

# Propositional Glue and the Projection Architecture of LFG

Avery D. Andrews

ANU, December 2010

final version, to appear in *Linguistics and Philosophy*

([www.springerlink.com](http://www.springerlink.com))

The syntactic theory of LFG is largely based on the idea of multiple structural levels, with distinct properties, related by correspondence relations. The c-structure-to-f-structure correspondence standardly called  $\phi$  was a prominent feature of the original theoretical architecture in Kaplan and Bresnan (1982), further developed in later presentations of the theory (Kaplan (1995), Dalrymple (2001), Bresnan (2001)). But structural correspondences are not such a prominent feature of ‘glue semantics’, the most developed approach to semantic composition in LFG.<sup>1</sup>

In this paper I will show how to remove some obstacles to integrating glue into the correspondence architecture, most importantly by simplifying it to use purely propositional linear logic rather than linear logic with (universal) quantifiers, and also by having a correspondence running from the glue proofs to the f-structures, rather than in the usual surface-to-abstract direction. Not only does this result in better support for proposals to connect the structure of the glue-proofs to (the rest of) the syntactic structure, but it also permits both the ‘semantic projection’ of standard glue and a level of ‘argument-structure’ developed in recent LFG to be eliminated (the latter subsumed into certain aspects of the structure of the glue-proof), leading to a substantial simplification of the framework.

## 1 The Role of Quantification in Glue

That most of glue is effectively propositional linear logic can be seen by looking at a simple example such as the analysis of a sentence such as *Bert likes Ernie*. Given standard phrase-structure rules, together with the lexical entries of (1), which include meaning-constructors, this sentence gets the f-structure (2):<sup>2</sup>

- (1) a. *Bert*: N, ( $\uparrow$ PRED) = ‘Bert’, *Bert* :  $\uparrow_e$
- b. *Ernie*: N, ( $\uparrow$ PRED) = ‘Ernie’, *Bert* :  $\uparrow_e$
- c. *likes*: B, ( $\uparrow$ PRED) = ‘Likes<( $\uparrow$ SUBJ)( $\uparrow$ OBJ)>,  
*Like* : ( $\uparrow$ OBJ)<sub>e</sub>  $\multimap$  ( $\uparrow$ SUBJ)<sub>e</sub>  $\multimap$   $\uparrow_p$

---

\* I would like to acknowledge an anonymous reviewer for thoughtful and interesting questions and observations, the late Robert K. Meyer for help with some issues concerning logic, Ash Asudeh and Miltiadis Kokkonidis for discussion of various issues, and the students in my 2006 Syntactic Theory course at ANU, whose questions about an earlier version of this material helped provide motivation and guidance for this version. Needless to say, I am the one responsible for all errors and omissions.

<sup>1</sup> It is however worth noting that glue semantics is no longer used as such in the PARGRAM project grammars, having been replaced by f-structure rewriting (Crouch (2006), Crouch and King (2006)). But f-structure rewriting is clearly intended as a technology rather than as a theory, and it remains to be seen whether this technological change embodies any theoretical ideas, and, if so, what they are.

<sup>2</sup> We follow the convention of omitting rightmost parentheses with  $\multimap$ , and use  $p$  rather than the more usual  $t$  as the type symbol for propositions ( $e$  being for entities, as usual). The status of the types will be briefly discussed at the end of subsection 3.3.

$$(2) \left[ \begin{array}{l} \text{SUBJ } g: [\text{PRED 'Bert'}] \\ f: \text{PRED 'Likes'} \langle (\uparrow \text{SUBJ}) (\uparrow \text{OBJ}) \rangle \\ \text{OBJ } h: [\text{PRED 'Ernie'}] \end{array} \right]$$

As a side-effect, the meaning-constructors of the lexical entries will have their  $\uparrow$ -arrows instantiated to f-structure labels like this, where  $e$  and  $p$  are semantic type subscripts indicating ‘entity’ and ‘proposition’, respectively:

$$(3) \begin{array}{l} \text{Bert} : h_e \\ \text{Ernie} : h_e \\ \text{Like} : h_e \multimap g_e \multimap f_p \end{array}$$

These constructors can serve as assumptions for this labelled (natural deduction) proof-tree:<sup>3</sup>

$$(4) \frac{\frac{\text{Like} : h_e \multimap g_e \multimap f_p \quad \text{Ernie} : h_e}{\text{Like}(\text{Ernie}) : g_e \multimap f_p} \multimap \mathcal{E} \quad \text{Bert} : h_e}{\text{Like}(\text{Ernie})(\text{Bert}) : f_p} \multimap \mathcal{E}$$

The label on the final conclusion represents the desired meaning assembly for the sentence.

The atomic formulas of this deduction can be viewed in various ways. In the currently standard ‘new glue’ formulation (introduced in Dalrymple et al. (1999), and used in most recent work), they are constants of various types, as indicated by the semantic type subscripts, in the System F of Girard et al. (1989), which can be regarded as having second order quantification over propositional variables.

Kokkonidis (2008) shows that a formally simpler system can be produced by treating the semantic type symbols as 1-place predicates, which apply to the f-structure locations, which are treated as individuals. Then the quantification that we will shortly be looking at becoming ordinary first-order linear quantification.

But for the derivation above, and many others in glue, only the propositional inference rules are used, so that the atomic formulas could be regarded simply as pairs consisting of semantic type and f-structure location. The upgrade to some kind of quantified linear logic only becomes motivated when we consider quantifier scope.

It was a driving insight of the glue approach that the principles governing linear logic deduction would guarantee correct scopings of complicated combinations of NL quantifiers, where many other approaches produce wrong results (Dalrymple et al., 1997). The effect is that a meaning-constructor such as (5) below for *everybody* will work (given in fully explicit first-order format here, although we will later revert to the conventional format with semantic type subscripts), even though it doesn’t explicitly state any restriction on scope, which is represented by the variable  $H$ , bound by a linear universal quantifier:

$$(5) \lambda P. \text{Every}(z, \text{Person}(z), P(z)) : \forall H((e \uparrow) \multimap p(H)) \multimap p(H)$$

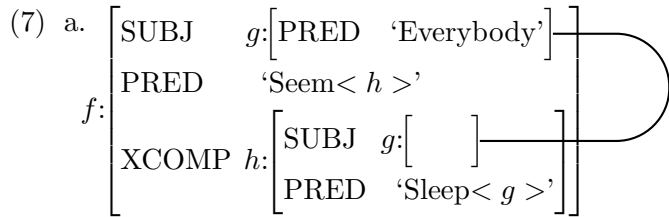
In a glue derivation, the variable  $H$  can be instantiated to any existing f-structure label by the rule or (linear) Universal Instantiation, thereby allowing multiple scopes in examples such as:

<sup>3</sup> Using the proof-rules of Asudeh (2005b). The labels are supposed to be terms in a typed lambda-calculus; I will normally omit the type information from the meanings.

(6) Everybody seems to sleep

But it is possible to eliminate this quantification, by in essence shifting its function to the instantiation component of LFG. In fact this has already been done, implicitly by Fry (1999), and explicitly by Lev (2007), but without discussion of the general significance of this move.

Consider first the issue of formulation. For (6), the f-structure would be as in (a) below, the instantiated constructors (following Asudeh (2005a)) as in (b):



- b. *Everybody* :  $\forall H((e(g) \multimap p(H)) \multimap p(H))$   
*Seem* :  $p(h) \multimap p(f)$   
*Sleep* :  $e(g) \multimap p(h)$

In the glue-proof, as discussed in greater detail by Asudeh, a Universal Instantiation step can instantiate  $H$  to either  $f$  or  $h$ , leading to two possible (normal) proofs representing the two quantifier scopings. As mentioned above, in more complicated examples, the constraints on the structure of proofs will block proofs that would otherwise representing meaningless readings.

But suppose that instead of a quantifier-bound variable in the uninstantiated meaning-constructor for *everybody*, we just had a ‘local name’ (Dalrymple, 2001, pp. 146-148):

- (8) *Everybody* :  $(e(\uparrow) \multimap p(\%H)) \multimap p(\%H)$

The desired effect is that  $\%H$  be able to be instantiated to any possible scope for the quantifier, but we don’t quite have this, since, by the notion of ‘minimal solution’ to an f-description in LFG,  $\%H$  would not in fact get identified with anything in the f-structure, for the reason that one aspect of a minimal solution is that it doesn’t make any identifications of substructure beyond what is required to satisfy the f-description.

But we can fix this problem by adding an additional specification to the lexical entry, which will use an inside-out-functional-uncertainty (iofu) equation (Dalrymple, 2001, pp. 143-146) to identify  $\%H$  with some containing f-structure (this is really just yet another way to implement Cooper Storage):

- (9) *Everybody*:N,  $(\uparrow\text{PREL}) = \text{‘Everybody’}$ ,  $(\text{GF}^* \uparrow) = \%H$ ,  
*Everybody* :  $(e(\uparrow) \multimap p(\%H)) \multimap p(\%H)$

A possible worry is that there might be some possible scope not accessible by a simple iofu path, but in the glue analyses that have so far been proposed, this does not appear to happen. To give the idea some empirical bite, one would also want to see the iofu paths used to impose some constraints, but we won’t attempt this here.

Therefore, by replacing linear quantification with LFG instantiation, we can make the glue derivations strictly propositional. This is certainly a mathematical simplification, but what

kinds of effects does it have? Dramatic and immediate empirical ones would seem unlikely, because as long as there is only universal quantification in glue, which appears to be sustainable, all possible arrangements of the quantifiers are logically equivalent to prenex position (Kokkonidis 2008, pg 53), and so can be omitted, yielding the same effective expressive power as the present proposal.

Theoretically, however, one potentially interesting consequence is that propositional glue is actually an instance of a kind of logical system which has been under investigation for a considerably longer time than linear logic. If we avoid the use of the tensors, we get a system called BCI, which goes back at least to the work of Carew Meredith in the 1950s (Prior, 1963, p. 316), and is still an active research subject today. Adding tensors doesn't essentially change this, rather we get what might be called 'BCI with fusion'.<sup>4</sup> One aspect of these consequences is that glue proofs can be viewed as arrows in a Symmetric Monoidal Closed Category, another extensively investigated kind of structure. Whether anything of genuine linguistic interest can gotten from these connections remains to be seen, but I think it is in general good for linguistics to be able to relate to as much pre-existing mathematics as possible.

Another is that the relationship between the glue proof and the resulting meanings is simplified: in standard glue, the meaning-terms elide without comment the UI steps whereby the quantifiers are removed, but if there are no such steps, this minor discrepancy between the proof and semantic structure does not arise.

But the consequence we'll be primarily concerned with in the sequel is that getting rid of the quantifiers makes it possible to integrate glue directly and completely into the correspondence architecture of LFG, with the semantic assemblies looking and acting like a normal level of linguistic representation, with one simple difference. However we must also make sure that this somewhat subtle formal change doesn't produce any bad empirical consequences (good ones would of course be fine). In the next section, we'll work through the integration of glue with the rest of the grammatical structure, and in the one after, we will look into some areas where problems might arise.

## 2 Glue by Correspondence

Glue is standardly presented by means of labelled deductions, originally in the Gentzen Sequent calculus, more often, recently in tree-style Natural Deduction. But it is a consequence of the Curry-Howard Isomorphism (CHI) that for the ND format at least, the labels are almost unnecessary, because the structure of proofs becomes largely the same as the structure of the lambda-terms that appear as labels on the proofs, to represent the 'logical forms' that sentences are supposed to receive (thinking of a logical form as a representation that's intended to be a good input to detection of semantically-based properties and relations such as entailment and contradiction). Universal Instantiation creates a slight problem with this, since it doesn't correspond to anything normally found in logical forms (the proof-term builders that it produces in for example the presentation of System F in Girard et al. (1989) are simply elided in the labelled deduction formats used in glue, with no discussion in the literature that I am aware of).

But when we get rid of the Universal Instantiation steps, this minor discrepancy is eliminated, and we can say that glue proofs are a (kind of) logical form, exactly. This is formally an immediate consequence of the CHI, but can be made more visually concrete, and homogenous with other aspects of linguistics, by borrowing some ideas from the proof-net literature. Proof-

---

<sup>4</sup> A name for the multiplicative version of conjunction often employed in Relevant Logic.

nets are a mainstay of the Type-Logical Grammar literature (e.g. (Moot (2002), Morrill (2005)), and have been occasionally invoked in LFG (Fry (1999), Andrews (2007a)), but not extensively utilized.

What they provide is a format for building proofs out of pre-assembled pieces, essentially identical in form to glue meaning-constructors, which can be assembled into full proofs by following relatively simple rules. Furthermore, although the usual proof-net format doesn't look like a standard logical form, it can be converted into a more conventional format by some fairly simple transformations, which furthermore support an intuitively easy version of an essential constraint on proof-nets, the 'Correctness Criterion'.

In the first subsection, we'll discuss the standard proof-net format and the technique for converting this into what we'll call the 'prefab' format, which provides a familiar looking representation of pieces of proofs/lambda terms. This section might be skipped, especially on a first reading, by readers without some familiarity with proof-nets. Then in the second subsection, we'll show how to construct the prefab format directly from the meaning-constructors, and in the third, show how the prefabs are to be assembled.

## 2.1. FROM PROOF-NETS TO PREFABS

In a proof-net, the premises and conclusion<sup>5</sup> of an argument are represented as tree-structures, whose nodes are labelled with formulas, and a proof is represented as a pairing of the leaves of the trees (labelled with atomic formulas) that satisfies certain constraints. Many of the presentations in the literature are rather difficult to decipher, due to arcane labeling conventions derived from the origins of proof-nets in the one-sided sequent calculus, but one can eventually work out that the structure of these trees is essentially that of Montagovian types. The root of a tree is labelled with the entire type, the branches with the subformulas, and the leaves with atomic formulas.

Glue proofs are subject to the Intuitionistic restriction that there be only one conclusion, and obey the further restriction that this be of type  $f_p$ , where  $f$  is the f-structure label of the entire f-structure (at least if we're parsing a declarative sentence). For purely implicational linear logic, this permits the tree-nodes to be labelled with implications rather than the 'tensor' 'par' and negation symbols originally used (see e.g. Moot (2002) or de Groote (1999) for the details). Pars can be made to disappear entirely, while tensors might still be needed for the analysis of anaphora, as discussed below.

An important feature of intuitionistic proof-nets labelled with implicational formulas is 'polarity', originated by Jaśkowski (1963). The premises of a proof-net are assigned negative polarity, the conclusion(s) positive, and polarity propagates through the tree according to the following rules:

- (10) a. The polarity of the consequent of an implication is the same as that of the implication, that of the antecedent the opposite
- b. The polarity of the components of a tensor is the same as that of the whole tensor

The polarities seem unnatural from the point of view of compositional semantics, and are sometimes reversed in the LFG literature, but it's probably best to go with the logical tradition here.

---

<sup>5</sup> Or conclusions, for non-Intuitionistic proof-nets.

Logicians write the component trees of proof-nets root-down, but for linguists, it's probably more natural to write them root-up. The roots are also labelled with the meaning-side of the constructor, which we'll call the 'semantic label'. We illustrate with these instantiated constructors for the sentence *everybody doesn't sleep*:

$$(11) \text{ Everybody} : g_e \multimap f_p \multimap f_p \\ \text{Not} : f_p \multimap f_p \\ \text{Sleep} : g_e \multimap f_p$$

The standard proof-net format for these will then be:

$$(12) \begin{array}{ccc} g_e^- & f_p^+ & \\ \swarrow & \searrow & \\ g_e \multimap f_p^+ & f_p^- & \\ \swarrow & \searrow & \\ (g_e \multimap f_p) \multimap f_p^- & & \end{array} \quad \begin{array}{cc} f_p^+ & f_p^- \\ \swarrow & \searrow \\ f_p \multimap f_p^- & \end{array} \quad \begin{array}{cc} g_e^+ & f_p^- \\ \swarrow & \searrow \\ g_e \multimap f_p^- & \end{array} \\ \text{Everybody} \qquad \text{Not} \qquad \text{Sleep}$$

Ambiguity will in this case be produced by the existence of multiple ways of connecting the atomic formulas, rather than alternative instantiations of local names.

To finish the set-up for a glue-proof, we add one additional premise of positive polarity, labelled with semantic type  $p$ , and the label of the entire f-structure. This represents the kind of conclusion we want our proof to produce. It has no meaning-label, since the meaning is supposed to be provided by the assembled net.

Next, rather surprisingly, if there is a proof of the conclusion from the premises, it can be expressed as a way of linking the (atomically labelled) leaves with 'axiom links'<sup>6</sup> that satisfies the following constraints:

- (13) a. The leaves are linked in non-overlapping pairs
- b. where the members of each pair are of the same semantic type and have the same f-structure label
  - c. but opposite polarity
  - d. subject to the Correctness Criterion (to come)

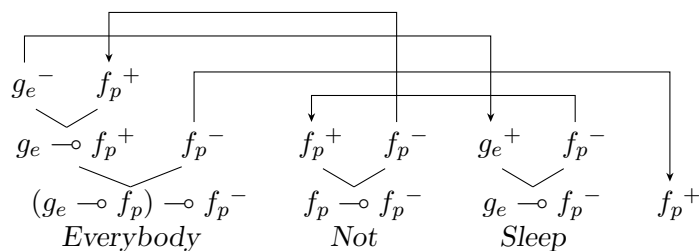
The links can be regarded as inherently undirected, but it will be intuitively helpful to regard them as directed from negative to positive.

There are two possible linkages of (12) that satisfy (13) as well as the Correctness Criterion:

$$(14) \text{ a.} \begin{array}{ccccccc} & & & & & & \\ & & & & & & \\ & & & & & & \\ g_e^- & f_p^+ & & & & & \\ \swarrow & \searrow & & & & & \\ g_e \multimap f_p^+ & f_p^- & & & & & \\ \swarrow & \searrow & & & & & \\ (g_e \multimap f_p) \multimap f_p^- & & & & & & \\ \text{Everybody} & & & & & & \end{array} \quad \begin{array}{cc} f_p^+ & f_p^- \\ \swarrow & \searrow \\ f_p \multimap f_p^- & \end{array} \quad \begin{array}{cc} g_e^+ & f_p^- \\ \swarrow & \searrow \\ g_e \multimap f_p^- & \end{array} \quad f_p^+ \\ \text{Not} \qquad \text{Sleep}$$

<sup>6</sup> So-called because they represent instances of the identity axiom in the one-sided sequent calculus.

b.



There's also a third, which violates the Correctness Criterion, and so represents neither a proof nor a sensible reading.

It isn't intuitively obvious that these things represent either proofs or readings, but that they represent proofs is a standard result in the linear logic literature, and conventional-looking lambda-calculus terms for the two readings can be extracted from them by proof-net reading methods such as those of de Groote and Retoré (1996), Perrier (1999), and other sources.

But there's an easier way for linguists at least to look at this, which is to reorganize the trees into what, thanks to the CHI, can be regarded as being either pre-assembled pieces of natural deduction proofs, or pre-assembled pieces of lambda-terms. One way of looking at this reorganization is as the relationship between a Montagovian type, and the shape of the kind of typed lambda-calculus structure that something of that type would fit into. So for example, an expression of type  $a \rightarrow b$  designates a function that applies to something of type  $a$  to produce something of type  $b$ . The relationship between the type-tree and the associated expression-tree for an implication/function can be represented as follows:

(15) type                      expression

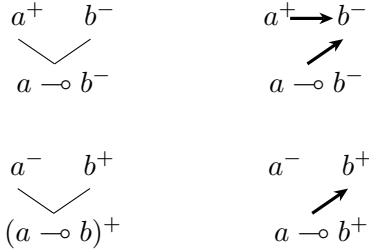


For lambda-abstractions, things are a bit more complicated due to the possibly long-distance binding-like relationship; we'll look at that shortly,

This tree-reorganization turns out to be closely related to a concept of proof-net theory, the 'dynamic graph' of de Groote (1999), a minor variant of the 'essential net' of Lamarche (1994).<sup>7</sup> This is a directed graph consisting of the axiom-links, oriented from negative to positive as we have depicted them, together with, for each implicational subtree, the links shown as thick arrows on the right below, where on the left appear positive and negative polarity implications in the usual form used in proof-nets (modulo being upside-down), and on the right, the corresponding dynamic graph links:

<sup>7</sup> But the deGroote paper is much more accessible; indeed, I have made very little progress in comprehending the contents of the Lamarche paper.

(16) premise subtree      dynamic graph



The dynamic graph was conceived of as something built on the basis of an entire proof-net, but it can also be constructed for individual premise-trees, that is, glue-sides of meaning-constructors.

Looking at the dynamic graph for a negative implication, it is evident that it is just like an expression tree, with the dynamic graph arrows going from the daughters to the mother. In the case of the positive implication, which represents a lambda-abstraction, things are a bit different: the arrow represents a link from the formula-component of the lambda-abstraction to the whole, with, in the dynamic graph, no representation of the link to the variable, which is represented by the consequent of the (positive) implication. To represent this, we'll need to retain the link from the implication to its antecedent in our derived representation; this will be represented below as a curved, dotted arrow.

The so-transformed constructors, which I will call 'prefabs' (on the basis that their processing from the type-like format is done prior to assembly), now represent pieces of conventional lambda-terms, which can be assembled by the usual proof-net rules into something that looks much more like a standard logical form than standard proof-nets.

As an exercise, one could work out what the transformed versions of the constructors for *Ernie*, *sleeps*, *not* and *everybody* would be; these will be given immediately below in the next section, where we show how to construct the transformed versions of the constructors directly from the meaning-constructors.

## 2.2. PREFABS FROM CONSTRUCTORS

We can construct our prefab proof-net format directly from the meaning-constructors, by a bottom-up method. One decision is whether to put the implication/function as the right or the left daughter; left seems more natural in the light of standard logical notation, but right makes the assembled structures less awkward to depict, so that's the convention we follow here.

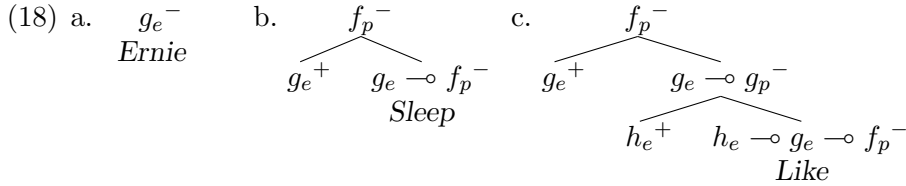
So our method is as follows:

- (17) a. Starting with a meaning-constructor, create a node with the meaning-side as 'semantic' label, glue-side as 'structural label', and give it negative polarity (for the benefit of readers who skipped the previous section. this is a property of a node that is to have values of  $-$  and  $+$ ).
- b. If the structural label of a node is an atomic formula, the construction from that node is finished.
- c. If a negative polarity node has an implication as its structural label, construct as its mother a new negative polarity node whose structural label is the consequent of

that implication, and as left-daughter of this new node a new positive polarity node whose structural label is the antecedent of that implication (leaving the starter node as right-daughter). Then apply whichever of (b,c,d) is appropriate to the new nodes.

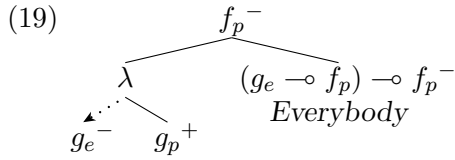
- d. If a positive polarity node has an implication as its structural label, construct a negative ‘left pseudo-daughter’ labelled with the antecedent (representing the pseudo-daughter relation with a curved, dotted arrow), and a right daughter labelled with the consequent. Then apply whichever of (b,c,d) is appropriate to the new nodes.

Some simple results of (a-c) are:



(a) sets up the starter node with (17a), then stops because it’s finished. (b), after the starter, has one application of (17c) to set up the mother and left-daughter, then stops because both of these new nodes are finished. (c) follows the starter with two successive applications of (17c) before it is finished.

Rule (d) gets to apply with the meaning-constructor for quantifiers. A notational point is because the structural label of a positive implication is trivial to read off from that of its daughters, and such an implication functions as a lambda-binder, it is convenient and suggestive just to write it as  $\lambda$ , so for *Everybody* we get:



The solid lines represent the dynamic graph links from the previous subsection, with assumed upward orientation, while the dotted line to the left pseudo-daughter represents the conventional proof-net link from a positive implication to its (negative) antecedent.

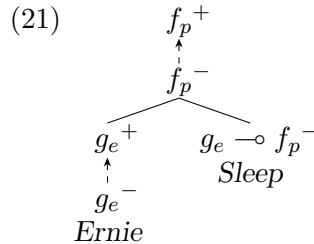
It is obvious that there is a great deal of redundancy in these representations, since everything is predictable from the labeling and polarity of the rightmost leaf, but this redundant detail is nevertheless useful for understanding the rules for assembly, to which we now turn. They are of course the same as in the previous subsection, but the new format allows them to produce things that look more like conventional logical forms, and, more importantly, the Correctness Criterion, which we have until now avoided formulating, gets a very intuitive interpretation.

The technique for assembly is to run ‘axiom links’, which we will now represent as dashed arrows, between the atom-labelled nodes according to the rules of (20) below, which are essentially the same of (13), but reformulated to fit the present circumstances. We also provide, as before, a single positive node with label  $f_p$ , where  $f$  is the f-structure label of the entire f-structure being interpreted:

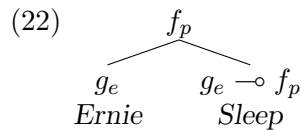
- (20) a. The nodes with atomic formula labels are linked in non-overlapping pairs
- b. where the members of each pair are of the same semantic type and have the same f-structure label

- c. but opposite polarity
- d. subject to the Correctness Criterion (to come).

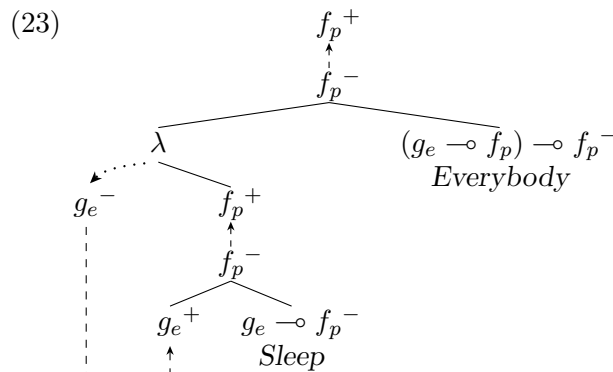
A pleasant consequence of our rearrangement is that the negative nodes can often be put more or less under the positives they are to be plugged into, creating a fairly ordinary-looking tree-structure. The assembly for *Ernie sleeps* would for example be:



If we ‘fuse’ the axiom-linked nodes and erase the polarities, we get a completely banal syntax tree for a logical form with the predicate in final position:

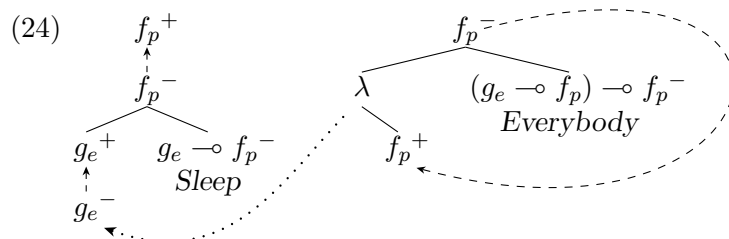


The topography is a bit more complex for quantifiers. Here the left pseudo-daughter is negative, so it needs to plug into a positive. A reasonable arrangement for *Everybody sleeps* would be:



Shortly below we’ll see an example with a negative operator coming in between the quantifier and predicate of which the ‘bound variable’ position/left pseudo-daughter is argument.

We’re finally ready to formulate the Correctness Criterion. It is needed, because in (23), there is an alternative way of setting up the axiom links that satisfies the rules so far (which is probably a bit easier to see in the standard proof-net format), but produces neither a valid proof nor a sensible meaning-assembly:



The Correctness Criterion for proof-nets has a large number of different-looking but equivalent formulations; see Moot (2002) for a discussion of some of them. The one given here follows from the algebraic criterion of de Groote (1999), and characterizes what's wrong with (24) essentially in terms of what appears to be wrong with it visually, which is that something with the function of a variable is being bound by something that doesn't have scope over it.

To formulate the criterion, we'll first restate the definition of the dynamic graph:

- (25) The dynamic graph consists of the tree nodes together with the axiom-links (dashed lines), oriented from negative to positive, and links from negative implications to their (mother) consequents, and from the antecedents of the negative implications to their consequents (the solid lines, oriented upwards).

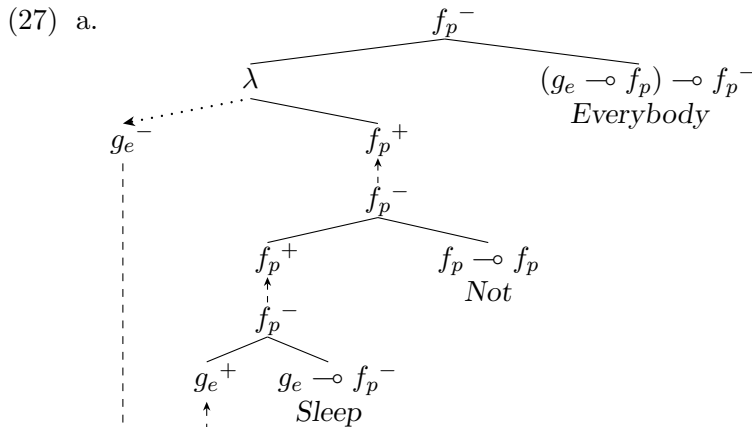
We now formulate the criterion:

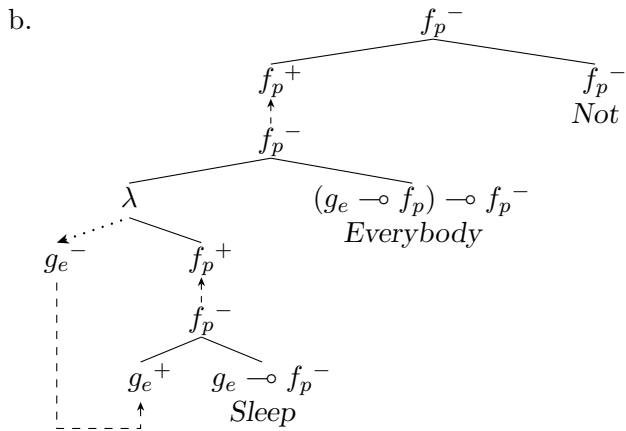
- (26) **Correctness Criterion:** The dynamic graph must be (a) rooted and acyclic, and (b) every path to the root that starts at the target of a dotted (pseudo left-daughter) link must pass through the source of that link.

(23) clearly satisfies (26), while (24) clearly doesn't, on two counts, absence of a root, and the existence of a path that starts at the target of a dotted link but doesn't pass through its source.

Condition (a) of the criterion is formulated a bit more generally than it has to be for glue-structures containing only implication; for these, it would suffice and be more intuitive to just say that the dynamic graph must form a tree (which would constitute a reasonable parse-tree for a formula). The formulation given covers the more complex case when tensors are included; we discuss this later. Condition (b) amounts to the requirement that variables be properly bound.

Now, given the obvious meaning-constructor for the negative, we can re-present the two assemblies for *everybody doesn't sleep*, given above in (14) in conventional proof-net format like this, where we've omitted the top positive node, leaving the top negative unpaired:





As before, we can get conventional-looking logical forms by fusing axiom-linked nodes.

We now have a propositional glue system for the core fragment of Dalrymple et al. (1999) which supports a system of assembly based on representations of a form that is reasonably familiar to linguists, and able to be assembled by fairly simple rules. It might seem that all we have done is taken a substantial trek through proof-nets in order to rediscover natural deduction, as in fact first noted by Perrier (1999), and in a sense we have done that, but this isn't quite all, because proof-nets are a very limited format for proofs; for example there does not yet appear to be any generally accepted way of using proof-nets for classical logic, and even the additives and units in 'non-rudimentary' linear logic cause substantial problems for proof-nets.

So we have found a version of natural deduction trees sufficient for quite a lot of semantic composition inside a more restrictive system, and, furthermore, shown that it is really a rather ordinary-looking format for grammatical representation. Note, for example, that aside from the 'modification' construction (which we would have to render as types of the form  $a \rightarrow a$ ), our prefabs and assembly rules are consistent with the proposals for 'Conceptual Structure' in Jackendoff (2002) and Culicover and Jackendoff (2005), but using worked out mathematics to suppress bizarre possibilities for assembling them.

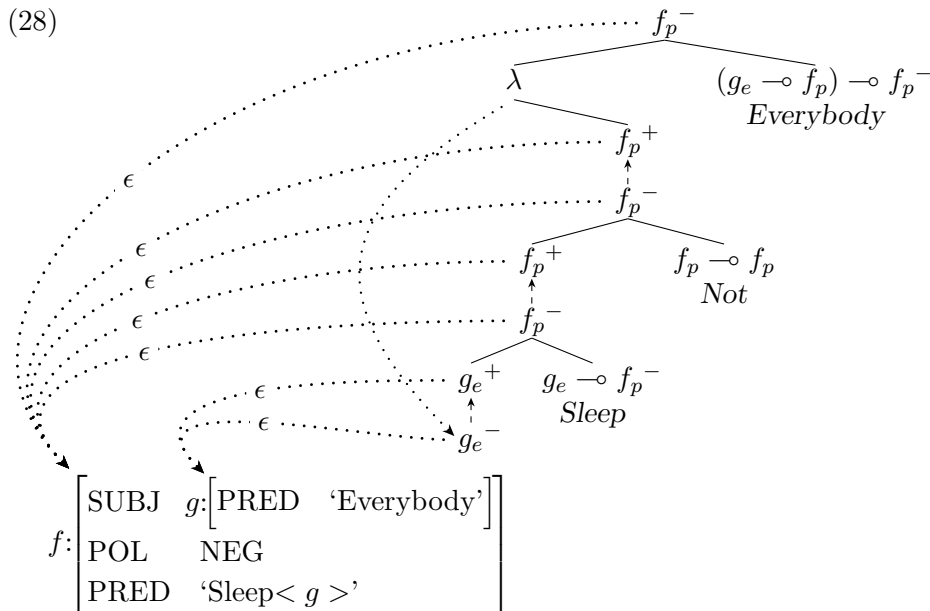
The relationship between natural deduction and proof-nets becomes looser when tensors are added, but before going on to this, we will reconsider the nature of the relationship between the glue assemblies and f-structure, and identify the respect in which the proposed architecture differs significantly from many others.

### 2.3. REVERSING $\sigma$

The connection between the glue-structure and the f-structure has been represented by the f-structure labels in the atomic formulas of the meaning-constructors. But the choice of labels is an arbitrary feature of the representation that we would want to eliminate at least on a conceptual level. This is in fact rather easy to do, using the ideas of Kaplan's (1995) correspondence-architecture, if we think of the f-structure labels as representations of a 'semantic projection'  $\epsilon$  that is essentially Kaplan's  $\sigma$  with its directionality reversed ( $\sigma^{-1}$ ). That is, rather than going from f-structures to their logical forms, it goes from certain nodes in the logical forms, specifically, the atomic nodes, to the f-structures.

The conceptual advantage of this is that  $\epsilon$  can then be a function, whereas traditional  $\sigma$  cannot be, due to the compression of structural layers that seems to be a basic characteristic of the relation between overt syntax and meaning, directly reflected in the LFG architecture.

On this conception, the f-structure and glue logical form of the wide-scope reading of *everybody doesn't sleep* can be represented as:



All of the different glue-nodes labelled  $f_p$  (three layers, if we fuse along the axiom-links) correspond to the single f-structure  $f$ . The other standard types of modification, discussed in Dalrymple (2001, ch. 11), show the same kind of many-to-one  $\epsilon$ -correspondence.

A useful consequence of using a functional  $\epsilon$  rather than  $\sigma$  appears when we consider the nature of the constraints on assembly. The requirement that the f-structure labels match simply becomes the requirement that the axiom-linked nodes share the same  $\epsilon$ -value. The  $\epsilon$ -value specification can be naturally seen as a basic property of the atomic subformulas. There is also the semantic type, which would be naturally viewed as a basic property of the meaning-side, propagating through the prefab or type tree of a constructor via type-assignment rules. The agreement in semantic type could be seen as a kind of semantic filtering. But we will note at the end of section 3.3 that there is a prospect for eliminating the semantic types.

We have now constructed a version of glue-semantics that is significantly cleaner and simpler than previous ones, and is more conformant with the correspondence-based nature of the LFG architecture. It will clearly produce the same results in implicational analyses not involving anaphora, but there are a number of more complex situations which we need to check over, to make sure that they don't produce problems. This is the task of the next section.

### 3 Some Possible Issues

The three things I will discuss here are intensional verbs, tensors and anaphora, and the potential for re-introducing the standard glue 'semantic projection', which we have provisionally discarded. In principle there might be other problems, but these are the ones that seem to emerge from the current literature.

#### 3.1. INTENSIONAL VERBS

So far, all of our meaning-constructors have been of 'antecedent depth' not greater than 2, meaning that, when rule (17d) applies, the antecedent of the positive implication it applies

to is atomic (that is, no lambda-binding of 2nd order variables). Most meaning-constructors that have appeared in actual analyses seem to have this property, but there is at least one interesting and important exception, the rendition into glue by Dalrymple et al. (1997) of Montague’s analysis of intensional verbs such as *seek*.

Here the idea is to capture the apparent scope ambiguity of examples such as *John seeks somebody/a unicorn* by having the object position associated with the semantic type  $(e \rightarrow p) \rightarrow p$ . The easiest way to render this in propositional glue is assign to *seek* a meaning-constructor like this:

$$(29) \text{ Seek} : (((\uparrow \text{OBJ})_e \multimap (\uparrow \text{OBJ})_p) \multimap (\uparrow \text{OBJ})_p) \multimap (\uparrow \text{SUBJ})_e \multimap \uparrow_p$$

Note in particular, that unlike in previous versions, there is no linear quantifier or corresponding functional uncertainty, because this is unnecessary and produces unwanted multiple analyses.

Given (29), the constructors for a quantified NP such as *a unicorn* will then be able to instantiate their local name  $\%_0H$  to either the f-structure of the object or to some higher structure, and the derivation will then go through smoothly, as in the standard treatment.

A final thing that might be useful is to look at the diagrams for a 2nd-order argument such as  $(h_e \multimap h_p) \multimap h_p$ . This will be a positive implication, so it should expand to this:

$$(30) \begin{array}{c} \lambda^+ \\ \swarrow \quad \searrow \\ h_e \multimap h_p^- \quad h_p^+ \end{array}$$

But the left-pseudo daughter is a negative implication, so it expands like this:

$$(31) \begin{array}{c} \lambda^+ \\ \swarrow \quad \searrow \\ h_p^- \quad h_p^+ \\ \swarrow \quad \searrow \\ h_e^+ \quad h_e \multimap h_p^- \end{array}$$

Which fits together properly with the other structures, even though it doesn’t look like a tree, as the ones we saw earlier do. Note that the Correctness Criterion will require the node labelled  $h_p^-$  to axiom-link to something from which the dynamic path will pass through the node labelled  $h_p^+$ . Sometimes it might be helpful to write in such path-requirements as yet another kind of link, but this would also increase visual clutter, and they are reasonably obvious from the geometry.

We have now shown how the standard glue analysis of intensional verbs works in propositional glue. Before moving on, however, it is perhaps worth noting that this analysis is not beyond question (e.g. (Deal, 2007)). If it can be eliminated, then presently investigated meaning-constructors will not have ‘antecedent depth’ greater than 2, something from which interesting proof-theoretic consequences can follow (Tatsuta, 1993), although not necessarily any that are useful for linguistics.

## 3.2. TENSORS AND ANAPHORA

The history of the use of tensors in glue has been a bit tangled. In the original ‘old glue’ formulation, they were the technique normally used to feed multiple arguments to transitive verbs, but in the ‘new glue’ format of Dalrymple et al. (1999), currying was adopted for this purpose, and tensors were excluded from the ‘core fragment’. Tensors for arguments will be discussed briefly below. Asudeh (2004) provides an extensive discussion of various other possible use of tensors in glue analyses, but the only one that seems to have been substantially developed is for treating bound anaphora. There have however been a fair range of alternatives to this proposed in glue.<sup>8</sup>

In spite of its competitors, the tensor analysis is relatively simple, used in analyses of further phenomena such as resumptive and copy pronouns (Asudeh (2004), Asudeh (2005b), Asudeh and Toivonen (2009)), and has some resemblance to analyses that have been proposed in Type-Logical Grammar (Jäger, 2005),<sup>9</sup> so propositional/correspondence-based glue probably ought to be able to support it.

Before beginning, it might be useful to discuss in general terms what a tensor is. Speaking roughly, it’s a way of combining two objects so that they can function with some degree of individual autonomy in a composite, but not necessarily in such a way that either can be discarded (if both discard and copying are possible, the tensor is a ‘product’). The essential idea is perhaps best conveyed by the natural deduction rules for tensor introduction and elimination:

$$(32) \quad \frac{A \quad B}{A \otimes B} \otimes \mathcal{I} \qquad \frac{A \otimes B \quad \begin{array}{c} [A]^i [B]^j \\ \vdots \\ C \end{array}}{C} \otimes \mathcal{E}^{i,j}$$

The introduction rule expresses the idea of combining two objects into a composite, while the elimination rule says that the composite can do the same job in combinations as the components it was built from.

But nothing in these rules licenses the discard of a component from a combination, as is the case for the standard elimination rules for logical conjunction:

$$(33) \quad \frac{A \wedge B}{A} \wedge \mathcal{E} \qquad \frac{A \wedge B}{B} \wedge \mathcal{E}$$

(However, if the (tree-style natural deduction equivalents of the) ‘structural rules’ of Weakening and Contraction are available, then the effect of discard can still be achieved with tensors.)

The rules of (32), in conjunction with ‘commuting conversions’ for identifying proofs (usefully reviewed in Benton et al. (1992, 1993) and Mackie et al. (1993)), turn out to characterize

<sup>8</sup> Which include an implication-only formulation in Lev (2007), which he adopts for reciprocals, but not, ultimately, for bound anaphora, where he follows Kokkonidis (2005) in proposing a DRT-based account. Other proposals resembling DRT include Crouch and van Genabith (1999) and Dalrymple (2001), although these also involve some substantial innovations to the glue framework itself.

<sup>9</sup> Jäger’s ‘restricted contraction’ can be regarded in the glue framework as ‘notational sugar’ for tensors, obeying the restriction that on the meaning-side, the meaning of the input must be returned unaltered to its original position (I’m indebted to Ash Asudeh for some discussion of Jäger’s approach).

the basic principles of monoidal categories (Troelstra, 1992), which describe some useful properties of certain ways of combining algebraic systems that are called ‘tensor products’, whence the name and the  $\otimes$  symbol.<sup>10</sup>

Returning to bound pronouns, the first point is that they have antecedents, which are treated in LFG as values of an ANT(ECEDENT) relation, as described in Dalrymple (1993). Then, their meaning constructor ‘takes’ content from the antecedent position, and uses tensoring to both put content (back) there, and in the position of the pronoun. The meaning-constructor for an anaphoric pronoun therefore looks like this (assuming conventionally that  $\otimes$  binds tighter than  $\multimap$ ):

$$(34) \quad \lambda x.[x, x] : (\uparrow \text{ANT})_e \multimap (\uparrow \text{ANT})_e \otimes \uparrow_e$$

An important aspect of this constructor is that from the point of glue, all that’s going on is that the constructor is consuming something in the antecedent position, and depositing something both there and in the pronoun position; that these things are actually copies is strictly internal to the meaning-side, which the glue analysis *per se* doesn’t know anything about (exponential-free linear logic can’t make copies).

The tensor rules can be used to construct the following proof for *Ernie washed himself*, where we’ve made various abbreviations to fit it onto the page, and also included a term-assignment for the  $\otimes$ -elimination rule which is different from the one used by Asudeh, which will be discussed shortly:

$$(35) \quad \frac{\frac{\lambda x.[x, x] : g_e \multimap g_e \otimes h_e \quad E : g_e}{\lambda x.[x, x](E) : g_e \otimes h_e} \quad \frac{W : h_e \multimap g_e \multimap f_p \quad [y : h_e]^i}{W(y) : g_e \multimap f_p} \quad [z : g_e]^j}{\frac{\lambda x.[x, x](E) : g_e \otimes h_e \quad W(y)(z) : f_p}{W(\pi^1(\lambda x.[x, x](E))) (\pi^2(\lambda x.[x, x](E))) : f_p} \otimes_{\mathcal{E}^{i,j}}}$$

The right subtree of this proof represents the fact that we can apply the transitive verb *W(ash)* to two arbitrary things in object and subject position to get something of type  $f_p$ , while the left subtree represents the fact that the pronoun *himself* takes something from subject position (its antecedent) and deposits something both there and in object position, using tensoring to represent the placement of content in two different places.

Then the final  $\otimes_{\mathcal{E}}$  step says that the combined placement on the left can be identified with the arbitrarily assumed objects on the right, delivering an OK proposition. An obvious question is why we didn’t  $\beta$ -reduce  $\lambda x.[x, x](\text{Ernie})$  on the left as soon as we got it, and the answer is that this would be a misleading thing to do within a linear logic proof, since linear logic can’t make the required copies. A more linearly honest representation would be to have the meaning-side of the reflexive pronoun be merely an opaque constant such as *Refl*.

And so we come the proof-term of the final step. The term-assignment we’re using is:

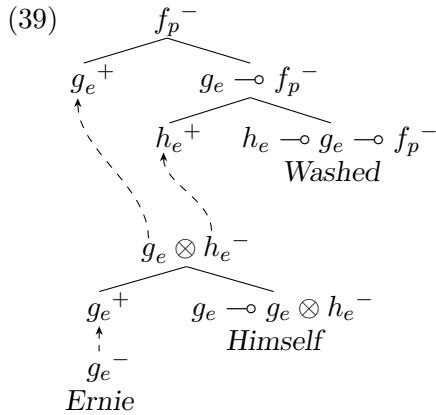
$$(36) \quad \frac{z : A \otimes B \quad \begin{array}{c} [x : A]^i [y : B]^j \\ \vdots \\ f : C \end{array}}{[\pi^1(z)/x, \pi^2(z)/y] f : C} \otimes_{\mathcal{E}^{i,j}}$$

<sup>10</sup> People intrigued by such things might enjoy looking at Baez and Stay (2011).



natural representation of the hierarchical argument-structures proposed by Alsina (1996), and others, as discussed by Andrews (2007b),<sup>11</sup> which have motivations from a variety of earlier work, such as the semantic role hierarchy of Jackendoff (1972). On the other hand, one could certainly envision a variant of glue in which no claims whatsoever were made concerning hierarchical relations between the arguments.

For negative polarity tensors, the only developed example is the analysis of anaphora. In the prefab notation, the most natural representation would be an upward-oriented (root-at-bottom) tree, but given the limited use of the structure, it doesn't seem worth working out how to typeset such trees, so just labeling nodes with things like ' $h_e \otimes g_e^-$ ' seems adequate. Then, by the rules for axiom-linking, we can run an axiom link from each atomic component of a negative tensor to some matching positive, giving a result like this:



Now it should be clear that if we embed the main predication of (39) under a negative or any other kind of operator that doesn't introduce any additional appropriate positive atomic nodes, it will make no difference to the possibilities for hooking up the components of the negative tensor.

And the proof-net of (39) is a natural representation of the proof-term of (35), and vice-versa. One can regard the projection notation as a linearization of proof-nets involving negative tensors, where  $\pi^1$  gives the 'view' of the tensor from its left component,  $\pi^2$  from its right. This can be easily used to extend Perrier's (1999) 'maximal labeling' to provide readings for proof-nets with tensors (although not necessarily a version of the Correctness Criterion).

This would be a good place to reconsider the Correctness Criterion. Observe that tensors have the capacity to 'fork' a dynamic path starting at a negative antecedent (of a positive implication), but condition (b) of (26) requires that all of these forks be recombined before the path goes through the positive consequent of that implication. It is also worth noting that it is the introduction of the tensors that require that the dynamic path be constrained to be acyclic (without forks, the requirement that every dynamic path go to the root is sufficient to suppress cycles).

Tensors then don't make any major problems to propositional/prefab glue and its reversed directionality of  $\sigma$ . They do however inject more complexity into some of the formulations. Therefore, proposals to eliminate them from meaning-constructors should certainly be taken seriously. Nevertheless, even with tensors, the proofs still fit unproblematically into the rather restrictive proof-net formulation; the fact that this seems to be Girard's favorite format for proofs should not, I think, be taken lightly.

<sup>11</sup> But there are alternatives to this style of argument-structure, e.g. Asudeh (2001).

### 3.3. IMPLICIT ARGUMENTS AND THE SEMANTIC PROJECTION

Our last point concerns a major feature of this proposal, the elimination of the ‘semantic projection’. In standard glue, this is an additional structural level beyond f-structure, but distinct from the glue-proof, which is to serve as the structural type component of the atomic formulas in the glue-proofs, rather than having the f-structures themselves fulfill that role, as proposed here.

Although a well-worked out empirical motivation for this level doesn’t seem to appear in the original literature, one can construct an intuitive-esthetic argument along the following lines, based on the analysis of common nouns and their interactions with quantifiers. On the basis that the standard semantic type for common nouns would be  $e \rightarrow p$ , we would want their glue types to be of the form  $X_e \multimap Y_p$ , for some values of  $X$  and  $Y$ . Quantificational determiners such as *some* and *every* should then apply to common nouns to produce quantified NPs of type  $(Z_e \multimap H_p) \multimap H_p$ , so the determiners should be of type  $(X_e \multimap Y_p) \multimap (Z_e \multimap H_p) \multimap H_p$ .

Workable results will in fact be produced if the common noun and quantificational determiners have uninstantiated glue-sides like this:

$$(40) \quad \begin{aligned} CN &: \uparrow_e \multimap \uparrow_p \\ QD &: (\uparrow_e \multimap \uparrow_p) \multimap (\uparrow_e \multimap \%H_p) \multimap \%H_p \end{aligned}$$

But having so many different semantic transactions going on at the f-structure designated by  $\uparrow$  is perhaps somewhat unsettling.<sup>12</sup>

One use of the semantic projection is to provide a place to put some non-syntactic attributes, so that each atomic subformula of the constructors in (40) involve a different structural location. Essentially following HPSG, the standard proposal has been to associate the common noun  $e$  and  $p$  arguments with attributes VAR and RESTR. One could put these directly in the f-structure, but they don’t have any independent role there, so it seems reasonable to put them in another projection, the ‘semantic projection’ linked to f-structure by the correspondence called ‘ $\sigma$ ’, which is not fully equivalent to Kaplan’s original  $\sigma$ , because it doesn’t go from f-structure to a representation of actual meaning, but only to an f-structure-like projection that functions as a ‘staging area’ for the composition of the meaning.

The meaning-constructors for common-nouns and quantifiers would then look like this (similar to Dalrymple (2001, p. 250), but not using a SPEC-value for the f-structure of NPs; note also the omission of the semantic type information, which can be hard to integrate typographically with the semantic projection specification):

$$(41) \quad \begin{aligned} CN &: \uparrow_\sigma \multimap \uparrow_\sigma \\ QD &: ((\uparrow_\sigma \text{ VAR}) \multimap (\uparrow_\sigma \text{ RESTR})) \multimap (\uparrow_\sigma \multimap \%H_\sigma) \multimap \%H_\sigma \end{aligned}$$

The intended f- and semantic structure for NPs is then like this:

$$(42) \quad \left[ \begin{array}{cc} \text{SPEC} & \text{Every} \\ \text{PRED} & \text{‘Muppet’} \end{array} \right] \cdots \sigma \cdots \rightarrow \left[ \begin{array}{cc} \text{VAR} & [ \ ] \\ \text{RESTR} & [ \ ] \end{array} \right]$$

<sup>12</sup> And, it might be relevant that in the early days of glue, the easier and more intuitive formulations of the Correctness Criterion appear to have either not existed, or not been widely known, so that it might not have been so easy to be sure that constructors such as these would behave properly. The extra attributes introduced below reduce the reliance on the Correctness Criterion to exclude bad proofs.

But in spite of the intuitive motivation, a solid argument was lacking.

Asudeh (2005b) fills this gap with an argument based on the properties of relational nouns, depending on the tensor analysis of bound anaphora introduced above, and also on his use of ‘resource managers’ to allow resumptive pronouns, from Asudeh (2004).

The first major point is that the implicit arguments of relational nouns such as *neighbor* are understood as if they introduced anaphors that can be bound by quantifiers:

(43) Every resident complained about a neighbor

But in languages that allow resumptive pronouns, relational nouns aren’t able to function as if they introduced such pronouns, as illustrated by these examples from Swedish (Asudeh, 2005b, p.379):

- (44) a. Varje förtsbo som Maria vet att hann arresterades försvann  
 Every suburbanite that Maria knows that he was-arrested vanished  
 Every suburbanite who Maria knows that he was arrested vanished
- b. \* Varje förtsbo som Maria vet att en granne arresterades försvann  
 Every suburbanite that Maria knows that a neighbor was-arrested vanished  
 Every suburbanite who Mary knows that one of his neighbors was arrested  
 vanished

This is a significant puzzle for binding theory, which Asudeh solves by locating the anaphoric implicit argument of the relational noun on the semantic projection, where the resource managers that he uses to allow resumptive pronouns can’t see it.

This provides an argument for the semantic projection, but we can neutralize it with our present analysis by allowing the meaning-constructor for a relational noun to involve a piece of f-structural material that isn’t linked into the rest of f-structure by any grammatical function other than the ANT(ECEDENT) relation.

This is achieved by the following constructors and side-equation for a relational noun such as *neighbor*:

$$(45) \text{Neighbor} : \%H_e \multimap \uparrow_e$$

$$\lambda x.[x, x] : (\%H \text{ ANT})_e \multimap (\%H \text{ ANT})_e \otimes \%H_e$$

$$(\%H \text{ ANT}) = (\text{GF}^* \uparrow)$$

$\%H$  will here construct a ‘free-floating’ piece of f-structure which bears no GF to the rest of the syntactic structure (since no equation forces it to be identified with anything else), but is connected to it by an ANT attribute. This allows the implicit argument of a relational noun to be bound by a quantifier.

And it also prevents it from being visible to a ‘manager’ resource, which has a somewhat formidable appearance, but merely picks up a pronoun that has a given antecedent, and throws it away. The general form of a manager resource is:

$$(46) \lambda P.\lambda x.x : (A_e \multimap A_e \otimes P_e) \multimap A_e \multimap A_e$$

where  $A$  is supposed to be the f-structure (or semantic projection, if this is assumed) of the antecedent, and  $P$ , that of the pronoun. The first argument on the glue-side is supposed

to match the glue-side of a pronoun, but the meaning-side discards it. Then, basically just to allow this subexpression be an argument, occupying a positive polarity position, we need some more material to occupy a negative polarity position, and the identity axiom  $(A_e \multimap A_e)$  is an obvious choice.

The full glue-side needs to specify the f-structure positions for  $A$  and  $P$ , which it does via functional uncertainty and the ANT(ECEDENT) relation. In our approach, Asudeh’s version (p. 430) would come out as:

$$(47) ((\uparrow \text{GF}^+ \text{ ANT})_e \multimap (\uparrow \text{GF}^+ \text{ ANT})_e \otimes (\uparrow \text{GF}^+)_e) \multimap (\uparrow \text{GF}^+ \text{ ANT})_e \multimap (\uparrow \text{GF}^+ \text{ ANT})_e$$

Recall that this is one of the meaning-constructors for a relative-clause construction, and that in successful derivations,  $\text{GF}^+$  will wind up instantiated to the GF-path of the relativized-on NP (it would probably be good to restate this using a local name equated to  $(\uparrow \text{GF}^+)$ ).

The anaphoric element introduced by a relational noun can’t be picked up by such a manager for the same reason as in Asudeh’s analysis, that it doesn’t lie at (the semantic projection of) the end of a  $\text{GF}^+$  path. Indeed, perhaps the account is a bit stronger in the present version, since there wouldn’t be any possibility of hacking the path to introduce a hop onto the semantic projection. But we won’t dwell on that possibility here, but simply claim to have transferred Asudeh’s account into a glue framework with no semantic projection.

What, then, of the intuitive-esthetic argument? Which might be able to be upgraded to more than intuitive, since it might prove useful to have constraints limiting the number of different things in the glue-structure that can  $\sigma$ -correspond to a given f-(sub)structure. An interesting possibility is afforded by the ‘INDEX’ and ‘CONCORD’ for the f-structures of NPs, fairly standard in the HPSG literature (Wechsler and Zlatić (2003) and other works), and also used in some recent LFG (Wechsler (2004), King and Dalrymple (2004), Hahn and Wechsler (2007)). Given that CONCORD seems to be associated more with NP-internal, and INDEX more with NP-external agreement, the following meaning-constructors for common nouns and quantificational determiners might be worth considering:

$$(48) \text{ CN} : (\uparrow \text{CONCORD})_e \multimap \uparrow_p \\ \text{ QD} : ((\uparrow \text{CONCORD})_e \multimap \uparrow_p) \multimap ((\uparrow \text{INDEX})_e \multimap \%H_p) \multimap \%H_p$$

A side effect of this proposal is that it ceases to be so clear that LFG needs two basic semantic types.

Partee (2006) discusses the general issue of whether two basic types are needed for semantics, and suggests that perhaps they are not. In LFG glue, on the other hand, it has been assumed that two basic types are needed in order to block a potentially bad assembly for *everyone sleeps*, discussed in Kokkonidis (2008, pg. 62), originally communicated by Mary Dalrymple:

$$(49) \text{ Everyone} : (g^1 \multimap g^1) \multimap g^2 \\ \text{ Sleep} : g^2 \multimap f$$

Here axiom-linking is represented by co-superscripting.

The distinction between entity and propositional types blocks this, by preventing the negative antecedent from linking to the positive consequent in the 2nd-order argument of *Everyone*, which would otherwise be legal. But the suggested localization of the positive antecedent at a INDEX- or CONCORD-value rather than the f-structure of the whole NP would also serve to block the bad reading.

Regardless of how this turns out, in the apparent absence of further motivation for the standard semantic projection, for example in the form of demonstrations of distinctive patterns of ‘spreading’ in the sense of Andrews and Manning (1999), it seems reasonable to dispense with it.<sup>13</sup>

#### 4 Conclusion

By reducing LFG’s glue-semantics to propositional linear logic, we have both allowed it to connect in a simpler way to the other parts of the syntactic representation, and assimilated it to a large body of pre-existing work in logic and mathematics. A further effect is to increase its resemblance to certain other contemporary approaches to syntax, in particular, the variants and descendants of Categorical Grammar, and the Minimalist Program.

Categorical Grammar and its variants and descendants (‘extended Categorical Grammar’, or ‘ECG’, one might call it) differ from LFG+glue in using a non-commutative logic extended with modalities to describe the syntax. This logic is then mapped into a commutative implicational logic to describe semantic composition. By eliminating the universal quantifiers from glue, we allow it to have the same kind of ‘semantic back end’ as ECG, and created a kind of hybrid<sup>14</sup> of techniques from Chomsky, Lambek and traditional grammar.<sup>15</sup>

An important goal of ECG is to uphold a rather stringent conception of ‘Direct Compositionality’ (Jacobson 1999, 2002, 2005), whereby the semantic interpretation of an expression is to be constructed directly by the syntactic rules that build it. LFG can be argued to obey direct compositionality from c-structure to f-structure (Asudeh, 2006), but it seems to me that the final step to the glue proof is not in accord with the spirit of DC, even if one managed to uphold some kind of technical conformity, perhaps on the basis that you can compile a proof-net into a combinator, if you want to. The issue lies in the treatment of NL quantifiers using either iofu or linear quantification to instantantiate their f-structure variables, which is described here in connection with example (9) as ‘yet another way to implement Cooper Storage’. The intent is clearly to think in terms of a long-distance relationship between the NP and some dominating constituent, rather than in terms of combinators or type shifting operations applying directly to the intervening material. And although one might chose to say that the glue-proof isn’t actually part of the syntax, there in fact appear to be too many interesting connections (Crouch and van Genabith (1999), Fry (1999), Asudeh and Crouch (2002), Andrews (2007c)) for this to be sustainable.

This might be counted as a point against the approach, but, on the other hand, LFG has been developed in confrontation with what has by now become a rather long list of what might be called ‘big descriptive problems from exotic languages’. Such as case-marking, agreement and grammatical relations in Icelandic (Andrews (1982), Andrews (1990)), case-marking, agreement and grammatical relations in Warlpiri and other Australian languages (Simpson (1983), Simpson and Bresnan (1983), Simpson (1991), Nordlinger (1998)), phrase-structure and grammatical relations in Tagalog (Kroeger, 1993), the typology and analysis of syntactic

---

<sup>13</sup> There is a further problem with the traditional semantic projection, communicated to me by Miltiadis Kokkonidis, originally from Mary Dalrymple, to the effect that, as noted by some presently unknown person, since the objects on this projection are mostly empty, they will all wind up being identified as the same object, under the standard set-theoretical interpretation of feature-structures in LFG. This problem can be avoided by using the graph-theoretical interpretation of Kuhn (2003) instead.

<sup>14</sup> Technically, I suppose, a chimera.

<sup>15</sup> But note that this view cannot extend to Lambek’s more recent framework of ‘pregroup grammars’ (e.g. Lambek (2008) and many other works), for the reason that their syntax is a kind of ‘compact’ monoidal category, in the commutative versions of which, at least, it is hard to see how to represent arguments of higher types, essential to many glue analyses.

ergativity (Manning, 1996), ‘case-stacking’ in Australian languages and apparently related phenomena (Andrews (1996), Nordlinger (1998), Nordlinger and Sadler (2004)), and more. ECG seems so far to have not been as extensively tested against a typologically diverse range of languages. The difference in typological coverage creates a difficulty for making a fair empirical comparison.

On the other hand, it is reasonable to claim that LFG+glue is variable-free, due to the fact that what it uses to fill the role of variables are well-understood graph-theoretical structures (the links from negative atomic antecedents to positive ones) rather than having an apparatus of type-identity between scattered tokens, definitions of  $\alpha$ -equivalence, etc. This is reflected in the fact that, if you want to, you can specify a model-theoretic interpretation as a category-theoretic functor, via the interpretation of a glue-proof as an arrow in an SMCC (see Troelstra (1992) for details; this is essentially the same thing as re-rendering the proof as a combinator, or in a Hilbert system for BCI, possibly with fusion). But the best treatment of pronouns remains unclear (for everybody, it seems to me).

Connecting glue to the Minimalist Program seems like a longer stretch, but there is a connection, noted by Andrews (2007a), in that the assembly of meaning-constructors is highly comparable to organizing a computation from a Numeration. One can, in fact, pull a collection of meaning-constructors from the lexicon (equivalent to forming a Numeration), decide how to hook up their atomic formulas (similar if not fully equivalent to applying external and internal merge), and run LFG in generation mode to find a c- and f-structure combination that produces the resulting glue-structure (comparable to sending it off to the Perceptual-Articulatory Interface, with no guarantee that it won’t crash).

A possibly useful fact is that the free assembly of meaning-constructors is simply an application of Carew Meredith’s rule of Condensed Detachment (Substitution and Modus Ponens combined into one step, in a Hilbert axiomatization), which is occasionally studied in the logic literature (Hindley and Meredith (1990), Hindley (1993)). Regular LFG meaning-assembly, construed as applying to the output of a parser, where instantiated constructors have their atomic nodes connected by axiom-links subject to the Correctness Criterion, can be viewed as a constrained version of CD, where the f-structure and semantic types limit what substitutions can be made. The version of OT-LFG proposed in Andrews (2007a) shares with the MP the property that it uses an unconstrained version of CD, which is then filtered by interfaces.

A more tangible kind of increased resemblance comes from the fact that with the glue-proofs, LFG acquires something that can be treated as a binary-branching level of structure that with overall topographical similarity to what is assumed in the MP (and there are possibilities for setting up counterparts to some of the more detailed machinery, such as vP and little v, if that should seem motivated). Unlike the situation with ECG, there are MP treatments of most of the major descriptive problems considered in LFG, although they differ considerably in the degree of formal explicitness and exact version of the MP employed. Furthermore, the MP has its own descriptive topics that the LFG community has not yet paid much attention to. But anything that the similarities between frameworks might help to identify the significance of the remaining differences.

## References

- Alsina, A.: 1996, *The Role of Argument Structure in Grammar*. Stanford, CA: CSLI Publications.
- Andrews, A. D.: 1982, ‘The Representation of Case in Modern Icelandic’.
- Andrews, A. D.: 1990, ‘Case-Structures and Control in Modern Icelandic’. In: *Modern Icelandic Syntax*. pp. 187–234.
- Andrews, A. D.: 1996, ‘Semantic Case-Stacking and Inside-Out Unification’. *Australian Journal of Linguistics* **16.1**, 1–55.

- Andrews, A. D.: 2007a, ‘Generating the Input in OT-LFG’. In: J. Grimshaw, J. Maling, C. Manning, and A. Zaenen (eds.): *Architectures, Rules, and Preferences: A Festschrift for Joan Bresnan*. Stanford CA: CSLI Publications, pp. 319–340.
- Andrews, A. D.: 2007b, ‘Glue Semantics for Clause-Union Complex Predicates’. In: M. Butt and T. H. King (eds.): *The Proceedings of the LFG ’07 Conference*. Stanford CA: CSLI Publications, pp. 44–65.
- Andrews, A. D.: 2007c, ‘Glue Semantics for Clause-Union Complex Predicates’. In: M. Butt and T. H. King (eds.): *The Proceedings of the LFG ’07 Conference*. Stanford CA: CSLI Publications, pp. 44–65. Retrieved November 15, 2010, from <http://csli-publications.stanford.edu/LFG/12/lfg07.html> (accessed Feb 19, 2010).
- Andrews, A. D. and C. D. Manning: 1999, *Complex Predicates and Information Spreading in LFG*. Stanford, CA: CSLI Publications.
- Asudeh, A.: 2001, ‘Linking, Optionality and Ambiguity in Marathi’. In: P. Sells (ed.): *Formal and empirical issues in optimality-theoretic syntax*. Stanford, CA: CSLI Publications, pp. 257–312.
- Asudeh, A.: 2004, ‘Resumption as Resource Management’. Ph.D. thesis, Stanford University, Stanford CA. Retrieved November 15, 2010, from <http://http-server.carleton.ca/~asudeh/>.
- Asudeh, A.: 2005a, ‘Control and Resource Sensitivity’. *Journal of Linguistics* **41**, 465–511.
- Asudeh, A.: 2005b, ‘Relational Nouns, Pronouns and Resumption’. *Linguistics and Philosophy* **28**, 375–446.
- Asudeh, A.: 2006, ‘Direct Compositionality and the Architecture of LFG’. In: M. Butt, M. Dalrymple, and T. H. King (eds.): *Intelligent Linguistic Architectures: Variations on Themes by Ronald M. Kaplan*. Stanford, CA: pp. 363–387.
- Asudeh, A. and R. Crouch: 2002, ‘Derivational Parallelism and Ellipsis Parallelism’. In: L. Mikkelsen and C. Potts (eds.): *WCCFL 21 Proceedings*. Somerville MA, pp. 1–14, Cascadilla Press. Retrieved November 15, 2010, from <http://http-server.carleton.ca/~asudeh/research/index.html>.
- Asudeh, A. and I. Toivonen: 2009, ‘Copy-Raising and Perception’. To appear in *Natural Language and Linguistic Theory*. Retrieved 15 Nov 2009 from <http://http-server.carleton.ca/~asudeh/>.
- Baez, J. and M. Stay: 2011, ‘Physics, Topology, Logic and Computation: A Rosetta Stone’. In: B. Coecke (ed.): *New Structures for Physics*. Springer, pp. 95–168. Retrieved November 17, 2010, from <http://math.ucr.edu/home/baez/rosetta.pdf>.
- Benton, N., G. M. Bierman, J. M. E. Hyland, and V. de Paiva: 1992, ‘Term Assignment for Intuitionistic Linear Logic’. Technical report. Retrieved November 15, 2010 from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.8666>.
- Benton, N., G. Biermann, V. de Paiva, and M. Hyland: 1993, ‘A Term Calculus for Intuitionistic Linear Logic’. In: M. Bezem and J. F. Groote (eds.): *Proceedings Intl. Conf. on Typed Lambda Calculi and Applications, TLCA ’93, Utrecht, The Netherlands, 16–18 March 1993*, Vol. 664. Berlin: Springer-Verlag, pp. 75–90.
- Bresnan, J. W.: 2001, *Lexical-Functional Syntax*. Blackwell.
- Crouch, R.: 2006, ‘Packed Rewriting for Mapping Text to Semantics and KR’. In: M. Butt, M. Dalrymple, and T. H. King (eds.): *Intelligent Linguistic Architectures: Variations on a Theme by Ronald M. Kaplan*. Stanford CA: CSLI Publications, pp. 389–416.
- Crouch, R. and T. H. King: 2006, ‘Semantics via F-structure Rewriting’. In: M. Butt and T. King (eds.): *Proceedings of LFG 2006*. Stanford, CA: CSLI Publications.
- Crouch, R. and J. van Genabith: 1999, ‘Context Change, Underspecification, and the Structure of Glue Language Derivations’. In: Mary Dalrymple (ed.): *Syntax and Semantics in Lexical Functional Grammar: The Resource-Logic Approach*. pp. 117–189.
- Culicover, P. W. and R. S. Jackendoff: 2005, *Simpler Syntax*. Oxford University Press.
- Dalrymple, M. (ed.): 1999, *Syntax and Semantics in Lexical Functional Grammar: The Resource-Logic Approach*. MIT Press.
- Dalrymple, M.: 2001, *Lexical Functional Grammar*. Academic Press.
- Dalrymple, M., V. Gupta, J. Lamping, and V. Saraswat: 1999, ‘Relating Resource-based Semantics to Categorical Semantics’. pp. 261–280. Earlier version published in *Proceedings of the Fifth Meeting on the Mathematics of Language*, Saarbrücken (1995).
- Dalrymple, M., R. M. Kaplan, J. T. Maxwell, and A. Zaenen (eds.): 1995, *Formal Issues in Lexical-Functional Grammar*. Stanford, CA: CSLI Publications. Retrieved Nov 15, 2010, from <http://standish.stanford.edu/bin/detail?fileID=457314864>.
- Dalrymple, M., J. Lamping, F. Pereira, and V. Saraswat: 1997, ‘Quantification, Anaphora and Intensionality’. *Logic, Language and Information* **6**, 219–273. appears with some modifications in Dalrymple (1999), pp. 39–90.
- Dalrymple, M. E.: 1993, *The Syntax of Anaphoric Binding*. Stanford CA: The Center for the Study of Language and Information.

- de Groote, P.: 1999, ‘An Algebraic Correctness Criterion for Intuitionistic Multiplicative Proof-Nets.’. *Theoretical Computer Science* **224**, 115–134. Retrieved 15 November, 2010, from <http://www.loria.fr/~degroote/bibliography.html>.
- de Groote, P. and C. Retoré: 1996, ‘On the Semantic Reading of Proof-Nets’. In: G. G.-J. Kruijff and D. Oehrlé (eds.): *Formal Grammar*. FOLLI Prague, pp. 57–70. URL: <http://citeseer.ist.psu.edu/degroote96semantic.html>.
- Deal, A. R.: 2007, ‘Property-type objects and modal embedding’. In: A. Grønn (ed.): *Proceedings of SuB 12*. Oslo, pp. 92–106, Department of Literature, Area Studies and European Languages, University of Oslo. Retrieved November 17, 2010, from <http://www.hf.uio.no/ilos/forskning/aktuelt/arrangementer/konferanser-seminarer/2007/SuB12/proceedings/>.
- Fry, J.: 1999, ‘Proof Nets and Negative Polarity Licensing’. In: M. Dalrymple (ed.): *Syntax and Semantics in Lexical Functional Grammar: The Resource-Logic Approach*. pp. 91–116.
- Girard, J.-Y., Y. Lafont, and P. Taylor: 1989, *Proofs and Types*. Cambridge: Cambridge University Press. Retrieved 15 November, 2010, from <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.85.5358&rep=rep1&type=pdf>.
- Hahm, H.-J. and S. Wechsler: 2007, ‘Untangling the Russian Predicate Agreement Knot’. In: M. Butt and T. H. King (eds.): *The Proceedings of the LFG '07 Conference*. Stanford CA: CSLI Publications, pp. 233–249. Retrieved November 15 2010 from <http://csli-publications.stanford.edu/LFG/12/lfg07.html>.
- Hindley, R.: 1993, ‘BCK and BCI Logics, Condensed Detachment and the 2-property’. *Notre Dame Journal of Formal Logic* **34**, 231–250.
- Hindley, R. and D. Meredith: 1990, ‘Principal Type-schemes and Condensed Detachment’. *Journal of Symbolic Logic* **55**, 90–105.
- Jackendoff, R. S.: 1972, *Semantic Interpretation in Generative Grammar*. MIT Press. extensive revision of 1969 MIT dissertation.
- Jackendoff, R. S.: 2002, *Foundations of Language*. Oxford: Oxford University Press.
- Jacobson, P.: 1999, ‘Towards a Variable-Free Semantics’. *Linguistics and Philosophy* **22**, 117–184.
- Jacobson, P.: 2002, ‘The (Dis)organization of the Grammar: 25 Years’. *Linguistics and Philosophy* **25**, 601–626.
- Jacobson, P.: 2005, ‘Direct Compositionality and Variable-Free Semantics: the Case of ‘Principle B’ Effects’. In: *Direct Compositionality*. Oxford University Press.
- Jäger, G.: 2005, *Anaphora and Type Logical Grammar*. Springer.
- Jaśkowski, S.: 1963, ‘Über Tautologien, in welchen keine Variable mehr als zweimal vorkommt’. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* **9**, 219–228.
- Kaplan, R. and J. Bresnan: 1982, ‘Lexical-Functional Grammar: a Formal System for Grammatical Representation’. In: J. Bresnan (ed.): *The Mental Representation of Grammatical Relations*. Also in Dalrymple *et al.* (eds) 1995 *Formal Issues in Lexical-Functional Grammar*, CSLI Publications; page number references to 1982 version.
- Kaplan, R. M.: 1987, ‘Three Seductions of Computational Psycholinguistics’. In: P. Whitelock, M.M.Wood, H. Somers, R. Johnson, and P. Bennet (eds.): *Linguistics and Computer Applications*. Academic Press, pp. 149–188. reprinted in Dalrymple *et al.* (1995), pp. 337–367.
- Kaplan, R. M.: 1995, ‘The Formal Architecture of LFG’. In: M. Dalrymple, R. M. Kaplan, J. T. Maxwell, and A. Zaenen (eds.): *Formal Issues in Lexical-Functional Grammar*. CSLI Publications, pp. 7–27.
- King, T. H. and M. Dalrymple: 2004, ‘Determiner agreement and noun conjunction’. *Journal of Linguistics* **40**, 69–104.
- Kokkonidis, M.: 2005, ‘Why Glue a Donkey to an F-structure when you can Constrain and Bind it Instead’. In: M. Butt and T. King (eds.): *Proceedings of LFG 2005*. Stanford, CA: CSLI Publications.
- Kokkonidis, M.: 2008, ‘First Order Glue’. *Journal of Logic, Language and Information* **17**, 43–68.
- Kroeger, P.: 1993, *Phrase-Structure and Grammatical Relations in Tagalog*. Stanford, CA: CSLI Publications. originally Stanford University Phd dissertation, 1991.
- Kuhn, J.: 2003, *Optimality-Theoretic Syntax—A Declarative Approach*. Stanford CA: CSLI Publications.
- Lamarche, F.: 1994, ‘Proof Nets for Intuitionistic Linear Logic 1: Essential Nets’. Technical Report, Imperial College London.
- Lambek, J.: 2008, *From Word to Sentence*. Milan: Polimetrica.
- Lev, I.: 2007, ‘Packed Computation of Exact Meaning Representations using Glue Semantics (with automatic handling of structural ambiguities and advanced natural language constructions)’. Ph.D. thesis, Stanford University. Retrieved November 15, 2010, from <http://sites.google.com/site/pulcproject/material>.
- Mackie, I., L. Román, and S. Abramsky: 1993, ‘An Internal Language for Autonomous Categories’. *Applied Categorical Structures* **1**, 311–343.
- Manning, C. D.: 1996, *Ergativity: Argument Structure and Grammatical Relations*. Stanford CA: CSLI Publications. originally Stanford PhD dissertation, 1994.

- Marantz, A.: 1984, *On the Nature of Grammatical Relations*. Cambridge MA: MIT Press.
- Moot, R.: 2002, 'Proof-Nets for Linguistic Analysis'. Ph.D. thesis, University of Utecht. Retrieved November 15, 2010, from <http://www.labri.fr/perso/moot/> and <http://igitur-archive.library.uu.nl/dissertations/1980438/full.pdf>.
- Morrill, G.: 2005, 'Geometry of Language and Linguistic Circuitry'. In: C. Casadio, P. J. Scott, and R. Seely (eds.): *Language and Grammar*. Stanford, CA: CSLI Publications, pp. 237–264.
- Nordlinger, R.: 1998, *Constructive Case*. Stanford CA: CSLI Publications.
- Nordlinger, R. and L. Sadler: 2004, 'Tense beyond the verb: encoding clausal tense/aspect/mood on nominal dependents'. *Natural Language and Linguistic Theory* **22**, 597–641.
- Partee, B. H.: 2006, 'Do We Need Two Basic Types'. In: H.-M. Gaertner, R. Eckardt, R. Musan, and B. Stiebels (eds.): *Puzzles for Manfred Krifka*. Berlin.
- Perrier, G.: 1999, 'Labelled Proof-nets for the Syntax and Semantics of Natural Languages'. *L.G. of the IGPL* **7**, 629–655.
- Prior, A. N.: 1963, *Formal Logic*. Oxford: Clarendon.
- Simpson, J. and J. Bresnan: 1983, 'Control and Obviation in Warlpiri'. *Natural Language and Linguistic Theory* **1**, 49–64.
- Simpson, J. H.: 1983, 'Aspects of Warlpiri Morphology and Syntax'. Ph.D. thesis, Massachusetts Institute of Technology. Mostly superseded by Simpson (1991).
- Simpson, J. H.: 1991, *Warlpiri Morpho-Syntax*. Kluwer Academic.
- Tatsuta, M.: 1993, 'Uniqueness of Normal Proofs of Minimal Formulas'. *Journal of Symbolic Logic* **58**, 789–799.
- Troelstra, A. S.: 1992, *Lectures on Linear Logic*. Stanford CA: CSLI Publications.
- Wechsler, S.: 2004, 'Number as Person'. In: O. Bonami and P. C. Hofherr (eds.): *Empirical Issues in Syntax and Semantics 5 (on-line Proceedings of the Fifth Syntax And Semantics Conference In Paris)*. pp. 255–274. Retrieved November 15, 2010, from <http://www.cssp.cnrs.fr/eiss5/> and <http://uts.cc.utexas.edu/~wechsler/NumberAsPerson.pdf>.
- Wechsler, S. and L. Zlatić: 2003, *The Many Faces of Agreement*. Stanford, CA: CSLI Publications.