

Propositional Glue and the Correspondence Architecture of LFG*

Avery D. Andrews
v3: ANU, Sept 2008

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 ϕ 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 emphasized ‘glue semantics’, the most developed approach to semantic composition in LFG,¹ although their existence is occasionally exploited (e.g. Crouch and van Genabith (1999), Asudeh and Crouch (2002)).

In this paper I will discuss and 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), thereby simplifying the framework.

1. The Role of (Linear) 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

* I would like to acknowledge Miltiadis Kokkonidis for helpful discussion of various aspects of glue proof terms.

¹ It is however worth noting that glue semantics is no longer used in the PAR-GRAM project grammars, having been replaced by f-structure rewriting (Crouch (2006), Crouch and King (2006)). But f-structure rewriting has so far been presented 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.

with the lexical entries of (1), which include meaning-constructors, this sentence gets the f-structure (2):²

- (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)_e \multimap (\uparrow SUBJ)_e \multimap \uparrow_p
- (2)
$$f: \begin{bmatrix} \text{SUBJ} & g: [\text{PRED} \text{ ‘Bert’}] \\ \text{PRED} & \text{‘Likes}<(\uparrow \text{SUBJ})(\uparrow \text{OBJ})> \\ \text{OBJ} & h: [\text{PRED} \text{ ‘Ernie’}] \end{bmatrix}$$

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) *Bert* : h_e
 Ernie : h_e
 Like : $h_e \multimap g_e \multimap f_p$

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

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

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,

² 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.

³ 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.

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 labels, which are treated as individuals. Then the quantification that we will shortly be looking at becomes 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 label. 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:

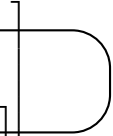
$$(5) \lambda Px. \text{Every}(z, \text{Person}(z), P(z)) : \forall H((e(\uparrow) \multimap p(H)) \multimap p(H))$$

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:

$$(6) \text{Everybody seems to sleep}$$

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):

$$(7) \text{ a. } \begin{array}{l} \left[\begin{array}{ll} \text{SUBJ} & g: [\text{PRED} \text{ 'Everybody'}] \\ \text{PRED} & \text{'Seem} < h > ' \\ \text{XCOMP} & h: \left[\begin{array}{ll} \text{SUBJ} & g: [\quad] \\ \text{PRED} & \text{'Sleep} < g > ' \end{array} \right] \end{array} \right] \end{array}$$


- 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 represent 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):

$$(8) \text{ Everybody} : (e(\uparrow) \multimap p(\%H)) \multimap p(\%H)$$

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 structure 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):

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

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 more 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 how significant is it?

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'.⁴ 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 pre-existing mathematics.

A consequence that might be of more concrete utility to linguistics is that once glue is made propositional, it becomes quite straightforward to formulate it as a fairly conventional form of linguistic structure-building, that integrates nicely into LFG. However we must also make sure that this somewhat subtle formal change doesn't produce any bad empirical consequences. 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 basis for a mathematical account of entailment and other meaning-based properties and relations). Universal Instantiation creates an issue for 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).

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 LFG and general linguistic practice, by borrowing some ideas from the proof-net literature. Proof-nets are frequently

⁴ 'Fusion' is a name for the multiplicative version of conjunction often employed in Relevant Logic.

used in 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 are closely connected to one of the formulations of an essential constraint on proof-nets, the 'Correctness Criterion'.

2.1. PROOF-NETS

In a proof-net, the premises and conclusion⁵ 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, to be discussed shortly. Glue-proofs are 'intuitionistic', which means that they derive a single conclusion, with 'positive' polarity, from a set of premises, which have 'negative' polarity.⁶ Furthermore, the only connectives that appear to be needed for glue are the implication (\multimap) and perhaps the tensor (\otimes). Without the tensor, the structure of the glue-formulas is essentially the same as that of Montogovian semantic types representing function-application and lambda-abstraction, supplemented by polarity. For example, the glue formula $h_e \multimap g_e \multimap f_p$ corresponds to the semantic type $\langle e, e, p \rangle$. Tensors represent pairing, as discussed later.

Glue-proofs are subject to the further restriction that the conclusion be of semantic type p (at least if we're parsing a declarative sentence), and have the f-structure label of the entire f-structure. So the setting for glue assembly is a collection of formulas of negative polarity, each of which corresponds to a semantic type, but also specifies grammatical information by means of the f-structure labels, plus one formula f_p of positive polarity, where f is the label of the f-structure of the whole sentence.

The proof-net rules depend on polarity being propagated to the literals of the negative polarity formulas, which happens in accord with 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.

⁵ Or conclusions, for non-Intuitionistic proof-nets.

⁶ See e.g. Moot (2002) or de Groote (1999) for details.

- 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.

The instantiated constructors for the sentence *everybody doesn't sleep* might be:

$$(11) \begin{array}{l} \text{Everybody} : (g_e \multimap f_p) \multimap f_p \\ \text{Not} : f_p \multimap f_p \\ \text{Sleep} : g_e \multimap f_p \end{array}$$

Writing the syntax trees for the glue-sides in logicians' standard root-down format, with polarity, and meaning-side at the bottom, a standard proof-net format for these constructors will then be:

$$(12) \quad \begin{array}{ccc} \begin{array}{c} g_e^- \quad f_p^+ \\ \diagdown \quad \diagup \\ g_e \multimap f_p^+ \quad f_p^- \\ \diagdown \quad \diagup \\ (g_e \multimap f_p) \multimap f_p^- \\ \text{Everybody} \end{array} & \begin{array}{c} f_p^+ \quad f_p^- \\ \diagdown \quad \diagup \\ f_p \multimap f_p^- \\ \text{Not} \end{array} & \begin{array}{c} g_e^+ \quad f_p^- \\ \diagdown \quad \diagup \\ g_e \multimap f_p^- \\ \text{Sleep} \end{array} \end{array}$$

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'⁷ 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

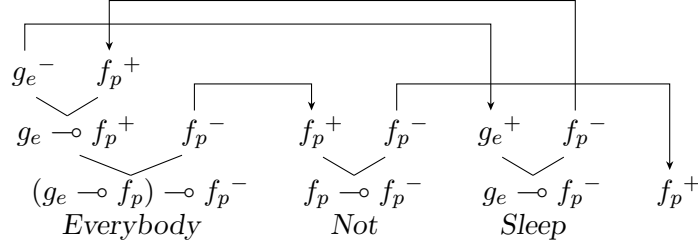
⁷ So-called because they represent instances of the identity axiom in the one-sided sequent calculus.

d. subject to the Correctness Criterion (to come)

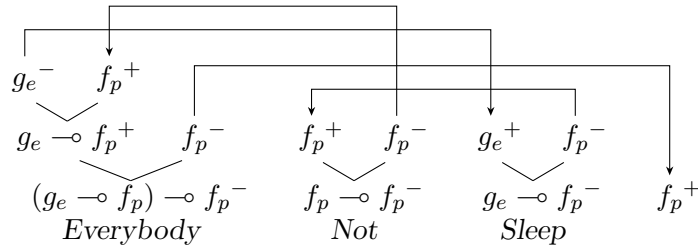
It is standard to regard these links as undirected, but it is intuitively helpful to regard them as directed from negative to positive.⁸

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

(14) a.



b.



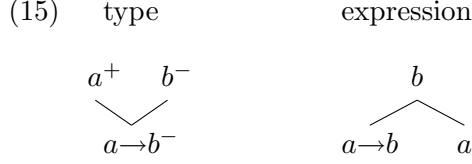
There's also a third, which violates the Correctness Criterion, and so represents neither a proof nor a sensible reading.

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

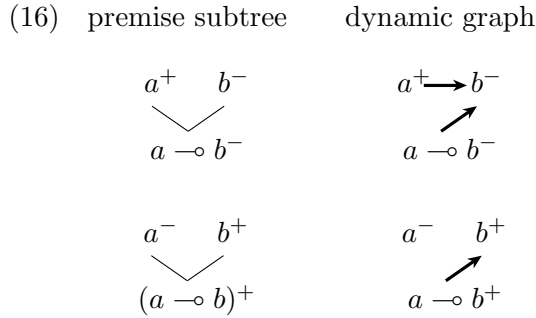
⁸ And in more sophisticated environments, there can be real reasons for doing this (Hughes, 2005).

and the associated expression-tree for an implication/function can be represented as follows, where the expression-tree on the right is root-up, following the customs of linguistics:



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 almost identical to a concept of proof-net theory, the 'dynamic graph' of de Groote (1999), a minor variant of the 'essential net' of Lamarche (1994).⁹ 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 negative (first row) and positive (second row) polarity implications in the usual form used in proof-nets, and on the right, the corresponding dynamic graph links:



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.

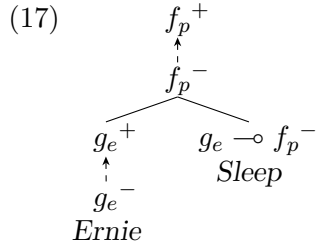
Looking at the dynamic graph for a negative implication, it is evident that it is just like an expression tree, with the dynamic graph arrows going from the daughters to the mother. In the case of the positive implication, which represents a lambda-abstraction, things are a bit different: the arrow represents a link from the formula-component of the lambda-abstraction to the whole, with, in the dynamic graph, no

⁹ But the deGroote paper is much more accessible; indeed I have made very little progress in comprehending the contents of the Lamarche paper.

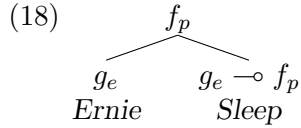
representation of the link to the variable, which is represented by the consequent of the (positive) implication. To represent this, we'll need to retain the link from the implication to its antecedent in our derived representation; this will be represented below as a dotted arrow, sometimes curved. It will sometimes be called a '(left) pseudo-daughter', because of its absence from the dynamic tree.

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. We will write them root-up, due to their similarity to standard linguistics tree-structures.

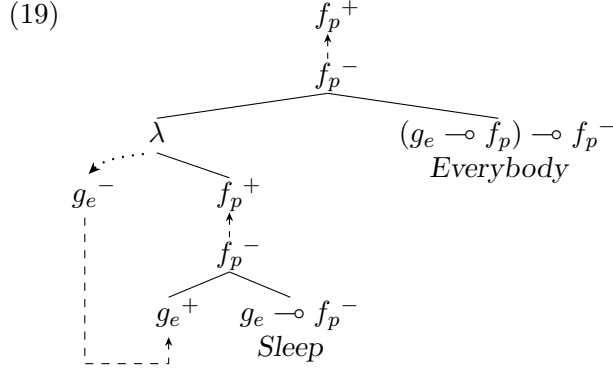
A pleasant consequence of the prefab format is that the negative nodes can often be put more or less under the positives they are to be plugged into, creating a fairly ordinary-looking tree-structure. The assembly for *Ernie sleeps* would for example be (17), where, for layout reasons, we've put arguments first and functions second:



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:

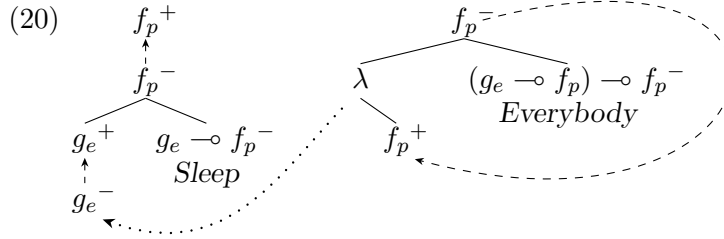


The topography is a bit more complex for quantifiers. Here the left pseudo-daughter is negative, so it needs to plug into a positive. A reasonable arrangement for *Everybody sleeps* would be:



This provides an intuitively obvious but mathematically disciplined graphical representation of linear lambda-binding.

It's now time to formulate the Correctness Criterion. It is needed, because in (19), 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:¹⁰



The problematic axiom-link here is the dashed line from the negative polarity f_p^- of *Everybody* to its positive one.

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 (20) 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. It is:

- (21) **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.

¹⁰ The possibility of unexpected ways of plugging inputs into outputs appears to be a frequent problem with intuitively-based systems of meaning-assembly, such as for example that of Jackendoff (2002).

(19) clearly satisfies (21), while (20) 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 implications; 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.¹¹

We now have a propositional glue system for the core fragment of Dalrymple et al. (1999) which supports a system of meaning-assembly based on representations of a form that is reasonably familiar to linguists, and able to be assembled by fairly simple rules. We now reconsider the nature of the relationship to f-structure.

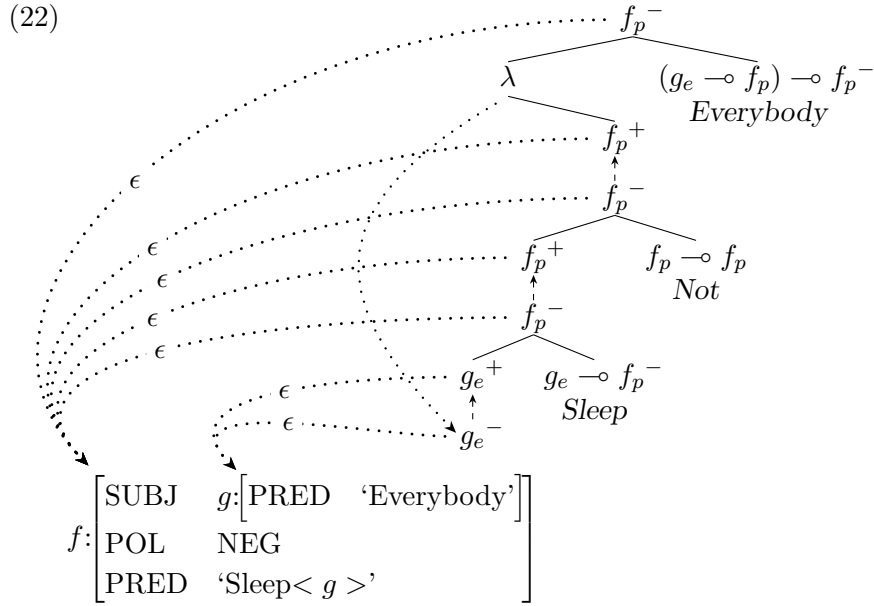
2.2. REVERSING σ

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 reversed 'semantic projection' going from the glue-structures to the f-structures rather than from f-structure towards glue, as is the case with the σ found in the LFG literature (both glue and earlier, at least back to Kaplan (1987)).

The trouble with σ is that, due to quantifiers and 'head switching' constructions such as sentence-adverbials, a projection from f-structures to genuinely semantic objects can't be a function (the standard σ -projection of glue stops short of the glue-assemblies). But it seems possible to have a close-to-inverse of σ going in the opposite way, which is a function that we will call ϵ (for 'expresses').

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

¹¹ (21) isn't identical to what de Groote proves, but it is obvious that a proof-structure (an axiom-linking satisfying all of the conditions except for Correctness) satisfying these conditions will satisfy his algebraic criterion, and he proves that any proof-structure satisfying this criterion will also satisfy (21).



All of the different glue-nodes labelled f_p (three, if we contract 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 ϵ -correspondence.

ϵ isn't quite the inverse of standard σ , because we've eliminated the 'semantic projection' that σ targets, as discussed later. ϵ can however be regarded as implicit in the construction of the formulas in glue-proofs from f-structure labels and semantic type symbols, modulo a bit of confusion caused by universal quantification.

A useful consequence of being explicit about ϵ manifests itself 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 ϵ -value. The ϵ -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 mathematically 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 an interesting range of cases, but there are

a number of more complex situations, which need to be investigated. 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 a potential argument for the semantic projection, which we have omitted from this architecture. 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, with the result that antecedents of positive implications (left pseudo-daughters) have always been 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 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 (some brackets square to clarify grouping):

$$(23) \text{ Seek} : [((\uparrow \text{OBJ})_e \multimap (\uparrow \text{OBJ})_p) \multimap (\uparrow \text{OBJ})_p] \multimap (\uparrow \text{SUBJ})_e \multimap \uparrow_p$$

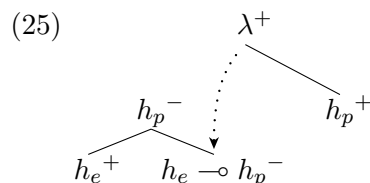
Note in particular, that unlike in previous versions of this constructor, there is no linear quantifier or corresponding functional uncertainty, because this is unnecessary and produces unwanted multiple analyses.

Given (23), the constructors for a quantified NP such as *a unicorn* will then be able to instantiate their local name $\%H$ 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.

It finally might be useful 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:

$$(24) \quad \begin{array}{c} \lambda^+ \\ \swarrow \quad \searrow \\ h_e \multimap h_p^- \quad h_p^+ \end{array}$$

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



Which fits together properly with the other structures (although the layout is awkward, since a dynamic path must run from the negative h_p^- literal to the positive one).¹² 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^+ . Observe also that with intensional verb-arguments, the structures cease to be trees, although the dynamic paths they provide constitute sets of trees.

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 (Deal, 2007). If it can be eliminated, then the meaning-constructors required by current analyses that I am aware of 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.

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.¹³

¹² Putting the predicates first would help here, or else using an unordered tree representation, which would be feasible, since the relative ordering of daughters is redundant if we know the formula labels and polarities).

¹³ 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.

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 (2007)), and has some resemblance to analyses that have been proposed in Type-Logical Grammar (Jäger, 2005)), so propositional glue needs 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 into a composite which can do the same job as the original contributors, but not necessarily in such a way that either can be discarded (if both discard and copying are possible, the tensor is also a 'product'). These ideas are embodied in the natural deduction rules for tensor introduction and elimination:

$$(26) \quad \frac{A \quad B}{A \otimes B} \otimes \mathcal{I} \qquad \frac{A \otimes B \quad \begin{array}{c} [A]^i [B]^j \\ \vdots \\ C \end{array}}{C} \otimes \mathcal{E}^{i,j}$$

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:

$$(27) \quad \frac{A \wedge B}{A} \wedge \mathcal{E} \qquad \frac{A \wedge B}{B} \wedge \mathcal{E}$$

(However, if the (tree-style natural deduction equivalents of the) 'structural rules' of Weakening and Contraction are available, then (26) and (27) have the same effects.)

The rules of (26), in conjunction with 'commuting conversions' for identifying proofs, turn out to characterize 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.¹⁴

Returning to bound pronouns, the first point is that they have antecedents, which are 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

¹⁴ People intrigued by such things might enjoy looking at Baez and Stay (2008). Benton et al. (1992) provides another useful discussion focussed on linear logic.

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):

$$(28) \quad \lambda x.[x, x] : (\uparrow \text{ANT})_e \multimap (\uparrow \text{ANT})_e \otimes \uparrow_e$$

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:

$$(29) \quad \frac{\frac{\lambda x.[x, x] : g_e \multimap g_e \otimes h_e \quad E : g_e}{\lambda x.[x, x](E) : g_e \otimes h_e} \quad \frac{\frac{W : h_e \multimap g_e \multimap f_p \quad [y : h_e]^i}{W(y) : g_e \multimap f_p} \quad [z : g_e]^j}{\frac{W(y)(z) : f_p}{W(\pi^1(\lambda x.[x, x](E))(\pi^2(\lambda x.[x, x](E))) : f_p} \otimes \mathcal{E}^{i,j}}$$

The right side of this proof represents the fact that we can apply the transitive verb $W(\text{ash})$ to two arbitrary things in object and subject position to get something of type f_p , while the left side 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 β -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 *Ref*.

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

$$(30) \quad \frac{[x : A]^i [y : B]^j \quad \vdots \quad f : C}{z : A \otimes B \quad \frac{f : C}{[\pi^1(z)/x, \pi^2(z)/y]f : C} \otimes \mathcal{E}^{i,j}}$$

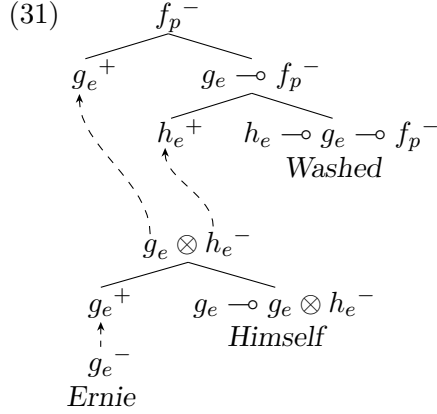
This says that if you can make f using x and y , and z is a tensor, and the types (formulas) are appropriate, then you can use z instead of x and y individually to make a variant of f with $\pi^1(z)$ (the left projection of z) substituted for x , and $\pi^2(z)$ (the right projection of z) substituted for y in f .

The glue literature on the other hand instead follows most of the linear logic literature in using the proof-term **let** z **be** $x \times y$ **in** f . This appears to differ from the projection-based term in a number of respects. One is that it discriminates proofs that the projection-based terms identify. For example, for a sentence such as *Bert doesn't wash himself*, we get two proofs which will have different terms using **let**, but the same with projections, due to the substitutions being able to insert things underneath the negative operators. But this is not actually a significant difference, since the distinct **let**-based proof terms wind up being equated by the usual commuting conversions (Mackie et al., 1993) used to group proofs into equivalence classes in Proof Theory.

Another apparent difference is that the **let**-based terms support a simple definition of linear lambda-term, wherein there is exactly one occurrence of the variable bound by a lambda within its scope. Orthographically, this doesn't work for the projection based terms, due to the copies appearing under π^1 and π^2 . Presumably this could be addressed with a more complicated definition, but I'm not aware of anybody having worked through the details (perhaps something could be based on the left and right square roots used by de Groote (1999) in his algebraic correctness criterion).

A third difference, which might be the reason that the Type Logical Grammarians use the projections, is that they map smoothly onto proof-nets, which the **let**-based terms do not. We can see this by looking at the proof-net for a negative sentence, but it will save a bit of exposition to do it in the prefab format from the outset, so we need to know how to represent tensors that way.

The underlying fact is that in a negative tensor, the dynamic graph links go from the tensor to its components. Therefore, the prefab presentation of (29) will look like this (the tensor should really appear as the root of small, root-down tree whose with its components as leaves, but it isn't worth the trouble of typesetting that):



Now it should be clear that if we embed the main predication of (31) under a negative or any other kind of operator (that doesn't introduce any other 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 (31) is a natural representation of the proof-term of (29), and vice-versa. One can regard the projection notation as a linearization of proof-nets involving negative tensors, where π^1 gives the 'view' of the tensor from its left component, π^2 from its right. This can be easily used to extend Perrier's (1999) maximal labelling 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 (21) requires that all of these forks be recombined before the path goes through the positive consequent of that implication.

Tensors then don't afford any serious problems to propositional glue and its reversed direction replacement ϵ for σ . They do however inject more complexity into some of the formulations, for example the 'real proof' objects cease to be natural deduction proof-trees, but rather equivalence classes of such trees under commuting conversions, and a more complex formulation of the Correctness Criterion becomes necessary (de Groote, 1999). Therefore, proposals to eliminate them from meaning-constructors should certainly be taken seriously.

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:

$$(32) \quad \begin{aligned} CN &: \uparrow_e \multimap \uparrow_p \\ QD &: (\uparrow_e \multimap \uparrow_p) \multimap (\uparrow_e \multimap \%H_p) \multimap \%H_p \end{aligned}$$

But having so many different semantic transactions going on at the f-structure designated by \uparrow is perhaps somewhat unsettling.

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 (32) 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', which is linked to f-structure by the (standard, from f-structure) σ -correspondence.

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):

$$(33) \quad \begin{aligned} CN &: \uparrow_\sigma \multimap \uparrow_\sigma \\ QD &: ((\uparrow_\sigma \text{ VAR}) \multimap (\uparrow_\sigma \text{ RESTR})) \multimap (\uparrow_\sigma \multimap \%H_\sigma) \multimap \%H_\sigma \end{aligned}$$

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

$$(34) \quad \begin{bmatrix} \text{SPEC} & \text{Every} \\ \text{PRED} & \text{'Muppet'} \end{bmatrix} \cdots \sigma \cdots \rightarrow \begin{bmatrix} \text{VAR} & [\] \\ \text{RESTR} & [\] \end{bmatrix}$$

But in spite of the intuitive motivation, a real 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 (Asudeh 2004, 2008).

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:

(35) 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):

- (36) a. Varje förtsbo som Maria vet att hann
 Every suburbanite that Maria knows that he
 arresterades försvann
 was-arrested vanished
 Every suburbanite who Maria knows that he was arrested
 vanished
- b. *Varje förtsbo som Maria vet att en granne
 Every suburbanite that Maria knows that a neighbor
 arresterades försvann
 was-arrested vanished
 Every suburbanite who Mary 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 our present analysis allows us to evade it. What we can do is formulate the meaning-constructor for a relational noun so 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*:

$$\begin{aligned}
 (37) \text{ Neighbor} &: \%H_e \multimap \uparrow_e \\
 \lambda x.[x, x] &: (\%H \text{ ANT})_e \multimap (\%H \text{ ANT})_e \otimes \%H_e \\
 (\%H \text{ ANT}) &= (\text{GF}^* \uparrow)
 \end{aligned}$$

%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:

$$(38) \lambda P.\lambda x.x : (A_e \multimap A_e \otimes P_e) \multimap A_e \multimap A_e$$

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, the version of Asudeh (2005b) would come out as:

$$(39) ((\uparrow \text{GF}^+ \text{ ANT})_e \multimap (\uparrow \text{GF}^+ \text{ ANT})_e \otimes (\uparrow \text{GF}^+)_e) \multimap (\uparrow \text{GF}^+ \text{ ANT})_e \multimap (\uparrow \text{GF}^+ \text{ ANT})_e$$

Recall that this is one of the meaning-constructors for a relative-clause construction, and that in successful derivations, 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, namely, that it doesn’t lie at (the semantic projection of) the end of a 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 correspond to a given f-(sub)structure. An interesting possibility is afforded by the ‘INDEX’ and ‘CONCORD’ attributes 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), Hahm and Wechsler (2007)).

Since 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:

$$(40) \quad \begin{aligned} CN &: (\uparrow \text{CONCORD})_e \multimap \uparrow_p \\ QD &: ((\uparrow \text{CONCORD})_e \multimap \uparrow_p) \multimap ((\uparrow \text{INDEX})_e \multimap \%H_p) \multimap \%H_p \end{aligned}$$

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:

$$(41) \quad \begin{aligned} \textit{Everyone} &: (g^1 \multimap g^1) \multimap g^2 \\ \textit{Sleep} &: g^2 \multimap f \end{aligned}$$

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 association of the positive antecedent to an INDEX-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.¹⁵

¹⁵ 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.

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. One useful effect is that we can use the structure of the glue-sides of meaning-constructors to do the work of traditional hierarchical argument-structures, as discussed in Andrews (2007b), therefore eliminating this as an autonomous level.

A further effect is to increase the resemblance of LFG 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’, we 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 hybrid¹⁶ of techniques from Chomsky, Lambek and traditional grammar.

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 instantiate their f-structure variables, which is described above 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 (2007b)) for this to be sustainable.

¹⁶ Technically, I suppose, a chimaera.

This might be counted as a point against the approach, but, on the other hand, LFG, like other theories used primarily by syntacticians, 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’. (Icelandic, Bantu, Australian, Austronesian, etc.). ECG seems to have made little if any attempt to deal this kind of material.

So if the debate about direct compositionality is to be conducted on an empirical basis, there need to be comparable studies in frameworks assuming full direct compositionality of the kinds of problems considered in LFG, for which glue is meant to provide an account of semantic composition.

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 α -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. 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 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, similar to what is assumed in the MP, corresponding to plausible ideas about the order of semantic composition (perhaps better described as ‘tightness of structural bonding’), as motivated for example by Marantz (1984). 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 big descriptive topics that

LFG has paid relatively little attention to. But anything that increases comparability of analyses across frameworks has some chance of being helpful.

References

- 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. URL: <http://arts.anu.edu.au/linguistics/People/AveryAndrews/Papers>.
- 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. URL: <http://cslipublications.stanford.edu/LFG/12/lfg07.html>.
- Andrews, A. D. and C. D. Manning: 1999, *Complex Predicates and Information Spreading in LFG*. Stanford, CA: CSLI Publications.
- Asudeh, A.: 2004, ‘Resumption as Resource Management’. Ph.D. thesis, Stanford University, Stanford CA. <http://http-server.carleton.ca/~asudeh/> (viewed June 2008).
- 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.: 2008, ‘Towards a Unified Theory of Resumption’. URL: <http://http-server.carleton.ca/>, <http://semanticsarchive.net/Archive/jNjZWRkM/asudeh08-rp.pdf>.
- Asudeh, A. and R. Crouch: 2002, ‘Coordination and Parallelism in Glue Semantics: Integrating Discourse Cohesion and the Element Constraint’. In: *Proceedings of the LFG02 Conference*. Stanford, CA, pp. 19–39, CSLI Publications. URL: <http://csli-publications.stanford.edu>.
- Asudeh, A. and I. Toivonen: 2007, ‘Copy-Raising and Perception’. submitted draft; URL: <http://http-server.carleton.ca/~asudeh/>.
- Baez, J. and M. Stay: 2008, ‘Physics, Topology, Logic and Computation: A Rosetta Stone’. URL: <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. URL: <http://citeseer.ist.psu.edu/article/benton92term.html>.
- 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.
- 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. URL: <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.'. *TCS* pp. 115–134. URL: <http://www.loria.fr/~degroote/bibliography.html> (viewed June 2008).
- 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'. URL: <http://people.umass.edu/amyrose/>.
- 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 University Press. URL: <http://www.monad.me.uk/stable/Proofs+Types.html> (viewed June 2008).
- 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. URL: <http://cslipublications.stanford.edu/LFG/12/lfg07.html>.
- Hughes, D.: 2005, 'Simple Multiplicative Proof Nets with Units'. <http://arxiv.org/abs/math.CT/0507003>.
- Jackendoff, R. S.: 2002, *Foundations of Language*. 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.
- 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. First distributed 2006; URL: <http://citeseer.ist.psu.edu/kokkonidis06firstorder.html>.
- 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.
- 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. URL: http://www.geocities.com/iddolev/pulc/current_work.html (viewed June 2008).
- Mackie, I., L. Román, and S. Abramsky: 1993, 'An Internal Language for Autonomous Categories'. *Applied Categorical Structures* **1**, 311–343.
- 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. URL: <http://www.labri.fr/perso/moot/> (viewed June 2008), also <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.
- 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. URL: www.zas.gwz-berlin.de/40-60-puzzles-for-krifka/.
- Perrier, G.: 1999, 'Labelled Proof-nets for the Syntax and Semantics of Natural Languages'. *L.G. of the IGPL* **7**, 629–655. URL: <http://www.loria.fr/~perrier/papers.html>.
- Prior, A. N.: 1963, *Formal Logic*. Oxford: Clarendon.
- 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. URL: <http://www.cssp.cnrs.fr/eiss5/> (viewed June 2008) and <http://uts.cc.utexas.edu/~wechsler/NumberAsPerson.pdf> (viewed June 2008).
- Wechsler, S. and L. Zlatić: 2003, *The Many Faces of Agreement*. Stanford, CA: CSLI Publications.