Chapter 1

Ambiguity, Neutrality, and Coordination in Higher-Order Grammar

CARL POLLARD AND JIŘÍ HANA

ABSTRACT. We show that the standard account of neutrality and coordination in type-logical grammar is untenable. However, when using as our framework a version of Lambek's categorical grammar with a type theory based on Lambek and Scott's higher order intuitionistic logic (the internal language of a topos) rather than the Lambek calculus, the account can largely be salvaged. Because of the difficulty of phonologically interpreting coordinated functors of differing directionality we need to handle both phonology and syntax within a single polymorphically typed lambda calculus.

1.1 Introduction

The standard type-logical grammar (hereafter TLG) account of neutrality and coordination (Morrill 1990, Bayer and Johnson 1995, Bayer 1996, hereafter MBJ) analyzes neutrality between two types A and B as the conjunction $A \wedge B$, and coordination of an A and a B as the disjunction $A \vee B$, where $\wedge$ and $\vee$ are Lambek's (1961) binary additive connectives. This account is problematic in several respects. First, as shown by Whitman 2002, it fails to distinguish argument ambiguity from argument neutrality, so that all instances of homophony between slots in a word paradigm are wrongly predicted to be syncretic. This is falsified by examples such as the following (Dyla 1984):

(1) * CO Janek zrobil a zmartwilo Marie? (Polish)
what.NOM/ACC did and upset
[Intended meaning, roughly: What was it that Janek did and that upset Mary?]

as compared with syncretic examples such as

(2) KOGO Janek lubi a Jerzy nienawidzi? (Polish)
who.ACC/GEN like$_{acc}$ and hate$_{gen}$
'Who does Janek like and Jerzy hate?'

Proceedings of Formal Grammar 2003
G. Jäger, P. Monachesi, G. Penn & S. Wintner (editors).

Likewise, the MBJ account fails to distinguish functor neutrality from functor ambiguity, deriving the ungrammatical (3) side by side with the grammatical (4):

(3) *Mary WANTS to go and John to go.

(4) I WOULD LIKE to leave town early tomorrow morning and for you to go with me.

Faced with this problem, Whitman (2002) and Morrill (p.c.) independently suggested the possibility of distinct phonologies/prosodies with no audible difference (more specifically, Morrill suggested using an ordered pair of a string and an integer instead of just a string). Whitman subsequently opted instead for abandoning the hypothesis that there is any purely syntactic distinction between neutrality and ambiguity, so that in principle all ambiguities are potentially neutralizable subject to pragmatic or processing constraints.

Second, the MBJ account does not account for examples of nonbinary coordination, where one or more of the coordinated constituents can themselves be coordinate structures:

(5) Kim is [drunk, stoned, or under the table] and an inveterate liar.

And third, on the MBJ account an unlike coordination such as *rich and an excellent cook* is analyzed as an $\mathrm{NP}\vee\mathrm{AP}$. But in the standard frame-semantical phonological interpretation (Heylen 1996 and 1997, Moortgat 1997, Carpenter 1997), if S is the stringset that interprets NP and T the stringset that interprets AP, then *rich and an excellent cook* is in their union. So either it is in S or it is in T; hence either it is derivable as an NP, or it is derivable as an AP. But it is neither. So the theory is inconsistent.

From what we have said so far, it may seem that the MBJ account is beyond salvation. However, we will argue that the intuitions behind it are right on the mark, but just need to be expressed in a more cooperative kind of type theory. An alternative account of (inter alia) the same phenomena due to Dalrymple and Kaplan (2000, hereafter DK) treats the f-structure of a coordinate structure as the set of f-structures of the coordinated constituents. In this paper we show how to import the desirable features of the DK analysis into a form of type-logical grammar so as to fix what is wrong with the MBJ account while preserving its main insights.

1.2 Higher Order Grammar

Our framework is higher-order grammar (hereafter HOG, Pollard 2001 and 2003), a version of Lambek's (1988, 1999) categorical (not categorial) grammar. In categorical grammar the role of proofs is different than in standard TLG. Instead of starting with a set of lexical signs (usually triples of a phonology/prosody, a syntactic type, and a meaning) and using proof theory to enlarge the set of triples, in categorical grammar the signs literally *are* the proofs, or more precisely, Prawitz (1965) equivalence classes of them. (Categorically, these are arrows which are in a natural one-to-one correspondence with equivalence classes of proofs, but for familiarity we will engage in a mild abuse of language and simply call them proofs.) On the Curry-Howard interpretation, this means that, e.g. the proper noun *John* is actually an inhabitant of the type NP. Logically, that means it is a proof of the proposition NP from the null premiss I (tensor identity):

$$\mathsf{John} : I \rightarrow \mathrm{NP}$$

Moreover, semantic interpretation is treated as a (categorical, not categorial) functor, which in logical terms amounts to a mapping from syntactic proofs into semantic proofs that preserves identity proofs ($p : A \rightarrow A$) and composition of proofs. In terms of the associated proof term calculi, this amounts to a translation from a bilinear lambda calculus (Mints 1977, Szabo 1978, Wansing 1992, Gabbay and de Queiroz 1992) into a more familiar (Church-Henkin-Gallin-Montague-style) classical higher-order logic.

HOG differs from Lambek's categorical grammar in employing as the syntactic type logic not the Lambek calculus but rather a full intuitionistic propositional logic with all three of weakening, contraction, and permutation. The basic types are sign types such as NP, S, and N, as well as types for the values of features such as CASE, VFORM, etc. In keeping with the formulas-as-types perspective we write $\times$, $+$, and 1 for conjunction, disjunction, and truth; categorically these are (cartesian, not tensor) product and coproduct and terminal type (in the presence of the structural rules the tensor null premiss I becomes cartesian 1). The type Bool is also provided by the logic: it is just $1 + 1$, with the truth values t and f being the canonical injections into the coproduct.

The proof term calculus, correspondingly, is not bilinear but rather a boolean, two-valued form of Lambek and Scott's (1986) higher order intuitionistic logic. Basic constants correspond to syntactic words (e.g. John of type NP denotes the word John: $1 \rightarrow \mathrm{NP}$), and also to feature values, e.g. acc of type Case denotes acc: $1 \rightarrow \mathrm{Case}$). In light of the Lambek-Scott equivalence discussed below, we don't bother to distinguish notationally between closed terms and the arrows/equivalence classes of proofs that they denote.

This logic, called SL (syntactic logic), has a robust form of subtyping: if x is a variable of type A and ϕ a propositional term with at most x free, then $[x \in A \mid \phi]$ is a subtype of A. It is important to note that in order for B to be a subtype of A, it is not enough just for there to be a proof from A to B; rather there must be a *monic* proof from A to B (the categorical generalization of an injective function from one set to another).

Because the proof term calculus for higher order intuitionistic logic is so much expressive than ones for the Lambek calculus, SL can be used in place of special purpose feature logics (such as Richter's (2000) RSRL formalism for HPSG) to impose feature constraints on types, and obviates the need to "layer" one logic over another (Doerre et al. 1996, Bayer and Johnson 1995). That is, the grammar (or at least the purely syntactic part of it) is written in SL. So it is relatively straightforward to implement a HOG grammar in a typed applicative programming language with a suitably robust form of subtyping. Note that unlike standard TLG, there is not a logic associated with the grammar but rather the grammar *is* a logical theory (since the constraints are nonlogical axioms). The canonical model of such a theory is a (bivalent boolean) topos (Lawvere 1971, Goldblatt 1984, Lambek and Scott 1986), which we call SYNTAX. Categorically, SL is the internal language of SYNTAX; the relationship between the two is one of adjoint equivalence (Lambek and Scott 1986).

A second important important difference between HOG and Lambek's categorical grammar is that it follows Curry (1961) in distinguishing phonology (phenogrammar) from syntax (tectogrammar). The way this is done is by having a second higher-order intuitionistic logic for the phonology (hereafter PL for phonological logic). The corresponding topos model is called PHONOLOGY, and (adopting Lambek's methodology for semantic interpretation) phonological interpretation is a logical functor from SYNTAX to PHONOLOGY (or equivalently, a translation from SL to PL that preserves all the logical connectives). Since the string realizations of signs are handled functorially, there is no need for the syntactic logic to be resource sensitive, and none of the structural rules in SYNTAX leads to undesirable results (Pollard 2003).

HOG makes use, both in SYNTAX and PHONOLOGY, of the fact that the Kleene-* type constructor is definable in a topos (as long as one follows Lambek and Scott in including a natural number type $\mathbb{N}$). That is, for every type A there is:

1. a type A*;
2. a monic proof $los_A : A \rightarrow A^*$ (the name is mnemonic for "length-one string");
3. an element $e_A : 1 \rightarrow A^*$ (the null A-string)
4. a proof $^\wedge{}_A : A^* \times A^* \rightarrow A^*$ (concatenation of A-strings) such that
   $\vdash \forall x(e^\wedge x = x)$
   $\vdash \forall x(x^\wedge e = x)$
   $\vdash \forall x, y, z((x^\wedge y)^\wedge z = x^\wedge(y^\wedge z))$

Also, for each A we define $A^+ = [s \in A^* | s \neq e_A]$.

It can be shown that Kleene-* is monotonic in the sense that if A is a subtype of B then A^* is a subtype of B^*. Also there is a natural embedding $h : (A \Rightarrow B)^*$ into $A \Rightarrow B^*$; intuitively, if $f_1, ..., f_n : A \rightarrow B$, then $h(f_1 \circ ... \circ f_n)$ is $\lambda x(f_1(x) \circ ... \circ f_n(x))$. (So h is a typed analog of Lisp's mapcar. This will be made more precise in the full paper.)

Thus, the PHONOLOGY topos can be obtained starting from the free bivalent boolean topos with Kleene-* over the basic type PHONEME (so that PHONEME* is the type of phonological words and PHONEME** the type of strings of phonological words) by using PL to define natural classes of phonemes and to impose phonotactic constraints. But Kleene-* will also play a crucial role in SYNTAX, as we will see.

It is crucially important that in a boolean topos, for each type A, the type $A \Rightarrow \mathrm{Bool}$ (the powertype of A) forms a boolean algebra, with local analogs of union, intersection, and complementation. It must be remembered that these are not global type constructors but only defined for the subtypes of a given type. For example, in the case of union what we have is, for each type A, a binary operation $\cup_A$ on the subtypes of A, but not a general binary type constructor like $\times$ or + that can produce a new type from any two arbitrary types.

In HOG, since directionality is handled by the phonological interpretation, there is no need for the implication constructor $\Rightarrow$ to split into $/$ and $\backslash$. Thus e.g. the transitive verb sees has type $\mathrm{NPacc} \Rightarrow (\mathrm{NPnom} \Rightarrow \mathrm{S})$ and the directionality is handled by the phonological functor $P : P(\textsf{sees}) = \lambda s \lambda t \,.\, t^\wedge/siz/^\wedge s$.

1.3 Argument Neutrality

We begin with an already solved problem, viz. how to handle cases of argument neutrality, such as (2) above. Following Levine et al. 2001, Daniels 2002, and Levy and Pollard 2002, we employ a nonstandard inventory of CASE values: pnom (pure nominative), pacc (pure accusative), and nom_acc (syncretic between nominative and accusative). Then NPacc and NPgen are defined as subtypes of NP as follows:

$$\mathrm{NPacc} =_{def} [x \in \mathrm{NP} \mid \textsc{Case}(x) = \textsf{pacc} \vee \textsc{Case}(x) = \textsf{acc\_gen}]$$
$$\mathrm{NPgen} =_{def} [x \in \mathrm{NP} \mid \textsc{Case}(x) = \textsf{pgen} \vee \textsc{Case}(x) = \textsf{acc\_gen}]$$

Also

$$\mathrm{NPacc\_gen} =_{def} [x \in \mathrm{N} \mid \textsc{Case}(x) = \textsf{acc\_gen}] = \mathrm{NPacc} \cap \mathrm{NPgen}$$

That is, case syncretism is handled by the same device as feature conjunction, just as in Bayer and Johnson 1995, except that our conjunction is (genuinely boolean) subtype intersection in a boolean topos, not additive conjunction.

By contrast, if we are given (homophonous or not) distinct signs, such as $\textsf{co}_1$ and $\textsf{co}_2$ of types NPnom and NPacc respectively, we can pair them to get $(\textsf{co}_1,\textsf{co}_2)$ of type $\mathrm{NPnom} \times \mathrm{NPacc}$, but there is no monic to get us from there to $\mathrm{NPnom} \wedge \mathrm{NPacc}$. So we do not have to posit diacritics in phonology (Morrill's suggested integers paired with strings), nor are we required to accept Whitman's conclusion that there is no syntactic distinction between ambiguity and neutrality.

1.4 Argument Coordination and Functor Neutrality I

Next we consider functor neutrality, e.g.

(6) John is rich and an excellent cook.

To get started, we need a type for the coordinated complement *rich and an excellent cook*. If Bayer is right about coordination being the lattice dual of neutralization, the obvious thing to try is $\cup$. Unfortunately however, as observed above, $\cup$ is not a global type constructor, but is only defined at each type Aas a binary operator on the subtypes of A.

The next obvious thing to try is the coproduct $+$ in SYNTAX, whose basic properties are dual to those of the product $\times$. But this will not work either, because in bivalent boolean higher order intuitionistic logic the only ways to prove $A + B$ are either (1) to prove A and then apply the injection $i : A \rightarrow \ A + B$ or (2) to prove B and then apply the injection $j : B \rightarrow \ A + B$. That is, the only ways *rich and an excellent cook* could be an $\mathrm{AP} + \mathrm{NP}$ are for it either to have been an AP to start with or to have been an NP to start with. This is a version of the same problem that arose in standard TLG in connection with analyzing coordination as additive $\vee$.

However it should be observed that $+$ is just what we want for functor ambiguity, as in:

(7) a. I canned the tuna.
b. I canned the incompetent employee.
c. *I canned the tuna and the incompetent employee.

Here, pairing of can_1 and can_2, both of type $\mathrm{NP} \Rightarrow \mathrm{VP}$, yields an $(\mathrm{NP} \Rightarrow \mathrm{VP}) \times (\mathrm{NP} \Rightarrow \mathrm{VP})$. But in intuitionistic propositional logic disjunctive syllogism and its converse are valid:[1]

$$(A + B) \Rightarrow C \equiv (A \Rightarrow C) \times (B \Rightarrow C)$$

So the ambiguity of *can* amounts to typing it to $(\mathrm{NP} + \mathrm{NP}) \Rightarrow \mathrm{VP}$.

Well, if neither the local coproduct nor the global one will model coordination, what is left? Another possibility is suggested by the analysis of Dalrymple and Kaplan (2000), which treats the f-structure of a coordinate phrase as something like the set[2] of the f-structures of the conjuncts. The natural way to formalize the DK analysis of coordination in HOG is to say that the coordination of an A and a B is a set each of whose elements is either an A or a B, that is it has type $(A + B) \Rightarrow \mathrm{Bool}$. (We really should refine things a bit to limit to nonempty finite subsets of $A + B$, but never mind; we are going to discard this analysis anyway.) More specifically, if a and b are of type A and B respectively, then $i(a)$ and $j(b)$ are both of type $A + B$ (so far this is following Morrill's account), but next we avail ourselves of the fact that (in any boolean topos), for any type C there is always an embedding $sing_C : C \rightarrow \ (C \Rightarrow \mathrm{Bool})$ mapping each element of C to the

[1] Categorically, these together constitute the Law of Exponents, which holds in any bicartesian closed category and therefore in any topos. The arithmetic law of exponents, where the types are natural numbers and the type constructors are arithmetic operations, is a special case.

[2] We say "something like" because we are not sure how much set theory is expressible in LFG's functional description language.

singleton containing it; so that, taking C to be $A + B$ and omitting the subscript, $sing(i(a))$ and $sing(j(b))$ are both of type $A + B \Rightarrow \mathrm{Bool}$. Now we have things that we can union together, so (picking up the Morrill narrative again) we form $sing(i(a))) \cup sing(b(j))$. Intuitively this is the doubleton set $\{a, b\}$. Applying this to example (6), we derive $\textsf{and}(sing(i(\textsf{rich})), sing(j(\textsf{an excellent cook})))$ as an $(\mathrm{AP} + \mathrm{NP}) \Rightarrow \mathrm{Bool}$.

Thus, we have shown that, at an appropriate level of abstraction, Morrill's analysis of coordination and Dalrymple and Kaplan's are essentially the same (modulo the identification of members of a set with their singleton subsets). Likewise (modulo the same thing) we have preserved Bayer's insight that neutralization and coordination are, respectively, intersection and union (though one must be careful about the types whose powersets these operations live on).

1.5 Argument Coordination and Functor Neutrality II

We are not out of the woods yet, though. For one thing, if we now try to analyze functor neutrality by saying that e.g. *is* is a $((\mathrm{AP}+\mathrm{NP}) \Rightarrow \mathrm{Bool}) \Rightarrow \mathrm{VP}$, it follows that it can never combine directly with a noncoordinate AP or a noncoordinate NP; instead such things must be shifted via a canonical injection followed by a singular embedding. More generally, since arguments always have the possibility of being coordinate structures, we need to change our general theory of argument selection, so that whenever a word was assigned type $A \Rightarrow B$ before, it will now be assigned $(A \Rightarrow \mathrm{Bool}) \Rightarrow B$. Second, we have not provided any account of how coordinate structures are phonologically interpreted. This is a nontrivial problem because sets are not linearly ordered, and yet the phonological realizations of the conjuncts must be. Third, there is no place in our account to locate the fact that *and* and *or* must be different signs that receive different phonological and semantic interpretations. Fourth, we have to generalize to nonbinary coordination (where some of the coordinated phrases may themselves be coordinate structures). Fifth, we need to handle so called "principled resolution" strategies for assigning agreement (person, number, and gender) features to coordinate structures (Corbett 1983), strategies which are often sensitive to the linear order of the coordinated phrases. For example, in Czech, if the coordinate subjects follow the verb, the agreement features of the verb can be dependent on the relative order of the subjects.

Because linear order enters into so many of the considerations just mentioned, rather than try to address them in terms of sets $(A \Rightarrow \mathrm{Bool})$, we prefer to refine the DK-like account sketched above by working with strings (lists) instead. o The gist of the string-based account of coordination is that for each syntactic type A, there is a type of "generalized A". As a first approximation, a generalized A can be thought of as something which is either an A or a coordination of A's. But that is too simple because it does not allow for the possibility that one or more of the coordinated phrases in a coordinate structure might themselves be coordinate structures. So instead we set things up in such a way that a generalized A is either an A or a coordination of generalized A's. More precisely, for each sign type A we have a type $\mathrm{GEN}[A]$ and a monic $gen_A : A \rightarrow \mathrm{GEN}[A]$. We now assign the conjunctions and$_A$ and or$_A$ not the type $A \Rightarrow (A \Rightarrow A)$, but rather the type $\mathrm{GEN}[A]^+ \Rightarrow (\mathrm{GEN}[A] \Rightarrow \mathrm{GEN}[A])$, with $P(\mathsf{and}) = \lambda s, t.s^\wedge/ænd/^\wedge t$. Modulo this change, the account now runs along the same lines as the hybrid Morrill-Dalrymple-Kaplan-style account given above.

By way of illustration, consider the example (5), repeated below:

(5) Kim is [drunk, stoned, or under the table] and an inveterate liar.

Such examples are also problematic for DK because the LFG functional description language only allows sets of f-structures, but not sets of sets, etc. We start by assigning is the type $\mathrm{GEN}[\mathrm{AP}+\mathrm{PP}+\mathrm{NP}] \Rightarrow \mathrm{VP}$, and using the coproduct injections composed with the *gen* injection to shift all the conjuncts to $\mathrm{GEN}[\mathrm{AP} + \mathrm{PP} + \mathrm{NP}]$. Next we use the los injections to shift each of the first two conjuncts to $(\mathrm{GEN}[\mathrm{AP} + \mathrm{PP} + \mathrm{NP}])^+$ which are then concatenated to form another $(\mathrm{GEN}[\mathrm{AP}+\mathrm{PP}+\mathrm{NP}])^+$. Then we apply or$_{\mathrm{AP+PP+NP}}$ to this and *under the table*, obtaining a $\mathrm{GEN}[\mathrm{AP} + \mathrm{PP} + \mathrm{NP}]$. (Note that we would have obtained a different one had we applied or rather than and, so that the identity of the conjunction is being taken into account.) Another application of the *los* injection shifts this to a $(\mathrm{GEN}[\mathrm{AP} + \mathrm{PP} + \mathrm{NP}])^+$, and finally and$_{\mathrm{AP+PP+NP}}$ combines with this and and *an inveterate liar* to produce a $\mathrm{GEN}[\mathrm{AP} + \mathrm{NP} + \mathrm{PP}]$.

But there is no way for a merely ambiguous functor to become neutral, e.g. for modal *can* of type $\mathrm{GEN}[\mathrm{VP}] \Rightarrow \mathrm{VP}$ and main verb *can* of type $\mathrm{GEN}[\mathrm{NP}] \Rightarrow \mathrm{VP}$ to somehow get together and cook up an $\mathrm{GEN}[\mathrm{VP} + \mathrm{NP}] \Rightarrow \mathrm{VP}$. If we pair them together we get something of type

$$(\mathrm{GEN}[\mathrm{NP}] \Rightarrow \mathrm{VP}) \times (\mathrm{GEN}[\mathrm{VP}] \Rightarrow \mathrm{VP})$$

which, by the Law of Exponentials, yields an

$$(\mathrm{GEN}[\mathrm{NP}] + \mathrm{GEN}[\mathrm{VP}]) \Rightarrow \mathrm{VP}$$

But (by a contravariance argument) to get from this to $\mathrm{GEN}[\mathrm{VP} + \mathrm{NP}] \Rightarrow \mathrm{VP}$, we would need a proof: $\mathrm{GEN}[\mathrm{VP} + \mathrm{NP}]$ to $\mathrm{GEN}[\mathrm{NP}] + \mathrm{GEN}[\mathrm{VP}]$. This cannot exist; if it did, it would let us shift the coordination of a VP and an NP either to an NP or to a VP. This is the desired result.

It may appear as though the preceding argument depends on the arguments of the homophonous functors having different types, but in fact this is not the case. Suppose instead we look at the two main verbs can$_1$ 'to put in cans' and can$_2$ 'to fire (an employee)', Then by pairing and disjunctive syllogism again, we get the copair [can$_1$,can$_2$]: $(\mathrm{GEN}[\mathrm{NP}] + \mathrm{GEN}[\mathrm{NP}]) \Rightarrow \mathrm{VP}$. To shift this to a $\mathrm{GEN}[\mathrm{NP}] \Rightarrow \mathrm{VP}$, by contravariance again we need a purely logical proof from $\mathrm{GEN}[\mathrm{NP}]$ to $\mathrm{GEN}[\mathrm{NP}] + \mathrm{GEN}[\mathrm{NP}]$. In fact there are two: the canonical injections i and j. But of course composing [can$_1$, can$_2$] with either of those just throws one of can$_1$ or can$_2$ away.

1.6 Functor Coordination.

Finally, we consider examples of functor coordination such as the celebrated

(8)	a.	Er findet	und hilft	FRAUEN
		he finds$_{acc}$	and helps$_{dat}$	women.ACC/DAT
		'He finds and helps women.'		
	b.	*Er findet	und hilft	MÄNNER.
		he finds$_{acc}$	and helps$_{dat}$	men.ACC
	c.	*Er findet	und hilft	KINDERN.
		he finds$_{acc}$	and helps$_{dat}$	children.DAT

This is less straightforward, for reasons that have nothing to do with neutrality. The reason is that the theory of coordination as sketched above does not even handle simple cases of coordination of like functors, as in

(9) John walks, talks, and smokes.

Here, starting with *walks* and *talks* both of type GEN[NP] $\Rightarrow$ S, we can easily produce *walks and talks* as a GEN[GEN[NP] $\Rightarrow$ S], but what enables this to combine with an NP, or more generally, a GEN[NP]? Note we don't have to require that the result be an S; it would be good enough for the result to be a GEN[S], the same type as

(10) John walks, John talks, and John smokes.

Ignoring the extra complication of conjuncts which are themselves coordinate structures, the problem here is that empirically, in functor coordination, the coordinated functors must all have the same directionality (i.e. be looking for their arguments in the same direction, and the phonology of the conjunction must be different for the leftward case than for the rightward one. Roughly what we need is, for the rightward case,

$$P(\mathsf{and}_r) = \lambda s.\lambda x.mapcar_r(begin(s), e)^\wedge/\text{æ}nd/^\wedge[last(s)](x)$$

and for the leftward case

$$P(\mathsf{and}_l) = \lambda s.\lambda x.[first(s)](x^\wedge/\text{æ}nd/^\wedge mapcar_l(rest(s), e)$$

where e is the null string and $mapcar_r$, $mapcar_l$ are functions like Lisp mapcar except that they are only defined on strings of functors with the same directionality. That is, in the phonology the (left or right) peripheral conjunct combines with the argument as if the other conjuncts were not there, and then the other conjuncts are appeased by having the null string fed to them.

The reason this is problematic is that, since our type theory is nondirectional, $mapcar_r$ (or $mapcar_l$) cannot tell by looking at a string of functors what their directionality is. Why? It is because the directionality is given by the functor P : SYNTAX $\rightarrow$ PHONOLOGY, which is external to SYNTAX. In order to solve the problem, it is necessary to revise the grammar architecture so that SYNTAX and PHONOLOGY are subtoposes of a common topos whose internal language talks about both syntax and phonology, as well as the interface between them. In that case P becomes a (partial) endofunctor. Thus the problem is reduced to representing the endofunctor P internally as a polymorphic function. In order to do that, we need to be working inside a topos where polymorphic functions are are definable, that is, we need our logic of phonology and syntax to be a polymorphic lambda calculus.

1.7 Conclusion

We showed that the standard TLG account of neutrality and coordination is untenable. Using as our framework a version of Lambek's categorical grammar with a type theory based on Lambek and Scott's higher order intuitionistic logic (the internal language of a topos) rather than the Lambek calculus, we showed that this account can largely be salvaged. However, the difficulty of phonologically interpreting coordinated functors of differing directionality suggests a need to handle both phonology and syntax within a single polymorphically typed lambda calculus. We intend to explore this approach next.

We suspect this is the right direction to go in, though we have yet to work out the details. We are not worried about working in an expressive formalism, given the competition (see Kepser 2001 for the undecidability of finite model checking (!) in RSRL and Carpenter 1999 on the Turing equivalence of multimodal categorial grammar). To look at the issue another way, we have never met a programmer who declined to employ a language because it was capable of expressing undecidable problems; one must just take care that the programs one wants to terminate actually do. Still another take on it is that when one formalizes a scientific theory, one gets at the real-world constraints being analyzed by the theory one writes, not by requiring that the language in which one writes the theory be inexpressive. Otherwise physicists studying gravitation might insist on writing their equations in languages where differential equations of order higher than two were not just empirically wrong but rather were syntax errors. In any case, programming language theory has long since made the move to formalisms that can express polymorphism; natural languages are *more* complicated than programming languages, so why should the math be simpler?

References

Bayer, S. 1996. The coordination of unlike categories. Language 72(3): 579-616.
Bayer, S. and M. Johnson. 1995. Features and agreement. ACL 33:70-76.
Carpenter, B. 1997. Type-Logical Semantics. Cambridge, MA: MIT Press.

Carpenter, B. 1999. The Turing-completeness of multimodal categorial grammar. In Papers Presented to Johan van Benthem in honor of his 50th Birthday. Utrecht: ESSLLI.
Corbett, G. 1983. Resolution rules: agreement in person, number, and gender. In G. Gazdar, ed., Order, Concord, and Constituency. Dordrecht: Foris, pp. 175-206.
Curry, H. 1961. Some logical aspects of grammatical structure. In R. Jakobson, ed., pp. 56-68.
Dalrymple, M. and R. Kaplan. 2000. Feature indeterminacy and feature resolution. Language 76:759-798.
Daniels, M. 2002. On a type-based treatment of feature neutrality and the coordination of unlikes. In van Eynde et al., eds., pp. 137-147.
Doerre, J., E. Koenig, and D. Gabbay. 1996. Fibred semantics for feature based grammar logic. Journal of Logic, Language, and Information 5:382-422.
Dyla, S. 1984. Across-the-board dependencies and case in Polish. Linguistic Inquiry 15(4):701-705.
Goldblatt, R. 1984. Topoi: the Categorial Analysis of Logic, 2nd edition. Amsterdam: North-Holland
Heylen, D. 1996. On the proper use of booleans in categorial logic. Formal Grammar, Prague.
Heylen, D. 1997. Generalization and coordination in categorial grammar. Formal Grammar, Aix-en-Provence.
Kepser, S. 2001. On the complexity of RSRL. FG-MOL 2001.
Lambek, J. 1961. On the calculus of syntactic types. In R. Jakobson, ed., pp. 166-178.
Lambek, J. 1988. Categorial and categorical grammars. In R. Oehrle, E. Bach, and D. Wheeler, eds., Categorial Grammars and Natural Language Structures. Dordercht: Reidel, pp. 297-317.
Lambek, J. 1999. Deductive systems and categories in linguistics. In H. Ohlbach and U. Reyle, eds., Logic, Language, and Reasoning. Essays in Honor of Dov Gabbay. Dordrecht: Kluwer, pp. 279-294.
Lambek, J. and P. Scott. 1986. Introduction to Higher-Order Categorical Logic. Cambridge: Cambridge University Press.
Lawvere, W. 1971. Quantifiers and sheaves, In Actes du Congre‘s International des Mathématique, Nice, tome I. Paris: Gautier-Villars, pp. 229-334.
Levine, R., T. Hukari, and M. Calcagno. 2001. Parasitic gaps in English: some overlooked cases and their theoretical implications. In P. Culicover and P. Postal, eds., Parasitic Gaps. Cambridge, MA: MIT Press, pp. 181-222.
Levy, R. and C. Pollard. 2002. Coordination and Neutralization in HPSG. In van Eynde et al., eds., pp. 221-234.
Moortgat, G. 1997. Categorical type logics. In J. van Benthem and A. ter Meulen, eds.
Morrill, G. 1990. Grammar and logical types. In M. Stohof and L. Torenvliet, eds., Proceedings of the Seventh Amsterdam Colloquium. Amsterdam: Institute for Logic, Language, and Information, pp. 429-450.
Morrill, G. 1994. Type Logical Grammar: Categorial Logic of Signs. Dordrecht: Kluwer.
Pollard, C. 2001. Cleaning the HPSG garage: some problems and some proposals. Unpublished manuscript, The Ohio State University. Presented at HPSG 2001, Trondheim.
Pollard, C. In review. Remarks on categorical grammar. In Casadio, C., P. Scott, and R. Seely. eds., Essays Presented to Joachim Lambek on the Occasion of his 80th Birthday. Stanford: CSLI.

Prawitz, D. 1965. Natural Deduction: A Proof-Theoretical Study. Uppsala: Almqvist and Wiksell.
Pullum, G. and A. Zwicky. 1986. Phonological resolution of syntactic feature conflict. Language 62(4):751-773.
Richter, F. 2000. A Mathematical Formalism for Linguistic Theories, with an Application in Head-Driven Phrase Structure Grammar. Ph.D. dissertation, University of Tuebingen.
van Benthem, J. and A. ter Meulen, eds. 1997. Handbook of Logic and Language. New York: Elsevier.
van Eynde, F., L. Hellan, and D. Beerman, eds. 2002. Proceedings of the Eighth International Conference on Head-Driven Phrase Structure. Stanford: CSLI Publications.
Whitman, P. 2002. Category Neutrality: a Type-Logical Investigation. Ph.D. dissertation, Department of Linguistics, The Ohio State University.
Zaenen, A. and L. Karttunen. 1984. Morphological non-distinctiveness and coordination. Proceedings of the First Eastern States Conference on Linguistics, pp. 309-320.