Distinguishing Phenogrammar from Tectogrammar Simplifies the Analysis of Interrogatives

Vedrana Mihaliček and Carl Pollard

The Ohio State University, Department of Linguistics, Columbus OH 43210, USA
{vedrana, pollard}@ling.ohio-state.edu

Abstract. Up to you, Vedrana, I ran out of time!

1 Introduction

Oehrle (1994) introduced a categorial grammar (CG) architecture in which word order (not just meaning) is represented using the terms of a typed λ-calculus. Variants of this architecture have since been employed in a variety of CG frameworks, including abstract categorial grammar (ACG, de Groote 2001), λ-grammar (λG, Muskens 2003, 2007b), higher-order grammar (HOG, Pollard 2004), and pheno-tecto distinguished CG (PTDCG, Smith 2010). Some salient commonalities of these approaches include the following (i) a clear separation of tectogrammar (roughly, abstract syntactic combinatorics) and phenogrammar (roughly, word order)[1]; (ii) an implementation of Montague's (1974) 'lowering' analysis of quantification in terms of β-reduction in the phenogrammatical calculus; (iii) uniform treatment of medial and peripheral extraction by phenogrammatical lowering of the null string into the 'trace' position; and (iv) a tectogrammatical type system based on linear logic, made possible by the 'outsourcing' to the phenogrammar of much of the work done by directionality and/or multimodality of the tectogrammar in other CG frameworks (e.g. Moortgat 1997, Morrill et al 2007, Baldridge 2002).

Works such as Oehrle 1994, de Groote 2001 and Muskens 2003, 2007b are largely programmatic in nature. However, Smith 2010 shows by example that the pheno-tecto-distinguished style of CG (hereafter, PTG) can also be a practical framework for describing highly complex linguistic phenomena (specifically, remnant comparative and correlational comparative constructions) in a way that highlights the underlying simplicity of the combinatorics and compositional semantics, by representing the difference between lowering and extraction (or, in mainstream generative-grammar terms, between covert and overt movement) as purely phenogrammatical. In this paper, we try to make the same point, with

[1] The terms 'tectogrammar' and 'phenogrammar' are meant to suggest an affinity with the programmatic suggestions of Curry (1961), who employed the terms 'tectogrammatical structure' and 'phenogrammatical structure'. The analogous ACG (λG) notions are 'abstract syntax' ('combinatorics') and 'concrete syntax' ('syntax').

respect to a different set of phenomena, namely cross-linguistic variation in the form of interrogative sentences, with special attention to multiple constituent questions and so-called Baker ambiguities. Because of space limitations, we here omit inessential details and discuss only two languages, English and Chinese

Unlike other CG approaches, both mainstream and PT, we drop the traditional requirement that there be a function mapping tectogrammatical types to semantic types. Demanding that every semantic difference be reflected in tectogrammar, in our opinion, lacks empirical motivation and unnecessarily complicates the tectogrammar. Additionally, we foresake the standard (and usually, mostly extensional) Montague semantics in favor of a hyperintensional form of possible-worlds semantics with propositions (rather than worlds) as a basic type.[2]

What emerges is a surprisingly simple and uniform analysis of interrogatives in the two languages. On our analysis, English and Chinese constituent questions are essentially identical semantically and tectogrammatically, with phenogrammar identified as the sole locus of variation. That is, the difference between *wh* fronting and *wh in situ* is analyzed as essentially a purely phenogrammatical difference.

The rest of the paper is organized as follows. In Section 2 we introduce the framework in more detail, including a brief review of some standard PTG features. In Section 3 we present the relevant data, which are analyzed in Section 4. Section 5 evaluates the framework and the analysis, and outlines some directions for future research.

2 An Overview of the Framework

The version of PTG we employ resembles λG (not ACG), in making no use of tectogrammtical terms. Thus, our 'signs' (the things the grammar proves) are triples consisting of a typed pheno term, a tecto type, and a typed semantic term (the pheno and semantic types are suppressed whenever no confusion results from so doing). However we depart from λG (as described in Muskens (2003 2007b) by (i) adopting a slightly different notation, (ii) dropping Muskens' Kripke models for the phenogrammatical terms in favor of standard (Henkin) models for the higher-order theory of a free monoid; and (iii) allowing for a relational interface between tectogrammar and semantics whereby a single tectogrammatical type may correspond to multiple semantic types. The first two departures are nonessential and are simply a matter of preference. The third departure, however, is substantive and bears directly on the linguistic results reported in this paper.

2.1 Phenogrammar

Phenogrammar is implemented s a classical higher-order theory (and therefore a typed λ calculus) with one basic type for strings, `Str` (besides the truth-value

[2] We think this choice greatly simplifies the compositional semantics of interrogatives, but space limitations prevent us from defending that belief here.

type `e` provided by higher-order logic (HOL)). The pheno components of words (lexical signs) are treated in terms of nonlogical constants of type `Str`, e.g. `chris`, `robin`, `liked`, `slept`, `who`, `whether`, etc. The constants $\circ$:`Str`$\rightarrow$`Str`$\rightarrow$`Str` and `e` : `Str` are axiomatized as the (binary, associative) operation of a free monoid and its two-sided identity respectively.
We use `p`, `q`, `r` as variables of type `Str`, and `f`, `g` as variables of type `Str`$\rightarrow$`Str`. We call terms of this calculus *pheno terms* (cf. Oehrle's (1994) ϕ *terms*).

A transitive English verb such as *liked* is associated with the following pheno term:

$$\vdash \lambda_p \lambda_q \mathtt{q} \circ \mathtt{liked} \circ \mathtt{p} : \mathtt{Str} \rightarrow \mathtt{Str} \rightarrow \mathtt{Str}$$

This pheno term requires that the first argument of *liked* – its object – concatenate to the right of `liked`, and its second argument – its subject – to the left of `liked`, so we get the expected subject–verb–object order.

2.2 Tectogrammar

The tectogrammatical signature is obtained by closing the set of basic types **N**, **NP**, **S** and $\bar{\mathbf{S}}$ under the linear implication $\multimap$. Type **N** is associated with common nouns (*book*, *dog*), and **NP** with noun phrases (*Chris*, *Robin*). Type **NP**$\multimap$**S** corresponds to intransitive finite verbs and verb phrases (*slept*, *liked Robin*), and type **NP**$\multimap$**NP**$\multimap$**S** to transitive finite verbs (*liked*). Type **S** is reserved for root clauses, declarative (*Chris liked Robin.*) or interrogative (*Who did Chris like?*), while $\bar{\mathbf{S}}$ corresponds to embedded clauses, again — declarative (*that Chris liked Robin*) or interrogative (*who Chris liked*).

Note that the tecto type of a transitive verb merely requires that it combine with two noun phrases, but does not determine the relative word order of the verb and its arguments since this is handled entirely within the phenogrammar.

We ignore the distinction between declaratives and interrogatives in the tectogrammar because we assume a nonfunctional relation between tecto types and semantic types. So it is possible for, say, the tectogrammatical type **S** to correspond to the semantic type of declaratives or interrogatives. The two kinds of utterances are then distinguished in terms of semantic types and that is how overgeneration is prevented. At the same type, the tectogrammar is kept maximally simple.

2.3 Semantics

We assume a hyperintensional semantic theory along the lines of Pollard 2008a (see Thomason 1980 and Muskens 2005, 2007a for versions of hyperintensional semantics with somewhat different technical assumptions). While we believe this choice to be well motivated (we direct the reader to Pollard 2008a for a detailed discussion of problems with traditional possible world semantics), the analysis of interrogatives presented here does not hinge on assuming this particular semantic theory; our account is compatible with a more mainstream Montague-style possible world semantics.

The most important departure from the standard possible world semantics is treating propositions as primitive and constructing possible worlds as certain sets of propositions, instead of the other way around. On our approach, propositions are modelled as members of a *pre*-boolean algebra *pre*-ordered by entailment. Entailment is axiomatized as a reflexive, transitive, but *not* antisymmetric relation on propositions. This way, it is possible for equivalent (mutully entailing) propositions to be distinct.

This hyperintensional semantic theory is expressed in a classical HOL with e (entities) and p (propositions) as the basic types (other than the truth-value type t provided by the logic).[3] The kind of HOL we employ follows Lambek and Scott (1986) in also having a basic type n (natural numbers) and machinery for forming (separation-style) subtypes.[4] Additionally, we make use of dependent product and coproduct types parametrized by the natural number type.

We recursively define the function Ext mapping hyperintensional types (i.e. e, p and any implicative types constructed out of these) to the corresponding extensional types. Here, A and B are metavariables over hyperintensional types:

(1) a. $\mathsf{Ext}(\mathsf{e}) = \mathsf{e}$
b. $\mathsf{Ext}(\mathsf{p}) = \mathsf{t}$
c. $\mathsf{Ext}(A \to B) = A \to \mathsf{Ext}(B)$

The type of possible words w is constructed out of the basic types in such a way that the interpretation of the type w is the set of ultrafilters of the pre-boolean prealgebra that interprets the type p. Specifically. $\mathsf{w} =_{def} [\mathsf{p} \to \mathsf{t}]_u$, where $u : (\mathsf{p} \to \mathsf{t}) \to \mathsf{t}$ is a predicate on sets of propositions that picks out those sets of propositions that are ultrafilters (see Pollard (2008a) for details of this construction).

Concomitantly, we introduce a family of constants $\mathsf{ext}_A : A \to \mathsf{w} \to \mathsf{Ext}(A)$ (where the type variable A ranges over the hyperintensional types) interpreted as a polymorphic function that maps a hyperintension and a world to the extension of that hyperintension at that world, as follows:

(2) a. $\vdash \forall_{x:\mathsf{e}} \forall_{w:\mathsf{w}} [\mathsf{ext}_{\mathsf{e}}(x)(w) = x]$
b. $\vdash \forall_{p:\mathsf{p}} \forall_{w:\mathsf{w}} [\mathsf{ext}_{\mathsf{p}}(p)(w) = p@w]$
c. $\vdash \forall_{f:A\to B} \forall_{w:\mathsf{w}} [\mathsf{ext}_{A\to B}(f)(w) = \lambda_{x:A} \mathsf{ext}_B(f(x))(w)]$

Here the notation '$p@w$' abbreviates $\mu_u(w)(p)$, where μ_u denotes the embedding of the set of worlds into the set of sets of propositions.

[3] For expository simplicity, we depart from Pollard 2008a in not distinguishing between the extensional type e and the corresponding hyperintensional type i (individual concepts). In particular, the meaning of a name is the same as its reference.

[4] Thus if A is a type and a an A-predicate (term of type $A \to \mathsf{t}$), then there is a type A_a interpreted as the subset of the intepretation of A that has the interpretation of a as its characteristic function; and there is a constant μ_a that denotes the subset embedding.

As for question meanings, we adopt an elaboration of the general approach of Pollard 2008b, which in turn modifies and refines the 'set of true answers' approach of Karttunen 1977. We present the details in Section 4.

2.4 A Small Example

Before moving on to the analysis of questions, we illustrate how the grammar works with a toy example. As in λG, a representation of a linguistic expression (or a *sign*) consists of a pheno term, a tecto typo, and a semantic term. Lexical entries are written in the form:

(3) $\vdash \texttt{pheno term}; \mathbf{TectoType}; \mathsf{semantic\ term}$

We make use of three logical rules[5]:

(4) $$\frac{}{\boldsymbol{v}; \mathbf{T}; v \vdash \boldsymbol{v}; \mathbf{T}; v}\ [\text{Ax}]$$

(5) $$\frac{\Gamma \vdash \boldsymbol{f}; \mathbf{T} \multimap \mathbf{T}'; f \qquad \Delta \vdash \boldsymbol{a}; \mathbf{T}; a}{\Gamma, \Delta \vdash \boldsymbol{f}(\boldsymbol{a}); \mathbf{T}'; f(a)}\ [\multimap\text{E}]$$

(6) $$\frac{\Gamma, \boldsymbol{a}; \mathbf{T}; a \vdash \boldsymbol{f}; \mathbf{T}'; f}{\Gamma \vdash \lambda_{\boldsymbol{a}}\boldsymbol{f}; \mathbf{T} \multimap \mathbf{T}'; \lambda_a f}\ [\multimap\text{I}]$$

These correspond roughly to trace. merge, and move respectively in mainstream generative grammar.
Here is a toy lexicon for English:

(7)
$\vdash \texttt{chris}; \mathbf{NP}; \mathsf{chris}$
$\vdash \texttt{robin}; \mathbf{NP}; \mathsf{robin}$
$\vdash \lambda_{\texttt{p}}\texttt{p} \circ \texttt{slept}; \mathbf{NP} \multimap \mathbf{S}; \mathsf{sleep}$
$\vdash \lambda_{\texttt{r}}\lambda_{\texttt{p}}\texttt{p} \circ \texttt{liked} \circ \texttt{r}; \mathbf{NP} \multimap \mathbf{NP} \multimap \mathbf{S}; \mathsf{like}$
$\vdash \texttt{dog}; \mathbf{N}; \mathsf{dog}$
$\vdash \lambda_{\texttt{p}}\lambda_{\texttt{f}}\texttt{f}(\texttt{every} \circ \texttt{p}); \mathbf{N} \multimap (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{every}$
$\vdash \lambda_{\texttt{p}}\lambda_{\texttt{f}}\texttt{f}(\texttt{a} \circ \texttt{p}); \mathbf{N} \multimap (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{exists}$
$\vdash \lambda_{\texttt{f}}\texttt{f}(\texttt{everyone}); (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{every(person)}$
$\vdash \lambda_{\texttt{f}}\texttt{f}(\texttt{someone}); (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{exists(person)}$

Given the rules and these lexical entries we can derive the following by means of [—∘ E] and β reduction in the pheno logic (to enhance readability, we freely β-reduce pheno and tecto terms in derivations):

(8) a. $\vdash \texttt{chris} \circ \texttt{slept}; \mathbf{S}; \mathsf{sleep(chris)}$

[5] Compare [—∘E] to *pointwise application* and [—∘I] to *pointwise abstraction* in Muskens 2007b.

b. $\vdash \mathtt{chris} \circ \mathtt{liked} \circ \mathtt{robin}; \mathbf{S}; \mathsf{like}(\mathsf{robin})(\mathsf{chris})$

The hyperintensional generalized quantifiers $\vdash \mathsf{every} : (\mathsf{e} \to \mathsf{p}) \to (\mathsf{e} \to \mathsf{p}) \to \mathsf{p}$ and $\vdash \mathsf{exists} : (\mathsf{e} \to \mathsf{p}) \to (\mathsf{e} \to \mathsf{p}) \to \mathsf{p}$ given in the lexicon above are related to their extensional counterparts via the following meaning postulates:

(9) a. $\forall_P \forall_Q \forall_w [\mathsf{every}(P)(Q)@w = \forall_x (P(x)@w \to Q(x)@w)]$
b. $\forall_P \forall_Q \forall_w [\mathsf{exists}(P)(Q)@w = \exists_x (P(x)@w \wedge Q(x)@w)]$

Below we show the entire proof of *Robin liked a dog*, to illustrate the mechanism for scoping *in situ* semantic operators which will be relevant to our analysis of interrogatives. First we assemble the quantificational noun phrase *a dog*:

$$(10) \quad \frac{\vdash \lambda_{\mathtt{p}} \lambda_{\mathtt{f}} \mathtt{f}(\mathtt{a} \circ \mathtt{p}); \mathbf{N} \multimap (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{exists} \qquad \vdash \mathtt{dog}; \mathbf{N}; \mathsf{dog}}{\vdash \lambda_{\mathtt{f}} \mathtt{f}(\mathtt{a} \circ \mathtt{dog}); (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{exists}(\mathsf{dog})} \; [\multimap \mathrm{E}]$$

We make use of [Ax] to introduce a hypothesis corresponding to the object argument of the verb. Intuitively, this is the slot that the quantificational object will eventually lower itself into.

$$(11) \quad \frac{\vdash \lambda_{\mathtt{r}} \lambda_{\mathtt{p}} \mathtt{p} \circ \mathtt{saw} \circ \mathtt{r}; \mathbf{NP} \multimap \mathbf{NP} \multimap \mathbf{S}; \mathsf{see} \qquad \dfrac{}{\mathtt{q}; \mathbf{NP}; \mathtt{x} \vdash \mathtt{q}; \mathbf{NP}; \mathtt{x}} \; [\mathrm{Ax}]}{\mathtt{q}; \mathbf{NP}; \mathtt{x} \vdash \lambda_{\mathtt{p}} \mathtt{p} \circ \mathtt{liked} \circ \mathtt{q}; \mathbf{NP} \multimap \mathbf{S}; \mathsf{like}(\mathtt{x})} \; [\multimap \mathrm{E}]$$

Then we proceed to combine the verb phrase with a missing object, with its subject *Robin*. In a step of [$\multimap$I] we discharge the object hypothesis, λ abstracting on the free variables in the pheno and the semantic term. The quantificational noun phrase *a dog* can now scope over $\lambda_x \mathsf{see}(\mathtt{x})(\mathsf{robin})$, but its pheno term ensures that it is lowered into the object gap in one step of β reduction in the pheno logic. By *gap* we simply mean a λ bound variable in a pheno term.

$$(12) \quad \frac{(11) \qquad \dfrac{(12) \qquad \vdash \mathtt{robin}; \mathbf{NP}; \mathsf{robin}}{\mathtt{q}; \mathbf{NP}; \mathtt{x} \vdash \mathtt{robin} \circ \mathtt{liked} \circ \mathtt{q}; \mathbf{S}; \mathsf{like}(\mathtt{x})(\mathsf{robin})} \; [\multimap \mathrm{E}]}{\vdash \lambda_q \mathtt{robin} \circ \mathtt{liked} \circ \mathtt{q}; \mathbf{NP} \multimap \mathbf{S}; \lambda_x \mathsf{like}(\mathtt{x})(\mathsf{robin})} \; [\multimap \mathrm{I}]}{\vdash \mathtt{robin} \circ \mathtt{liked} \circ \mathtt{a} \circ \mathtt{dog}; \mathbf{S}; \mathsf{exists}(\mathsf{dog})(\lambda_x \mathsf{like}(\mathtt{x})(\mathsf{robin}))} \; [\multimap \mathrm{E}]$$

We easily predict the ambiguity of a sentence with two quantificational expressions such as *Everyone saw a dog*. Since the context is a multiset and not a list, the subject and the object hypothesis that would be introduced in the proof of this sentence can be discharged in either order. If the object hypothesis is discharged first, we get the reading in (14). If the subject hypothesis is discharged first, we get the reading in (15). In both cases, the word order is the same since the quantificational expressions just lower themselves into the appropriate gap of their argument.

(13) $\vdash \mathtt{everyone} \circ \mathtt{saw} \circ \mathtt{a} \circ \mathtt{dog}; \mathbf{S}; \mathsf{every}(\mathsf{person})(\lambda_x \mathsf{exists}(\mathsf{dog})(\lambda_y \mathsf{saw}(\mathtt{y})(\mathtt{x})))$

(14) $\vdash \mathtt{everyone} \circ \mathtt{saw} \circ \mathtt{a} \circ \mathtt{dog}; \mathbf{S}; \mathsf{exists}(\mathsf{dog})(\lambda_y \mathsf{every}(\mathsf{person})(\lambda_x \mathsf{saw}(\mathtt{y})(\mathtt{x})))$

3 The Data

In this section, we briefly describe the data we will account for in Section 4. Due to considerations of space, we mainly focus on embedded interrogatives.

3.1 Interrogatives in Chinese

Like English, Chinese is an SVO language:

(15) Zhangsan xihuan Lisi.
Zhangsan like Lisi
'Zhangsan likes Lisi.'

Unlike English, Chinese has distinct interrogative verb forms which reduplicate the first syllable of the verb, with the morpheme *bu* 'not' separating the two copies. These forms are employed in polar questions, both root and embedded. The only difference between declaratives and polar interrogatives is the form of the finite verb (e.g. *xihuan* vs. *xi-bu-xihuan*).

(16) a. Zhangsan xi-bu-xihuan Lisi?
Zhangsan like? Lisi
'Does Zhangsan like Lisi?'

b. Chunsheng xiang-zhidao Zhangsan xi-bu-xihuan Lisi?
Chunsheng wonder Zhangsan like? Lisi
'Chunsheng wonders whether Zhangsan likes Lisi.'

Constituent questions contain interrogative (*wh*) expressions such as *shenme* 'what' or *shei* 'who'. These interrogative expressions appear *in situ*, i.e. in the same place in the clause where their non-interrogative counterparts appear. This is true of both main and embedded clauses:

(17) a. Zhangsan xihuan shenme?
Zhangsan like what
'What does Zhangsan like?'

b. Shei xihuan shenme?
who like what
'Who likes what?'

c. Zhangsan xiang-zhidao Lisi xihuan shei.
Zhangsan wonder Lisi like who
'Zhansang wonders who Lisi likes.'

Chinese *wh* expressions can have arbitrarily wide scope, constrained solely by the properties of the embedding verb(s).

(18) Zhangsan xiang-zhidao shei xihuan shenme./?
Zhangsan wonder who like what
'Zhangsan wonders who likes what.'
'Who does Zhangsan wonder what (that person) likes?'
'What does Zhangsan wonder who likes?'

The preceding example is three-ways ambiguous. Both embedded *wh* expressions can have embedded scope. On this interpretation, the main clause is declarative, and the embedded clause is a binary constituent question. Alternatively, either of the embedded *wh* expressions can have root scope, resulting in n interpretation where both the main and the embedded clause are unary constituent questions. It is impossible, however, for both embedded *wh* expressions to have root scope, since the embedding verb *xiang-zhidao* 'wonder' can only take interrogative but not declarative complements (much like its English counterpart).

3.2 Interrogatives in English

In English, embedded polar interrogatives are formed by means of the interrogative 'complementizer' *whether*, which takes a sentential complement, e.g. *Chris wonders whether Robin likes Sandy.*

In constituent questions, in contrast to Chinese, *wh* expressions are not all *in situ*. Rather, a *wh* expression must occur on the extreme left periphery of a clause, and take scope at that clause. in order for the clause to be interpreted as a constituent question. Adapting an HPSG usage, we call such a left-peripheral *wh* expression a *filler*.[6]

(19) a. Chris wonders who Robin likes.
b. * Chris wonders Robin likes who.

(20) a. Chris wonders who Robin gave what.
b. * Chris wonders Robin gave who what.
c. * Chris wonders who what Robin gave.

By definition, a filler *wh* expression can only have surface scope. By contrast, an *in situ wh* expression can scope at or wider than the minimal clause in which it occurs, but only if the clause t which it scopes aslo hs a filler:

(21) Chris wonders who$_x$ likes what$_y$.

[6] In mainstream generative grammar, such *wh* expressions are analyzed as having undergone overt *wh* movement (string-vacuous movement in case the *wh* expression in question is the subject of the clause). Note that not every extreme left-peripheral *wh* expression is a filler; for example, in

1. Who thought which dog barked?

which dog is not a filler, but rather an *in situ wh* expression with root scope.

a. 'Chris wonders who likes what'

b. # 'For which person x does Chris wonder which thing y is such that x likes y?'

c. # 'For which thing y does Chris wonder which person x is such that x likes y?'

(22) Who$_x$ wonders who$_y$ likes what$_z$?

a. 'Which person x is such that x wonders which person y and which thing z are such that y likes z?'
possible answer: Chris.

b. 'Which person x and which thing z are such that x wonders which person y is such that y likes z?'
possible answer: Chris wonders who likes beer.

c. # 'Which person x and which person y are such that x wonders which thing z is such that y likes z?'
impossible answer: Chris wonders what Robin likes.

4 The Analysis

4.1 Polar Questions

Semantic assumptions. Like Karttunen 1977, we analyze polar questions (meanings of both root and embedded polar interrogative clauses) as having extensions which are singleton sets of true answers. On our hyperintensional approach, this means that polar questions have the type $\mathsf{p} \to \mathsf{p}$; so that the extension at some world w is then a set of propositions ($\mathsf{p} \to \mathsf{t}$) – intuitively, the set of true answers to it. Thus, e.g. *whether Chris slept* or *Did Chris sleep?* denotes at some w a set with exactly one member: either the proposition that Chris slept or that he didn't, whichever is true at w. We abbreviate the polar question type $\mathsf{p} \to \mathsf{p}$ as k_0.

(23) $\mathsf{k}_0 = \mathsf{p} \to \mathsf{p}$

Now we introduce the constant $\mathsf{whether} : \mathsf{p} \to \mathsf{k}_0$ together with the following meaning postulate (nonlogical axiom):

(24) $\vdash \mathsf{whether} = \lambda_q \lambda_p [p \ \mathsf{and} \ ((p \ \mathsf{eq_p} \ q) \ \mathsf{or} \ (p \ \mathsf{eq_p} \ (\mathsf{not} \ q)))]$

In the definition of whether we made use of the propositional connectives and, or and not that translate the English sentential connectives *and*, *or* and *it's not the case that*. The following theorems (which follow directly from the facts that (1) the propositions form a preboolean algebra, and (2) worlds are ultrfilters) relate these propositional connectives to their extensional counterparts:

(25) a. $\vdash \forall_p \forall_q \forall_w [(p \ \mathsf{and} \ q)@w = (p@w \wedge q@w)]$

b. $\vdash \forall_p \forall_q \forall_w [(p \ \mathsf{or} \ q)@w = (p@w \vee q@w)]$

c. $\vdash \forall_p \forall_w [(\mathsf{not}\ p)@w = \neg(p@w)]$

We also made use of the constant eq_p (we omit the subscript when the type is clear from the context). This is one of a family of constants eq_A of type $A \to A \to \mathsf{p}$ which are used to express, for each hyperintensional meaning type A, propositions that two meanings of type A are one and the same meaning. The following meaning postulate states that at any world w, the extension of eq_A at w is the ordinary equality relation on things of type A:

(26) $\vdash \forall_w \forall_x \forall_y [(x\ \mathsf{eq}\ y)@w = (x = y)]$

English polar questions. The constant whether is used as the semantics of the English interrogative complementizer *whether*, which has the following lexical entry:

$\vdash \lambda_{\mathtt{p}}(\mathtt{whether} \circ \mathtt{p}); \mathbf{S} \multimap \bar{\mathbf{S}}; \mathsf{whether}$

Now we can generate embedded polar questions in English, such as:

(27) $\vdash \mathtt{whether} \circ \mathtt{chris} \circ \mathtt{slept}; \bar{\mathbf{S}}; \mathsf{whether}(\mathsf{sleep}(\mathsf{chris}))$

The semantic term denotes a singleton set of propositions, as desired:

(28) $\vdash \forall_w \mathsf{whether}(\mathsf{sleep}(\mathsf{chris})))@w =$
$\lambda_p[p@w \wedge ((p = \mathsf{sleep}(\mathsf{chris})) \vee (p = (\mathsf{not}(\mathsf{sleep}(\mathsf{chris})))))]$

Embedded interrogatives in English are assigned a distinct tectogrammatical type from root questions, since they are not interchangeable - *whether Chris slept* cannot be a root question, and *Did Chris sleep?* cannot be an embedded question, hence we must distinguish between $\bar{\mathbf{S}}$ and $\mathbf{S}$. (Of course the same tectogrmmatical distinction will be made for declarative clauses.)

Chinese polar questions. In Chinese, root and embedded polar interrogatives are interchangeable so t hey are both assigned to the same tectogrammatical type $\mathbf{S}$. Since there is no interrogative complementizer in Chinese, whether is packaged into the semantic term of the interrogative verb forms, which as we saw has a distinct form from its declarative forming counterpart. We give the following toy lexicon for Chinese:

$\vdash \mathtt{zhangsan}; \mathbf{NP}; \mathsf{zhangsan}$

$\vdash \mathtt{lisi}; \mathbf{NP}; \mathsf{lisi}$

$\vdash \lambda_{\mathtt{p}} \lambda_{\mathtt{q}} \mathtt{q} \circ \mathtt{xihuan} \circ \mathtt{p}; \mathbf{NP} \to \mathbf{NP} \to \mathbf{S}; \mathsf{like}$

$\vdash \lambda_{\mathtt{p}} \lambda_{\mathtt{q}} \mathtt{q} \circ \mathtt{xi\text{-}bu\text{-}xihuan} \circ \mathtt{p}; \mathbf{NP} \to \mathbf{NP} \to \mathbf{S}; \lambda_x \lambda_y \mathsf{whether}(\mathsf{like}(\mathsf{x})(\mathsf{y}))$

Now we can derive the following examples:

(29) a. $\vdash \mathtt{zhangsan} \circ \mathtt{xihuan} \circ \mathtt{lisi}; \mathbf{S}; \mathsf{like}(\mathsf{lisi})(\mathsf{zhangsan})$

b. $\vdash$ $\texttt{zhangsan} \circ \texttt{xi-bu-xihuan} \circ \texttt{lisi}; \mathbf{S}; \mathsf{whether}(\mathsf{like}(\mathsf{lisi})(\mathsf{zhangsan}))$

In sum, the difference between English and Chinese embedded polar interrogatives is that (i) English distinguishes (tectogrammatically) between embedded and root interrogatives, while Chinese does not, and (ii) the form that contributes the interrogative meaning is the interrogative complementizer *whether* in English, while in Chinese it is the distinct verbal form.

4.2 Constituent Questions

Semantic assumptions. We analyze n-ary questions as having extensions which are (curried) functions from n individuals to a singleton set of propositions. Recall that k_0 is the type of polar questions. We define the type of n-ary constituent question as follows:

(30) $\mathsf{k}_{n+1} = \mathsf{e} \rightarrow \mathsf{k}_n$

For example, the hyperintensional meaning type of a unary constituent question such as *who slept* is a function from individuals to properties of propositions (type $\mathsf{e} \rightarrow \mathsf{k}_0$). Such a question denotes at some world w a function from individuals to a singleton set of propositions (type $\mathsf{e} \rightarrow \mathsf{p} \rightarrow \mathsf{t}$), mapping each individual x to the singleton set whose member is either the proposition that x slept or the proposition that x didn't sleep, whichever is true at w.

The type of questions, k, is then defined to be the dependent coproduct of all question types, indexed by the natural numbers:

(31) $\mathsf{k} =_{def} \coprod_{n:\mathsf{n}} \mathsf{k_n}$

Like quantificational expressions, *wh* expressions take scope. But unlike quantificational expressions, which bind an entity variable in a proposition to yield a proposition, *wh* expressions do not have a fixed result type. For example, in a unary constituent question, the unique *wh* expression binds an entity variable in proposition to yield a term of type $\mathsf{k_1}$; while in a binary constituent question, one *wh* expression will yield a term of type $\mathsf{k_1}$ while the other one will bind an entity variable in that term to yield a term of type $\mathsf{k_2}$.

We first define the *wh* expression that scopes over propositions to yield a unary wh question. Its semantic argument type is the same as for quantificational noun phrases ($\mathsf{e} \rightarrow \mathsf{p}$), but its result type is $\mathsf{k_1}$.

(32) $\vdash \mathsf{who'} = \lambda_P \lambda_x \lambda_p [(\mathsf{person}\ x)\ \mathsf{and}\ (\mathsf{whether}\ (Px)\ p)]$

Combining $\mathsf{who'}$ with $\lambda_x(\mathsf{sleep\ x}) : \mathsf{e} \rightarrow \mathsf{p}$ we get the desired meaning:

(33) $\vdash \mathsf{who'}(\lambda_x(\mathsf{sleep\ x})) = \lambda_x \lambda_p [(\mathsf{person}\ x)\ \mathsf{and}\ (\mathsf{whether}\ (\mathsf{sleep\ x})\ \mathsf{p})]$

Unlike $\mathsf{who'}$. which combines with individual properties to yield unary questions, *wh* expressions that form (n+2)-ary constituent questions must combine

with (n+1)-ary constituent questions "missing" an enity argument. So they combine with terms of type $\mathsf{e} \rightarrow \mathsf{k_{n+1}}$ to yield terms of type $\mathsf{k_{n+2}}$. Note that $\mathsf{e} \rightarrow \mathsf{k_{n+1}}$ is exactly the type $\mathsf{k_{n+2}}$.

More formally, we recursively define a family of constants $\mathsf{who}_n : \mathrm{k}_{n+2} \rightarrow \mathrm{k}_{n+2}$ for *wh* expressions that scope over constituent questions and yield constituent questions. In the recursion clause, we make use of the polymorphic function $\mathsf{per}_{A,B,C} : (A \rightarrow B \rightarrow C) \rightarrow (B \rightarrow A \rightarrow C)$ that permutes the first two arguments of a function:

(34) $\quad \vdash \mathsf{per} = \lambda_f \lambda_x \lambda_y (f\ y\ x)$

(35) a. $\vdash \mathsf{who}_0 = \lambda_k \lambda_x \lambda_y \lambda_p [(\mathsf{person}\ x)\ \mathsf{and}\ (k\ x\ y\ p)]$

b. $\vdash \mathsf{who}_{n+1} = \lambda_k [\mathsf{per}\ \lambda_x . \mathsf{who}_n (\mathsf{per}\ k\ x)]$

Essentially, all that $\mathsf{who_n}$ does is require of its argument's first argument that it be a person. We package all the constants $\mathsf{who_n}$ together into a single dependent product type:

(36) $\quad \vdash \mathsf{who} = \lambda_{n:\mathrm{n}} . \mathsf{who}_n : \prod_{n:\mathrm{n}} (\mathrm{k}_{n+2} \rightarrow \mathrm{k}_{n+2})$

For ease of exposition, we started with the meanings of the interrogative pronoun *who*. The generalization to the interrogative determiner *which* is straightforward:

(37) $\quad \vdash \mathsf{which}' = \lambda_Q \lambda_P \lambda_x \lambda_p [(Qx)\ \mathsf{and}\ (\mathsf{whether}\ (Px)\ p)]$

(38) a. $\vdash \mathsf{which}_0 = \lambda_Q \lambda_k \lambda_x \lambda_y \lambda_p [(Q\ x)\ \mathsf{and}\ (k\ x\ y\ p)]$

b. $\vdash \mathsf{which}_{n+1} = \lambda_Q \lambda_k [\mathsf{per}\ \lambda_x . \mathsf{which}_n (Q) (\mathsf{per}\ k\ x)]$

(39) $\quad \vdash \mathsf{which} = \lambda_{n:\mathrm{n}} . \mathsf{which}_n : \prod_{n:\mathrm{n}} [(\mathrm{e} \rightarrow \mathrm{p}) \rightarrow (\mathrm{k}_{n+2} \rightarrow \mathrm{k}_{n+2})]$

We leave it it the reader to formulate the semantic constants needed for the interrogative pronoun *what*.

English constituent questions. Recall that in English there is exactly one filler *wh* expression per constituent question. Any other ones appear *in situ*. We straightforwardly account for this fact by associating the filler with the semantic term who', and the *in situ* ones with the semantic term who, intuitively one of the $\mathsf{who_n}$ constants.

Since the semantic argument and result type of who' are distinct ($(\mathsf{e} \rightarrow \mathsf{p})$ vs. $\mathsf{k_1}$), it is impossible for two fillers to occur in the same clause. And since all $\mathsf{who_n}$ constants require that there already be a constituent question for them to scope over, and who' is the unique constant that turns propositions into constituent questions, the presence of a filler *wh* expression is necessary for any *in situ* ones to occur. So, we guarantee that there is exactly one filler per constituent question.

Following Muskens 2007b, the fronting of the filler to the left periphery is accomplished entirely in the phenogrammar. We give the following lexical entries for English *wh* expressions:

(40) English filler *wh* expressions:

$\vdash \lambda_f.\texttt{who} \circ \texttt{f(e)}; (\mathbf{NP} \multimap \mathbf{S}) \multimap \bar{\mathbf{S}};\mathsf{who}'$

$\vdash \lambda_f.\texttt{what} \circ \texttt{f(e)}; (\mathbf{NP} \multimap \mathbf{S}) \multimap \bar{\mathbf{S}};\mathsf{what}'$

$\vdash \lambda_p\lambda_f.\texttt{which} \circ \texttt{p} \circ \texttt{f(e)}; \mathbf{N} \multimap (\mathbf{NP} \multimap \mathbf{S}) \multimap \bar{\mathbf{S}};\mathsf{which}'$

(41) English *in situ wh* expressions:

$\vdash \lambda_f.\texttt{f(who)}; (\mathbf{NP} \multimap \mathbf{S}) \multimap \bar{\mathbf{S}};\mathsf{who}$

$\vdash \lambda_f.\texttt{f(what)}; (\mathbf{NP} \multimap \mathbf{S}) \multimap \bar{\mathbf{S}};\mathsf{what}$

$\vdash \lambda_p\lambda_f.\texttt{f(which} \circ \texttt{p)}; \mathbf{N} \multimap (\mathbf{NP} \multimap \mathbf{S}) \multimap \bar{\mathbf{S}};\mathsf{which}$

Note that the pheno terms of *in situ wh* expressions have the same structure as those of quantificational noun phrases. So, the *in situ wh* expressions just lower themselves into the appropriate gap of their argument.

The pheno terms of the filler expressions are different. Instead of lowering themselves into the gap of their argument, they concatenate themselves to the left of their argument after feeding it the empty string e which effectively plugs the existing gap. This is how preposing of the fillers is accomplished. So, here are the kinds of embedded questions that our grammar can now generate:

(42) unary constituent questions

a. $\vdash \texttt{who} \circ \texttt{liked} \circ \texttt{robin};\bar{\mathbf{S}};\mathsf{who}'(\lambda_{\mathsf{x}}(\mathsf{like}(\mathsf{robin})(\mathsf{x}))$

b. $\vdash \texttt{which} \circ \texttt{dog} \circ \texttt{slept};\bar{\mathbf{S}};\mathsf{which}'(\mathsf{dog})(\lambda_{\mathsf{x}}(\mathsf{sleep}(\mathsf{x}))$

c. $\vdash \texttt{which} \circ \texttt{dog} \circ \texttt{chris} \circ \texttt{liked};\bar{\mathbf{S}};\mathsf{which}'(\mathsf{dog})(\lambda_{\mathsf{x}}(\mathsf{like}(\mathsf{x})(\mathsf{chris}))$

(43) binary constituent questions

a. $\vdash \texttt{who} \circ \texttt{liked} \circ \texttt{who};\bar{\mathbf{S}};\mathsf{who}_0(\lambda_{\mathsf{y}}\mathsf{who}'(\lambda_{\mathsf{x}}(\mathsf{like}(\mathsf{y})(\mathsf{x}))))$

b. $\vdash \texttt{which} \circ \texttt{dog} \circ \texttt{who} \circ \texttt{liked};\bar{\mathbf{S}};\mathsf{who}_0(\lambda_{\mathsf{y}}\mathsf{which}'(\mathsf{dog})(\lambda_{\mathsf{x}}(\mathsf{like}(\mathsf{x})(\mathsf{y}))))$

Given the pheno terms of filler *wh* expressions, we automatically predict that they must have surface scope: they must scope over the clause on whose left periphery they occur. The *in situ* expressions, on the other hand, can scope higher than their surface position would suggesr, since they can lower themselves into the right gap from virtually anywhere.

However, because of the semantic typing, the *in situ* wh expressions are dependent on there already being some filler in the clause over which they are to scope. So we predict that they can only scope higher than their surface position would suggest in case the matrix clause already contains a filler *wh* expression (in accordance with the data laid out in the preceding section).

Suppose we have the following lexical entry for *wonders*:

$\vdash \lambda_p\lambda_q\texttt{q} \circ \texttt{wonders} \circ \texttt{p};\bar{\mathbf{S}} \multimap \mathbf{NP} \multimap \mathbf{S};\mathsf{wonder}$

where wonder has type $\mathsf{k} \to \mathsf{e} \to \mathsf{p}$. Then we correctly predict the ambiguity of embedded clauses such as *who wonders which dog liked what* (as in *Robin asked me who wonders which dog liked what*), depending on whether *what* has embedded or matrix scope. Below we show the two derivable semantic terms:

(44) *who wonders which dog liked what*

a. $\vdash \mathsf{who}'(\lambda_{\mathsf{z}}\mathsf{wonder}(\mathsf{what}_0(\lambda_{\mathsf{y}}(\mathsf{which}'(\mathsf{dog})(\lambda_{\mathsf{x}}(\mathsf{like\ y\ x})))(\mathsf{z}))) : \mathsf{k}_1$

b. $\vdash \mathsf{what}_0(\lambda_{\mathsf{y}}\mathsf{who}'(\lambda_{\mathsf{z}}\mathsf{wonder}((\mathsf{which}'(\mathsf{dog})(\lambda_{\mathsf{x}}(\mathsf{like\ y\ x})))(\mathsf{z}))) : \mathsf{k}_2$

Chinese constituent questions. Unlike English, Chinese has no interrogative filler *wh* expressions, but only *in situ* ones. So, all *wh* expressions are assigned to the same kind of pheno term and are systematically ambiguous between who' and who (what' and what). We add the following lexical entries:

(45) Chinese *wh* pronouns

$\vdash \lambda_f.\mathtt{f(shei)}; (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{who}'$

$\vdash \lambda_f.\mathtt{f(shei)}; (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{who}$

$\vdash \lambda_f.\mathtt{f(shenme)}; (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{what}'$

$\vdash \lambda_f.\mathtt{f(shenme)}; (\mathbf{NP} \multimap \mathbf{S}) \multimap \mathbf{S}; \mathsf{what}$

Now we can generate examples like the following:

(46) a. 'what Zhangsan likes'/'What does Zhangsan like?'

$\vdash \mathtt{zhangsan} \circ \mathtt{xihuan} \circ \mathtt{shenme}; \mathbf{S}; \mathsf{what}'(\lambda_{\mathsf{x}}\mathsf{like}(\mathsf{x})(\mathsf{zhangsan}))$

b. 'who likes Lisi'/'Who likes Lisi?'

$\vdash \mathtt{shei} \circ \mathtt{xihuan} \circ \mathtt{lisi}; \mathbf{S}; \mathsf{who}'(\lambda_{\mathsf{x}}\mathsf{like}(\mathsf{lisi})(\mathsf{x}))$

c. 'who likes what'/'Who likes what?'

$\vdash \mathtt{shei} \circ \mathtt{xihuan} \circ \mathtt{shenme}; \mathbf{S}; \mathsf{what}_0(\lambda_{\mathsf{y}}\mathsf{who}'(\lambda_{\mathsf{x}}\mathsf{like}(\mathsf{y})(\mathsf{x})))$

$\vdash \mathtt{shei} \circ \mathtt{xihuan} \circ \mathtt{shenme}; \mathbf{S}; \mathsf{who}_0(\lambda_{\mathsf{x}}\mathsf{what}'(\lambda_{\mathsf{y}}\mathsf{like}(\mathsf{y})(\mathsf{x})))$

Note the insignificant ambiguity of binary questions such as the one in (46c) depending on which *wh* expression is scoped first.[7]

Since all *wh* expressions in Chinese lower themselves into their argument's gap, they can scope arbitrarily high. We give the following lexical entry for *xiang-zhidao* 'wonder':

$$\vdash \lambda_p\lambda_q\mathtt{q} \circ \mathtt{xiang\text{-}zhidao} \circ \mathtt{p}; \mathbf{S} \multimap \mathbf{NP} \multimap \mathbf{S}; \mathsf{wonder}$$

Our grammar predicts that embedded *wh* expressions can in fact have root scope. All of the following semantic terms are derivable for the sentence in (47):

(47) Zhangsan xiang-zhidao shei xihuan shenme./?

a. 'Zhangsan wonders who likes what.'

$\vdash \mathsf{wonder}(\mathsf{what}_0(\lambda_{\mathsf{y}}\mathsf{who}'(\lambda_{\mathsf{x}}\mathsf{like}(\mathsf{y})(\mathsf{x}))))(\mathsf{zhangsan})$

$\vdash \mathsf{wonder}(\mathsf{who}_0(\lambda_{\mathsf{x}}\mathsf{what}'(\lambda_{\mathsf{y}}\mathsf{like}(\mathsf{y})(\mathsf{x}))))(\mathsf{zhangsan})$

b. 'Who does Zhangsan wonder what (that person) likes?'

[7] Insignificant in the sense that, at any world w, the two redings have the same extension at w.

$\vdash \mathsf{who}'(\lambda_{\mathsf{x}}\mathsf{wonder}(\mathsf{what}'(\lambda_{\mathsf{y}}\mathsf{like}(\mathsf{y})(\mathsf{x})))(\mathsf{zhangsan}))$

c. 'What does Zhangsan wonder who likes?'

$\vdash \mathsf{what}_0(\lambda_{\mathsf{y}}\mathsf{wonder}(\mathsf{who}'(\lambda_{\mathsf{x}}\mathsf{like}(\mathsf{y})(\mathsf{x})))(\mathsf{zhangsan}))$

It is however impossible for both embedded *wh* expressions to have root scope, not because of the tectogrammatical type of *xiang-zhidao*, but because of its *semantic* type: it needs an argument of type k.

5 Discussion and Conclusion

Hi Vedrana, I'm out of time so this is up to you. Here are some random thoughts. The final version should mention the following. (1) Comparison with mainstream generative (whoever invented the 'tucking in' analysis of multiple wh in minimalism?)and type-logical analyses (Vermaat); (2) some remarks on adapting our analysis to languages typologically different from both English and Chinese (especially Japanese and Serbo-Croatian); and (3) speculative remarks on recoding our analysis into a 'direct-style' framework where in-situ operators are not type-raised in the pheno component, but rather are scoped via (nonconfluent) reduction in the semantic calculus using polymorphic shift operators ($\lambda\mu$-calculi are insuffiently expressive because the relevant operators have to be schematized across return types, which also differ from there scope types). Because in direct style you have some control (via the formultion of the reduction rules for the shift opertors) over how far up and to what types of enclosing expressions an operator cn scope to, this might give us a handle on, e.g. how wh-expressions can get trapped in the minimal containing ka-clause (if in fact this is what happens) and other 'scope island' effectsw.

References

Baldridge, Jason. 2002. *Lexically Specied Derivational Control in Combinatory Categorial Grammar*. Ph.D. thesis, University of Edinburgh.

Curry, Haskell. 1961. Some logical aspects of grammatical structure. In Roman Jakobson, ed., *Structure of Language and Its Mathematical Aspects*.

de Groote, Philippe. 2001. Towards Abstract Categorial Grammar. *Proceedings of ACL*, 2001.

Karttunen, Lauri. 1977. Syntax and semantics of questions. *Linguistics and Philosophy* **1**:1.

Lambek, Joachim. and Philip J Scott. 1986. *Introduction to Higher Order Categorical Logic*. Cambridge University Press.

Montague, Richard. 1974. The proper treatment of quantification in English. In R. Thomason, ed., *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press, pp. 247-270.

Moortgat, Michael. 1996. Generalized quantification and discontinuous type constructors. In H. Bunt, and A. van Horck, eds., *Discontinuous Constituency*. Mouton de Gruyter.

Moortgat, Michael. 1997. Categorial type logics. J. van Benthem and A. ter Meulen, eds., *Handbook of Logic and Language*. Amsterdam: Elsevier.

Morrill, Glyn and Teresa Solias. 1993. Tuples, discontinuity and gapping in Categorial Grammar. *Proceedings of the European Chapter Association for Computational Linguistics*, EACL93, Utrecht, 287-297.

Morrill, Glyn, Mario Fada and Oriol Valentin. 2007. Nondeterministic discontinuous Lambek calculus. *Proceedings of the Seventh International Workshop on Computational Semantics*, IWCS7, Tilburg.

Muskens, Reinhard. 2003. Language, lambdas, and logic. In Geert-Jan Kruijff and Richard Oehrle, eds., *Resource Sensitivity in Binding and Anaphora, Studies in Linguistics and Philosophy*. Kluwer.

Muskens, Reinhard. 2005. Sense and the computation of reference. *Linguistics and Philosophy* **28**:4.

Muskens, Reinhard. 2007a. Intensional models for the theory of types. *The Journal of Symbolic Logic* **72**:1.

Muskens, Reinhard. 2007b. Separating syntax and combinatorics in categorial grammar. *Research on Language and Computation* **5**:3.

Oehrle, Richard. 1994. Term-labeled categorial type systems. *Linguistics and Philosophy* **17**:6.

. Pollard, Carl. 2004. Type-Logical HPSG. In G. Jäger, P. Monachesi, G. Penn, and S. Wintner, eds. *Proceedings of Formal Grammar 2004*. Nancy, France: European Summer School in Language, Logic, and Information, pp. 107-124.

Pollard, Carl. 2008a. Hyperintensions. *Journal of Logic and Computation* **18**:2.

Pollard, Carl. 2008b. Hyperintensional questions. In W. Hodges and R. de Queiroz, eds., *Proceedings of the 15th Annual Workshop on Logic, Language, Information, and Computation (WoLLIC 08): Heriot-Watt University 2008*. Springer Lecture Notes in Artificial Intelligence 5110::261-274.

Smith, E. Allyn. 2010. *Correlational Comparison in English*. Ph.D. dissertation, Department of Linguistics, The Ohio State University.

Thomason, Richmond. 1980. A model theory for propositional attitudes. *Linguistics and Philosophy* **4**:1.

Vermaat, Willemijn. 2005. *The Logic of Variation. A Cross-Linguistic Account of Wh-Question Formation*. Ph.D. thesis, Utrecht Institute of Linguistics OTS, Utrecht University.