
\[ \frac{((s/n)\s)\s \Rightarrow n\s}{s/(n\s), (s/n)\s, ((s/n)\s)\s \Rightarrow s} \]

This entails that with β = λTΛU. U(λy. SEEK(T)(y))) as lexical translation of zoekt, only (i) and (iii) are represented. Alternatively, we could also assume the lexical (wide scope object) translation λTΛU. T(λx.U(λy. SEEK(λP.P(x))(y))), but that would leave (iii)

---

The same holds when both subject and object = s/(n\s). Both possibilities with subject = (s/n)\s give rise to just one derivation (and reading).
unaccounted for. The third possibility, to assume the lexical translation \( \beta = \lambda T \lambda U. U(\lambda y.T(\lambda x.\text{SEEK}(T)(y))) \), would only give (i). We are hence forced to adopt two lexical translations and to conclude that "it seems then, that the Lambek calculus will only be able to predict quantifier scope ambiguities in a restricted number of cases" (Bouma (1986), p. 9).

Since it is the directionality of L which thwarts us, it is an obvious strategy to take a closer look at LP (for: Lambek calculus with Permutation, also known as Lambek-Van Benthem calculus), the non-directional version of the calculus which has been studied by Van Benthem (cf., e.g., his (1988)). (72) shows a possible formalization of LP (plus semantics). The two kinds of compound categories in L, the left- and right-searching functors \( x/y \) and \( y/x \), are collapsed into one category \( (y,x) \), a non-directional functor from argument categories \( y \) to value categories \( x \). (Note the direct correspondence between syntactic categories and semantic types here.) In (72), \( T, U, \) and \( V \) are finite sequences of categories, where \( T = T_1, T_2 \) is non-empty. LP preserves the axioms and the Cut rule of L; it contains rules (L and R) for the introduction of the connective (\( , \)) in the left- and the right-hand side of sequents; moreover, a permutation rule, \( P \), is added, which allows one to interchange adjacent categories (every permutation is a composition of interchanges between neighbours).

\[
\begin{align*}
(72) & \quad \text{AX} \quad x : \delta \Rightarrow x : \delta \\
& \quad \text{L} \quad T \Rightarrow y : \alpha \quad \quad U, x : \beta(\alpha), V \Rightarrow z \\
& \quad \quad U, (y,x) : \beta, T, V \Rightarrow z \\
& \quad \quad T_1, y : w, T_2 \Rightarrow x : \delta \\
& \quad \quad P \quad U, y, x, V \Rightarrow z \\
& \quad \quad T \Rightarrow (y,x) : \lambda w. \delta \\
& \quad \quad U, x, y, V \Rightarrow z \\
& \quad \quad \text{Cut} \quad T \Rightarrow x : \gamma \\
& \quad \quad \quad U, x : \gamma, V \Rightarrow y \\
& \quad \quad \quad U, T, V \Rightarrow y
\end{align*}
\]

Conceived of as a syntactic theory, LP is not really interesting, because it has the following property: whenever a sequent \( T \Rightarrow x \) is LP-derivable, then for all permutations \( \pi(T) \) of \( T \), also \( \pi(T) \Rightarrow x \) is LP-derivable. Nevertheless, it might be useful as a theory of semantic type change. We might, for example, restrict ourselves to sequents of categories that are derivable in our favourite syntactic theory (L, say), and let LP take care of their semantic interpretation. More formally:

\[
(73) \quad \text{The compound expression } e_1, \ldots, e_n \text{ is of category } x \text{ and has a translation } \alpha \text{ of type } t \text{ if and only if: (i) for all } 1 \leq i \leq n, \text{ } e_i \text{ is assigned category } x_i \text{ and translation } \alpha_i \text{ of type } t_i \text{ in the lexicon; (ii) } x_1, \ldots, x_n \Rightarrow_L x; \text{ and (iii) } t_1; \alpha_1, \ldots, t_n; \alpha_n \Rightarrow_{LP} t; \alpha.
\]

But now the permuting effects of LP will turn up in the semantics. Cf. (74) and (75). The sequent \( np, (np \backslash s)/np, np \Rightarrow s \) is L-derivable. In LP there are two non-equivalent derivations of \( e, (e, (e, t)), e \Rightarrow t \); (76) and (77) – one too many.
John loves Paris
(75) EXPRESSION CATEGORY TYPE TRANSLATION
John np e j
loves (np's)/np (e,(e,t)) LOVE
Paris np e p

(76) e:p ⇒ e:p
    t:LOVE(j)(p) ⇒ t:LOVE(j)(p)
    e:j ⇒ e:j
    (e,(e,t)):LOVE, e:p ⇒ t:LOVE(j)(p)
    e:j, (e,(e,t)):LOVE, e:p ⇒ t:LOVE(j)(p)
(77) e:j ⇒ e:j
    t:LOVE(p)(j) ⇒ t:LOVE(p)(j)
    e:p ⇒ e:p
    (e,(e,t)):LOVE, e:j ⇒ t:LOVE(p)(j)
    (e,(e,t)):LOVE, e:j, e:p ⇒ t:LOVE(p)(j)
    e:j, (e,(e,t)):LOVE, e:p ⇒ t:LOVE(p)(j)

Moreover, given the assignments in (79) (α = λQλP.∀x[Q(x)→P(x)]), sentence (78) gets two readings of type t, cf. (80) and (81).

(78) Every man walks
(79) EXPRESSION CATEGORY TYPE TRANSLATION
   every np/cn (e,(e,t),(e,(e,t),t)) α
   man cn (e,t) MAN = β
   walks np's (e,t) WALK = γ

(80) (e,t):β ⇒ (e,t):β
    t:α(β)(γ) ⇒ t:α(β)(γ)
    (e,(e,t),((e,(e,t),t))):α(β), (e,t):γ ⇒ t:α(β)(γ)
    (e,(e,t),((e,(e,t),t))):α(β), (e,t):β, (e,t):γ ⇒ t:α(β)(γ) = δ

(81) (e,t):β ⇒ (e,t):β
    t:α(γ)(β) ⇒ t:α(γ)(β)
    (e,(e,t),((e,(e,t),t))):α(γ), (e,t):β ⇒ t:α(γ)(β)
    (e,(e,t),((e,(e,t),t))):α(γ), (e,t):β, (e,t):γ ⇒ t:α(γ)(β) = ε

δ is equivalent to ∀x[MAN(x)→WALK(x)], but ε yields ∀x[WALK(x)→MAN(x)]. Similarly, eight readings are predicted for the sentence Every man finds a woman. They differ as regards the scope of the quantifiers (∃∀ versus ∀∃), but also with respect to the ‘θ-roles’ of the verb and the order of the argument places of the determiner every. (In this particular case the symmetry of the determiner a avoids another multiplication of the number of readings by two.)
In the sequel we shall develop a calculus that is not thwarted by directionality (since it is a theory of semantic – i.e., non-directional – types), but at the same time avoids the permuting effects of LP.\textsuperscript{13}

3.2 Flexible Montague grammar

We now give a formal definition of our flexible semantics. Each syntactic category $C$ is not associated with one type, but with an infinite set of types: the type set of $C$: $\mathcal{T}(C)$. This set consists of a basic type, $t_b(C)$, together with types which are derived from that type by general rules. For the flexible version of the fragment introduced in section 1, we choose the basic types as in (82).

\begin{align*}
(82) \quad C & \quad t_b(C) \\
S & \quad t \\
T & \quad e \\
CN, IV & \quad ((s,e),(t)) \\
TV & \quad ((s,e),((s,e),(t))) \\
PV & \quad ((s),(s,e),(t)) \\
Det & \quad ((s)((s,e),(t)),((s,e),(t)))
\end{align*}

$\mathcal{T}(C)$ is the smallest set such that:

\begin{align*}
(83) \quad & \text{(i) } t_b(C) \in \mathcal{T}(C); \\
& \text{(ii) if } (\vec{a},b) \in \mathcal{T}(C), \text{ then } (\vec{a},((s,(s,b),(t)),t)) \in \mathcal{T}(C) \\
& \quad \text{(cf. value raising } = \text{ (92))}; \\
& \text{(iii) if } (\vec{a},((s,b),(c),(t))) \in \mathcal{T}(C), \text{ then } (\vec{a},((s,(s,b),(t)),(c),(t)),t)) \in \mathcal{T}(C) \\
& \quad \text{(cf. argument raising } = \text{ (93))}; \\
& \text{(iv) if } (\vec{a},((s,(s,(s,b),(t)),(c),(t)))) \in \mathcal{T}(C), \text{ then } (\vec{a},((s,b),(c),(t))) \in \mathcal{T}(C) \\
& \quad \text{(cf. argument lowering } = \text{ (94))}. 
\end{align*}

In this definition, $s$ and $t$ are specific IL types; $b$ is an arbitrary type; and $\vec{a}$ and $\vec{c}$ are sequences of types: $a_1, ..., a_m, c_1, ..., c_n$ with $m$ and $n \geq 0$. $(\vec{a},b)$ represents the type $(a_1, ..., (a_n,b), ...)$. If we apply (ii) to $t_b(T) = e$ ($\vec{a}$ being empty), we see that TYPE(T) of PTQ, $((s,(s,e),(t)),t)$, is among the (derived) types in the type set of category $T$, $\mathcal{T}(T)$. If we

\textsuperscript{13} In order to be independent from specific syntactic frameworks, we will not use a sequent format for this semantic theory. We have, of course, not shown that it is altogether impossible to treat quantifier scope within flexible categorial grammar – only that current (sub-)systems (of L(P)) are inadequate (contrary to what has been suggested in Van Benthem (1986), Dowty (1988), a.o.). Thus, Moortgat (1988) simply incorporates Hendriks (1988) into L; Moortgat (1990) presents a more principled sequent-approach; and Emms (1989) offers an interesting alternative using polymorphic quantifiers and variable categories.
apply (iii) to the first (s,e) argument of \( \text{t}_3(\text{TV}) = ((s,e),((s,e),t)) \) (\( \text{a} \) again being empty), then also PTQ's \( \text{type}(\text{TV}) = ((s,((s,(s,e),t)),t)),((s,e),t)) \) appears to be an element of \( \text{T(IV)} \). The UG type of IVs (see above), \(((s,((s,(s,e),t)),t)),t)\), is in \( \text{T(IV)} \): apply (ii) or (iii) to \( \text{t}_3(\text{IV}) = ((s,e),t) \).

Similarly, every (lexical or compound) expression \( \alpha \) of category C is not assigned one translation, but a translation set, \( \text{Tr}(\alpha) \): a set consisting of one or more basic translations plus derived translations:

### 3.2.1 Basic translations

#### 3.2.1.1 Basic translations of lexical expressions.

Every lexical expression \( \alpha \) is assigned a set of basic translations, \( \text{Tr}_B(\alpha) \), in the lexicon. In general – i.e., for all lexically unambiguous expressions – \( \text{Tr}_B(\alpha) \) will only contain one member: \( \text{tr}_B(\alpha) \). The other elements of \( \text{Tr}(\alpha) \) are derived from \( \text{tr}_B(\alpha) \) by general rules (cf. section 3.2.2, below). It is important to stress that \( \text{tr}_B(\alpha) \) is not necessarily of the basic type \( \text{t}_B(C) \). The fragment contains the following basic translations (\( \text{Tr}_B(a) = \{\text{tr}_B(a)\} \)):

\[
\begin{align*}
\text{tr}_B(a)_{\text{type}(\text{tr}_B(a))} & \quad \text{PRICE}_{((s,e),t)} \\
[\text{CNprice}] & \quad \lambda x. [\text{MAN}(\gamma x) \land \exists u(x = \gamma u)]((s,e),t) \\
[\text{CNman}] & \quad \lambda x. [\text{WOMAN}(\gamma x) \land \exists u(x = \gamma u)]((s,e),t) \\
[\text{CNwoman}] & \quad \lambda x. [\text{UNICORN}(\gamma x) \land \exists u(x = \gamma u)]((s,e),t) \\
[\text{CNunicorn}] & \quad \lambda x. [\text{WALK}(\gamma x)]((s,e),t) \\
[\text{IVwalk}] & \quad \lambda x. [\text{RISE}(\gamma x)]((s,e),t) \\
[\text{IVrise}] & \quad \lambda x. [\text{READ}(\gamma x)]((s,e),t) \\
[\text{TV\_Mary}] & \quad \lambda x. [\text{BELIEVE}(\gamma x)]((s,e),t) \\
[\text{TV\_believe}] & \quad \lambda x. [\text{CLAIM}(\gamma x)]((s,e),t) \\
[\text{TV\_claim}] & \quad \lambda x. [\text{SEEK}(\gamma x)]((s,e),t) \\
[\text{TV\_seek}] & \quad \lambda x. [\text{EVERY}(\gamma x)]((s,e),t) \\
[\text{TV\_every}] & \quad \lambda x. [\text{ONE}(\gamma x)]((s,e),t) \\
[\text{TV\_one}] & \quad \lambda x. [\text{PRED}(\gamma x)]((s,e),t) \\
[\text{TV\_pred}] & \quad \lambda x. [\text{PRED}(\gamma x)]((s,e),t)
\end{align*}
\]

Note that the fragment does not contain syntactic variables \( hei \); that the basic translation of \( \text{seek} \) is not of the basic type of transitive verb phrases, \( t_9(\text{TV}) = ((s,e),((s,e),t)) \); that the basic translation of \( \text{be missing} \) is not of the basic type assigned to intransitive verb phrases, \( t_9(\text{IV}) = ((s,e),t) \); and that the functional application of the basic translation of \( a \)
to the intension of the basic translation of *woman* yields a translation of *a woman* which is not of the basic type assigned to noun phrases, $\text{tr}_B(T) = e$, but of the derived type $((s, ((s, e), t)), t)$.

### 3.2.1.2 Basic translations of compound expressions

Compound expressions $\alpha_0$ are also assigned a set of basic translations: $\text{Tr}_B(\alpha_0)$. $\text{Tr}_B(\alpha_0)$ is a function of (i) the translation sets of the parts, $\alpha_1, \ldots, \alpha_n$, of $\alpha_0$: $\text{Tr}(\alpha_1), \ldots, \text{Tr}(\alpha_n)$ (the sets consisting of both the basic translations and the derived translations of $\alpha_1, \ldots, \alpha_n$); and (ii) the semantic operation $R$ associated with the syntactic rule which constructs $\alpha_0$ from its immediate parts $\alpha_1, \ldots, \alpha_n$. The fragment only contains binary rules: in all cases, $\alpha_0$ is built up from two parts, $\alpha$ and $\beta$. Hence the above can be simplified.

$$\text{(85)} \quad \text{Tr}_B(\alpha_0) = \{ R(\alpha', \beta'): \alpha' \in \text{Tr}(\alpha) \text{ and } \beta' \in \text{Tr}(\beta) \}$$

There are three **semantic operations** $R$ in the fragment: FA, GC and GD.

$$\text{(86)} \quad \text{FA (Intensional Functional Application): for } \gamma \text{ of type } ((s, a), b) \text{ and } \delta \text{ of type } a:}$$

$$\text{FA}(\gamma, \delta) = \gamma(\delta)$$

$$\text{(87)} \quad \text{GC (Generalized Conjunction): for } \gamma \text{ and } \delta \text{ of type } (\bar{a}, t):}$$

$$\text{GC}(\gamma, \delta) = \lambda x^a.\left[ \gamma(x) \land \delta(x) \right]$$

$$\text{(88)} \quad \text{GD (Generalized Disjunction): for } \gamma \text{ and } \delta \text{ of type } (\bar{a}, t):}$$

$$\text{GD}(\gamma, \delta) = \lambda x^a.\left[ \gamma(x) \lor \delta(x) \right]$$

$$\text{(86), (87) and (88) play a role in the rules of the fragment:}$$

(A) **Rules of application:** $\text{Tr}_B(\alpha_0) = \{ \text{FA}(\alpha', \beta'): \alpha' \in \text{Tr}(\alpha) \text{ and } \beta' \in \text{Tr}(\beta) \}$

(i) $\beta_T + \alpha_{IV} \rightarrow \alpha_0 = [S \beta_T \alpha_{IV}]$

(ii) $\alpha_{TV} + \beta_T \rightarrow \alpha_0 = [IV \alpha_{TV} \beta_T]$

(iii) $\alpha_{Det} + \beta_{CN} \rightarrow \alpha_0 = [T \alpha_{Det} \beta_{CN}]$

(iv) $\alpha_{PV} + \beta_S \rightarrow \alpha_0 = [IV \alpha_{PV} \beta_S]$

(B) **Rules of conjunction:** $\text{Tr}_B(\alpha_0) = \{ \text{GC}(\alpha', \beta'): \alpha' \in \text{Tr}(\alpha) \text{ and } \beta' \in \text{Tr}(\beta) \}$

(v) $\alpha_S + \beta_S \rightarrow \alpha_0 = [S \alpha_S \text{ and } \beta_S]$

(vi) $\alpha_{IV} + \beta_{IV} \rightarrow \alpha_0 = [IV \alpha_{IV} \text{ and } \beta_{IV}]$

(vii) $\alpha_T + \beta_T \rightarrow \alpha_0 = [T \alpha_T \text{ and } \beta_T]$

(viii) $\alpha_{TV} + \beta_{TV} \rightarrow \alpha_0 = [TV \alpha_{TV} \text{ and } \beta_{TV}]$
(C) Rules of disjunction: \( \text{Tr}_B(\alpha_0) = \{ \text{GD}(\alpha',\beta') : \alpha' \in \text{Tr}(\alpha) \text{ and } \beta' \in \text{Tr}(\beta) \} \)

(ix) \( \alpha_S + \beta_S \rightarrow \alpha_0 = [S \alpha_S \text{ or } \beta_S] \)

(x) \( \alpha_T + \beta_T \rightarrow \alpha_0 = [T \alpha_T \text{ or } \beta_T] \)

(xi) \( \alpha_P + \beta_P \rightarrow \alpha_0 = [P \alpha_P \text{ or } \beta_P] \)

(xii) \( \alpha_T + \beta_TV \rightarrow \alpha_0 = [TV \alpha_T \text{ or } \beta_{TV}] \)

Note that there are no rules of quantification in the fragment. Their task will be taken over by the derived translations (see section 3.2, below). Also note that we have added rules for the conjunction and disjunction of transitive verb phrases ((viii) and (xii)), cf. sentence (29) and rule (32) above.

An example: application rule (iii) allows us to construct the noun phrase a fish from the determiner a and the common noun fish.

(89) \( \text{tr}_B(a) = \lambda P \lambda Q. \exists y[\forall P(y) \wedge Q(y)], \text{ and } \text{tr}_B(\text{fish}) = \lambda x. [\text{FISH}(\gamma x) \exists u(x = u)] \)

Rule (iii) states that \( \text{tr}_B(a \text{ fish}) = \{ FA(\alpha',\beta') : \alpha' \in \text{Tr}(a) \text{ and } \beta' \in \text{Tr}(\text{fish}) \} \). Because \( \text{tr}_B(a) \in \text{Tr}(a) \) and \( \text{tr}_B(\text{fish}) \in \text{Tr}(\text{fish}) \), we know that (90) of type \((s,((s, (s,e), t), t)), t\) is a member of \( \text{tr}_B(a \text{ fish}) \).

(90) \( FA(\text{tr}_B(a), \text{tr}_B(\text{fish})) = \lambda P \lambda Q. \exists y[\forall P(y) \wedge Q(y)](\forall \lambda x. [\text{FISH}(\gamma x) \exists u(x = u)]) \equiv \lambda Q. \exists y[\forall \text{FISH}(\gamma) \wedge Q(\gamma)] \)

3.2.2 Derived translations

The translation set \( \text{Tr}(\alpha) \) of a (lexical or compound) expression \( \alpha \) is defined inductively: \( \text{Tr}(\alpha) \) is the smallest set such that:

(91) For all \( \alpha' \in \text{Tr}_B(\alpha) : \alpha' \in \text{Tr}(\alpha) ; \)

(92) value raising (VR): \((\bar{a}, b) \Rightarrow (\bar{a}, ((s, ((s, (s, b), t), t)), t))\)

if \( \alpha' \in \text{Tr}(\alpha) \) and \( \alpha' \) is of type \((\bar{a}, b)\), then

\( \lambda x^a_x \lambda w((s, ((s, (s, b), t), t)), t)^w(\alpha(\bar{x})) \in \text{Tr}(\alpha) ; \)

(93) argument raising (AR): \((\bar{a}, ((s, (s, (s, b), t), t)), (\bar{c}, t)) \Rightarrow (\bar{a}, ((s, ((s, (s, b), t), t)), (\bar{c}, t)))\)

if \( \alpha' \in \text{Tr}(\alpha) \) and \( \alpha' \) is of type \((\bar{a}, ((s, (s, (s, b), t), t)), (\bar{c}, t))\), then

\( \lambda x^a_x \lambda w((s, ((s, (s, b), t), t)), (\bar{c}, t))^w(\lambda z((s, (s, b), t), \alpha(\bar{x}) (z)(\bar{c}))) \in \text{Tr}(\alpha) ; \) and

(94) argument lowering (AL): \((\bar{a}, ((s, ((s, (s, b), t), t)), (\bar{c}, t))) \Rightarrow (\bar{a}, ((s, ((s, (s, b), t), t)), (\bar{c}, t)))\)

if \( \alpha' \in \text{Tr}(\alpha) \) and \( \alpha' \) is of type \((\bar{a}, ((s, ((s, (s, b), t), t)), (\bar{c}, t)))\), then

\( \lambda x^a_x \lambda w((s, (s, b), (\bar{c}, t))^w(\lambda z((s, (s, b), t), \alpha(\bar{x}) (z)(\bar{c}))) \in \text{Tr}(\alpha) . \)

In (92), (93) and (94) \( \bar{x}^a_x \) and \( \bar{y}_c^c \) are sequences of variables, \( \bar{x} = x_{a_1}, \ldots, x_{a_n} \) and \( \bar{y} = y_{c_1}, \ldots, y_{c_m} \), of type \( \bar{a} = a_1, \ldots, a_n \) and \( \bar{c} = c_1, \ldots, c_m \) respectively, with \( m \) and \( n \geq 0 \).

\( \lambda x^a_x \tau \) abbreviates \( \lambda x_{a_1} \ldots x_{a_n} \tau \), and \( \tau(\bar{x}^a_x) \) abbreviates \( \tau(x_{a_1}) \ldots (x_{a_n}) \). All variables in
the above terms are different. \(\lambda x_{n_1}, \ldots, \lambda x_{n_n}, \lambda x_{n+1}(b,c), \forall x_{n+1}(\forall'(x_{n_1}, \ldots, x_{n_n}))\) is, for example, the term assigned by value raising.

First some lexical examples. When value raising is applied to the basic translation (\(\text{tr}_B\)) of John, \(j\), we get \(\lambda P. \forall P(j)\) as a derived translation, i.e., the translation PTQ assigns to this noun phrase; application of argument raising to (the first argument of) the basic translation of find, \(\lambda y \lambda x. \text{FIND}(\forall y)(\forall x)\), leads to \(\lambda T \lambda x. \forall^T(\forall^y. \text{FIND}(\forall y)(\forall x))\), the PTQ translation; application of argument lowering to (the first argument of) the basic translation of seek, \(\lambda T \lambda x. \text{SEEK}(T)(\forall x)\), gives us a derived translation that can be applied to the intension of the basic translations of proper names (which are of type e):

\[
\lambda y \lambda z. \text{SEEK}(\forall^y P \forall^2 P(y))(z).
\]

It is worth noting that the semantic operations FA, GC and GD are partial with respect to \(\text{Tr}(\alpha)\) and \(\text{Tr}(\beta)\). Sometimes, \(\text{FA}/\text{GC}/\text{GD}(\alpha', \beta')\) fails to be defined for \(\alpha'\in \text{Tr}(\alpha)\) and \(\beta'\in \text{Tr}(\beta)\). For instance, \(\text{Tr}_B(\text{John and Mary})\) does not contain the GC of \(j_e\) and \(m_e\), simply because \(j\) and \(m\) are not of a conjoinable type (a, t). Neither does \(\text{Tr}_B(\text{seek John})\) contain the FA of \(\lambda T \lambda x. \text{SEEK}(T)(\forall x)\) – an expression of type ((s, (s, (s, e), t)), t), (s, (s, e), t) – and \(j_e\), for the reason that \(\text{FA}(\lambda T \lambda x. \text{SEEK}(T)(\forall x), j)\) is undefined: \(\lambda T \lambda x. \text{SEEK}(T)(\forall x)\) is not of the required type ((s, \text{TYPE}(j)), b) = (s, e, b). But this is no problem. Since we have also the derived translations \(\lambda P. \forall P(j)\) and \(\lambda P. \forall P(m)\) at our disposal, we can be sure that \(\text{Tr}_B(\text{John and Mary})\) and \(\text{Tr}_B(\text{seek John})\) are non-empty:

\[
(95) \quad \text{GC}(\lambda P. \forall P(j), \lambda P. \forall P(m)) = \\
\lambda Q. [\lambda P. \forall P(j)(Q) \land \lambda P. \forall P(m)](Q) \in \text{Tr}_B(\text{John and Mary}), \quad \text{and} \\
(96) \quad \text{FA}(\lambda T \lambda x. \text{SEEK}(T)(\forall x), \lambda P. \forall P(j)) = \\
\lambda T \lambda x. \text{SEEK}(T)(\forall x)(\forall^P(j)) \in \text{Tr}_B(\text{seek John}).
\]

Let us now consider a compound example, which shows that we are able to account for the de dicto wide scope-or reading (24) of sentence (19): rule (iii) enables us to construct a fish from the determiner a and the common noun fish. We have just seen that

\[
(97) \quad \text{FA}(\text{tr}_B(a), \text{tr}_B(\text{fish})) = \lambda P \lambda Q. \forall y[\forall P(y) \land \forall Q(y)](\forall^x. \text{FISH}(\forall x \land \exists u(x = ^u))) \iff \\
\lambda Q. \forall y[\forall FISH(v) \land \forall Q(v)] \quad (= 90)
\]

of type ((s, (s, e), t), t) = \(\pi\) (\(\pi\) is Montague’s TYPE(T)) is a member of \(\text{Tr}_B(\text{a fish})\). Hence, according to (91), it is also a member \(\text{Tr}(\text{a fish})\). Due to (92), value raising, we know that \(\text{Tr}(\text{a fish})\) furthermore contains the derived translation (98), of type ((s, ((s, (s, e), t), t), t), t): \(\omega:\)

\[
(98) \quad \lambda Q. (\forall^2 (s, ((s, (s, e), t), t), t)). \forall Q(\forall P. \exists v[FISH(v) \land \forall P(v)]))
\]

Analogously, we can show that the noun phrase a bike has the translation

\[
(99) \quad \lambda Q. (\forall^2 (s, ((s, (s, e), t), t), t)). \forall Q(\forall P. \exists v[BIKE(v) \land \forall P(v)]).
\]
Therefore, the rule of disjunction (x) specifies GD((98),(99)) = (100) as one of the basic translations of a fish or a bike:

\[(100) \lambda I. [\lambda Q. \forall Q[\exists \lambda P. \exists v[\text{fish}(v) \land P(v)]](I) \lor \lambda Q. \forall Q[\exists \lambda P. \exists v[\text{bike}(v) \land P(v)]](I)] \Rightarrow \lambda I. [\forall I[\exists \lambda P. \exists v[\text{fish}(v) \land P(v)]] \lor \forall I[\exists \lambda P. \exists v[\text{bike}(v) \land P(v)]]]]\]

(100) is also of type \(\omega\). Note that we can use argument raising (applying it to the first argument) to blow up the basic translation of seek, \(\lambda T \lambda x. \text{SEEK}(T)(^x)\), which is of type \(((s,(s,(s,e),t)),(s,e),t))\), to (101), which is of the higher type \(((s,\omega),(s,e),t)) = ((s,(s,(s,(s,(s,e),t)),t)),t),(s,e),t)):\n
\[(101) \lambda J_{(s,\omega)} \lambda y. \forall J[\lambda T. [\lambda T \lambda x. \text{SEEK}(T)(^x)](T)(y)) \Rightarrow \lambda J \lambda y. \forall J[\lambda T. \text{SEEK}(T)(^y)]\]

Because (100) \(\in \text{Tr}(a \text{fish or a bike})\) and (101) \(\in \text{Tr}(\text{seek})\), we can apply rule (ii), which gives us FA((101),(100)) = (102) – of type \(((s,e),t)\) – as one of the basic translations of seek a fish or a bike:

\[(102) [\lambda J \lambda y. \forall J[\lambda T. \text{SEEK}(T)(^y)]]
\langle \lambda J. [\forall J[\exists \lambda P. \exists v[\text{fish}(v) \land P(v)]] \lor \forall J[\exists \lambda P. \exists v[\text{bike}(v) \land P(v)]]]\rangle \Rightarrow
\lambda y. [\text{SEEK}[\exists \lambda P. \exists v[\text{fish}(v) \land P(v)]](\forall y) \lor \text{SEEK}[\exists \lambda P. \exists v[\text{bike}(v) \land P(v)]](\forall \gamma)] \rangle \]

Finally, when (102) is applied to \(\forall J\) (rule (i)), we arrive at reading (24):

\[(103) [\lambda y. [\text{SEEK}[\exists \lambda P. \exists v[\text{fish}(v) \land P(v)]](\forall y)] \lor \text{SEEK}[\exists \lambda P. \exists v[\text{bike}(v) \land P(v)]](\forall y)](\forall y) \Rightarrow \text{SEEK}[\exists \lambda P. \exists v[\text{fish}(v) \land P(v)]](\forall y) \lor \text{SEEK}[\exists \lambda P. \exists v[\text{bike}(v) \land P(v)]](\forall y)](\forall y)\]

4. APPLICATIONS

This section shows a number of applications of the flexible Montague grammar defined above. In section 4.1 we shall see that we can represent ambiguities due to the occurrence of quantified noun phrases, not only in ‘flat’ structures (where the quantifiers are all immediate arguments of one relation (4.1.1)), but also in embedded application structures (4.1.2) of arbitrary complexity. In 4.2 we shall consider ambiguities regarding the scope of coordination. In 4.3 it will be shown that certain restrictions on possible scopings can be readily accounted for within the present framework. Let us first settle some notation conventions.

(104) An application of VR (AR, AL) on a type \((\vec{a},c)\) will be called an \textbf{m-application} of VR (AR, AL) (written as \(m\text{VR} (m\text{AR}, m\text{AL})\)) iff length \((\vec{a}) = m-1. \) \([m \in \{1,2,3,...\}\), cf. (92), (93) and (94) for \(\vec{a}.)\]
Example: \((a,b) \Rightarrow^2 \text{VR} (a,((s,(s,b),t)),t))\); \((a,b) \Rightarrow^1 \text{VR} ((s,((s,(a,b)),t)),t)\).

(105) For any type a: \(a^0 = a\); \(a^{n+1} = ((s,a^n),t)\).

Example: \(e^2 = ((s,((s,e),t)),t)\).

(106) A reading of an expression \(\alpha\) is an interpretation of \(\alpha\) of type t.

4.1 The scope of quantification

4.1.1 Flat structures

Sentence (107) consists of an extensional transitive verb and two proper names.

(107) John finds Mary

If we apply rule (ii), and use the basic translations of *find* and *Mary*, we obtain

\(\lambda y \lambda x. \text{FIND}(\forall y)(\forall x)(\forall m) \Leftrightarrow \lambda x. \text{FIND}(\forall m)(\forall x) \Leftrightarrow \lambda x. \text{FIND}(m)(\forall x)\).

If we then apply (108) to the intension of the basic translation \(j\) of *John* (with the help of rule (i)), the resulting translation is

\(\lambda x. \text{FIND}(m)(\forall x)(\forall j) \Leftrightarrow \text{FIND}(m)(j)\).

It can be shown that this is the only reading (in the sense of (106)) that is assigned to sentence (107) by the flexible grammar. I.e., if we, instead of using the basic translations, were to combine arbitrary derived translations, the result would be equivalent (the proofs of claims about numbers of readings can be found in or deduced from the Appendix).

If an intensional transitive verb is combined with two proper names, the grammar also assigns one reading, regardless of the translations used:

(110) John seeks Mary

We shall sketch two possible ways to obtain this reading. One way involves the application of the basic translation of *seek*, \(\lambda T \lambda x. \text{SEEK}(T)(\forall x)\), to the intension of the derived translation \(\lambda P. \forall P(\forall m)\) of *Mary* (the result of value-raising m) and the combination of the result with \(j\), just as we did in the case of (107). But we can also apply argument lowering to the first argument of \(\lambda T \lambda x. \text{SEEK}(T)(\forall x)\),

\(\lambda y. \lambda T \lambda x. \text{SEEK}(T)(\forall x)(\lambda P. \forall P(y)) \Leftrightarrow \lambda y \lambda x. \text{SEEK}(\lambda P. \forall P(y))(\forall x)\).
and combine the result (via (ii) and (i)) with the basic translations of the proper names:
\[ \lambda y \lambda x.\text{SEEK}(\lambda P.\text{\`P}(y))(\lambda x)(\lambda x)(\lambda x) \equiv \text{SEEK}(\lambda P.\text{\`P}(u))(v) \].

If, however, the noun phrases surrounding the transitive verb are quantified noun phrases, then the grammar assigns a different number of readings to the sentences with *seek* and *find*:

(112) Every man finds a woman

(i) \( \forall v[\text{MAN}(v) \rightarrow \exists u[\text{WOMAN}(u) \land \text{FIND}(u)(v)]] \)

(ii) \( \exists u[\text{WOMAN}(u) \land \forall v[\text{MAN}(v) \rightarrow \text{FIND}(u)(v)]] \)

(113) Every man seeks a woman

(i) \( \forall v[\text{MAN}(v) \rightarrow \exists u[\text{WOMAN}(u) \land \text{SEEK}(\lambda P.\text{\`P}(u))(v)]] \)

(ii) \( \exists u[\text{WOMAN}(u) \land \forall v[\text{MAN}(v) \rightarrow \text{SEEK}(\lambda P.\text{\`P}(u))(v)]] \)

(iii) \( \forall v[\text{MAN}(v) \rightarrow \text{SEEK}(\lambda P.\exists u[\text{WOMAN}(u) \land \text{\`P}(u)])(v)] \)

(112) gets two readings, whereas (113) can be understood in three ways. The basic translation \( \lambda y \lambda x.\text{FIND}(\lambda y)(\lambda x)(\lambda x) \) is of type \( (s,e)((s,e),t) \), and there are two non-equivalent ways, (114) and (115), to argument-raise it in two steps to type \( ((s,((s,e),t)),(s,((s,e),t)),t),t) = ((s,e),(s,e),(s,e),t) \):\(^{14}\)

(114) \( ((s,e),((s,e),t)) \Rightarrow ((s,e),(s,e),t),\lambda T_1 T_2.\forall T_1 (\lambda x.\text{FIND}(\lambda x)(\lambda y)(\lambda y)) \Rightarrow \alpha \); \( ((s,e),(s,e),t) \Rightarrow ((s,e),(s,e),t),\lambda T_1 T_2.\forall T_2 (\lambda x.\beta(T_1)(\lambda x.\text{FIND}(\lambda y)(\lambda y)))(T_2) \)

(115) \( ((s,e),((s,e),t)) \Rightarrow ((s,e),(s,e),t),\lambda x.\lambda T_2.\forall T_2 (\lambda y.\text{FIND}(\lambda x)(\lambda y)(\lambda y)) \Rightarrow \beta \); \( ((s,e),((s,e),t)) \Rightarrow ((s,e),(s,e),t),\lambda T_1 T_2.\forall T_1 (\lambda x.\text{FIND}(\lambda y)(\lambda y))(T_2) \)

The combination of (114) and the basic translations *every man* and *a woman* yields (112) (i); (112) (ii) is the result of combining (115) with these basic translations. Again, additional type change does not increase the number of readings. In the case of (113) we can let the argument-lowered translation of *seek*, \( \lambda y \lambda x.\text{SEEK}(\lambda P.\text{\`P}(y))(\lambda x) \), undergo (a) and (b) as well. We then get (113) (i) and (113) (ii), respectively. But there is also a third possibility here. We can use the basic translation of *seek*, \( \lambda T \lambda x.\text{SEEK}(T)(\lambda x) \), and raise its second argument to type \( (s,e) \): \( \lambda T_1 T_2.\forall T_2 (\lambda y.\text{SEEK}(T_1)(\lambda y)) \). In this way we account for the *de dicto* reading, (113) (iii).

In general, the combination of an n-ary relation of type \( (s,e_1,\ldots,((s,e_n),t)) \) with n quantified noun phrases – i.e., noun phrases of type \( (s,((s,e),t),t) \) – yields maximally \( n! = n \cdot (n-1) \cdot \ldots \cdot 2 \cdot 1 \) readings of type t (n! is a maximum here, because quantified noun phrases that are value-raised terms of type e (proper names), for instance do not help multiplying the number of readings), which is what we need.

\(^{14}\) Different from AL, the order in which VR and AR are applied is important.
4.1.2 Embedded structures

The flexible grammar of section 3.2 is able to represent scope ambiguities in arbitrarily complex structures of functional application. Consider the syntactic structure of the compound sentence *Fred claims that every schoolboy believes that a mathematician wrote Through the Looking Glass.*

\[(116) \quad [S \text{ Fred}_T [\text{TV claims that}_T \exists [\text{everyDet schoolboy}_C N] [\text{TV believes that}_T \exists [\text{aDet mathematician}_C N] \text{ [TV wrote}_T \text{TV Through the looking glass}_T ] ] ] ] \]

With the basic translations indicated below, (116) gets exactly the eight readings which result from interchanging the mutual scopings of the quantified noun phrases *every schoolboy* and *a mathematician* on the one hand, and interpreting these noun phrases more or less *de dicto* or *de re* within the compound intensional context (*...believe that(...claim that (...))*) on the other:

\[
\begin{align*}
\text{Fred} & \quad f_e, \\
\text{believe that} & \quad \lambda p \lambda x. \text{BELIEVE}(\forall v [\text{BOY}(v) \rightarrow \forall v [\text{P}(v)]]), \\
\text{every schoolboy} & \quad \lambda P. \forall v [\text{BOY}(v) \rightarrow \forall v [\text{P}(v)]](s, t, ((s, e), t)), \\
\text{claim that} & \quad \lambda P. \forall v [\text{P}(v)], \\
\text{a mathematician} & \quad \lambda P. \exists u [\text{MATH}(u) \wedge \forall v [\text{P}(v)]](s, t, ((s, e), t)), \\
\text{write} & \quad \lambda y \lambda x. \text{WRITE}(\forall v [\text{P}(v)], t), \\
\text{Through the looking glass} & \quad L_e. 
\end{align*}
\]

\[(117) \quad \text{CLAIM}(f, \forall v [\text{BOY}(v) \rightarrow \exists u [\text{MATH}(u) \wedge \text{WRITE}(u, L)])] \\
(118) \quad \text{CLAIM}(f, \exists u [\text{MATH}(u) \wedge \forall v [\text{BOY}(v) \rightarrow \exists u [\text{MATH}(u) \wedge \text{WRITE}(u, L)])]) ] \\
(119) \quad \text{CLAIM}(f, \exists u [\text{MATH}(u) \wedge \forall v [\text{BOY}(v) \rightarrow \exists u [\text{MATH}(u) \wedge \text{WRITE}(u, L)])]) ] \\
(120) \quad \exists u [\text{MATH}(u) \wedge \text{CLAIM}(f, \forall v [\text{BOY}(v) \rightarrow \exists u [\text{MATH}(u) \wedge \text{WRITE}(u, L)])]) ] \\
(121) \quad \forall v [\text{BOY}(v) \rightarrow \text{CLAIM}(f, \exists u [\text{MATH}(u) \wedge \text{BELIEVE}(v, \exists u [\text{MATH}(u) \wedge \text{WRITE}(u, L)])]) ] \\
(122) \quad \forall v [\text{BOY}(v) \rightarrow \text{CLAIM}(f, \exists u [\text{MATH}(u) \wedge \text{BELIEVE}(v, \exists u [\text{MATH}(u) \wedge \text{WRITE}(u, L)])]) ] \\
(123) \quad \forall v [\text{BOY}(v) \rightarrow \exists u [\text{MATH}(u) \wedge \text{CLAIM}(f, \forall v [\text{MATH}(u) \wedge \text{BELIEVE}(v, \forall v [\text{MATH}(u) \wedge \text{WRITE}(u, L)])]) ] \\
(124) \quad \exists u [\text{MATH}(u) \wedge \forall v [\text{BOY}(v) \rightarrow \text{CLAIM}(f, \forall v [\text{MATH}(u) \wedge \text{BELIEVE}(v, \forall v [\text{MATH}(u) \wedge \text{WRITE}(u, L)])]) ]
\]
By way of illustration, we show the derivation of (124), semantically the most ‘dislocated’ reading of sentence (116).\footnote{The remaining seven readings can be obtained in the following way:}

The noun phrase \textit{a mathematician} must ‘escape from’ its original scope domain, the one embedded in \textit{believe that}. This can be effectuated by subsequently applying 3\textit{VR} (creating the ‘escape hatch’) and 2\textit{AR} (letting the subject argument ‘go out’ of it) to the basic translation of \textit{write}, the main functor in that domain: \((s,e),((s,e),t) \Rightarrow ((s,e),((s,e),t^2)) \Rightarrow ((s,e),((s,e),t^2))\):

\begin{align*}
\lambda y \lambda x. \text{WRITE}(\forall x,y) &\Rightarrow 3\text{VR} \lambda y \lambda x \lambda w. \forall w(\forall \text{WRITE}(\forall x,y)) \Rightarrow 2\text{AR} \\
\lambda y \lambda T \lambda w. \forall T(\forall \lambda x. \forall w(\forall \text{WRITE}(\forall x,y)))
\end{align*}

The noun phrase \textit{every schoolboy} is to have wider scope than \textit{claim that}. The main functor of the domain embedded in \textit{claim that} is \textit{believe that}. We can perform the same trick here: first apply 3\textit{VR}, and then 2\textit{AR}, in this case to \textit{believe that}. Due to our application of value raising to \textit{write}, the compound first argument of \textit{believe that (a mathematician wrote Through the Looking Glass)} is too large (being of type \(t^2\) instead of \(t\)). This calls for an additional application of \textit{AR} to \textit{believe that}, 1\textit{AR} this time. Since \textit{a mathematician} is going to have wider scope than \textit{every schoolboy}, 1\textit{AR} should be applied \textit{after} 3\textit{VR} and 2\textit{AR}.

\begin{align*}
\lambda p \lambda x. \text{BELIEVE}(\forall x,p) &\Rightarrow 3\text{VR} \lambda p \lambda x \lambda w. \forall w(\forall \text{BELIEVE}(\forall x,p)) \Rightarrow 2\text{AR} \\
\lambda p \lambda T \lambda w. \forall T(\forall \lambda x. \forall w(\forall \text{BELIEVE}(\forall x,p))) &\Rightarrow 1\text{AR} \\
\lambda Q \lambda T \lambda w. \forall Q(\forall p. \forall T(\forall \lambda x. \forall w(\forall \text{BELIEVE}(\forall x,p))))
\end{align*}
As a consequence of our applying value raising to believe that, the compound first argument of claim that (every schoolboy believes that a mathematician wrote Through the Looking Glass) does not 'fit' any more. This necessitates an application of 1AR to claim that.

(127) \( \lambda p \lambda x. \text{CLAIM}(\langle x, p \rangle) \Rightarrow 1\text{AR} \lambda Q \lambda x. \forall Q(\langle \lambda p. \text{CLAIM}(\langle x, p \rangle) \rangle) \)

With this amount of type change, we get the following translation for a mathematician wrote Through the Looking Glass:

(128) \[ \lambda y \lambda T \lambda w. \forall T(\langle \lambda x. \forall w(\langle \text{WRITE}(\langle x, y \rangle) \rangle) \rangle)(L)(\langle \lambda P. \exists u[\text{MATH}(u) \land \forall w(\langle \text{WRITE}(u, L) \rangle) \rangle) \Rightarrow \lambda w. \exists u[\text{MATH}(u) \land \forall w(\langle \text{WRITE}(u, L) \rangle) \rangle] \]

Every schoolboy believes that a mathematician wrote Through the Looking Glass is translated as:

(129) \[ \lambda Q \lambda T \lambda w. \forall Q(\langle \lambda p. \forall T(\langle \lambda x. \forall w(\langle \text{BELIEVE}(\langle x, p \rangle) \rangle) \rangle) \rangle)(\langle \lambda w. \exists u[\text{MATH}(u) \land \forall w(\langle \text{WRITE}(u, L) \rangle) \rangle)(\langle \lambda P. \forall v[\text{BOY}(v) \rightarrow \forall w(\langle \text{BELIEVE}(v, \langle \text{WRITE}(u, L) \rangle) \rangle) \rangle) \Rightarrow \lambda w. \exists u[\text{MATH}(u) \land \forall v[\text{BOY}(v) \rightarrow \forall w(\langle \text{BELIEVE}(v, \langle \text{WRITE}(u, L) \rangle) \rangle)[\rangle] \]

And we get the following translation for sentence (116) as a whole:

(130) \[ \lambda Q \lambda x. \forall Q(\langle \lambda p. \text{CLAIM}(\langle x, p \rangle) \rangle)[\rangle)(\langle \lambda w. \exists u[\text{MATH}(u) \land \forall v[\text{BOY}(v) \rightarrow \forall w(\langle \text{BELIEVE}(v, \langle \text{WRITE}(u, L) \rangle) \rangle) \rangle) \Rightarrow \exists u[\text{MATH}(u) \land \forall v[\text{BOY}(v) \rightarrow \text{CLAIM}(\langle f, \langle \text{BELIEVE}(v, \langle \text{WRITE}(u, L) \rangle) \rangle) \rangle] \]

4.2 The scope of coordination

We showed in section 3.2.2 that the de dicto wide scope-or reading (iii) of sentence (131) can be represented.

(131) John seeks a fish or a bike
(i) \( \lambda P.[\exists v[\text{FISH}(v) \land \forall P(v)]] \lor \exists v[\text{BIKE}(v) \land \forall P(v)] \]
(ii) \( \exists v[\text{FISH}(v) \land \exists v[\langle \lambda P. \forall P(v) \rangle(j)] \lor \exists v[\text{BIKE}(v) \land \exists v[\langle \lambda P. \forall P(v) \rangle(j)] \]
(iii) \( \exists v[\langle \lambda P. \exists v[\text{FISH}(v) \land \forall P(v)] \rangle(j)] \lor \exists v[\langle \lambda P. \exists v[\text{BIKE}(v) \land \forall P(v)] \rangle(j)] \)

It is obvious that the 'standard' de dicto reading (i) is accounted for too: for this reading, it is sufficient to combine the basic translations of the expressions. The de re reading (ii) can be realised by building up (132),

(132) \( \lambda P.[\exists v[\text{FISH}(v) \land \forall P(v)] \lor \exists v[\text{BIKE}(v) \land \forall P(v)] \]

the generalized disjunction of the basic translations of a fish and a bike, and applying (133) to the intension of this expression:

(133) $\lambda T\lambda x.\neg T(\lambda y.\text{SEEK}(\lambda P.\forall P(y))(\forall x))$

(133) is a derived translation of seek which results from applying $^1$AL and $^1$AR (in that order) to the basic translation: $((s,e^2),((s,e),t)) \Rightarrow (s,e),((s,e),t) \Rightarrow ((s,e^2),((s,e),t))$.

(134) $\lambda T\lambda x.\text{SEEK}(T)(\forall x) \Rightarrow^1$AL $\lambda y\lambda x.\text{SEEK}(\lambda P.\forall P(y))(\forall x) \Rightarrow^1$AR

$\lambda T\lambda x.\forall T(\lambda y.\text{SEEK}(\lambda P.\forall P(y))(\forall x))$

Also the scope ambiguity in sentence (135) is represented:

(135) John caught and ate a fish

(i) $\exists v[FISH(v) \land \text{CATCH}(v)(j)] \land \exists v[FISH(v) \land \text{EAT}(v)(j)]$

(ii) $\exists v[FISH(v) \land \text{CATCH}(v)(j) \land \text{EAT}(v)(j)]$

The extensional transitive verbs caught and ate have basic translations of the minimal type, $t_B(TV) = ((s,e),((s,e),t))$, and translations of type $((s,e^2),((s,e),t))$ are derivable from the basic translations by raising the first argument of $t_B(TV)$:

(136) $\lambda T\lambda x.\forall T(\lambda y.\text{CATCH}(\forall y)(\forall x))$ and $\lambda T\lambda x.\forall T(\lambda y.\text{EAT}(\forall y)(\forall x))$

Reading (i) employs the generalized conjunction of these derived translations:

(137) $\lambda T\lambda x.[\forall T(\lambda y.\text{CATCH}(\forall y)(\forall x)) \land \forall T(\lambda y.\text{EAT}(\forall y)(\forall x))]$

Reading (ii) is achieved via generalized conjunction of the basic translations, (138), followed by raising the first argument of the conjunction: (139).

(138) $\lambda y\lambda x.[\text{CATCH}(\forall y)(\forall x) \land \text{EAT}(\forall y)(\forall x)]$

(139) $\lambda T\lambda x.\forall T(\lambda y.[\text{CATCH}(\forall y)(\forall x) \land \text{EAT}(\forall y)(\forall x)])$

Partee and Rooth (1983) in fact only accept reading (ii), 'unless the sentence [is] given a very marked intonation or the context is heavily loaded' (p. 365), but this observation is at least questionable, in consideration of the natural continuation in (140):

(140) John caught and ate a fish. The fish he caught was inedible, and the fish he ate caught his eye.

Be this as it may, less probable but possible readings should be predicted by the grammar as well, preferably together with an explanation of their lesser probability. An explanation of cases like these will be given below.
The flexible grammar also supplies a desired de dicto wide scope-or reading of sentence (141), viz., (142), a problematic case for which Partee and Rooth, surprisingly, suggest 'a rule quantifying in terms of a higher type' (p. 376):

(141) John believes that a man or a woman walks
(142) \text{BELIEVE}(j,^\exists u[\text{MAN}(u)\land \text{WALK}(u)]) \lor \text{BELIEVE}(j,^\exists u[\text{WOMAN}(u)\land \text{WALK}(u)])

To get (142), we value-raise the noun phrases a man and a woman: ((s,((s,e),t)),t) \Rightarrow VR ((s,((s,e),t)),t^2) = e^4, and take the generalized disjunction of the derived translations (I is of type ((s,((s,e),t)),t))^1 = e^3:

(143) \lambda I,^\forall I(^\forall I(^\lambda P,^\exists u[\text{MAN}(u)\land ^\forall P(^u)])\land ^\forall I(^\lambda P,^\exists u[\text{WOMAN}(u)\land ^\forall P(^u)]))

Next we apply 1AR, 2VR, and 1AR to \lambda x.\text{WALK}(\langle x \rangle), the basic translation of walk. ((s,e),t) \Rightarrow 1AR ((s,e^2),t) \Rightarrow 2VR ((s,e^2),t^2) \Rightarrow 1AR ((s,e^4),t^2):

(144) \lambda x.\text{WALK}(\langle x \rangle) \Rightarrow 1AR \lambda T.\langle T(\lambda x.\text{WALK}(\langle x \rangle)) \rangle \Rightarrow 2VR
\lambda T\lambda w.\langle w(\langle T(\lambda x.\text{WALK}(\langle x \rangle)) \rangle) \rangle \Rightarrow 1AR \lambda Q\lambda w.\langle Q(\langle T(\lambda x.\text{WALK}(\langle x \rangle)) \rangle) \rangle

(144) is applied to the intension of (143):

(145) [\lambda Q\lambda T.\langle Q(\langle T(\lambda x.\text{WALK}(\langle x \rangle)) \rangle) \rangle]
\langle ^\forall I(^\forall I(^\lambda P,^\exists u[\text{MAN}(u)\land ^\forall P(^u)])\land ^\forall I(^\lambda P,^\exists u[\text{WOMAN}(u)\land ^\forall P(^u)])) \rangle \Leftrightarrow
\lambda T.\langle ^\forall I(^\exists u[\text{WOMAN}(u)\land \text{WALK}(u)])\land ^\forall I(^\exists u[\text{WOMAN}(u)\land \text{WALK}(u)]) \rangle \rangle

Finally, the first argument of believe that is raised, and the result (146) is subsequently applied to the intension of (145) and j:

(146) \lambda p\lambda x.\text{BELIEVE}(p)(\langle x \rangle) \Rightarrow 1AR \lambda R\lambda x.\langle R(\lambda p.\text{BELIEVE}(p)(\langle x \rangle)) \rangle
(147) [\lambda R\lambda x.\langle R(\lambda p.\text{BELIEVE}(p)(\langle x \rangle)) \rangle]
\langle ^\forall I(\forall I(^\exists u[\text{WOMAN}(u)\land \text{WALK}(u)])\land ^\forall I(^\exists u[\text{WOMAN}(u)\land \text{WALK}(u)]))\rangle\langle j \rangle \Leftrightarrow
\text{BELIEVE}(\langle ^\exists u[\text{WOMAN}(u)\land \text{WALK}(u)]\rangle)(j) \lor \text{BELIEVE}(\langle ^\exists u[\text{WOMAN}(u)\land \text{WALK}(u)]\rangle)(j)

The flexible grammar of section 3 assigns all possible coordination scopes.\footnote{Except for the readings where quantifiers in conjuncts and disjuncts have wider scope than the conjunctions and disjunctions coordinating them. Those readings seem to be really impossible. cf. the discussion on the sentences (9) and (14) below.}

For example, (148) is assigned the readings (i), (ii) and (iii):

(148) John said that every student lost or won
(i) \text{SAY}(j,^\forall \forall v[\text{STUDENT}(v)\rightarrow \text{LOOSE}(v) \lor \text{WIN}(v)])
(ii) \text{SAY}(j,^\forall \forall v[\text{STUDENT}(v)\rightarrow \text{LOOSE}(v)] \lor \forall \forall v[\text{STUDENT}(v)\rightarrow \text{WIN}(v)])
(iii) \text{SAY}(j,^\forall \forall v[\text{STUDENT}(v)\rightarrow \text{LOOSE}(v)] \lor \text{SAY}(j,^\forall \forall v[\text{STUDENT}(v)\rightarrow \text{WIN}(v)])
'While intuitions are far from clear' (1983, p. 376), only (i) and (iii) are considered okay by Partee and Rooth, whereas in their fragment, (i) and (ii) are accounted for. A similar case is (149).

(149) Every student failed or got a D

This sentence does not seem to have the reading 'every student failed or every student got a D', a reading that seems to be predicted by all available theories of type ambiguity, including the present one: apply $^1$VR and generalized disjunction to the intransitive verb phrases, $\lambda T.[\forall T(\forall \lambda x.\text{FAIL}('x')) \lor \forall T(\forall \lambda x.\text{GET-D}('x'))]$, and combine this with the basic translation of every student. It is, however, obvious that the data are far from clear. Consider (150).

(150) Every player of our team is wearing a red shirt or a green shirt

That this sentence does have the reading 'every player of our team is wearing a red shirt or every player of our team is wearing a green shirt', can be supported by the possibility (if we add a 'heavily loaded' context: the speaker is colour-blind) of the continuation '… but I can't tell you what the exact colour is'.

That the context must be heavily loaded is what we should expect. For, under normal circumstances the Gricean maxim 'avoid ambiguity' requires us not to utter (151), (152) or (153) to convey the information expressed by the 'every … or every …' readings; there are less ambiguous sentences that express them.\footnote{The same pragmatic explanation applies to (135) (ii): there is an unambiguous sentence available for expressing this reading: \textit{John caught a fish and ate a fish}. With respect to (148), \textit{John said that every student lost or won}, Rooth and Partee (1982) note that in a parallel sentence like \textit{John said that a student lost and won}, with a and and replacing every and or, the parallel wide scope-and reading is absolutely excluded: \textit{John said that a student lost and won} cannot express 'John said that a student lost and John said that a student won'. They attribute the difference to discourse properties of indefinite noun phrases. Something like this might very well be the case, for note that with a different noun phrase, the wide scope-and reading is not altogether impossible: \textit{John said that exactly one student lost and won} does have the reading 'John said that exactly one student lost and John said that exactly one student won'.}

(151) John said that every student lost or every student won
(152) Every student failed or every student got a D
(153) Every player of our team is wearing a red shirt or every player of our team is wearing a green shirt

Only a colour-blind person entangled in the cumbersoness of his disjunctive worldview or a colour-blind person entangled in the cumbersoness of her disjunctive worldview may be expected to let being short prevail over being unambiguous here. Besides, an explanation in terms of a pragmatic conflict between the Gricean maxims 'be short' and 'avoid ambiguity' can contribute to an explanation why reading (iii) for sentence
(148) = (151) is more plausible than reading (ii), whereas the former is semantically more ‘dislocated’ than the latter (i.e., it requires more type change) — a fact that otherwise would remain mysterious: the gain of time and saving of energy obtained by uttering John said that every student lost or won as ‘short for’ John said that every student lost or John said that every student won (reading (iii)), exceeds the profit made by using the former sentence instead of John said that every student lost or every student won (reading (ii)).

In section 1.3.4 we noted that PTQ assigns the following translations to (9) and (14):

(154) Mary walks and seeks every unicorn (= (9))
(i) \text{WALK}(m) \land \text{SEEK}(\forall x.\forall y.\forall z. [\text{UNICORN}(y) \rightarrow y^P(x)](m))
(ii) \text{WALK}(m) \land \forall y. [\text{UNICORN}(y) \rightarrow \text{SEEK}(\forall x. y^P(x))(m)]
(iii) \forall y. [\text{UNICORN}(y) \rightarrow [\text{WALK}(m) \land \text{SEEK}(\forall x. y^P(x))(m)]]

(155) John runs and no unicorn walks (= (14))
(i) \text{RUN}(j) \land \neg \exists y. [\text{UNICORN}(y) \land \text{WALK}(y)]
(ii) \neg \text{RUN}(j) \lor \neg \exists y. [\text{UNICORN}(y) \land \text{WALK}(y)]

The flexible grammar only generates the readings (154)(i), (ii), and (155)(i). Reading (154)(i) does not require any noteworthy type change. For (154)(ii) we change the type of seek via 1\text{AL} and 1\text{AR}, respectively: ((s,e^2),((s,e),t)) \Rightarrow 1\text{AL} ((s,e),((s,e),t)) \Rightarrow 1\text{AR} ((s,e^2),((s,e),t), which leads to the derived translation

(156) \lambda T \lambda y. \forall x. \exists y. [\text{UNICORN}(y) \rightarrow \text{SEEK}(\forall x. y^P(x))](y).

Application to the intension of the basic translation of every unicorn yields

(157) \lambda y. \forall y. [\text{UNICORN}(y) \rightarrow \text{SEEK}(\forall x. y^P(x))](y),

Generalized intransitive verb conjunction of (157) and \lambda y. \text{WALK}(y) and application to the intension of m results in (ii). The generation of (154) (i) is straightforward. We shall not prove here that the non-readings (154) (ii) and (155) (ii) are not assigned within the flexible grammar (cf. the Appendix), but we shortly sketch why this is the case. We saw above that, anticipating the semantic structure in which this expression is embedded, a quantified noun phrase like every unicorn can be given wider scope than its original scope domain by applying value raising to the main functor of the scope domain (seek and walk in the cases under consideration), thus creating an ‘escape hatch’, and next letting it escape by applying argument raising to the relevant argument position. In the case of (9) ((14) is completely analogous) the semantic structure roughly looks like this: WALK + SEEK. We see that the left conjunct (WALK) is part of the semantic structure of

---

18 This might also explain why the reading of the sentences in (28) with or having widest scope are more plausible than the readings where or has intermediate scope (that is, narrower than claim that, but wider than seek).
the sentence, but that the functor of the scope domain of every unicorn, SEEK, is not embedded in it. Walk ‘c-commands’ seek, but does not dominate this expression. This entails that there is no escape from walk – and reading (iii) cannot be constructed.\textsuperscript{19}

4.3 Structural restrictions

The flexible grammar defined here is best seen as creating a ‘logical space’ of possibilities which is not necessarily exhausted by the facts of natural language. It should therefore meet the requirement that existing structural restrictions on possible readings can be implemented. In this section we shall show that this holds for scope islands and scope sieves.

It is well-known that some syntactic categories function as scope islands: quantified noun phrases or coordinations occurring inside expressions of island categories cannot have their scope outside of these environments. Relative clauses constitute a clear and relatively undisputed example of the phenomenon. Consider, e.g., (158) and (159).

(158) A girl who lives in every Italian city walks
(i) $\exists u[\text{GIRL}(u) \land \forall v[\text{ITALIAN-CITY}(v) \rightarrow \text{LIVE}(v)(u)] \land \text{WALK}(u)]$
(ii) $\forall v[\text{ITALIAN-CITY}(v) \rightarrow \exists u[\text{GIRL}(u) \land \text{LIVE}(v)(u)] \land \text{WALK}(u)]$

(159) Every girl who lives in Florence or Pisa walks
(i) $\forall u[\text{GIRL}(u) \land [\text{LIVE}(f)(u) \lor \text{LIVE}(p)(u)] \rightarrow \text{WALK}(u)]$
(ii) $\forall v[\text{GIRL}(u) \land \text{LIVE}(f)(u)] \rightarrow \text{WALK}(u)] \lor \forall v[\text{GIRL}(u) \land \text{LIVE}(p)(u)] \rightarrow \text{WALK}(u)]$

(158) can only have the (strange) interpretation (i); interpretation (ii) is impossible (though much more normal). Similarly, (159) can be interpreted as (i), but not as (ii).

Nonetheless, the sentences are each assigned two readings by the flexible grammar if we assume a (simplified) more or less standard treatment of relative clauses: (a) the relative pronoun who is assigned the basic translation $\lambda F \lambda Q \lambda x. [\sim Q(x) \land \sim P(x)]$ of type $((s,((s,e),t)),((s,((s,e),t)),((s,((s,e),t)))); (b)$ the basic type of relative clauses, $t_B(\text{REL})$, equals $((s,((s,e),t)),((s,((s,e),t))));$ and (c) two more rules of application are added:

(xiii) $\alpha_{\text{RELPRO}} + \beta_{IV} \rightarrow \alpha_0 = [\text{REL} \alpha_{\text{RELPRO}} \beta_{IV}]$
$\text{Tr}_B(\alpha_0) = \{(FA(\alpha',\beta') : \alpha' \in \text{Tr}(\alpha) \land \beta' \in \text{Tr}(\beta)\}$

\textsuperscript{19} This is a consequence of our having separate rules of conjunction and disjunction. When or and and are treated as polymorphic functors (of syntactic category $X\lor X/X$) and semantic type $(a,(a,a))$, the usual analysis in flexible categorial grammars, cf. e.g., Moortgat (1988)), which combine with their arguments via functional application, the situation is different. Under that analysis, (154) (iii) and (155) (ii) are possible interpretations.

The analysis in terms of generalized conjunction and disjunction has the effect of turning coordinations into some kind of scope islands (cf. the treatment of relative clauses below), but, having scope themselves, they are ‘floating’ scope islands.
(xiv) \[ \beta_{CN} + \alpha_{REL} \rightarrow \alpha_0 = [CN \alpha_{CN} \beta_{REL}] \]
\[ \text{Tr}_B(\alpha) = \{(FA(\alpha',\beta') : \alpha' \in \text{Tr}(\alpha) \text{ and } \beta' \in \text{Tr}(\beta) \} \]

We shall restrict ourselves to sentence (158). ((159) can be treated analogously.) Reading (i) is obtained by doing as little as possible: merely applying 1AR to \textit{live in}, and combining the result with the basic translation of every Italian city and the basic translation of who. This leads to the following translation of the relative clause who lives in every Italian city.

(160) \[ \lambda Q \lambda x. [\forall x [\text{ITALIAN-CITY}(v) \rightarrow \text{LIVE}(v)(\forall x)]] \]

In order to get reading (ii) of (158), we apply 2VR and 1AR (in that order) to \( \lambda x \lambda y. \text{LIVE}(\forall x)(\forall y) \). The result is of type ((s,e^2),e^3):

(161) \[ \lambda T \lambda W. \forall T[(\forall x.\forall W[(\forall y.\text{LIVE}(\forall x)(\forall y))] \]

This is applied to \( \forall P. \forall v [\text{ITALIAN-CITY}(v) \rightarrow \forall P(\forall v)] \), and the result reduces to:

(162) \[ \lambda W. \forall v [\text{ITALIAN-CITY}(v) \rightarrow \forall W[(\forall y.\text{LIVE}(v)(\forall y))] \]

Now we let the basic translation of who undergo 3AR and 1AR, respectively, and apply the outcome, \( \lambda U \lambda Q \lambda T. \forall U[(\forall P. \forall T[(\forall x.\forall P(x))] \]

and then to the basic translation of girl, \( \lambda x. [\text{GIRL}(\forall x) \land \exists u(x = u)] = \text{GIRL}^* \). Note that in this case the translation of the relative clause is not of the basic type \( t_B(\text{REL}) = ((s,((s,e),t)),((s,e),t)) \), but of the derived type ((s,((s,e),t)),((s,e^2),t)). Thus, the compound common noun girl who lives in every Italian city is translated as:

(163) \[ [\lambda U \lambda Q \lambda T. \forall U[(\forall P. \forall T[(\forall x.\forall P(x))] \]
\[ \forall T[(\forall x.\forall [\text{GIRL}^*(x) \land \text{LIVE}(v)(\forall x))]]] \]
\[ \forall T[(\forall x.\forall [\text{GIRL}^*(x) \land \text{LIVE}(v)(\forall x))]] \]

Next, 1AR is applied to the basic translation of the subject determiner \( a \): \( \lambda B \lambda Q. \forall B[(\forall P. \exists x[\forall P(x) \land \forall Q(x))] \); this is combined with (163) and \( \text{WALK}^* = \lambda x. \text{WALK}(\forall x) \).

(164) \[ [\lambda B \lambda Q. \forall B[(\forall P. \exists x[\forall P(x) \land \forall Q(x))]] \]
\[ \forall T[(\forall x.\forall [\text{GIRL}^*(x) \land \text{LIVE}(v)(\forall x))]] \]
\[ \forall T[(\forall x.\forall [\text{GIRL}^*(x) \land \text{LIVE}(v)(\forall x))]] \]
\[ \forall T[(\forall x.\forall [\text{GIRL}^*(x) \land \text{LIVE}(v)(\forall x))]] \]

How is the fact that relative clauses constitute scope islands to be accounted for? The solution is very simple: we just add a condition which requires that all basic translations of relative clauses be of type \( t_B(\text{REL}) = ((s,((s,e),t)),((s,e),t)) \).

(165) For all \( \alpha_{REL} \) and \( \alpha' \in \text{Tr}_B(\alpha) \): \( \text{TYPE}(\alpha') = ((s,((s,e),t)),((s,e),t)) \)
This is sufficient for turning relative clauses into barriers for quantification and coordination scope. Moreover, an interesting by-product of imposing (165) as a general restriction on the basic translations of relative clauses is that relative clauses are also predicted to be barriers for unbounded dependencies like topicalization and constituent question formation, witness (166) and (167).

(166) *Every Italian city a girl who loves walks
(167) *Which Italian city a girl who loves walks?

In all semantically interpreted theories of unbounded dependencies, the semantic type assigned to gap-containing expressions of category $\alpha$ differs (of course) from the type assigned to their gapless namesakes and, hence, from the basic type of $\alpha$, $t_8(\alpha)$. In Gazdar, Klein, Pullum and Sag (1985, p. 230), for instance, an expression of category $\alpha$ containing a gap of category $\beta$ is assigned the type $(\text{TYPE}(\beta), \text{TYPE}(\alpha))$. Therefore, condition (165) will rule out the existence of gap-containing relative clauses.21

Another kind of structural restriction on the availability of scope readings is the phenomenon of scope sieves: quantified noun phrases occurring inside expressions of sieve categories cannot have their scope at the level of the sieve. The existence of scope sieves has been pointed out by Asher and Bonevac (1985, 1987), Muskens (1989)21 and Van der Does (1990). A sentence like (168), which contains a neutral perception verb (i.e., see plus a naked infinitival complement), cannot be interpreted as in (i). The embedded subject must have scope in the matrix sentence, cf. (ii).

(168) John sees every man swim
(i) $\text{SEE}(j, \forall v[\text{MAN}(v) \rightarrow \text{SWIM}(v)])$
(ii) $\forall v[\text{MAN}(v) \rightarrow \text{SEE}(j, \text{SWIM}(v))]$

Note that we cannot characterize the correct readings by claiming that neutral perception verbs may not take scope over quantified noun phrases. E.g., in a perfectly acceptable *de dicto* reading of (169), (i), the verb *see* has scope over the determiner *a*. (169) also has a

---

20 (Sequel to footnote 19.) We saw that an account using separate rules of generalized conjunction and disjunction has the effect of turning coordinations into 'floating' scope islands.

Given the observation about unbounded dependencies made here, we shall also predict the existence of 'Across-the-Board' phenomena (cf. Gazdar c.a. (1985), pp.176–178):

*The doctor who Kim worked for and Sandy relied on died*

*I.e., coordinations may contain gaps, but if they do, then both coordinated constituents must contain them. (The presence of only one gap would lead to different types of $\alpha'$ and $\beta'$, and hence lead to inapplicability of generalized conjunction and disjunction.)*

21 In Muskens (1989), the bad readings are ruled out with the help of a semantic condition: the interpretation of the naked infinitival complement must be persistent. This works because syntactic variables are used in the representation of scope ambiguities, and these do not disturb the persistence of the interpretation of the complement.
correct de re reading, (ii). The intermediate reading (iii), however, in which a pen is interpreted de dicto with respect to see, but de re as regards seek, is out.

\[(169)\] John sees Mary seek a pen
\[(i)\] \(\text{SEE}(j,\lambda\text{SEEK}(\lambda P. \exists u[\text{PEN}(u) \land \neg P(u)])(m))\)
\[(ii)\] \(\exists u[\text{PEN}(u) \land \text{SEE}(j,\lambda\text{SEEK}(\lambda P. \neg P(u))(m))]\)
\[(iii)\] \(\text{SEE}(j,\lambda\exists u[\text{PEN}(u) \land \text{SEE}(\lambda P. \neg P(u))(m)])\)

Within our flexible framework, the representation of this phenomenon is straightforward. The only assumption needed is that the naked infinitival forms of verbs which are translated as \(\alpha\) of type \((\vec{a},t)\), are lexically translated as \(\lambda \vec{x} \lambda w. \lambda (s,(s,(t,t),t)) \cdot w([\alpha(\vec{x})])\) of type \((\vec{a},t^2) = (\vec{a},((s,(t,t),t),t)).\) I.e., their basic translation looks like a VR term. (Maybe this is the right translation for verbal stems in general; the usual type \((\vec{a},t)\), then, results from morphologically filling the last \((s,(t,t),t))\)-argument with tenses and aspects like to, -ed or -ing.)

For example, the naked infinitive seek is assigned the basic translation \(\lambda T \lambda \vec{x} \lambda w. \lambda (\text{SEEK}(T)(\vec{y}))\). If this is applied to the intension of \(\lambda P. \exists u[\text{PEN}(u) \land \neg P(u)]\) and \(m\), respectively, and the result is combined with \(\lambda R \lambda x. \lambda R(\lambda p. \text{SEE}(p)(\vec{x}))\) (the term obtained by raising the first argument of \(\lambda p \lambda x. \text{SEE}(p)(\vec{x})\), the basic translation of see) and the basic translation \(j\) of John, then reading (i) of sentence (169) is generated. If, in addition, the first argument of the naked infinitive translation is lowered and raised, we get (ii). It is impossible to obtain (iii). This approach also immediately accounts for principles like conjunction and disjunction distribution which have been claimed to hold for naked infinitives (as early as Barwise (1981)). In sentences like John sees Mary walk and (or) Bill talk, the ‘higher order’ translations of Mary walk and Bill talk will force the verb sees to distribute over the conjunction (disjunction), and hence John sees Mary walk and (or) John sees Bill talk will be entailed without further assumptions.

5. DISCUSSION: COMPOSITIONALITY AND CONTEXTUALITY

We have tried to show above that the adoption of flexible type assignment in Montague grammar leads to an empirically adequate semantic theory of quantification and coordination that can do without rules of quantification: the rules for flexible type assignment – value raising, argument raising and argument lowering – are empirically motivated. But one might also ask oneself whether there is a conceptual motivation for them: why these rules?

As for Montague’s rules of quantification, the situation is more or less clear. The principle of compositionality of meaning/reference entails that in all cases where a non-lexical ambiguity is found that cannot be reduced to different syntactic structures, a syntactic ambiguity needs to be ‘forced’. Montague implemented this as a derivational ambiguity: whenever an ambiguity cannot be accounted for in terms of the syntactic
structure (de dicto/de re and other scope ambiguities constitute clear examples), apparently different ways of constructing one and the same syntactic structure are at issue.

The grammar presented here is different in this respect. It is clear that non-lexical ambiguities do not have syntactic repercussions within this grammar: instead, one syntactic object is assigned a set of interpretations. However, in representing scope ambiguities no use is made of ‘artificial’ alternative ways of construction. We can restrict ourselves to the ‘intuitive’ syntactic structure, which is respected by all the syntactic operations of the fragment: constituent expressions are real parts, so the flexible grammar is even ‘more compositional’ than its predecessors (cf. Janssen (1983), pp. 65-66). Just as in PTQ, the meanings of a compound expression are determined by (i) the meanings of its constituent parts, and (ii) the semantic operation associated with the compounding syntactic operation, but with the difference that we can no longer speak about the meanings of the constituent parts. These expressions each have a set of interpretations, of which (and this is what matters) only the ‘mutually fitting’ are allowed to join (cf. the remark on partiality in section 3.2.2).

This is an important difference. We noted before that value raising can be used to anticipate, so to speak, the global semantic structure in which expressions are embedded. The same can be said with respect to argument raising and lowering, and the local semantic structure. These rules enable expressions to adapt themselves to the format of the arguments or functor they are combined with. In other words: besides the principle of compositionality, the principle of contextuality seems to play a role: ‘(...) nach der Bedeutung der Wörter muss in Satzzusammenhängen, nicht in ihrer Vereinzelung gefragt werden (...)’ (Frege (1884), p. XXII).22

A strong interpretation of the principle of contextuality is involved here. Dummett discusses the status of the principle in Frege’s work, and arrives at what we might call a weak interpretation of the principle: ‘In a certain sense, sentences have a primacy within language over other linguistic expressions: a sentence is determined as true under certain conditions, which conditions are derivable from the way in which the sentence is constructed out of its constituent words; and the senses of the words relate solely to this determination of the truth conditions of the sentences in which the words may occur. Of course, looked at in one way, the word has a sense independently of any particular sentence in which it occurs: but its sense is something relating entirely to the occurrence of the word in a sentence (...).’ (Dummett (1976), pp.194-195, italics added). Under this weak interpretation (which, for that matter, seems to be the interpretation intended by Frege) the principle of contextuality is best considered a heuristic principle: in order to find out what the meaning of an expression is (‘independently of any particular sentence in which it occurs’), we must study the occurrence of that expression in the context of sentences. The fact that within the grammar of PTQ all expressions are assigned

22 ‘The meaning of words can only be found in the context of sentences, not when they are studied in isolation.’
denotations of a relational type $\langle \bar{a}, t \rangle$ illustrates the point: the denotation of expressions is identified with their contribution to the truth conditions of the sentences in which they occur.

The strong interpretation of the principle of contextuality explicitly denies the existence of the meaning of an expression, independent of the sentences in which that expression occurs. This is the interpretation which (in spite of himself) is expressed by Frege’s statement ‘Nur im Zusammenhange eines Satzes bedeuten die Wörter etwas.’ (Frege (1884), p. 73). In different sentences an expression can contribute in different ways to the truth conditions of the whole in which it occurs, and those contributions do not share a ‘greatest common divisor’. In the flexible grammar presented above this is the case: take, for instance, the expression finds. In sentence (180), the contribution of this expression can be formalized as (181). In sentence (182), however, it contributes in two ways: (183) and (184).

(180) John finds Mary
(181) $\lambda y \lambda x. \text{FIND}(\langle y \rangle)(\langle x \rangle)$
(182) Every man finds a woman
(183) $\lambda T_1 \lambda T_2. \langle T_2(\langle \lambda y. \langle T_1(\langle \lambda x. \text{FIND}(\langle x \rangle)(\langle y \rangle)\rangle)\rangle)\rangle$
(184) $\lambda T_1 \lambda T_2. \langle T_1(\langle \lambda x. \langle T_2(\langle \lambda y. \text{FIND}(\langle x \rangle)(\langle y \rangle)\rangle)\rangle)\rangle$

The fact that (181), the contribution to (180), is called the ‘basic translation’ in the grammar, is doubly misleading. First, this basic translation is not more essential or necessary than the derived translations in that it plays a role in all sentences in which the expression occurs: in (182), only derived translations are used. And second, in all cases where basic translations play a role, this role could just as well be taken over by derived translations. For example, in the case of (180) we could proceed as in (185) or (186),

(185) $\lambda T_1 \lambda T_2. \langle T_2(\langle \lambda y. \langle T_1(\langle \lambda x. \text{FIND}(\langle x \rangle)(\langle y \rangle)\rangle)\rangle)\rangle (\langle \lambda P. \langle P(\langle m \rangle)\rangle) (\langle \lambda P. \langle P(\langle j \rangle)\rangle)$
(186) $\lambda T_1 \lambda T_2. \langle T_1(\langle \lambda x. \langle T_2(\langle \lambda y. \text{FIND}(\langle x \rangle)(\langle y \rangle)\rangle)\rangle)\rangle (\langle \lambda P. \langle P(\langle m \rangle)\rangle) (\langle \lambda P. \langle P(\langle j \rangle)\rangle)$,

but also in infinitely many different ways, which would without exception result in something equivalent to FIND(m)(j).

---

23 ‘It is only in the context of a sentence that words have a meaning.’
APPENDIX: SEMANTIC PROPERTIES OF THE FRAGMENT

The fragment

(1) \( \text{t}_B(S): \quad t \quad \text{t}_B(TV): \quad ((s,e),((s,e),t)) \)
\( \text{t}_B(T): \quad e \quad \text{t}_B(PV): \quad ((s,t),((s,e),t)) \)
\( \text{t}_B(CN), \text{t}_B(IV): \quad e^1 \quad \text{t}_B(Det): \quad ((s,e^1),e^2) \)

(2) \( T(C) \) is the smallest set such that \( \text{t}_B(C) \in T(C) \) and \( \text{length}(\alpha') = i - 1 \):
\( [^1\text{VR}] \) if \((\alpha',b) \in T(C)\), then \((\alpha',b^2) \in T(C)\);
\( [^1\text{AR}] \) if \((\alpha',((s,b),(c',t))) \in T(C)\), then \((\alpha',((s,b^2),(c',t))) \in T(C)\);
\( [^1\text{AL}] \) if \((\alpha',((s,b^2),(c',t))) \in T(C)\), then \((\alpha',((s,b),(c',t))) \in T(C)\).

(3) \( \text{Tr}(\alpha) \) is the smallest set such that \( \text{Tr}_B(\alpha) \subseteq \text{Tr}(\alpha) \) and \( \text{length}(\alpha') = i - 1 \):
\( [^1\text{VR}] \) if \( \alpha' \in \text{Tr}(\alpha) \), then \( \lambda x\lambda w. \gamma^{\text{w}}(\gamma'(x)) \in \text{Tr}(\alpha) \);
\( [^1\text{AR}] \) if \( \alpha' \in \text{Tr}(\alpha) \), then \( \lambda x\lambda w. \gamma^{\text{w}}(\gamma z. \alpha'(x)(z)(y)) \in \text{Tr}(\alpha) \);
\( [^1\text{AL}] \) if \( \alpha' \in \text{Tr}(\alpha) \), then \( \lambda x\lambda w. \gamma^{\text{w}}(\gamma z. z)(y) \in \text{Tr}(\alpha) \).

(4) For lexical expressions: \( \text{Tr}_B(\alpha) = \{ \text{tr}_B(\alpha_{\text{type}}(\text{tr}_B(\alpha)))\} \): cf. (84).

(5) \( \gamma \) of type \((s,a),b) \) and \( \delta \) of type \( a \): \( \text{FA}(\gamma,\delta) = \gamma^{\text{\delta}} \)
\( \gamma \) and \( \delta \) of type \((\alpha',t)\): \( \text{GC}(\gamma,\delta) = \lambda x. [\gamma(x) \land \delta(x)] \)
\( \gamma \) and \( \delta \) of type \((\alpha',t)\): \( \text{GD}(\gamma,\delta) = \lambda x. [\gamma(x) \lor \delta(x)] \)

(6) \( \text{Tr}_B(\alpha_0) = \{ \text{FA}(\alpha',\beta'): \alpha' \in \text{Tr}(\alpha) \) and \( \beta' \in \text{Tr}(\beta)\} \):
\( (i) \beta_T + \alpha_{IV} \rightarrow \alpha_0 = [ S \beta_T \alpha_{IV} ] \)
\( (ii) \alpha_{TV} + \beta_T \rightarrow \alpha_0 = [ IV \alpha_{TV} \beta_T ] \)
\( (iii) \alpha_{Det} + \beta_{CN} \rightarrow \alpha_0 = [ T \alpha_{Det} \beta_{CN} ] \)
\( (iv) \alpha_{PV} + \beta_S \rightarrow \alpha_0 = [ IV \alpha_{PV} \beta_S ] \)
\( \text{Tr}_B(\alpha_0) = \{ \text{GC}(\alpha',\beta'): \alpha' \in \text{Tr}(\alpha) \) and \( \beta' \in \text{Tr}(\beta)\} \):
\( (v) \alpha_S + \beta_S \rightarrow \alpha_0 = [ S \alpha_S and \beta_S ] \)
\( (vi) \alpha_{IV} + \beta_{IV} \rightarrow \alpha_0 = [ IV \alpha_{IV} and \beta_{IV} ] \)
\( (vii) \alpha_T + \beta_T \rightarrow \alpha_0 = [ T \alpha_T and \beta_T ] \)
\( (viii) \alpha_{TV} + \beta_{TV} \rightarrow \alpha_0 = [ TV \alpha_{TV} and \beta_{TV} ] \)
\( \text{Tr}_B(\alpha_0) = \{ \text{GD}(\alpha',\beta'): \alpha' \in \text{Tr}(\alpha) \) and \( \beta' \in \text{Tr}(\beta)\} \):
\( (ix) \alpha_S + \beta_S \rightarrow \alpha_0 = [ S \alpha_S or \beta_S ] \)
\( (x) \alpha_{IV} + \beta_{IV} \rightarrow \alpha_0 = [ IV \alpha_{IV} or \beta_{IV} ] \)
\( (xi) \alpha_T + \beta_T \rightarrow \alpha_0 = [ T \alpha_T or \beta_T ] \)
\( (xii) \alpha_{TV} + \beta_{TV} \rightarrow \alpha_0 = [ TV \alpha_{TV} or \beta_{TV} ] \)
Value raising \cap Argument raising

VR and AR overlap. If \((\bar{a}, b) \Rightarrow^{i} VR (\bar{a}', b^2)\) with \(b = ((s, c), t)\), then the type transition can also be achieved using \(1^{i} AR: (\bar{a}', ((s, c), t)) \Rightarrow (\bar{a}', ((s, c), t^2))\). The translations assigned are equivalent: \(\lambda \bar{x} \lambda \bar{w}. \forall w(\langle\alpha'(\bar{x})\rangle) \iff \lambda \bar{x} \lambda \bar{w}. \forall w(\langle\lambda z. \alpha'(\bar{x})(z)\rangle)\). Therefore, we can at will assume that applications of VR are ‘real VR’, i.e., \(b \neq ((s, c), t)\).

Argument lowering

It can be shown that in the derivation of a translation all applications of AL can be ordered before AR and VR (a corollary of (I)); that AL can be assumed to have applied only to lexical translations (a corollary of the conjunction of (I) and (II)); and that the relative order of different AL-applications to a translation is semantically irrelevant (III).

(I) (a) \(1^{i} AL(i^{i} VR(\alpha)) \iff \alpha\),
(b) \(1^{i} AL(i^{i} AR(\alpha)) \iff \alpha\),
(c) If \(i \neq j\), then \(1^{i} AL(i^{j} VR(\alpha)) \iff 1^{j} VR(1^{i} AL(\alpha))\),
(d) If \(i \neq j\), then \(1^{i} AL(i^{j} AR(\alpha)) \iff 1^{j} AR(1^{i} AL(\alpha))\).

(a) \(1^{i} AL\) is applied immediately after \(i^{i} VR\): then \(type(i^{i} VR(\alpha)) = (\bar{a}', ((s, c), t))\), with \(length(\bar{a}') = i-1\). So \(i^{i} VR\) is not an instance of real VR, and the case reduces to case (b):
(b) \(1^{i} AL\) is applied immediately after \(i^{i} AR\): \(type(i^{i} AR(\alpha)) = (\bar{a}', ((s, b^2), (c', t')))\), and \(type(\alpha) = type(1^{i} AL(i^{i} AR(\alpha))) = (\bar{a}', ((s, b), (c', t)))\), with \(length(\bar{a}') = i-1\). Prove that \(1^{i} AL(i^{i} AR(\alpha)) \iff \alpha\) by \(\lambda\)-conversion and \(\forall\)-elimination:
\[1^{i} AL(i^{i} AR(\alpha)) = \lambda \bar{x} \lambda \bar{w}. \forall \bar{y}. [\lambda \bar{x}' \lambda \bar{w}' \lambda \bar{y}' . \forall w(\langle\lambda z. \alpha'(\bar{x}')(z')(\bar{y}')(\bar{y})\rangle)(\bar{x})(\langle\lambda z. \forall z(\bar{w})(\bar{y})\rangle)(\bar{y}) \iff \alpha].\]
(c) \(1^{i} AL\) is applied immediately after \(i^{j} VR\), \(i \neq j\): \(type(i^{j} VR(\alpha)) = (\bar{x}, d^2)\), length(\(\bar{x}\)) = \(j-1\). Note that \(i < j\), for the type \((\bar{x}, d^2)\) has only length(\(\bar{x}\)) + 1 = \(j\) arguments. Hence \((\bar{x}, d^2) = (\bar{a}', ((s, b^2), (c', d^2)))\) and \(type(\alpha) = type(1^{i} AL(i^{j} VR(\alpha))) = (\bar{a}', ((s, b), (c', d^2)))\).

Prove that \(1^{i} AL(i^{j} VR(\alpha)) \iff 1^{j} VR(1^{i} AL(\alpha))\) by \(\lambda\)-conversion and \(\forall\)-elimination.

(d) \(1^{i} AL\) is applied immediately after \(i^{j} AR\), \(i \neq j\). Distinguish the cases \(i > j\) and \(i < j\). Prove that \(1^{i} AL(i^{j} AR(\alpha)) \iff 1^{j} AR(1^{i} AL(\alpha))\) by \(\lambda\)-conversion and \(\forall\)-elimination.

(II) (a) \(1^{i} AL(FA(\alpha, \beta)) = FA(1^{i+1} AL(\alpha, \beta))\)
(b) \(1^{i} AL(GC(\alpha, \beta)) = GC(1^{i} AL(\alpha), 1^{i} AL(\beta))\)
(c) \(1^{i} AL(GD(\alpha, \beta)) = GD(1^{i} AL(\alpha), 1^{i} AL(\beta))\)

Proofs by \(\lambda\)-conversion and \(\forall\)-elimination. E.g. (b): \(1^{i} AL(GC(\alpha, \beta)) = \lambda \bar{x} \lambda \bar{w}. \lambda \bar{y}. [\lambda \bar{x}' \lambda \bar{w}' . \langle\alpha(\bar{x}')(\bar{w})(\bar{y})\rangle . \langle\beta(\bar{x}')(\bar{w})(\bar{y})\rangle]\), with \(\forall \bar{w}. [\lambda \bar{x}' . \langle\alpha(\bar{x}')\rangle . \langle\beta(\bar{x}')\rangle]\), and \(GC(1^{i} AL(\alpha), 1^{i} AL(\beta)) = \langle\lambda z. \forall z(\bar{w})(\bar{y})\rangle \iff \langle\lambda z. \forall z(\bar{w})(\bar{y})\rangle\).

(III) \(1^{i} AL(1^{j} AL(\alpha)) \iff 1^{i} AL(1^{j} AL(\alpha))\).

Prove that \(1^{i} AL(1^{j} AL(\alpha)) \iff 1^{i} AL(1^{j} AL(\alpha))\) by \(\lambda\)-conversion and \(\forall\)-elimination.
Combinability of types

(IV) If types \(a', f' = (s,a'), b\) have been derived from types \(a, f\), respectively, by means of VR and AR, then (i) the derivation \(a \Rightarrow a'\) does not contain any real \(^1\)VR, or (ii) the derivation \(f \Rightarrow f'\) does not contain any real \(^1\)VR.

Proof: Suppose both derivations contain real \(^1\)VR. Consider the last real \(^1\)VR applied in the derivation \(a \Rightarrow a'\): \(a_j \Rightarrow \) \(^1\)VR \(a_{j+1} = a_j^2\). Since \(^1\)VR is real, \(a_j\) is not of the form \(((s,a_j^2p+1),w)\). Consider the last real \(^1\)VR applied in \(f \Rightarrow f'\): \(f_k \Rightarrow \) \(^1\)VR \(f_{k+1} = f_k^2\). Since \(^1\)VR is real, \(f_k\) is not of the form \(((s,x),t)\). After that, only \(^1\)VR, \(^1\)AR \((i > 1)\) and \(^1\)AR \((p, \text{ pe } \mathbb{N})\) have applied, so \(a' = ((s,a_j^2p+1),w)\). Consider the last real \(^1\)VR applied in \(f \Rightarrow f'\): \(f_k \Rightarrow \) \(^1\)VR \(f_{k+1} = f_k^2\). Since \(^1\)VR is real, \(f_k\) is not of the form \(((s,x),t)\). After that, only \(^1\)VR, \(^1\)AR \((i > 1)\) and \(^1\)AR \((q, \text{ times})\) have applied, so \(f' = ((s,f_k^2q+1),y)\). Since \(f_k^2q+1 = a_q = ((s,a_j^2p+1),w)\), we know that \(w = t\), and \(f_k^2q = a_j^2p+1\). But then it must hold that either \(f_k\) (if \(q \leq p\)) or \(a_j\) (if \(q > p\)) is of the form \(((s,x),t)\). Contradiction.

(V) If types \(a', f' = (s,a'), c\) have been derived from types \(x, (s,x), b\) [call such types 'fitting'], respectively, by VR and AR, then \(f \Rightarrow f'\) does not contain real \(^1\)VR.

Proof: If \(x \Rightarrow a'\) contains real \(^1\)VR, then \(f \Rightarrow f'\) does not contain real \(^1\)VR (use (IV)). So, assume that \(x \Rightarrow a'\) does not contain real \(^1\)VR, but that \(f \Rightarrow f'\) does. Note that the latter entails that \(a' = ((s,x),t)\). Now, if \(x \Rightarrow a'\) contains \(^1\)VR, \(^1\)AR \((i > 1)\) and \(\text{ no } \(^1\)VR\), we have that \(a' = ((s,x),t)\). Hence \(x \Rightarrow a'\) does not contain real \(^1\)VR and \(^1\)VR, \(^1\)AR for \(i > 1\), but only \(^1\)AR \((r, \text{ times})\), leading to a type \(((s,x),t)\), so \(a'' = ((s,x^2r),t)\). On the other hand, the first real \(^1\)VR (after \(q\) times \(^1\)AR) in \(f = ((s,x), b) \Rightarrow f' = ((s,c'),c)\) entails that \(a'' = ((s,(s,x^2q),d),t)\) (where \(d \neq t\)) as a subtype. Contradiction.

(VI) If expression \(\alpha\) of category C translates as \(\alpha'\) of type \(a'\) \([\alpha'_a\]\), then \(t_B(C) \Rightarrow a'\).

Proof by induction on the complexity of \(\alpha\): (i) the claim is trivial for lexical expressions.

(\(ii\)) \(GC/GD:\ \alpha_C = \beta C and \gamma_C\) or \(\alpha_C = \beta C or \gamma C\) and \(\alpha'_C\) is derived from \(GC/GD(\beta',\gamma'_C)\). a \(\Rightarrow a'.\) By induction hypothesis \((IH)\): \(t_B(C) \Rightarrow \text{type}(\beta')\) and \(t_B(C) \Rightarrow \text{type}(\gamma')\). Note that type(\(\beta'\)) = type(\(\gamma'\)) = a. So \(t_B(C) \Rightarrow a'\). \(\alpha''\) is derived from \(FA(\beta',\gamma'_C)\) a \(\Rightarrow a'.\) IH: \(t_B(IV) = (s,(x),t) \Rightarrow \text{type}(\beta') = (s,(s,(x),t)) \Rightarrow \text{type}(\beta') = (s,(s,(x),t))\)

(\(\alpha\)) \(\alpha_S = \gamma_S T_B(1)\) and \(\alpha'_S\) is derived from \(FA(\beta',\gamma'_C)\) a \(\Rightarrow a'.\) IH: \(t_B(TV) = (s,(s,(x),t)) \Rightarrow \text{type}(\beta') = (s,(s,(x),t))\)

(c) \(\alpha_T = [\beta_T \gamma_T C_T]\) and \(\alpha'_T\) is derived from \(FA(\beta',\gamma'_C)\) a \(\Rightarrow a'.\) IH: \(t_B(TD) = (s,(s,(x),t)) \Rightarrow \text{type}(\beta') = (s,(s,(x),t))\)

(d) \(\alpha_I = [\beta_I \gamma_I C_I]\) and \(\alpha'_I\) is derived from \(FA(\beta',\gamma'_C)\) a \(\Rightarrow a'.\) IH: \(t_B(TF) = (s,(s,(x),t)) \Rightarrow \text{type}(\beta') = (s,(s,(x),t))\)

Note that in (a) through (d) (V) can be applied: the types \(t_B(C_B)\) and \(t_B(C_P)\) are fitting [and not subject to \(^1\)AL]: \(t_B(C_B) = (s,(s,C_B),d)\), so \(t_B(C_B) \Rightarrow \text{type}(\beta') = (s,(s,(x),t))\) contains no \(^1\)VR. But this entails that \(d = a [\text{ since the application of } \text{to a type } ((s,x),y) \text{ does not affect } y]\), and hence \(d \Rightarrow a'.\) Thus: (a) \(d = t = t_B(S) \Rightarrow a'\); (b) \(d = (s,(x),t) = t_B(IV) \Rightarrow a'\); (c) \(e^2 = ((s,(s,(x),t)),t) \Rightarrow \text{type}(\beta') = (s,(s,(x),t))\) and \(e = t_B(T) \Rightarrow \text{type}(\beta') = (s,(s,(x),t))\) when \(a'\).
Corollary: apart from AL, the fact that the fragment is fitting entails that the argument categories of the fragment [S,T,CN] only undergo 1VR; that unary non-main functor categories [IV,Det] only undergo 1AR and 2VR; and that binary non-main functor categories [TV, PV] only undergo 1AR, 2AR, and 3VR. Moreover, observe that \( c \Rightarrow t \) iff \( c = t \) (all type change leads to compound types), so in sentence readings [interpretations of type t, cf. (106) above] unary main functor categories [IV,Det] only undergo 1AR, and binary main functor categories [TV, PV] only undergo 1AR and 2AR.

**Value raising and argument raising**

Applications of AR and VR within functional application structures can be shifted back to the functor. It follows that in pure application structures (structures which contain no generalized conjunction and disjunction) only lexical type change is needed.

(VII) \( i^1\text{AR}(FA(\alpha, \beta)) \Leftrightarrow FA(\text{i}^1\text{AR}(\alpha), \beta) \)  
\( i^1\text{VR}(FA(\alpha, \beta)) \Leftrightarrow FA(\text{i}^1\text{VR}(\alpha), \beta) \)

Proof: \( i^1\text{AR}(FA(\alpha, \beta)) = \lambda \bar{x}\lambda w\bar{y}.\gamma.\psi.\psi(\lambda z.\alpha(\bar{y}))(\bar{x})(\bar{y}) \). Note that since \( \alpha \Leftrightarrow \lambda v.\bar{x}\lambda w\lambda\bar{y}.\alpha(v)(\bar{x})(\bar{y}) \), we have \( i^1\text{AR}(\alpha) \Leftrightarrow \lambda v.\bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda z.\alpha(v))(\bar{x})(\bar{y}) \), and \( FA(\text{i}^1\text{AR}(\alpha), \beta) = \text{i}^1\text{AR}(\alpha)(\bar{y}) \Leftrightarrow [\lambda v.\bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda z.\alpha(v))(\bar{x})(\bar{y})] \Leftrightarrow \lambda v.\bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda z.\alpha(\bar{y}))(\bar{x})(\bar{y}) \). (b) is analogous to (a).

When translations \( \varphi(d,(\bar{s},b_2),d) \) and \( \alpha(c,b_2) \) of a functor and (the \( j-1 \)-ary main functor of) its \( i \)-th argument are such that a final iVR-application to (the main functor of) that argument ‘answers’ an iAR-application to the functor, both applications can be omitted.

(VIII) \( \varphi = X_1(\ldots(X_n(\text{iAR}(\varphi_0)))\ldots) \) \( [X_1 = \text{PVR}, p > i \text{ or } \text{iAR}, q \neq i] \) and \( \alpha = \text{iAR}(\alpha_0) \), then for \( \varphi' = X_1(\ldots(X_n(\varphi_0)))\ldots \): \( \varphi(\bar{y})(\lambda \alpha(\bar{y})) \Leftrightarrow \varphi'(\bar{y})(\lambda \alpha(\bar{y})) \) \( [\text{length}(\bar{y})=i-1, \text{length}(\bar{y})=j-1] \). (\[\ldots\] will be used for gaps in contexts, <...> for substitutions.)

Proof: Note that \( \alpha \Leftrightarrow \lambda \bar{x}\lambda v.\psi.\psi(\lambda \alpha(\bar{y})), \varphi \Leftrightarrow \lambda \bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda \alpha(\bar{y})), \) and \( \varphi' \Leftrightarrow \lambda \bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda \alpha(\bar{y})), \) where \( \bar{y} \) does not occur in \( \alpha_0 \) and \( \bar{w} \) does not occur in \( \Phi \).

\( \varphi(\bar{y})(\lambda \alpha(\bar{y})) \Leftrightarrow \lambda \bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda \alpha(\bar{y})) \)
\( \Leftrightarrow \lambda \bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda \alpha(\bar{y})) \)
\( \Leftrightarrow \lambda \bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda \alpha(\bar{y})) \)
\( \Leftrightarrow \lambda \bar{x}\lambda w\lambda\bar{y}.\psi.\psi(\lambda \alpha(\bar{y})) \)
When translations \( \varphi(d,(\bar{s},b_2),d) \) and \( \alpha(c,b_2) \) of a functor and (the \( j-1 \)-ary main functor of) its \( i \)-th argument are such that two consecutive applications of JVR to (the main functor of) that argument ‘answer’ two – possibly non-consecutive – applications or iAR to the functor, we can omit one JVR and the ‘innermost’ iAR. [Similarly in the case where translations \( \varphi(d,(\bar{s},b_2),d) \) and \( \alpha(c,b_2) \) of a functor and (the \( j-1 \)-ary main functor of) its \( i \)-th argument contain two consecutive applications of iAR to the functor which ‘answer’ two – possibly non-consecutive – applications or JVR to (the main functor of) that argument. There we can skip one iAR and the ‘outermost’ JVR. Proof omitted.]
(IX) If \( \varphi = X_1(\ldots(X_k(\ldots(X_{k+i}(\ldots(X_{k+i+j}(\ldots(AR(X_{k+i+j}(\ldots(X_{k+i+1}(\ldots(X_{k+i}(\varphi_0(\ldots))\ldots))))\ldots))))\ldots))\ldots) [X_1=AR\ or\ pV\ (p > i)] \) and \( \alpha = Y_1(\ldots(Y_{m+i}(\ldots(Y_{m+i+j}(\ldots(VR(VR(\alpha_0(\ldots))\ldots))))\ldots)) [Y_1=AR\ or\ qV\ (q \geq j)] \), where

\( \#(iAR\ in\ X_{k+i+1}\ldotsX_{k+i+j}) = 0 \) and \( \#(iAR\ in\ X_1\ldotsX_k) = \#(iVR+iAR\ in\ Y_1\ldotsY_{m+i}) = r \), then for

\( \varphi' = X_1(\ldots(X_k(\ldots(X_{k+i+1}(\ldots(X_{k+i}(\varphi_0(\ldots))\ldots))))\ldots)) \) and \( \alpha' = Y_1(\ldots(Y_{m+i}(\ldots(Y_{m+i+j}(\ldots(AR(\varphi_0(\ldots))\ldots))))\ldots) \):

\( \varphi(\psi)^{\langle\alpha(\chi)\rangle} \iff \varphi'(\psi)^{\langle\alpha'(\chi)\rangle} \) [length(\( \psi \)) = i−1, length(\( \chi \)) = j−1].

\[ \text{Proof by induction on } r. \]

\( r = 0 \): Note that \( \varphi \iff \lambda\bar{x}\alpha\lambda\bar{y}\alpha\beta.\Phi_1[\psi(\bar{x}(\alpha(\psi)).\Phi_2[\psi(\bar{y}(\alpha(\psi)).\varphi_0)])] \), and that \( \varphi' \iff \lambda\bar{x}\alpha\lambda\bar{y}\alpha\beta.\Phi_1[\psi(\bar{y}(\alpha(\psi)).\varphi_0)] \), where \( \psi \) does not occur in \( \Phi_2 \). Moreover, \( \alpha \iff \lambda\bar{z}\alpha\lambda\bar{f}.\Lambda[\psi(\bar{f}(\alpha(\lambda\bar{f}.\alpha)))] \) and \( \alpha' \iff \lambda\bar{z}\alpha\lambda\bar{f}.\Lambda[\psi(\bar{f}(\alpha(\lambda\bar{f}.\alpha)))] \).
Generalized conjunction and disjunction

(VIII) generalizes to cases where the translation \( \varphi \) of the functor consists of (possibly iterated) coordinations (conjunctions and/or disjunctions) of translations \( \varphi_1, \ldots, \varphi_s \); when translations \( \varphi(\mathcal{A}_{(s,b)},d) \) and \( \alpha(\mathcal{B}_{b}) \) of a functor with coordinates \( \{ \varphi_1, \ldots, \varphi_s \} \) and \( \{ \alpha_1, \alpha_2 \} \) of a functor with coordinates \( \{ \varphi_1, \ldots, \varphi_s \} \) and \( \alpha(\mathcal{B}_{b}) \), respectively, of a functor and (the \( j \)-ary main functor of) its \( i \)-th argument are such that a final JVR-application to (the main functor of) that argument ‘answers’ \( \text{iAR} \)-applications to all coordinates of the functor, these applications can be omitted.

(X) If \( \varphi = X_1(\ldots(X_n(\text{iAR}(\{ \varphi_1, \ldots, \varphi_s \})) \ldots)) \) \( \text{[X}_j = \text{GD, GC, PVR, p > i or \text{iAR}, q \neq j \text{]} \) and \( \alpha = \text{iVR}(\alpha_0) \), then for \( \varphi' = X_1(\ldots(X_n(\{ \varphi_1, \ldots, \varphi_s \})) \ldots) \): \( \varphi'((\alpha(\mathcal{B})) \Leftrightarrow \varphi((\alpha(\mathcal{B}))) \) \( \text{[length}(\mathcal{B}) = i - 1, \text{length}(\mathcal{B}) = j - 1 \).

Proof: \( \varphi \Leftrightarrow \lambda\mathcal{X}\lambda\mathcal{W}\lambda\mathcal{Y} \cdot \Phi[\mathcal{V}(\lambda\mathcal{W}\cdot \varphi_1), \ldots, \mathcal{V}(\lambda\mathcal{W}\cdot \varphi_s)] \), \( \alpha \Leftrightarrow \lambda\mathcal{X}\lambda\mathcal{V} \cdot \mathcal{V}(\alpha(\mathcal{B})) \), and \( \varphi' \Leftrightarrow \lambda\mathcal{X}\lambda\mathcal{W}\lambda\mathcal{Y} \cdot \Phi[\mathcal{V}(\lambda\mathcal{W}\cdot \varphi_1), \ldots, \mathcal{V}(\lambda\mathcal{W}\cdot \varphi_s)] \), \( \mathcal{Y} \) does not occur in \( \alpha_0 \) and \( \mathcal{W} \) does not occur in \( \Phi. \)

(IX) generalizes to cases where both translations \( \varphi \) and \( \alpha \) of a functor and (the \( j \)-ary main functor of) its \( i \)-th argument are (possibly iterated) coordinations of translations \( \varphi_1, \ldots, \varphi_s \) and \( \alpha_1, \ldots, \alpha_t \); when translations \( \varphi(\mathcal{A}_{(s,b)},d) \) and \( \alpha(\mathcal{B}_{b}) \) of coordinates \( \{ \varphi_1, \ldots, \varphi_s \} \) and \( \{ \alpha_1, \ldots, \alpha_t \} \), respectively, of a functor and (the \( j \)-ary main functor of) its \( i \)-th argument are such that two consecutive applications of \( \text{JVR} \) to all coordinates of (the main functor of) that argument ‘answer’ two – possibly non-consecutive – applications of \( \text{iAR} \) to all coordinates of the functor, we can omit one \( \text{JVR} \) and the ‘innermost’ \( \text{iAR}. \) Similarly in the case where translations \( \varphi(\mathcal{A}_{(s,b)},d) \) and \( \alpha(\mathcal{B}_{b}) \) of a functor and (the \( j \)-ary main functor of) its \( i \)-th argument contain two consecutive applications of \( \text{iAR} \) to all coordinates of the functor which ‘answer’ two – possibly non-consecutive – applications or \( \text{JVR} \) to all coordinates of (the main functor of) that argument. There we can skip one \( \text{iAR} \) and the ‘outermost’ \( \text{JVR}. \)

(XI) If \( \varphi = X_1(\ldots(X_k(\text{iAR}(X_{k+1}(\ldots(X_n(\text{iAR}(\{ \varphi_1, \ldots, \varphi_s \})) \ldots)))) \ldots) \) \( \text{[X}_j = \text{GD, GC, AR or \text{PVR, p > i \}] \) and \( \alpha = Y_1(\ldots(Y_m(\text{JVR}(\{ \alpha_1, \ldots, \alpha_t \}))) \ldots) \) \( \text{[Y}_j = \text{GD, GC, AR or \text{PVR, q \geq j \}] \) where \( \#(\text{iAR} \text{in } X_{k+1} \ldots X_n) = 0 \) and \( \#(\text{iAR} \text{in } X_{k+1} \ldots X_k) = \#(\text{JVR} + \text{iAR} \text{in } Y_{1} \ldots Y_m) = r, \) \text{then } \varphi((\alpha(\mathcal{B})) \Leftrightarrow \varphi'(\alpha(\mathcal{B}))) \) \( \text{for } \varphi' = X_1(\ldots(X_k(\text{iAR}(X_{k+1}(\ldots(X_n(\{ \varphi_1, \ldots, \varphi_s \})) \ldots)))) \ldots) \) and \( Y_1(\ldots(Y_m(\text{JVR}(\{ \alpha_1, \ldots, \alpha_t \}))) \ldots) \) \( = \alpha' \) \( \text{[length}(\mathcal{B}) = i - 1, \text{length}(\mathcal{B}) = j - 1 \).\)

Proof by induction on \( r \) (analogous to (IX)).

\( r=0: \) Since \( \varphi \Leftrightarrow \lambda\mathcal{X}\lambda\mathcal{V}\lambda\mathcal{Y} \cdot \Phi[\mathcal{V}(\lambda\mathcal{V}\cdot \varphi_1), \ldots, \mathcal{V}(\lambda\mathcal{V}\cdot \varphi_s)] \), \( \mathcal{V} \) does not occur in \( \alpha_0 \) and \( \mathcal{V} \) does not occur in \( \Phi. \)
\( \varphi' \leftrightarrow \lambda \vec{x} \lambda \nu \lambda \vec{y}. \Phi_1[\nu'('\lambda \nu'. \Phi_2[\varphi_1, \ldots, \varphi_3])] \) (where ' does not occur in \( \Phi_2 \)),
\( \alpha \leftarrow \lambda \vec{z} \lambda f. A[\nu'('\lambda \nu'. 'f('\lambda \nu'. \alpha_1')'), \ldots, 'f('\lambda \nu'. 'f('\lambda \nu'. \alpha_t')')], \) and
\( \alpha' \leftarrow \lambda \vec{z} \lambda f. A[\nu'('\lambda \nu'. \alpha_1'), \ldots, 'f('\lambda \nu'. \alpha_t')], \) we have \( \Phi(\psi)(\varphi(\vec{z})) \leftrightarrow \delta = \lambda \vec{y}. \Phi_1[A[\Phi_2[\varphi_1 <\nu'='':=':\lambda \nu'. \alpha_1]>], \ldots, \Phi_2[\varphi_3 <\nu'='':=':\lambda \nu'. \alpha_3]>]][\vec{z}:=':\vec{x}>][\vec{z}:=':\vec{y}>], \)
\( \ldots, A[\Phi_2[\varphi_8 <\nu'='':=':\lambda \nu'. \alpha_8]>], \ldots, \Phi_2[\varphi_8 <\nu'='':=':\lambda \nu'. \alpha_8]>][\vec{z}:=':\vec{x}>][\vec{z}:=':\vec{y}>], \)
and \( \Phi(\psi)(\varphi(\vec{z})) \leftrightarrow \varepsilon = \lambda \vec{y}. \Phi_1[A[\Phi_2[\varphi_1 <\nu'='':=':\lambda \nu'. \alpha_1]>], \ldots, \Phi_2[\varphi_8 <\nu'='':=':\lambda \nu'. \alpha_8]>][\vec{z}:=':\vec{x}>][\vec{z}:=':\vec{y}>]. \)

But because ' does not occur in \( \Phi_2 \): \( \delta \leftrightarrow \varepsilon. \)

\( r > 0 \): Since \( r > 0 \), we can assume that \( \alpha \leftarrow \lambda \vec{z} \lambda \nu \lambda \vec{x}. A[\nu'('\lambda \nu'. \varphi_1(\vec{x}')(\vec{y}')), \ldots, 'f('\lambda \nu'. \varphi_8(\vec{x}')(\vec{y}'))], \) and
\( \varphi \leftarrow \lambda \vec{x} \lambda \nu \lambda \vec{y}. \Phi[\nu'('\lambda \nu'. \varphi_1(\vec{x}')(\vec{y}')), \ldots, 'f('\lambda \nu'. \varphi_8(\vec{x}')(\vec{y}'))], \) for \( \nu \leq s, w \leq t, \) length(\( \vec{x} \)) = \( i-1 \), and length(\( \vec{z} \)) = \( j-1 \). Note that \( \varphi_1, \ldots, \varphi_v \) and \( \alpha_1, \ldots, \alpha_w \) are translations that have undergone \( r \) times \( 1 \)AR and \( 1 \)VR, respectively. So, we can apply the induction hypothesis: there are simpler terms \( \varphi_1', \ldots, \varphi_v' \) and \( \alpha_1', \ldots, \alpha_w' \) (with \( r-1 \) times \( 1 \)AR and \( 1 \)VR) such that for all \( i \) and \( j \), \( 1 \leq i \leq v, 1 \leq j \leq w: \varphi_i(\vec{z})(\alpha_j(\vec{z})) \leftrightarrow \varphi_i(\vec{x})(\alpha_j(\vec{x}').) \).

But then for \( \varphi'' = \lambda \vec{x} \lambda \nu \lambda \vec{y}. \Phi[\nu'('\lambda \nu'. \varphi_1(\vec{x}')(\vec{y}')), \ldots, 'f('\lambda \nu'. \varphi_8(\vec{x}')(\vec{y}'))], \) \( \alpha'' = \lambda \vec{z} \lambda w. A[\nu'('\lambda \nu'. \varphi_1(\vec{x}')(\vec{y}')), \ldots, 'f('\lambda \nu'. \varphi_8(\vec{x}')(\vec{y}'))], \) we have \( \Phi(\psi)(\varphi(\vec{z})) \leftrightarrow \Phi(\psi)(\varphi(\vec{z})). \)

These facts suffice to check the claims about numbers of readings for the sentences treated in the paper. They also suggest minimal ways of obtaining these readings: (i) apply \( 1 \)AL only to basic lexical translations; (ii) apply \( 1 \)VR and \( 1 \)AR only to translations of lexical items and coordinations; (iii) apply only \( 1 \)VR to \( 1 \)S, \( 1 \)T, and \( 1 \)CN; \( 1 \)AR, \( 2 \)AR and \( 3 \)VR to \( 2 \)TV and \( 2 \)PV; and \( 1 \)AR and \( 2 \)VR to \( 1 \)V and Det; (iv) apply no \( 1 \)AR to (coordinates of) the main functor of the sentence; (v) do no final applications of \( 1 \)VR to (the \( 1-\mathit{ary} \) main functor of) an \( i \)-th argument which are ‘answered’ by an application of \( 1 \)AR to (the coordinates of) its functor; (vi) do no consecutive applications of \( 1 \)VR to (the coordinates of) (the \( 1-\mathit{ary} \) main functor of) an \( i \)-th argument which are ‘answered’ by two applications of \( 1 \)AR to (the coordinates of) its functor; (vii) do no consecutive applications of \( 1 \)AR to (the coordinates of) a functor which are ‘answered’ by two applications of \( 1 \)VR to (the coordinates of) (the \( 1-\mathit{ary} \) main functor of) its \( i \)-th argument. Two examples:

(9) Mary walks and seeks every unicorn

(126) Fred claims that every schoolboy believes that a mathematician wrote Through the looking glass.

If we assume (for simplicity) that every unicorn is a lexical item, we see that there are five candidates for type change: the four lexical items and the coordination walks and seeks every unicorn. Seeks can undergo \( 1 \)AL. The proper name only gives rise to final \( 1 \)VR applications which have to be answered by \( 2 \)AR (on both conjoined verb phrases or on their conjunction), so no type change is needed here (v). The quantifier every unicorn necessitates \( 1 \)AR for lowered seeks. \( 1 \)VR for this noun phrase would be ‘final’ and ‘answered’ by \( 1 \)AR on seeks, hence the verb needs at most one \( 1 \)AR. For the conjoined
verb phrases and their conjunction, VR is ruled out (iv). We saw that 2AR on the conjoined verb phrases or their conjunction would have to be answered by a final application of 1VR to the proper name. So the choice is between <> (nothing) and <1AL,1AR> for seeks: two readings.

Sentence (126) does not give rise to argument lowerings; there are no coordinations, so the type change can be kept lexical. As for the term phrases, we see that the proper names Fred and Through the looking glass only give rise to final 1VR applications which have to be answered by their functors claims and wrote – no type change is needed here (v). However, the quantified noun phrases every schoolboy and a mathematician necessitate (‘unanswered’) applications of 2AR in their functors believes and wrote, respectively. Iteration of 2AR on these functors would involve applications of final 1VR to the quantifiers (v). For wrote, then, we have exactly one 2AR and zero 1AR. The remaining option, 3VR, is necessarily answered; hence it must be non-final (v) and non-consecutive (vi). So, we are left with (α) <2AR> and (β) <3VR,2AR>. For claims, we have zero 2AR. Besides, 3VR is impossible for this main functor of the sentence: value raising would not lead to a translation of type t (iv). 1AR will be answered; hence it must be non-consecutive (vii). I.e., (a) <> and (b) <1AR> remain. For believes, we have exactly one 2AR and two possibilities, 1AR and 3VR. The facts about wrote entail that 1AR will be applied at most once. 3VR is necessarily answered; hence it must be non-final. Moreover, we saw that applying it more than once would violate (vii). Therefore, the remaining options are (A) <2AR>, (B) <1AR,2AR>, (C) <2AR,1AR>, (D) <3VR,2AR>, (E) <2AR,3VR,1AR>, (F) <1AR,3VR,2AR>, (G) <3VR,2AR,1AR> and (H) <3VR,1AR,2AR>. Only the combinations (A)(α)(α), (B)(α)(β), (C)(α)(β), (D)(β)(α), (E)(β)(β), (F)(β)(β), (G)(β)(β) and (H)(β)(β) fit: eight readings.
REFERENCES


*Department of Philosophy*
*University of Amsterdam*
*Nieuwe Doelenstraat 15*
*1012 CP Amsterdam*
*The Netherlands*
The ITLI Prepublication Series

1990

Logic, Semantics and Philosophy of Language
LP-90-01 Jaap van der Does
LP-90-02 Jeroen Groenendijk, Martin Stokhof
LP-90-03 Renate Bartsch
LP-90-04 Aarne Ranta
LP-90-05 Patrick Blackburn
LP-90-06 Gennaro Chierchia
LP-90-07 Gennaro Chierchia
LP-90-08 Herman Hendriks
LP-90-09 Paul Dekker
LP-90-10 Theo M.V. Janssen

Mathematical Logic and Foundations
ML-90-01 Harold Schellinx
ML-90-02 Jaap van Oosten
ML-90-03 Yde Venema
ML-90-04 Maarten de Rijke
ML-90-05 Domenico Zambella

Computation and Complexity Theory
CT-90-01 John Tromp, Pieter van Emde Boas
CT-90-02 Sieger van Denneheuvel
Gerard R., Renardel de Lavalette
CT-90-03 Ricardo Gavaldà, Leen Torenvliet
Osamu Watanabe, José L. Balcázar

CT-90-04 Harry Buhrman, Leen Torenvliet

Other Prepublications
X-90-01 A.S. Troelstra
X-90-02 Maarten de Rijke
X-90-03 L.D. Beklemishev
X-90-04
X-90-05 Valentin Stehtman
X-90-06 Valentin Goranko, Solomon Passy
X-90-07 V.Yu. Shavrukov
X-90-08 L.D. Beklemishev
X-90-09 V.Yu. Shavrukov
X-90-10 Sieger van Denneheuvel
Peter van Emde Boas
X-90-11 Alessandra Carbone

A Generalized Quantifier Logic for Naked Infinitives
Dynamic Montague Grammar
Concept Formation and Concept Composition
Intuitionistic Categorial Grammar
Nominal Tense Logic
The Variability of Impersonal Subjects
Anaphora and Dynamic Logic
Flexible Montague Grammar
The Scope of Negation in Discourse, towards a flexible dynamic Montague grammar
Models for Discourse Markers

Isomorphisms and Non-Isomorphisms of Graph Models
A Semantical Proof of De Jongh's Theorem
Relational Games
Unary Interpretability Logic
Sequences with Simple Initial Segments

Associative Storage Modification Machines
A Normal Form for PCSJ Expressions

Generalized Kolmogorov Complexity
in Relativized Separations
Bounded Reductions

Remarks on Intuitionism and the Philosophy of Mathematics,
Revised Version
Some Chapters on Interpretability Logic
On the Complexity of Arithmetical Interpretations of Modal Formulae
Annual Report 1989
Derived Sets in Euclidean Spaces and Modal Logic
Using the Universal Modality: Gains and Questions
The Lindenbaum Fixed Point Algebra is Undecidable
Provability Logics for Natural Turing Progressions of Arithmetical Theories
On Rosser's Provability Predicate
An Overview of the Rule Language RL/1

Provable Fixed points in IΔ₀+Ω₁, revised version