Input and Glue in OT-LFG[*]

Avery D Andrews, ANU

draft, May 2005

In glue logic (Dalrymple 1999, 2001, Asudeh 2004), LFG has found a reasonably stable but developing method for connecting syntactic structures to formal semantic intepretation. However current formulations of glue logic make the rather unsatisfactory move of divorcing grammatical features from any intrinsic connection with what are usually regarded as their meanings. Rather than directly connecting a grammatical feature attribute-value combination such as [NUM PL] to a meaning-constructor that produces a plurality meaning, current glue simply co-introduces a grammatical feature and any associated meaning constructors in the entries of all lexical items introducing the feature. Not only does this lose any account of important traditional observations about how morphology and semantics are related, it also creates an incompatibility with current OT-LFG, which depends on the idea of f-structures having intrinsic semantic content (Kuhn, 2003, 64-65,74). The lack of a means to connect OT-LFG to a formal semantics leaves it incomplete not only in comparison with its contemporary competitors, such as Categorial Grammar, HPSG, and conventional LFG, but even in its own terms, in that there are significant constraints which can't be formulated without explicit semantics, most noticeably scope-based ones such as Bresnan's (2001b) FAITH$^{\mathrm{NEG}}$, or the 'respect-the-tree' constraints that one would plausibly use to account for scoping modifier and light verb constructions such as those discussed by Andrews (1983) and Alsina (1997).

In this paper I will develop a proposal to add glue-logic based semantic interpretation to OT-LFG, and show how it supports formulations of scope-based constraints, as well as a better account of the relationship between grammatical features and meanings. The basic idea is that an input will consist of two components, an f-description delimiting a range of potential candidate feature-structures, and an assembled collection of meaning-constructors determining a meaning. The f-description determines the candidate-set in the same way that an underspecified f-structure does in conventional OT-LFG,[*] while the meaning-constructors specify the semantic content that is

[*]This is a prepublication draft correcting some typoes detected by a reviewer and myself, but not containing substantive corrections.

[*]There is no substantive consequence to using descriptions rather than structures, although they keep the formalism simpler and closer to the description-based tendency in

associated with the structure.

In conventional glue, a lexical item has a single lexical entry which specifies its morphological, c-structural, f-structure and semantic (glue) information all at once. For example, assuming a simple operator semantics for the past tense, the entry for a verb form such as *went* might be:

(1) *went*:V, PRED ‘Go$_{motion}$’, TENSE PAST, $\lambda x.go(x) : (\uparrow\text{SUBJ})_e \multimap \uparrow_t$,
$\lambda P.Past(P) : \uparrow_t \multimap \uparrow_t$

Here both the TENSE and PRED-features are stipulatively co-introduced with appropriate meaning-constructors.

Such a treatment can’t be used in OT-LFG, because we need to associate a feature-structure with a meaning independently of its morphological realization (at least if we want to apply the ideas of OT to any aspects of the latter). In order to produce an input that specifies a feature-structure and a meaning, but not a morphological realization, it is necessary to split the lexicon into two components, a ‘Semantic Lexicon’ connecting features to meaning, and a ‘Morphological Lexicon’ connecting features to pronunciations and other morphologically relevant information, such as inflectional diacritics (conjugation and declension classes). The Semantic Lexicon will then be used to generate the inputs, and the Morphological Lexicon to produce the alternative realizational candidates. In addition to allowing OT-LFG to have a formal semantics at all, the splitting of the lexicon will be seen to have some desirable properties in accounting for certain properties of the interface between semantics, syntax and morphology.

In the first section below, I will provide some general motivation for the split lexicon, and in the second, I will make a specific proposal for using it to generate inputs for OT-LFG. In the third section I will show how the proposal supports the formulation of scope-based constraints such as Bresnan’s FAITH$^{\text{NEG}}$ and other ‘tree-respecting’ constraints.

1 The Split Lexicon

LFG and virtually all other grammatical theories have a reasonable account of the one-to-many paradigmatic relations between features and their morphological spellings (alternations, usually conditioned but sometimes free),

LFG formalization.

as well as many-to-one syntagmatic relations (fusional/portmanteau realizations). But LFG in its various forms has tended not to provide a clean and explicit account of the similar properties of the features-to-meaning relationship.

Consider for example the morphological and semantic properties of the verb *go*. We find what appears to be a single morphological unit, with conditioned alternate spellings /gow/, /wɛnt/, /gɔ/, but quite a wide range of meanings, partially dependent on what other morphological units *go* appears together with. Here some uses of the past tense form /wɛnt/ are illustrated:[*]

(2) a. Mary went to the store

b. The milk went off

c. The alarm went off

d. Susan went off at the children about the state of the kitchen

e. Mike went 'what on earth is going on here??'

f. Joe went nuts

g. Fred underwent an unpleasant operation

The shared morphological irregularity is evidence that all of these forms involve some sort of common feature, but this clearly cannot be a meaning as such, since the meanings are quite different, ranging from somewhat related to apparently completely distinct.

In standard LFG, the meanings are coded into different PRED-features. The morphological relationships are not represented in any theoretically explicit way, although perhaps alluded to by using subscripted variants of 'Go' as the PRED-values, such as 'Go$_{say}$' for the *go* of (e). Lacking an explicit formal semantics, OT-LFG would appear to be stuck with the classic treatment, but with standard LFG plus glue-logic, things are potentially better. Most of the functions of PRED-features can be taken over by the meaning-constructors (Completeness, Coherence, perhaps even Predicate Uniqueness (Dalrymple (2001, pg. 220), Kuhn 2001), which leaves the PRED-feature

[*]A rather puzzling gap in my dialect at least is *forego*, which appears to have the nonfinite past *foregone*, but note the finite **forewent* (which does however appear in the dictionary).

with no clear function other than to interface with the morphology.[*] If the PRED-feature is no longer tied to the meaning, it becomes possible for all of the uses of *go* to introduce the same PRED-feature. The morphology can then be arranged to spell this out in different ways in different situtations, such as /wɛnt/ when in combination with a [TENSE PAST] feature.

However LFG+glue has a difficulty with another aspect of the feature-to-meaning interface, due to its failure to directly express generalizations relating the presence of features to the presence of meanings. Since features and meaning-constructors are stipulatively co-introduced in the lexical entries of words or formatives, it is an accident that there are any general correlations between the presence of features and the presence of meanings, independently of the morphological expressions of the features. Of course the association between features and meanings is not a simple one-to-one or even one-to-many relationship, as witnessed by phenomena such as *pluralia tanta* (non-compositional interpretation of lexical stems and the plural feature), the rise of 'preterite present' verbs (non-compositional interpretation of past tense verb forms as present time reference), and idiomatic/quirky case-marking. Nonetheless, the association of features with meanings is quite pervasive, and any theory which does not express it is severely deficient.

To address this issue, we need to do something to associate meaning-constructors in some direct way with the features that they tend to cooccur with. In a conventional LFG architecture, this could be accomplished with a fairly low level splitting of the lexcion whereby the meaning-constructors would be attached at a late stage in the construction of lexical entries, on the basis of what features those entries contained. But this won't work for OT-LFG, where we need to attach meanings to structures which lack information about morphological realization. So we propose a deeper split, where the Semantic Lexicon is used to produce interpreted feature-structures, and the Morphological Lexicion to produce morphologically spelled-out realizations.

The Morphological Lexicon presents no novelties; it looks like a conventional LFG-lexicon, except for the removal of subcategorization information from the PRED-values. The Semantic Lexicon will however need some kind of new formal device, in order to connect features to meaning constructors. Since OT-LFG wants an underspecified feature-structure to constrain the possible morphosyntactic realizations of the meaning of an utterance, a rea-

[*]Therefore it might be better to replace PRED-features with FORM-features, although we won't do this here.

sonable thing to try for would be a device that co-generated f-descriptions (which can do the work of underspecified f-structures) with sets of meaning-constructors. Using a ‘ $\Leftrightarrow$ ’ connective to pair f-structural and semantic information with the former on the left and the latter on the right, we can formulate some sample Semantic Lexical Entries (SLEs) as follows, where $\mathcal{G}$ is a convenient abbreviation for a functional uncertainty expression ranging over the grammatical functions that may be universally used to express arguments:

(3) a. $(f\,\mathrm{TENSE}) = \mathrm{PAST} \;\Leftrightarrow \lambda P.Past(P) : f_t \multimap f_t$

b. $(f\,\mathrm{PRED}) = \text{‘Go’},\ (f\,\mathcal{G}) = g \;\Leftrightarrow \lambda x.Go(x) : g_e \multimap f_t$

In the next section we will see why only f-structure variables, not GF-application terms, appear on the glue-side of the meaning constructors.

Although we assume f-descriptions as the underlying formalization of the f-structural component of an SLE, it is sometimes more readable to present it as a structure; when we do this, we'll use the standard GF_i notation for underspecified term grammatical functions:

(4) a. $f{:}\begin{bmatrix}\mathrm{TENSE} & \mathrm{PAST}\end{bmatrix} \;\Leftrightarrow \lambda P.Past(P) : f_t \multimap f_t$

b. $f{:}\begin{bmatrix}\mathrm{PRED} & \text{‘Go’} \\ \mathrm{GF}_1 & g{:}\begin{bmatrix}\ \end{bmatrix}\end{bmatrix} \;\Leftrightarrow \lambda x.Go(x) : g_e \multimap f_t$

But this should be understood as an informal version of a description-based formulation using functional uncertainty.

SLEs in the format of 3 clearly express facts about the connections between features and meanings. In the next section we will present a mechanism for using them to produce inputs which are in effect underspecified f-structures correlated with meanings; in the remainder of this section I will discuss how they accomodate two further aspects of the relationships between features and meanings.

One important complexity of this relationship is that it is possible for a feature to have multiple meanings; this is rather obviously the case for the PRED-features of lexical items such as *go*, and also occurs with grammatical features, such as for example semantic cases, which typically express similar but significantly different ranges of meanings in different languages. This can be straightforwardly accomodated by having multiple SLEs associating different meanings with the same feature. For example, alongside of SLE (3b) for the motional interpretation 'Go' feature, we can have 5 for the quotative:

(5) $(f\,\mathrm{PRED}) = \text{`Go'},\ (f\,\mathcal{G}) =\ g,\ (f\,\mathcal{G}) =\ h\ \Leftrightarrow \lambda y.\lambda x.Say(x,y) : h_t \multimap g_e \multimap f_t$

The second complexity I'll consider is non-compositional interpretation, whereby several different features are interpreted together way to give an 'idiomatic' meaning. Conventional LFG manages non-compositional interpretation in a somewhat clunky but workable way with FORM-features and constraining equations (Kaplan and Bresnan, 1982, 213-214), but I am aware of no proposals for it in standard OT-LFG. In our present approach, we can accomodate it by referring to multiple features in the SLE. For example, following Toivonen's (2001) treatment of verb-particle constructions, we can propose an entry such as this for one of the noncompositional interpretations of *go off*:

(6) $$f{:}\begin{bmatrix} \mathrm{PRED} & \text{`Go'} \\ \mathrm{GF}_1 & g{:}\begin{bmatrix}\ \end{bmatrix} \\ \mathrm{XCOMP} & \begin{bmatrix}\mathrm{PRED} & \text{`Off'}\end{bmatrix} \end{bmatrix}$$

$$\Leftrightarrow \lambda x.become\_unpalatable(x) : g_e \multimap f_t$$

This mechanism seems appropriate for what Egan (2004) describes as the 'CHUNKS' style of noncompositional interpretation, in which an arrangement of co-occurring overt lexical items seem to constitute a 'chunk' whose meaning must simply be listed. If the chunks are defined over f-structure, a certain amount of syntactic manipulation can also be accomodated, but not some of the more complex phenomena discussed in the literature, such as by Nunberg et al. (1994). However SLEs may well be sufficient to account for idioms whose interpretation is accomplished by strictly grammatical means.

An intriguing feature of the split lexicon is that it produces a certain amount of symmetry between the workings of spellout of features in overt morphology, and their semantic interpretation as meanings. Corresponding to the phenomenon of variant morphological realization of the same feature, we have multiple meanings for the same feature, and corresponding to the phenomenon of portmanteau realization of multiple features as a single morphological form, we have noncompositional interpretation of multiple features as a single semantic unit. But there are major differences, most importantly, we'll see in the next section that 'semantic spellout' is resource-sensitive, while morphological spellout isn't. Freely variant realization furthermore seems to be rather less common than its interpretational counterpart, multiple meanings.

2 Generating and Using the Input

We now describe a process for generating the inputs. The basic idea is to take a multiset of SLEs, replace their variables, and see if the resulting f-description is satisfiable and meaning-constructors assembleable. If so, then the pair $<F, M>$ is an input, where F is the f-description and M the assembled set of meaning.constructors. For an illustration of how this is supposed to work, suppose we add to the Semantic Lexicon an entry for the proper name *Marvin*:

(7) $(f\,\text{PRED}) = \text{`Marvin'} \;\Leftrightarrow\; \mathit{Marvin} : f_e$

If we leave the variables of the SLEs in (3) untouched and replace f with g in 7, we get the following f-description and meaning-constructor set:

(8) a. $(f\,\text{TENSE}) = \text{PAST}$
$(f\,\text{PRED}) = \text{`Go'}$
$(f\,\mathcal{G}) = g$
$(g\,\text{PRED}) = \text{`Marvin'}$

b. $$\begin{array}{rcl} \lambda P.\mathit{Past}(P) & : & f_t \multimap f_t \\ \lambda x.\mathit{Go}(x) & : & g_e \multimap f_t \\ \mathit{Marvin} & : & g_e \end{array}$$

The f-description is satisfiable,[*] and the meaning-constructors are assembleable, so we have an input that is roughly equivalent to a conventional OT-LFG input such as:

(9) $$\begin{bmatrix} \text{GF}_1 & \begin{bmatrix} \text{PRED} & \text{`Marvin'} \end{bmatrix} \\ \text{TENSE} & \text{PAST} \\ \text{PRED} & \text{`Go'} \end{bmatrix}$$

[*]Satisfiability will be decideable as long as the SLEs don't introduce any inside-out functional uncertainty (Backofen, 1994).

but with the significant improvement that it comes with a precisely specified formal semantic interpretation $\mathit{Past}(\mathit{go}(\mathit{Marvin}))$, provided by the assembled meaning-constructors.

We can now see why we want only f-structure variables, not functional application terms, to appear in the meaning-constructors, as mentioned in the discussion of (3). The effect of having only variables, and not functional application terms, in the glue terms of the meaning-constructors is that a subsititution of variables for variables determines a range of possibilities for semantic assembly, as illustrated for example in (8b). But if the glue term for *go* were for example $(f\,\mathrm{SUBJ})_e \multimap f_t$, then some other technique would be needed to control assembly.

There are at least two constraints we will want to impose on this procedure. The first is that substitutions should not merge variables in a single meaning-constructor. For example we wouldn't want to replace both f and g in (3b) with f. To allow this would be to generate large numbers of inputs with f-structure fusions between predicates and arguments, and even distinct arguments. The former at least arguably occurs, for example with complex predicates, but should clearly be regulated in the candidates by the constraint-ranking, which won't be possible if the fusions are being produced in the input, rather than the input-candidate relationships.

The second constraint is that the f-descriptions must be able to be satisifed without merging atomic values. Suppose there were no such constraint. Then we could select the SLE (3a) for the past tense feature any number of times, stacking up the tense operator with no limit. It might be possible to avoid specific instances of this kind of problem with clever formulations of the associated meaning-constructors (perhaps the t type could be split into tensed and tenseless events, with *Past* mapping the latter onto the former), but we clearly want to prevent this for any feature without needing to engage in delicate engineering with the type system, which a general prohibition of atomic value merger achieves.

There are various ways in which the constraint could be implemented; an approach with properties that will be useful later is to assign to each meaning-constructor a unique index when it is selected from the lexicon, and mark this index on each of the atomic feature-values introduced in the f-descriptional component of the SLE-instance. The index will then keep features associated with different constructor instances from merging, thereby implementing the constraint. We will see later that the indexes and the relations they define will also be useful for stating constraints connecting constituent structure to semantic relationships.

A more general perspective on the non-merger constraint is that it is a kind of resource sensitivity (Asudeh, 2004, 87-99): under semantic interpretation, a feature can only be used once, and all features must be used. This is a major difference between the semantic interpretation of features and their morphological spellout, since the latter is clearly not resource-limited: one feature in the syntactic structure can be spelled out many times, or not at all.[*] This second form of resource-sensitivity in the grammar is inherently simpler than the one involved in semantic assembly, since it doesn't involve any assembly of the feature resources, merely their conversion into meaning-constructors.

But leaving this issue for now, we can state the proposed definition of how an arbitrary input is generated from the Semantic Lexicon as follows:

(10)
1. Select a multiset of Semantic Lexical Entries from the Semantic Lexicon.
2. Assign to each entry instance (in the multiset) a unique index, and attach that index to each atomic feature value in the f-descriptional component of the entry.
3. Rename the f-structure variables in each entry instance, subject to the constraint that distinct variables within an entry don't merge (of course, merger of variables across different entry instances is essential).
4. Check that the resulting f-description F is satisfiable.
5. Try to find a way to assemble the meaning-constructors to produce a meaning M.
6. If 4 and 5, succeed, return the resulting pair $<F, M>$ as an input.

Although this is a technique for defining a set, rather than a practical computational procecudure, it should be clear that it occasions no extraordinary computational difficulties for dealing with the set it defines. The two tasks we

[*]One might think that the use of glue-logic for semantic interpretation already captures the resource-based nature of the intepretation of features, but this isn't really the case in current LFG, since lexical entries merely stipulate the co-introduction of meaning-constructors together with feature values.

might wish to undertake are comprehension and generation. For comprehension, a method similar to the restriction-based analysis-by-description technique of Wedekind and Kaplan (1993) could be used, whereby features would be removed from a provisional copy of the structure as meaning-constructors were added. Production is a much deeper matter; it is in general quite unclear how we come up with coherent semantic representations which fulfill whatever purposes we might be attempting to achieve by producing language. But it should be obvious that formulating the problem as that of picking a set of SLEs from the Semantic Lexicon that can be assembled, and assembling them, adds no gratuitous difficulties.

Although 10 will suffice as a basic input-generation scheme, there is at least one elaboration that would be worth considering. Dalrymple (2001, pp. 265-269) shows that in order to capture some results originally due to Kasper (1995), 'intersective' and 'gradeable' adjectives will have to have two meaning-constructors, one specifying the distinctive meaning of the adjective, the other combining this intersectively with the meaning of the rest of the nominal that the adjective appears in. Having most adjectival lexical entries contain copies of the same 'intersective' meaning-constructor is rather awkward, and can be avoided by the suggestion of Asudeh and Crouch (2002) and Asudeh (2004) that certain meaning-constructors might be introduced by the phrase-structure rules. This possibility is not available in OT-LFG, but two alternatives are.

The first, and formally least problematic, is to suppose that there are constructors that introduce features that are frequently not spelled out in the morphology of languages. For example the constructor 10 below for intersective and gradeable adjectives would be (Dalrymple, 2001, pg. 266):

(11) $$g \in (f\,\mathrm{ADJUNCT}) \Leftrightarrow \lambda P.\lambda Q.\lambda x.P(x) \wedge Q(x) : \\ ((g\,\mathrm{VAR})_e \multimap g_t) \multimap ((f\,\mathrm{VAR})_e \multimap (f\,\mathrm{RESTR})_t) \multimap (f\,\mathrm{VAR})_e \multimap (f\,\mathrm{RESTR})_t$$

This could be co-introduced with a feature such as MOD +, which would be unexpressed in English, but perhaps spelled out as the *ezafet* of Farsi and similar languages.

An alternative would be to have a small list of universally available meaning-constructors that aren't associated with atomic feature-values, but can be freely added to the multiset of SLEs selected by step 1 of (10). The nature of these freely available 'syncategorematic' constructors will have to be constrained in order to avoid computational difficulty; a reasonable proposal would be that they must combine at least two inputs into an output, so that the number of them that can be used in an assembly will be limited by the number of ordinary ordinary SLE-instances chosen, which do introduce features. The choice between these alternatives is a subsidiary issue which we don't need to settle here.

Now that we have our inputs, the next step is to use them in OT-LFG. The main thing we have to do is modify the Restricted definition of **Gen** (Kuhn, 2003, pg. 74), where Φ_{in} is the underspecified input f-structure:

(12) *Restricted definition of **Gen***:
$Gen_{G_{inviol}}(\Phi_{in}) = \{<T, \Phi'> \in L(G_{inviol}) \mid \Phi_{in} \sqsubseteq \Phi'$, where Φ' contains no more semantic information than $\Phi_{in}\}$

Obvously, we need to replace the reference to subsumption with satisfaction, since the input is now an f-description rather than an (underspecified) f-structure (although we will continue to represent it as Φ_{in}, due to the minimal substantive difference between f-descriptions and f-structures). A more obscure question is what to do about the requirement that the candidate not contain more semantic information than the input.

The problem is that it is not clear what it would mean for the candidate to contain additional semantic information: the semantic content of something is dependent on how it is used and reacted to by the agents that encounter it, as much as by its intrinsic properties. If the candidate is taken as a representation of what is to be performed, as opposed to how that is to be understood, how could it have additional semantic content?

I think this can be seen as an unclarity flowing from the absence of an explicit account of semantics in standard OT-LFG. Kuhn acknowledges 'some leeway' in the definition of semantic information, but suggests that it can be specified by 'declaring particular features or feature-combinations as contributors to semantic information' (2003, pp. 74-75). One possible response would be to simply drop the requirement. This would allow the candidates to freely overparse the input by adding additional features. The difficulty with this is that OT-LFG already includes a facility for 'lexical overparsing' (Bresnan 2001b, Kuhn 2003, pp. 106-108), whereby not all the specifications in a (in our framework, morphological) lexical entry need to be used to build the candidate f-structure. Perhaps the lexical overparsing mechanism could be retired in favor of a structural one, but to achieve maximum compatibility with previous work we will for the present retain the former, which requires prohibiting the latter in order to avoid theoretical redundancy.

One way to do this would be to recognize a set of semantically interpretable feature-values as opposed to uninterpretable ones (mainly, if not exclusively, structural case), and require that the candidate not contain any interpretable features not required by the input. This would appear to be Kuhn's suggestion. A problem with it is that syntactic cases are frequently fully homophonous with semantic cases, indicating that they involve the same feature, which must therefore be sometimes semantically interpretable, and sometimes not. An easy solution to this would be to have a list of features which can fail to be semantically interpreted, and allow overparsing of the input only for features that appear on this list. This leads to the following Revised restricted definition of **Gen**:

(13) *Revised restricted definition of **Gen**:*
$Gen_{G_{inviol}}(\Phi_{in}) = \{<T, \Phi'> \,>\in L(G_{inviol}) \mid \Phi'$ satisfies Φ_{in}, and all features in Φ' that aren't required to satisfy Φ_{in} are on the list of features that don't require a semantic interpretation$\}$

Given 13, the rest of OT-LFG, including both the generation and evaluation of candidates, can now procede as conceived by Bresnan and formulated by Kuhn, except for the difference that we how have an explict level of compositional semantics that can be used for the formulation of constraints. Before developing this further, I'll make one final remark about the proposed form of the inputs, which is their lack of a 'semantic projection' as standardly assumed in LFG-with-glue (Dalrymple 2001, Asudeh 2004). As discussed in Andrews (2004), the semantic projection does not appear to have any clear function, so it is omitted here. Many earlier proposals for semantic interpretation in LFG made essential use of a semantic projection, such as those of Haloversen (1983) and Wedekind and Kaplan (1993), but in glue-logic, the meanings are contained within the meaning-constructors, and all the syntax has to provide is locations where they can be connected to each other. F-structures can do this job as well as objects on a semantic projection, so the latter should be eliminated, unless some substantial arguments for it are produced.

3 Scope-based Constraints

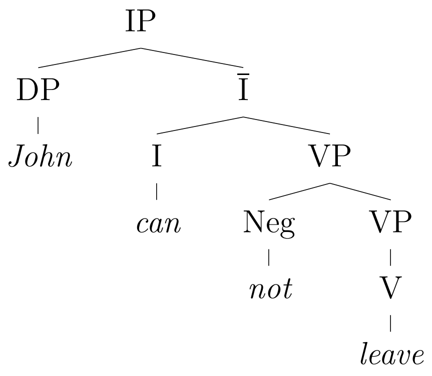
An example of a scope-based constraint in current OT-LFG literature is Bresnan's (2001b) FAITH$^{\mathrm{NEG}}$, which imposes restrictions on the ordering of negative markers and other material in the clause. For example the form *cannot* can have the *not* taking scope over the negative, while the separated *can not* requires the auxiliary to outscope the negative:

(14) a. John cannot leave

b. John can not leave

Bresnan proposes that the difference between (a) and (b) in (14) is the result of a structural difference, whereby in (a), the Neg is adjoined to the auxiliary's I, while for (b) it is to be adjoined to VP, yielding a structure such as this:

(15)



Inflectional and I-adjoined negatives appear to take scope with respect to the auxiliary in a manner determined by the lexical choice of the auxiliary; for *must not* and *mustn't* are interpreted as NEC NEG, while *need not* and *needn't* are NEG NEC (see de Haan (1997) for literature survey and discussion).

The VP-adjoined negative seems to scope consistently under the auxiliary in affirmative clauses; that a violable constraint is involved is indicated by the fact that a Q NEG POSS reading becomes possible in the stilted question form where the interrogative appears in C but the negative is left behind:[*]

(16) Can we not help them?

Another example is provided by the difference between:

(17) a. John often hasn't paid taxes

b. John hasn't often paid taxes

[*]This sentence-form can be obtained in the registers where it occurs, by penalizing inflected negatives in C (or perhaps everywhere).

(a) might be true and (b) false, if John's taxable income is 0 half the time on average.

Other potential applications would include the 'tree-respecting' constraints governing complex predicates (Alsina, 1997) and 'scoping modifiers' (Andrews, 1983), and the iconic ordering of markers in 'case-stacking' (Dench and Evans, 1988). Here, scope-sensitive constraints might be able to replace complex alternative mechanisms that have been proposed to account for the data, such as multiple projections for complex predicates and scoping modifiers (Andrews and Manning 1999, Andrews 2004).

To develop any such constraints, we need to establish some connections between scope in the semantic representation, and the arrangements of formatives in the syntactic tree. I believe that this can be most perspicuoulsy done (for beginners at any rate) by using a proof-net based formulation of glue logic semantic assembly (Fry 1999, Andrews 2004), rather than the more commonly encountered deductive formulation.[*] On a proof-net-based presentation, assembly is achieved not by combining meaning-constructors with conventional deduction rules, but by connecting their literals with links in patterns that correspond exactly to the structure of proofs, and can furthermore be used to calculate the desired output semantic form, as worked out by Perrier (1999) building on de Groote (1999), and applied to LFG by Andrews (2004). The presentational advantage is that meaning-constructors retain their visual integrity in the course of semantic assembly, so that semantically-based relationships can be easily correlated with morphosyntactic relationships.

In the proof-net format of Andrews (2004),[*] the two readings of (14) would be represented as the following patterns of linkage:

(18) a. *not* outscopes *can*:

[*]However, it is important to keep in mind the proof nets and deductions are formally equivalent, for glue, so the choice between them is strictly a matter of perceived convenience rather than empirical or mathematical substance, and, of course, tastes and preferences may differ.

[*]Which is considerably streamlined from previous notations.

$$\begin{tikzcd}[row sep=small, column sep=small]
\lambda S.\mathit{not}(S) & : & f_t & \multimap & f_t \\
\lambda S.\mathit{can}(S) & : & f_t & \multimap & f_t \arrow[ull, bend right] \\
\lambda x.\mathit{leave}(x) & : & g_e & \multimap & f_t \arrow[ull, bend right] \\
\mathit{John} & : & g_e \arrow[u] & &
\end{tikzcd}$$

b. *can* outscopes *not*:

$$\begin{tikzcd}[row sep=small, column sep=small]
\lambda S.\mathit{not}(S) & : & f_t & \multimap & f_t \arrow[dll, bend left] \\
\lambda S.\mathit{can}(S) & : & f_t & \multimap & f_t \\
\lambda x.\mathit{leave}(x) & : & g_e & \multimap & f_t \arrow[uull, bend right=90] \\
\mathit{John} & : & g_e \arrow[u] & &
\end{tikzcd}$$

There is a straightforward definition of relative scope in terms of hookup patterns; to develop it we need to say a bit more about the theory behind the hookups.

One ingredient of the account is the idea that the literals on the glue-side of the meaning-constructors (the f-structure designators, subscripted by their semantic type) have positive and negative polarity: the links flow from positive to negative, and always connect literals with the same f-structure location and semantic type (so they can be metaphorically understood as pipes through which content of that type flows). The relevant polarity rules are as follows:[*]

(19) a. The polarity of an entire glue-side is positive

b. The polarity of the antecedent of a positive implication is negative

c. The polarity of the consequent of a positive implication is positive

You can see that the links in (18) are properly oriented with respect to these polarity conventions and the positive-to-negative flow idea. For a full treatment, there need to be additional polarity rules for negative implications,

[*]Unfortunately, de Groote (1999) and Perrier (1999) adopt the opposite polarity convention; here I've followed some LFG work that uses polarity.

and for tensors if these are present, but we don't need these complexities here.[*]

So what we note about relative scope is that the lower scoped element has an output that feeds into the higher one. Complex configurations involving quantifiers and certain control verbs have rather complicated hookups with inputs and outputs being traded between pairs of constructors, but we can formulate a definition that copes with these complexities by means of a notion of 'Final Output', which amounts to the last positive polarity to be assigned by 19:

(20) a. The Final Output of an implication is the Final Output of its consequent.

b. The Final Output of a literal is the output of that literal.

We can now define 'Outscopes' recursively, with the aid of the concept 'Feeds Into':

(21) a. A meaning-constructor B Feeds Into a meaning-constructor A iff the final output of B is an input to A.

b. A meaning-constructor A Outscopes a meaning-constructor B if there is a sequence of meaning-constructors $C_1, \ldots, C_n$ such that $C_1 = A$, $C_n = B$, and for $1 \leq i < n - 1$, C_i Feeds Into C_{i+1}.

The conditions on well-formed hookups will guarantee that Outscopes is an asymmetric relation (Lemma 4.6, de Groote 1999). This Outscopes relation will apply in the expected way to quantifiers, as well as negatives and similar $t \multimap t$ operators, and would also be relevant to formulating constraints to capturing the 'respect of the tree' and iconic ordering of stacked cases phenomena mentioned above.

The next thing we need is some way to connect c-structure nodes to meaning-constructors, so that the c-structural relations between the former can be linked to the Outscopes relations between the latter. This can be done with the aid of the indexes that were originally introduced to impose resource-sensitivity on feature-interpretaton. We can provide that when a candidate is checked for satisfying the input f-description, the feature-values

[*]For negative implications, see Andrews (2004), and for a brief discussion of tensors, Andrews (2003).

in the candidate f-structure acquire the indexes from their correspondents in the input. The next step is to connect the features to c-structure nodes, in particular the preterminals where the feature are spelled out. In conventional LFG, there isn't really a formalized relationship between features and preterminals (although some principles, such as the Morphological Blocking Principle of Andrews (1990), do make at least implicit use of such a relation), but in OT-LFG there is, in the form of the 'λ-projection' introduced by Bresnan (2001a) to support OT-based accounts of morphosyntactic competition. The λ-projection assigns to each preterminal in the c-structure a mini-f-structure assembled on the basis of the f-description of the word introduced under this preterminal(Kuhn, 2003, 111-113). Therefore we can transmit indexes from the features of the full candidate f-structure to the matching ones in the mini-f-structures of the λ-projection, and thence, finally, to the preterminals themselves.

So now we have a many-to-many relationship between c-structure nodes and meaning-constructors, which can be used in various ways to formulate constraints. The approach I will take is to project c-structure relationships between preterminals back onto meaning-constructors, reminiscently of how the c-structure precedence relationships between c-structure nodes are projected back onto f-structures to give the 'f-precedence' relationship (Kaplan, 1995, 21). The c-structure relationship I will use is the one called 'm-command': a node A m-commands node B iff every maximal category dominating A also dominates B. Note that the relation is formulated in such as way that a negative or negatives cliticized or otherwise adjoined to an auxiliary will not m-command it, and will therefore not be required to take scope over it by a constraint formulated in terms of m-command. We can project m-command back to the meaning-constructors with the following definition:

(22) Meaning-constructor M m-commands meaning-constructor N iff every preterminal linked to M m-commands every preterminal linked to N.

We can use this in a preliminary formulation of FAITH$^{\text{NEG}}$:

(23) If a negative meaning-constructor M Outscopes a meaning-constructor N, it must m-command N.

Various issues arise in refining this formulation.

Perhaps the most important one is how to count multiple violations of the constraint (Kuhn, 2003, 89-90). Kuhn proposes that constraints contain a special metavariable, $\star$, such that one violation is accrued for each instantiation of the metavariable that falsifies the constraint (whereas there is no counting of the number of ways in which the so-instantiated constraint is violated). For a constraint such as 23, which regulates the relationships between two elements, I think the most plausible treatment would be to have two metavariables, one for each of the elements. As before, one violation mark will be accrued for each instantiation for which the constraint is not satisfied, but the instantiations will specify pairs of elements rather than singletons. Since the meaning-constructors are the unique elements being related, it seems best to take them to be the things represented by the metavariables, so we reformulate 23 as 24:

(24) If $\star_1$ is a sentential negative meaning-constructor (one with glue term $f_t \multimap f_t$, which introduces the negation operator), and $\star_2$ a meaning-constructor, and $\star_1$ Outscopes $\star_2$, then $\star_1$ m-commands $\star_2$.

24 is formulated in terms of specifically negative constraints due to some puzzling data whereby adverbs can reorder freely around positive auxiliaries but not negative ones:

(25) a. John probably can file his return in June

b. John can probably file his return in June

c. John certainly must file his return by June 15

d. John must certainly file his return by June 15

If the auxiliary is negated, the scoping becomes rigid, with post-auxiliary orderings become unacceptable:[*]

(26) a. John probably can't file his return in June

b. *John can't probably file his return in June

c. John certainly mustn't file his return after June 15

[*]See Nilsen (2002) for discussion of a semantic constraint that would rule out (b, d) above, as long as something will fix the scope of the adverb below the negative.

d. *John mustn't certainly file his return after June 15

One puzzling aspect of this data is what the c-structures should be. The post-modal orderings can be treated as adjoined to the following VP (Sells, 2001, 61), but the restrictive and well-motivated phrase-structure theory of Toivonen (2001) doesn't seem to provide an easy answer to the pre-modal ordering, since (a) adjunction to $\bar{\mathrm{I}}$ is not permitted by the theory (b) adjunction to I is not an option either, since the adverbs can be phrasal: *John almost certainly can('t) file in June.* So it remains to be seen how adverb-modal ordering facts will integrate with other scoping effects in the formulation of constraints related to (24). Nonetheless, it seems plausible that constraints connecting Outscopes to m-command would be effective in controlling the c-structure arrangements of scoping modifiers and light-verbs that appear in VP-nests, as argued by Alsina (1997) and (Manning 1992, Manning 1996).

4 Conclusion

We have shown how f-descriptions can be co-generated with sets of meaning-constructors, which can then be assembled, to provide OT-LFG with a form of input that has an explicitly specified semantic interpretation rather than merely an intuitively interpreted feature-structure. Such a proposal is particularly important for constraints related to scope, which is not well represented by LFG f-structures, due to their characteristically flat organization. But there are other kinds of meaning-related constraints that the proposal should be able to accomodate, such as for example Linking Theory constraints. It remains to be seen whether these can be accomodated without further modifications of the approach.

References

Alsina, Alex. 1997. A theory of complex predicates: Evidence from causatives in Bantu and Romance. In A. Alsina, J. Bresnan, and P. Sells, eds., *Complex Predicates*, pages 203–246.

Alsina, Alex, Joan Bresnan, and Peter Sells, eds. 1997. *Complex Predicates*. Stanford CA: Center for the Study of Language and Information.

Andrews, Avery D. 1983. A note on the constituent structure of modifiers. *Linguistic Inquiry* pages 695–7.

Andrews, Avery D. 1990. Unification and morphological blocking. *Natural Language and Linguistic Theory* pages 507–557.

Andrews, Avery D. 2003. Glue logic, projections, and modifiers. URL: `http://arts.anu.edu.au/linguistics/People/AveryAndrews/Papers`.

Andrews, Avery D. 2004. Glue logic vs. spreading architecture in LFG. In C. Mostovsky, ed., *Proceedings of the 2003 Conference of the Australian Linguistics Society*. URL: `http://www.newcastle.edu.au/school/lang-media/news/als2003/proceedings.html`.

Andrews, Avery D. and Christopher D. Manning. 1999. *Complex Predicates and Information Spreading in LFG*. Stanford, CA: CSLI Publications.

Asudeh, Ash. 2004. *Resumption as Resource Management*. Ph.D. thesis, Stanford University, Stanford CA. URL: `http://www.stanford.edu/~asudeh`.

Asudeh, Ash and Richard Crouch. 2002. Coordination and parallelism in glue semantics: Integrating discourse cohesion and the element constraint. In *Proceedings of the LFG02 Conference*, pages 19–39. CSLI Publications. `http://csli-publications.stanford.edu`.

Backofen, Rolf. 1994. Regular path expressions in feature logic. *Journal of Symbolic Computation* pages 421–455.

Bresnan, Joan. 2001a. Explaining morphosyntactic competition. In M. Baltin and C. Collins, eds., *In Handbook of Contemporary Syntactic Theory*, pages 11–44. Oxford.

Bresnan, Joan Wanda, ed. 1982. *The Mental Representation of Grammatical Relations*. Cambridge MA: MIT Press.

Bresnan, Joan Wanda. 2001b. *Lexical-Functional Syntax*. Blackwell.

Dalrymple, Mary, ed. 1999. *Syntax and Semantics in Lexical Functional Grammar: The Resource-Logic Approach*. MIT Press.

Dalrymple, Mary. 2001. *Lexical Functional Grammar*. Academic Press.

Dalrymple, Mary, Ronald M. Kaplan, John T. Maxwell, and Annie Zaenen, eds. 1995. *Formal Issues in Lexical-Functional Grammar*. Stanford CA: The Center for the Study of Language and Information.

de Groote, Philippe. 1999. An algebraic correctness criterion for intuitionistic multiplicative proof-nets. *TCS* pages 115–134. URL: `http://www.loria.fr/~degroote/bibliography.html`.

de Haan, Ferdinand. 1997. *The Interaction of Modality and Negation*. New York & London: Garland.

Dench, Alan and Nicholas Evans. 1988. Multiple case-marking in Australian languages. *Australian Journal of Linguistics* pages 1–47.

Egan, Andy. 2004. Pretense for the complete idiom. under review; URL: `http://www.geocities.com/eganamit/papers.html`.

Fry, John. 1999. Proof nets and negative polarity licensing. In M. Dalrymple, ed., *Syntax and Semantics in Lexical Functional Grammar: The Resource-Logic Approach*, pages 91–116.

Haloversen, Per-Kristian. 1983. Semantics for Lexical-Functional Grammar. *Linguistic Inquiry* pages 567–615.

Kaplan, R.M. and Joan 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; page number references to 1982 version.

Kaplan, Ronald 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*, pages 7–27. CSLI Publications.

Kasper, R. 1995. Semantics of recursive modification. URL: `ftp://ling.ohio-state.edu/pub/HPSG/Workshop.Tue.85/Kasper/`. (handout for HPSG Workshop Tuebingen, June 1995).

Kuhn, Jonas. 2001. Resource sensitivity in the syntax-semantics interface and the german split np construction. In W. D. Meurers and T. Kiss, eds., *Constraint-Based Approaches to Germanic Syntax*. Stanford CA: CSLI Publications.

Kuhn, Jonas. 2003. *Optimality-Theoretic Syntax–A Declarative Approach*. Stanford CA: CSLI Publications.

Manning, Christopher David. 1992. Romance is so complex. Tech. Rep. CSLI-92-168, Stanford University, Stanford CA. URL: `http://nlp.stanford.edu/~manning/papers/romance.ps`.

Manning, Christopher D. 1996. Romance complex predicates: In defence of the right-branching structure. paper presented at the Workshop on Surfaced-Base Syntax and Romance Languages, 1996 European Summer School on Logic, Language and Information, Prague. Draft available at URL: `http://nlp.stanford.edu/~manning/papers/`.

Nilsen, Oystein. 2002. Domains for adverbs. Paper submitted to *Lingua* for special edition on adverbs, edited Artemis Alexiadou. URL: `http://www.let.uu.nl/\\7Eoystein.nilsen/personal/domains.pdf`.

Nunberg, Jeff, Ivan Sag, and Thomas Wasow. 1994. Idiom. *Language* pages 491–538.

Perrier, Guy. 1999. Labelled proof-nets for the syntax and semantics of natural languages. *L.G. of the IGPL* pages 629–655. URL: `http://www.loria.fr/~perrier/papers.html`.

Sells, Peter. 2001. *Structure, Alignment and Optimality in Swedish*. Stanford CA: CSLI Publications.

Toivonen, Ida. 2001. *The Syntax of Non-Projecting Words*. Ph.D. thesis, Stanford University, Stanford CA.

Wedekind, Jürgen and Ronald M. Kaplan. 1993. Type-driven semantic interpretation of f-structures. In *Proceedings of the 6th Conference of the European Chapter of the Association for Computational Linguistics (EACL-1993)*, pages 404–411.