A Resource Sensitive Interpretation of Lexical Functional Grammar

Mark Johnson[*]
Brown University

(Received ..... ; Accepted in final form .....)

Abstract. This paper investigates whether the fundamental linguistic insights and intuitions of Lexical Functional Grammar (LFG), which is usually presented as a "constraint-based" linguistic theory, can be reformulated in a "resource sensitive" framework using a substructural modal logic. In the approach investigated here, LFG's f-descriptions are replaced with expressions from a multi-modal propositional logic (with permutation and possibly limited contraction). In effect, the feature structure "unification" basis of LFG's f-structures is replaced with a very different resource based mechanism. It turns out that some linguistic analyses that required non-monotonic devices in LFG (such as the "constraint equations" in the Andrews (1982) analysis of Icelandic) can be straightforwardly expressed in the framework presented here. Moreover, a Curry-Howard correspondence between proofs in this logic and λ-terms provides a semantic interpretation as a by-product of the process of showing syntactic well-formedness.

Key words: Lexical Functional Grammar, feature-structures, "unification" grammar

1. Introduction

This paper describes a re-interpretation of Lexical Functional Grammar (Kaplan and Bresnan, 1982) in resource sensitive terms. Lexical Functional Grammar (LFG) is a kind of "unification-based" or "constraint-based" theory of grammar (Shieber, 1986). Constraint-based theories of grammar specify the well-formed linguistic structures in terms of a system of constraints that these structures must satisfy.

Recently a different conceptualization of grammar has begun to be explored in which resource sensitivity, rather than constraint satisfaction, is the central organizing theme. To date, these ideas have been primarily explored within categorial grammar; see Carpenter (1996) and Morrill (1994) for examples of this work. This paper explores the role that they might play in the area of feature structures, and shows that some of the central linguistic insights of LFG can be expressed in

[*] This paper was written during 1995–96 while I was on sabbatical leave from Brown University at the Rank Xerox Research Laboratory (RXRC) in Grenoble. I would like to thank my colleagues at RXRC, Xerox Parc and in the categorial grammar/ linear logic community for providing me with such a stimulating environment, and the thoughtful JoLLI reviewers for their helpful comments. I would especially like to thank Ron Kaplan and Dick Oehrle for their helpful suggestions.

this kind of way as well. As explained below, the resulting system is architecturally simpler than the standard LFG model augmented with a linear logic interpretation mechanism, and so may be preferred on grounds of parsimony.

The point of this paper is not that the standard feature structure interpretations of LFG provided by Kaplan and Bresnan (1982) are wrong, but that other, quite different, formal systems can also express the linguistic intuitions of Lexical Functional grammarians. This alternative interpretation of the linguistic theory highlights different aspects of LFG linguistic analyses than the feature structure interpretation does. For example, the resource sensitive treatment of agreement features permits a straightforward treatment of linguistic phenomena such as Icelandic quirky case marking, which seems to require a substantial (non-monotonic) extension to standard feature structure systems.

One of the attractions of resource sensitive approaches to grammar is the simple and elegant treatment of the syntax-semantics interface they permit. The Curry-Howard relationship between the basic "plugging" operations of a resource sensitive system and the structure of expressions in the lambda calculus permits a proof of syntactic well-formedness to determine (together with the lexicon) the semantic interpretation of the utterance.

In a series of papers, Mary Dalrymple and colleagues propose to extend LFG with a Linear Logic component in order to provide semantic interpretation. This extension provides LFG with a variety of resource sensitive devices, which they exploit to provide a semantic interpretation for LFG analyses. Once such resource sensitive devices are introduced into a grammatical framework it is natural to ask if they can be applied to other linguistic tasks as well. The primary goal of this paper is to investigate the extent to which such resource sensitive devices can capture phenomena described by feature structure constraints in LFG.

This paper shows that many of the tasks filled by LFG's f-structure constraints can be re-expressed in a resource sensitive framework, and suggests that a much simpler architecture may be linguistically adequate. The revised theory is given a new name—Resource Sensitive Lexical Functional Grammar or R-LFG—to distinguish it from the theory proposed by Kaplan and Bresnan (1982). While no reduction of LFGs to R-LFGs is offered here, it does seem that many LFG analyses can be re-expressed in R-LFG in a simple, linguistically adequate manner.

Most attempts at accounting for the non-monotonic extensions of LFG, such as existential constraints and constraint equations, have proceeded by grafting these extensions on top of a monotonic feature structure architecture. For example, Johnson (1995) uses circumscription to restrict attention to the appropriate minimal models in which such constraints should hold. This paper can be viewed as espousing a more radical approach in which the non-monotonic constraints, expressed as resource dependencies, are taken to be primary. Thus this paper differs from other attempts to import categorial grammar intuitions into LFG, such as König (1995), in that feature structure unification is replaced with a different kind of formal device.

The linguistic phenomena which motivate LFG's existential constraints and constraint equations receive a simple account when feature values are treated as resources. But some things which are simple to describe in a constraint-based framework, such as the transitivity of agreement, become more complicated to express in a resource sensitive framework. Johnson (1997) argues that most linguistic relationships in LFG have a natural resource sensitive character, so R-LFG is not so much a new linguistic theory as a new way of expressing the intuitions behind LFG linguistic analyses, using simpler formal tools than standard LFG. Johnson (to appear) discusses much the same material as this paper, but from a less formal and more linguistic perspective.

2. Architecture of R-LFG

Figure 1 sketches the architecture of LFG as described in Kaplan and Bresnan (1982). In their system a grammar consists of a set of syntactic rules and a lexicon.[1] The lexicon pairs the phonological forms of words with a c-structure category (e.g., N, V, etc.) and a f-description, which is a formula that describes the f-structure elements that can be associated with this word. The syntactic rules are phrase structure rules annotated with instructions on how to form the f-description of the parent node from the f-descriptions associated with its children.

The lexical entries and syntactic rules generate a set of trees, called *c-structures*, each of which is paired with an f-description. The yield of the c-structure is a string of phonological words, which comprises the phonological form of this utterance. The f-descriptions describe *f-structures*, which are attribute-value structures (Johnson, 1988; Johnson, 1991). In order for an utterance to be well-formed its f-description must be consistent (i.e., satisfied by some f-structure). Moreover, the f-description may contain additional constraints that a *minimal* f-structure that satisfies the "standard" attribute-value constraints must satisfy as well, such as the completeness constraints requiring the existence of their arguments and "constraint equations" which require that their

[1] The lexicon itself consists of a set of lexical entries closed under a set of lexical rules, but this additional structure is not discussed here.

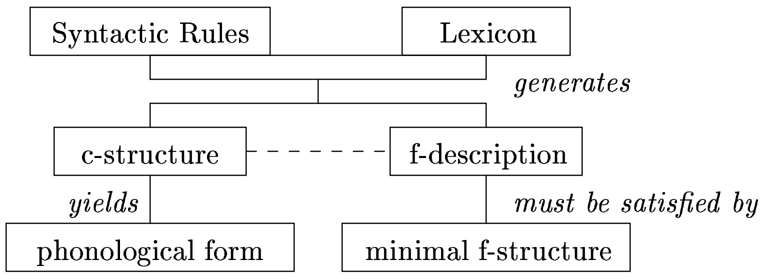


Figure 1. The architecture of Lexical Functional Grammar of Kaplan and Bresnan (1982)

arguments are forced to be equal by other constraints. These additional constraints that the minimal model must satisfy make LFG's system of f-structure constraints non-monotonic (Shieber, 1986).

The resulting system is hard to model formally. Kaplan and Bresnan (1982) describe a two-step process in which the "defining" equations specify a minimal model in which the existential constraints and "constraint equations" must hold. Johnson (1995) suggested formalizing this account using circumscription, and pointed out that disjunction does

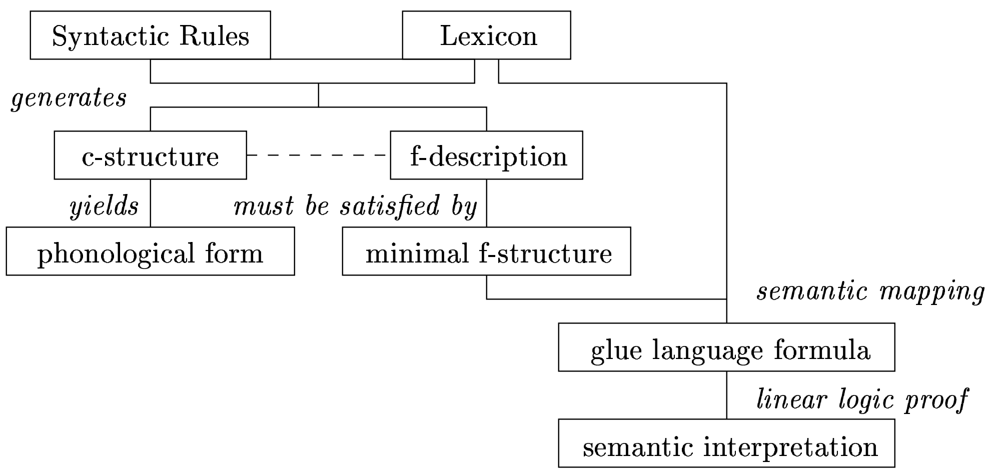


Figure 2. The architecture of the Linear Logic extension of Lexical Functional Grammar proposed by Dalrymple and colleagues

not have its intended properties in such a system.[2] R-LFG has no such problems.

[2] For example, consider a treatment of Case marking in which subject NPs are optionally assigned a nominative Case feature *nom*, such as the Andrews (1982) analysis of Icelandic quirky case marking discussed in section 6, using the following LFG syntactic rule.

$$S \rightarrow \begin{array}{c} \text{NP} \\ (\uparrow \textsc{subj}) = \downarrow \\ ((\uparrow \textsc{subj case}) = \mathit{nom}) \end{array} \quad \begin{array}{c} \text{VP} \\ \uparrow = \downarrow \end{array}$$

The parentheses surrounding the lower equation annotating the NP indicates that this defining equation is optional, reflecting the fact that the subject NP is only optionally assigned nominative case (as it may be assigned a 'quirky' non-nominative case by the verb, as explained in section 6). This annotation presumably abbreviates the following disjunction:

$$(\uparrow \textsc{subj case}) = \mathit{nom} \lor \mathit{true}$$

Clearly replacing this disjunction with *true* does not change the set of minimal models for any f-description which contains it, so the equation itself has no effect on the minimal models. This is obviously not the interpretation that Kaplan and Bresnan intend.

Kaplan and Bresnan (1982) do not discuss disjunction, but it appears they intend disjunctions to be interpreted as an abbreviatory convention, i.e., that their process applies only to individual conjunctions after expansion to a Disjunctive Normal Form (DNF). Thus their treatment, while not falling foul of the problem just noted, involves a rather curious mixture of proof-theoretic devices (e.g., DNF expansion) and model-theoretic devices (e.g., focussing on minimal models).

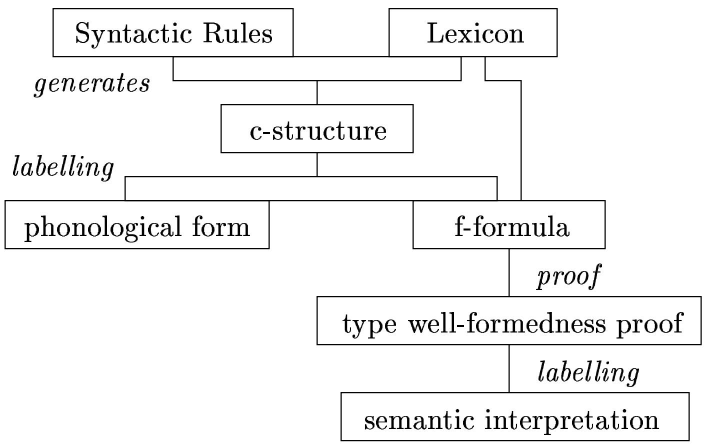


Figure 3. The architecture of R-LFG

In a series of papers, Mary Dalrymple and colleagues extend the Kaplan and Bresnan model to account for semantic interpretation by adding another component called a "semantic projection", as shown in Figure 2 (Dalrymple et al., 1995b; Dalrymple et al., 1996a; Dalrymple et al., 1996b; Dalrymple et al., 1996c). The f-structure and the lexical entries appearing in the utterance together determine a formula in a "glue language", which is a first-order language with linear logic connectives that describes how meanings are associated with elements of the minimal f-structure. In order for the utterance to be well-formed this glue language formula must derive a single atomic consequence of the form $f_\sigma \rightsquigarrow \varphi$ using the proof rules of linear logic, where f_σ is the semantic projection of the root f-structure element and φ contains no linear logic connectives. This indicates that every predicate is semantically saturated and that all of the semantic values associated with lexical items and phrases have combined into the single semantic value φ. The term φ, interpreted as a formula from a classical higher-order logic, is then a semantic interpretation of the utterance.

The system proposed in this paper is a simplification of the one in Dalrymple et al. (1995a), as shown in Figure 3, and hence might be preferable on the grounds of parsimony. It consists of two labelled deductive systems (Gabbay, 1994), one associating c-structures and f-formulae (the correlate of f-descriptions here), and another associating f-formulae with semantic interpretations.[3] The syntactic rules and lex-

[3] It is possible to combine these into a single labelled deductive system, but the dual system presented here emphasises the connection with LFG.

ical entries are formulated as proof rules, and the c-structures are just the proofs generated by these rules. Phonological forms and f-formulae are "read off" these c-structures; they are homomorphic images of the c-structures defined via labelling relations.[4] The f-formulae play the roles of both the f-description and the glue language formula in the system of Dalrymple et al. (1995a). The f-formulae associated with a c-structure must derive the atomic type t, which indicates that all predicates are saturated and that all semantic values have combined to form an entity of type t (the type of saturated propositions). The semantic interpretation is given by another labelling function associating these proofs with lambda terms, which when given a standard interpretation yield semantic interpretations for utterances.

This system differs from the one described by Dalrymple and colleagues in the following ways:

- F-formulae play the role of both f-descriptions and linear logic glue formulae, making redundant the f-description satisfiability requirement, the f-structure minimization and the associated non-monotonic devices.
- The lambda terms that provide semantic interpretation are determined by a Curry-Howard style labelling relation, rather than constructed via a "glue language". This cleanly separates the type information which is used to control the linear logic deduction from the lambda expressions which denote the semantic interpretations.
- The proof system itself is purely propositional. Variables do appear in the lambda terms labelling the f-formulae, but these labels merely record, rather than determine, the structure of the proof.

3. C-structures and F-formulae

This section begins the formal presentation of R-LFG. We start with the f-formulae, show how they can be labelled with lambda terms, and then discuss how the labelled f-formulae themselves are derived from c-structures (see Figure 3).

3.1. Labelled f-formulae

The f-formulae in R-LFG express the grammatical relation structure of an utterance. The f-formula language is a residuated multi-modal

[4] The relationship between c-structures, phonological forms and f-descriptions in LFG could be expressed in terms of a labelling relationship as well.

language with one binary connective '$\otimes$' and corresponding residuated implication '$\multimap$', and a family of modal operators indexed by grammatical functions.

Following Moortgat (1997), the set $\mathcal{F}$ of *f-formulae* is constructed from a finite set $\mathcal{A}$ of atomic types closed under implication, conjunction and modal prefixation with operators indexed by $\mathcal{G}$:

$$\mathcal{F} \rightarrow \mathcal{A} \mid (\mathcal{F} \multimap \mathcal{F}) \mid (\mathcal{F} \otimes \mathcal{F}) \mid (\mathcal{G}\ \mathcal{F}) \qquad (1)$$

Informally, modal operators are used to encode features in a manner reminiscent of Kasper and Rounds (1990), in that a f-formula $(g\ F)$ where $g \in \mathcal{G}$ and $F \in \mathcal{F}$ can be read as "F appears in the context g". More precisely, a modal operators $g \in \mathcal{G}$ behave as a unary modal operator $\langle g \rangle$ in the logics described by Moortgat (1997); the corresponding residuals $[\cdot]^{\downarrow}$ are not used in linguistic applications. The operators '$\otimes$' and '$\multimap$' are interpreted as in linear logic. Parentheses are minimized by assuming that modal operators bind most tightly, followed by implication '$\multimap$', and then the product '$\otimes$'.

For example, if $\mathcal{A} = \{e, t\}$ (the types of individuals and saturated propositions respectively)[5] and $\mathcal{G} = \{\textsc{subj}, \textsc{obj}, \textsc{comp}, \textsc{xcomp}\}$, then

$$\textsc{obj}\ e \multimap (\textsc{subj}\ e \multimap t)$$

is an f-formula. Informally, this f-formula is the type of a transitive verb; i.e., something which maps an object phrase of type e into a function from a subject phrase of type e to a phrase of type t.

An f-formula both specifies the type of the lambda term that denotes the semantic interpretation of a phrase, and encodes the linkage between the arguments of that lambda term and the grammatical relations borne by the phrases that will provide the semantic values that saturate those arguments. The modal operators themselves are "semantically impotent" in the sense of Morrill (1994), since they serve merely to express these linkage constraints.

Thus f-formulae can be interpreted as decorated type expressions for a typed lambda calculus (where the product '$\otimes$' is interpreted as the type corresponding to a pair-forming operation '$\bullet$' with corresponding first and second projections 'π_1' and 'π_2' respectively). More explicitly, define the mapping $(\cdot)^{\natural}$ from $\mathcal{F}$ to type expressions of the typed lambda

[5] This paper is not concerned with issues of intensionality, and for simplicity all propositions are assigned type t. It is relatively straightforward to modify the analyses presented here to use the type assignments of any of the standard strongly typed treatments of intensionality, though.

calculus (with product) as follows:

$$\begin{array}{rcll} (a)^\natural & = & a & \text{where } a \in \mathcal{A} \\ (A \otimes B)^\natural & = & (A^\natural) \otimes (B^\natural) & \text{where } A, B \in \mathcal{F} \\ (A \multimap B)^\natural & = & (A^\natural) \multimap (B^\natural) & \text{where } A, B \in \mathcal{F} \\ (g\, A)^\natural & = & A^\natural & \text{where } g \in \mathcal{G}, A \in \mathcal{F} \end{array}$$

For example, $(\text{OBJ}\, e \multimap (\text{SUBJ}\, e \multimap t))^\natural = e \multimap (e \multimap t)$, which is the type usually associated with lambda expressions that express the denotation of transitive verbs.

A *labelled f-formula* is a pair $s : f$ where $f \in \mathcal{F}$ is a f-formula and s is an expression of the typed lambda calculus of type $f^\natural$. For example

$$\lambda x.\lambda y.\mathit{like}'(y, x) : \text{OBJ}\ e \multimap (\text{SUBJ}\ e \multimap t)$$

is a labelled f-formula consisting of a lambda term that denotes the semantic interpretation of the verb *likes*, together with an f-formula that describes its grammatical valence.

3.2. Labelled deduction using f-formulae

This section provides a Gentzen-style sequent calculus presentation of the basic logic of f-formulae. This logic is the Lambek calculus with Permutation (LP) (van Benthem, 1995) together with proof rules for modal operators as described in Moortgat (1997). It will be extended with additional proof rules below.

It is presented as a sequent calculus, although other presentations are possible. In order to present the sequent calculus we need a binary structural operator '$(\cdot,\cdot)^\otimes$' and a family of unary structural operators '$(\cdot)^g$' indexed by $g \in \mathcal{G}$. Let $\mathcal{F}_l$ be the set of labelled f-formulae. The set $\mathcal{F}_t$ of *labelled f-terms* is defined as follows:

$$\mathcal{F}_t \;\rightarrow\; \mathcal{F}_l \mid (\mathcal{F}_t, \mathcal{F}_t)^\otimes \mid (\mathcal{F}_t)^g \qquad (2)$$

A (labelled, intuitionistic) *f-sequent* is a pair of a labelled f-term and a labelled f-formula, written $\Delta \vdash X$, where $\Delta \in \mathcal{F}_t$ is called the *antecedent* and $X \in \mathcal{F}_l$ is called the *consequent.*

Informally, a sequent $\Delta \vdash \alpha : A$ can be read as "the configuration of objects Δ can combine to form the object α of type A". The structure of a proof is determined solely by the f-formulae; the labels merely serve to map this proof structure into a lambda term. In our application Δ will consist of labelled f-formulae, usually derived from the lexical items of an utterance and structured in a way determined by the utterance's c-structure. The proof shows that these can be combined to form an object of type t, corresponding to a saturated proposition.

$$\frac{}{\alpha : A \vdash \alpha : A}\ \text{Ax} \qquad \frac{\Delta \vdash \beta : B \quad \Gamma[\beta : B] \vdash \alpha : A}{\Gamma[\Delta] \vdash \alpha : A}\ \text{Cut}$$

$$\frac{\Gamma[(\beta : B, \gamma : C)^{\otimes}] \vdash \alpha : A}{\Gamma[\beta \bullet \gamma : B \otimes C] \vdash \alpha : A}\ \otimes\text{L} \qquad \frac{\Gamma \vdash \alpha : A \quad \Delta \vdash \beta : B}{(\Gamma, \Delta)^{\otimes} \vdash \alpha \bullet \beta : A \otimes B}\ \otimes\text{R}$$

$$\frac{\Delta \vdash \beta : B \quad \Gamma[\gamma(\beta) : C] \vdash \alpha : A}{\Gamma[(\Delta, \gamma : B \multimap C)^{\otimes}] \vdash \alpha : A}\ \multimap\text{L} \qquad \frac{(\Gamma, x : B)^{\otimes} \vdash \alpha : A}{\Gamma \vdash \lambda x.\alpha : B \multimap A}\ \multimap\text{R}$$

$$\frac{\Gamma[(\Delta_2, \Delta_1)^{\otimes}] \vdash \alpha : A}{\Gamma[(\Delta_1, \Delta_2)^{\otimes}] \vdash \alpha : A}\ \text{P} \qquad \frac{\Gamma[((\Delta_1, \Delta_2)^{\otimes}, \Delta_3)^{\otimes}] \vdash \alpha : A}{\Gamma[(\Delta_1, (\Delta_2, \Delta_3)^{\otimes})^{\otimes}] \vdash \alpha : A}\ \text{A}$$

$$\frac{\Gamma[(\beta : B)^{g}] \vdash \alpha : A}{\Gamma[\beta : g\, B] \vdash \alpha : A}\ \Diamond\text{L} \qquad \frac{\Gamma \vdash \alpha : A}{(\Gamma)^{g} \vdash \alpha : g\, A}\ \Diamond\text{R}$$

Figure 4. The sequent calculus proof rules for f-sequents

The proof also determines the typed lambda term labelling the consequent via a Curry-Howard correspondence. This lambda term, as a syntactic object, records the abstract structure of the proof, and β-reduction of that lambda term corresponds to proof normalization (Girard, Lafont, and Taylor, 1989). It turns out that this lambda term, interpreted in the standard way as denoting a function in a given intended interpretation, also provides an adequate truth-conditional semantics for the natural language utterances that the labelled f-terms correspond to (van Benthem, 1995). (In effect, this amounts to claiming that the proof of syntactic well-formedness of an utterance also determines its semantic interpretation). Since we are interested in the lambda terms labelling f-formulae for the latter reason, we freely apply α-conversion and β-reduction to simplify lambda terms during the course of a proof.

The sequent calculus proof rules presented in Figure 3.2 are those for the pure residuation calculus of Moortgat (1997). In these rules A, B, C range over f-formulae; α, β, γ range over typed lambda expressions; x is a fresh lambda calculus variable not appearing elsewhere in the proof; and Γ, Δ range over labelled f-terms. In addition, $\Gamma[\Delta]$ indicates an f-term Γ with a distinguished occurence of the subterm Δ.

The *identity axioms* 'Ax' and the Cut rule express the reflexivity and transitivity of the proof system. (Moortgat (1997) shows that the Cut rule is eliminable in such a multi-modal system; i.e., removing it does not affect the set of provable sequents).

$$\begin{tikzcd}
\lambda x.sleep'(x) : \textsc{subj}\ e \multimap t \arrow[d, "\textsc{subj}"] \\
kim' : e
\end{tikzcd}$$

Figure 5. A graphical representation of the antecedent in the proof (3)

The *logical rules* ‘⊗L’ and ‘⊗R’ define the properties of the product ‘⊗’, while ‘⊸L’ and ‘⊸R’ define the properties of its residuated implication ‘⊸’. Similarly, the rules ‘◇L’, ‘◇R’ define the properties of the modals ‘g’. Note that the semantic labelling counterpart to the modal rules ‘◇L’ and ‘◇R’ is identity, since modals are semantically impotent as discussed above.

The *structural rules* ‘P’ and ‘A’ permit us to restructure binary structural connectives in the antecedent; these rules make the product connective ‘⊗’ commutative and associative. In effect, in a product of formulae only the number of times each subformula appears is important, i.e., products encode *multi-sets* of resources.

Because of this, it is convenient to adopt a “sugared” syntactic notation in which a comma-separated sequence $\Delta_1, \ldots, \Delta_n$ of f-terms in an antecedent abbreviates an arbitrary binary branching term structure using the binary structural connective $(\cdot,\cdot)^{\otimes}$ with (some permutation of) the Δ_i as leaves.

We close this section with some example proofs. The first example (3) demonstrates that the subject phrase *Kim* can combine with the intransitive verb *sleeps*. Notice the way that the ‘◇R’ rule cancels the modal operator SUBJ, permitting evaluation in this grammatical context.

$$\dfrac{\dfrac{\dfrac{}{kim' : e \vdash kim' : e}\,\text{Ax}}{(kim' : e)^{\textsc{subj}} \vdash kim' : \textsc{subj}\ e}\,\Diamond\text{R} \qquad \dfrac{}{sleep'(kim') : t \vdash sleep'(kim') : t}\,\text{Ax}}{(kim' : e)^{\textsc{subj}}, \lambda x.sleep'(x) : \textsc{subj}\ e \multimap t \vdash sleep'(kim') : t}\,\multimap\text{L} \tag{3}$$

The antecedent in this proof can be understood as having the tree structure depicted in Figure 5. In general this kind of tree-structured graphical representation can depict the bracketting structure imposed by the unary structural connectives in f-terms.

The second example (4) shows how hypothetical reasoning using the right hand rules permits us to reorder arguments in this system. This example shows that standard type assignment for a transitive verb, viz., as a functor that combines first with its object and then with its subject, can be transformed into a type assignment in which these arguments are permuted. The lambda terms labelling the types reflect this permutation. It is clear that this system permits arguments to be arbitrarily permuted, and that the grammatical functions encoded via the modal operators, rather than the order of arguments in some argument list, determine how predicates are linked with their appropriate arguments.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x : \textsc{obj}\ e \vdash x : \textsc{obj}\ e}\,\text{Ax} \qquad \dfrac{\dfrac{}{y : \textsc{subj}\ e \vdash y : \textsc{subj}\ e}\,\text{Ax} \quad \dfrac{}{\beta(x)(y) : t \vdash \beta(x)(y) : t}\,\text{Ax}}{y : \textsc{subj}\ e, \beta(x) : \textsc{subj}\ e \multimap t \vdash \beta(x)(y) : t}\,{\multimap}\,\text{L}}{x : \textsc{obj}\ e, y : \textsc{subj}\ e, \beta : \textsc{obj}\ e \multimap (\textsc{subj}\ e \multimap t) \vdash \beta(x)(y) : t}\,{\multimap}\,\text{L}}{y : \textsc{subj}\ e, \beta : \textsc{obj}\ e \multimap (\textsc{subj}\ e \multimap t) \vdash \lambda x.\beta(x)(y) : \textsc{obj}\ e \multimap t}\,{\multimap}\,\text{R}}{\beta : \textsc{obj}\ e \multimap (\textsc{subj}\ e \multimap t) \vdash \lambda y.\lambda x.\beta(x)(y) : \textsc{subj}\ e \multimap (\textsc{obj}\ e \multimap t)}\,{\multimap}\,\text{R} \qquad (4)$$

3.3. C-STRUCTURE CATEGORIES

The previous subsections presented the logic of f-formulae and showed how they are labelled with lambda terms. This subsection and the ones following show how labelled f-formulae are associated with c-structures. A separate labelled deduction system is used here to do this. It is nothing more than a deductive presentation of the usual method for annotating phrase structure grammar parse trees used in "unification-based" grammars such as LFG.

Let $\mathcal{C}$ be a finite set of *c-structure categories* (these correspond to non-terminals in a context free grammar). A *labelled c-structure category* is a triple $\phi : C : \tau$, where $C \in \mathcal{C}$ is a c-structure category, ϕ is a string of phonological forms and $\tau \in \mathcal{F}_t$ is a labelled f-term.

For example, if $\mathcal{C} = \{\textsc{s}, \textsc{np}, \textsc{vp}, \textsc{v}\}$ then the following are labelled c-structure categories. The first two might be used to describe the words *Kim* and *sleeps*, while the third describes the sentence *Kim sleeps.*

$$\text{Kim} : \text{NP} : (\mathit{kim}' : e) \qquad (5)$$

$$\text{sleeps} : \text{V} : (\lambda x.\mathit{sleep}'(x) : \textsc{subj}\ e \multimap t) \qquad (6)$$

$$\text{Kim sleeps} : \text{S} : ((\mathit{kim}' : e)^{\textsc{subj}}, \lambda x.\mathit{sleep}'(x) : \textsc{subj}\ e \multimap t)^{\otimes} \qquad (7)$$

Let $\mathcal{C}_l$ be the set of labelled c-structure categories. A *lexicon* is a finite subset $L \subset \mathcal{C}_l$; informally a lexical entry $\phi : C : \tau \in L$ asserts that the phonological form ϕ can be analysed as an instance of the c-structure category C associated with the f-term τ.

3.4. Deduction with labelled c-structure categories

This subsection develops the proof system for c-structure categories. The proof system itself consists of proof rule schemata that correspond to context free productions annotated with instructions that specify how to construct the phonological form and f-term labellings of the parent node from those on its children

There are many ways in which this can be done, and the choice between them is largely irrelevant to the main point of this paper. The method presented here was chosen because it closely resembles the treatment in Kaplan and Bresnan (1982), especially in its use of meta-variables and meta-terms.[6]

The instructions for the f-term labelling are presented using "meta f-terms". We start with a sequence $\mathcal{V} = v_1, v_2, \ldots$ of *meta-variables* (distinct from the f-terms) indexed by positive integers, and define the set $\mathcal{F}_m$ of (labelled) *meta f-terms* as follows:

$$\mathcal{F}_m \; \rightarrow \; \mathcal{V} \mid \mathcal{F}_t \mid (\mathcal{F}_m, \mathcal{F}_m)^{\otimes} \mid (\mathcal{F}_m)^g$$

That is, a meta f-term is just like an f-term, except that it may contain a meta-variable v_i in place of an f-term. Informally, the meta-variable v_i is a place holder in a meta f-term that will be replaced by the f-term associated with the ith child in a syntactic rule.

Then a *syntactic rule* is a triple $\langle C, \chi, \mu \rangle$, where $C \in \mathcal{C}$ is the *parent c-structure category*, $\chi \in \mathcal{C}^*$ is the sequence of *children c-structure categories* and μ is a meta f-term such that if the meta-variable v_i appears in μ then $i \leq |\chi|$. Informally, such a syntactic rule is to be understood as a phrase structure rule $C \rightarrow \chi$, with the meta f-term specifying how the f-term labelling the parent C is to be constructed out of the f-terms labelling its children χ.

In order to keep the size of the proofs managable, the proof system is presented in a natural deduction format. This proof system manipulates trees; the leaf nodes of the tree being read as antecedents and the root node as the consequent.

The axioms are all single node trees

$$c$$

[6] LFG permits a wider class of productions than those described in this paper. For example, in LFG the right hand side of productions are permitted to contain arbitrary regular expressions, rather than just the finite sequence of labelled categories permitted here. It does not seem to be particularly problematic to extend the class of syntactic rules presented below to permit such regular expressions, but this extension is not considered here because it does not seem germane to this paper's primary focus, i.e., the extent to which LFG linguistic analyses usually expressed in terms of f-structure constraints can be re-expressed using f-formulae.

where $c \in \mathcal{C}_l$.

Let R be a set of syntactic rules. For each syntactic rule $\langle C, C_1 \cdots C_n, \mu\rangle \in R$ there is a proof rule:

$$\frac{\begin{matrix} \vdots & & \vdots \\ \phi : C_1 : \tau_1 & \cdots & \phi_n : C_n : \tau_n \end{matrix}}{(\phi_1 \cdots \phi_n) : C : \mu[\tau_1/v_1, \ldots, \tau_n/v_n]}$$

where $\phi_1 \cdots \phi_n$ is the concatenation of $\phi_1, \ldots, \phi_n$, and $\mu[\tau_1/v_1, \ldots, \tau_n/v_n]$ denotes the simultaneous replacement of the v_i with the τ_i in μ. (Because of the restriction on the meta-variables that can occur in μ the result must be an f-term).

A *c-structure proof* of c from assumptions $c_1, \ldots, c_n$ is a tree with root c and yield $c_1, \ldots, c_n$ constructed using the proof rules just given.

For example, suppose R contains the following syntactic rules:

$$\langle \mathrm{S}, \mathrm{NP}\,\mathrm{VP}, ((v_1)^{\textsc{subj}}, v_2)^{\otimes}\rangle \tag{8}$$

$$\langle \mathrm{VP}, \mathrm{V}, v_1\rangle \tag{9}$$

Then the following is a c-structure proof using R. It shows that the labelled c-structure constituents in (5) and (6) that were associated with the words *Kim* and *sleeps* can be combined to yield the c-structure constituent associated with *Kim sleeps* in (7).

$$\frac{\mathrm{Kim} : \mathrm{NP} : (\mathit{kim}' : e) \quad \dfrac{\mathrm{sleeps} : \mathrm{V} : (\lambda x.\mathit{sleep}'(x) : \textsc{subj}\ e \multimap t)}{\mathrm{sleeps} : \mathrm{VP} : (\lambda x.\mathit{sleep}'(x) : \textsc{subj}\ e \multimap t)}\,(9)}{\mathrm{Kim\ sleeps} : \mathrm{S} : ((\mathit{kim}' : e)^{\textsc{subj}}, \lambda x.\mathit{sleep}'(x) : \textsc{subj}\ e \multimap t)^{\otimes}}\,(8) \tag{10}$$

Note that this proof is essentially just the non-terminal part of a standard LFG c-structure (drawn inverted) with nodes labelled with labelled f-terms. The corresponding standard c-structure tree is depicted in Figure 6.

It is convenient to devise another notation for syntactic rules that is closer to the one used by Kaplan and Bresnan (1982). This notation is not capable of expressing all syntactic rules, but it is very useful in practice.[7] In this notation syntactic rules are written as phrase structure rules in which the c-structure categories on the right hand side are optionally annotated with labelled f-terms that may contain the symbol '$\downarrow$', which abbreviates the meta-variable corresponding to this category. A category without any annotation is assumed to be annotated with the term '$\downarrow$'. The meta f-term associated with the rule as

[7] For example, this notation cannot express syntactic rules labelled with meta-terms in which two meta-variables appear embedded within the same modal.

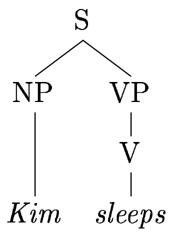


Figure 6. The standard LFG c-structure tree corresponding to the c-structure proof in (10)

a whole is obtained by taking the product of the terms labelling the right hand c-structure categories. For example, the two syntactic rules in (8) and (9) would be written as (11) and (12) respectively in this notation.

$$\mathrm{S} \to \underset{\textsc{subj}\ \downarrow}{\mathrm{NP}}\ \mathrm{VP} \tag{11}$$

$$\mathrm{VP} \to \mathrm{V} \tag{12}$$

In addition, annotated c-structure categories in such rules may themselves appear bracketted, which means that they and their associated annotations are optional. Thus the notation

$$\mathrm{VP} \to \mathrm{V} \left(\underset{\textsc{obj}\ \downarrow}{\mathrm{NP}}\right) \left(\underset{\textsc{xcomp}\ \downarrow}{\mathrm{VP}}\right) \tag{13}$$

abbreviates the following four syntactic rules:

$$\begin{gathered}
\langle \mathrm{VP}, \mathrm{V}, v_1 \rangle \\
\langle \mathrm{VP}, \mathrm{V\,NP}, (v_1, (v_2)^{\textsc{obj}})^{\otimes} \rangle \\
\langle \mathrm{VP}, \mathrm{V\,VP}, (v_1, (v_2)^{\textsc{xcomp}})^{\otimes} \rangle \\
\langle \mathrm{VP}, \mathrm{V\,NP\,VP}, (v_1, (v_2)^{\textsc{obj}}, (v_3)^{\textsc{xcomp}})^{\otimes} \rangle
\end{gathered}$$

3.5. Derivations of R-LFG

We are now in a position to define a resource sensitive Lexical Functional Grammar. An R-LFG is a septuple $G = \langle \mathcal{A}, \mathcal{G}, \mathcal{C}, L, R, s, t \rangle$ where $\mathcal{A}$ and $\mathcal{G}$ are the atoms and modal operators defining f-formulae $\mathcal{F}$ as described above, $\mathcal{C}$ is a set of c-structure categories, L is a lexicon, R

is a set of syntactic rules, $s \in \mathcal{C}$ is the *c-structure start category* and $t \in \mathcal{F}$ is the *saturated f-formula.*[8]

Then G associates a phonological form ϕ with a semantic interpretation given by the denotation of the lambda term α iff:

1. there is a c-structure proof of a labelled c-structure category $\phi : s : \tau$ using R from assumptions in L, and

2. the f-term τ labelling this proof validates $\tau \vdash \alpha : t$.

For example, suppose the start category is S and the saturated f-formula is t. We saw in (10) above that there is a proof of:

$$\text{Kim sleeps} : \text{S} : ((kim' : e)^{\text{SUBJ}}, \lambda x.sleep'(x) : \text{SUBJ}\ e \multimap t)^{\otimes}$$

from the lexicon L given earlier. Moreover, (3) showed that

$$(kim' : e)^{\text{SUBJ}}, \lambda x.sleep'(x) : \text{SUBJ}\ e \multimap t \vdash sleep'(kim') : t$$

so we can conclude that *Kim sleeps* has an interpretation given by the denotation of $sleep'(kim')$.

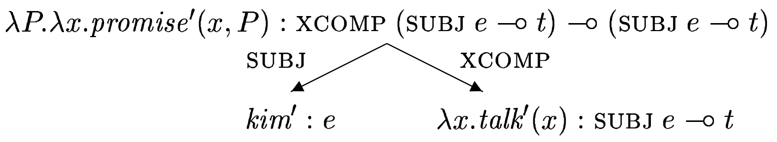
We close this section with a more complex example based on Chierchia's analysis of Control constructions (specifically, what used to be called "Equi-NP deletion" in classical transformational grammar), such as *Kim promised to talk.* This example shows how higher-order types can be manipulated in R-LFG, and serves as the starting point for the analysis of Raising and the LFG analysis of Control discussed below.

Chierchia (1984) observed that Control constructions such as *Kim promised to talk* can be analysed as the main verb *promised* taking two arguments: the subject noun phrase *Kim* and a semantically unsaturated verb phrase *to talk.* The semantic implicature that Kim will be the one talking is not expressed syntactically, but is part of the meaning of the verb *promise.*

One way of doing this is to restrict attention to a certain class of interpretations. If we require that our semantic interpretations always satisfy the formula (14) then the denotation of the Control predicate $promise'$ of type $(e \otimes e \multimap t) \multimap t$ must be appropriately related to the denotation non-Control predicate $promise'_*$ of type $(e \otimes t) \multimap t$, where $promise'_*(x, Q)$ is interpreted as "x promises that Q".

$$\forall x. \forall P.\, promise'(x, P) \leftrightarrow promise'_*(x, P(x)). \tag{14}$$

[8] In the current application the saturated f-formula t is an atomic f-formula, i.e., $t \in \mathcal{A}$. However, it is not necessary that this is the case, and in R-LFGs which deal with more complex natural language semantic phenomena such as intensionality or dynamic interpretation it may be more natural to take the saturated f-formula to be non-atomic.



$$\begin{tikzcd}
& \lambda P.\lambda x.\mathit{promise}'(x,P) : \mathrm{XCOMP}\ (\mathrm{SUBJ}\ e \multimap t) \multimap (\mathrm{SUBJ}\ e \multimap t) \arrow[dl, "\mathrm{SUBJ}"'] \arrow[dr, "\mathrm{XCOMP}"] & \\
\mathit{kim}' : e & & \lambda x.\mathit{talk}'(x) : \mathrm{SUBJ}\ e \multimap t
\end{tikzcd}$$

Figure 7. A depiction of the f-term (15) labelling the c-structure proof of *Kim promised to talk.*

If we restrict attention to interpretations in which (14) is true, then the desired implicature will hold. Formulae such as (14) which express such constraints on interpretations are called *meaning postulates.*

The Chierchia analysis can be formulated in R-LFG as follows. The following lexical entries are used in this analysis. We assume that *to talk* is a lexical entry here, but in a more complete fragment this phrase would be decomposed into its separate words.

Kim : NP : $(\mathit{kim}' : e)$
promised : V : $(\lambda P.\lambda x.\mathit{promise}'(x,P) : \mathrm{XCOMP}\ (\mathrm{SUBJ}\ e \multimap t) \multimap (\mathrm{SUBJ}\ e \multimap t))$
to talk : VP : $(\lambda x.\mathit{talk}'(x) : \mathrm{SUBJ}\ e \multimap t)$

Notice that *promised* is assigned a higher type than normal complement-taking verbs; it combines with an *unsaturated* XCOMP argument before combining with its subject.

The syntactic rules (11) and (13) suffice here, yielding a c-structure proof of category S associating the phonological form *Kim promised to talk* with the labelled f-term:

$$\begin{pmatrix} (\mathit{kim}' : e)^{\mathrm{SUBJ}}, \\ \lambda P.\lambda x.\mathit{promise}'(x,P) : \mathrm{XCOMP}\ (\mathrm{SUBJ}\ e \multimap t) \multimap (\mathrm{SUBJ}\ e \multimap t), \\ (\lambda x.\mathit{talk}'(x) : \mathrm{SUBJ}\ e \multimap t)^{\mathrm{XCOMP}} \end{pmatrix}^{\otimes} \tag{15}$$

This f-term can be depicted as shown in Figure 7. The following proof shows that this f-term derives the saturated type t. The lambda term labelling is not shown here to keep the proof of managable size.

$$\dfrac{\dfrac{\dfrac{}{\mathrm{SUBJ}\ e \multimap t \vdash \mathrm{SUBJ}\ e \multimap t}\,\mathrm{Ax}}{(\mathrm{SUBJ}\ e \multimap t)^{\mathrm{XCOMP}} \vdash \mathrm{XCOMP}\ (\mathrm{SUBJ}\ e \multimap t)}\,\Diamond\mathrm{R} \qquad \dfrac{\dfrac{\dfrac{}{e \vdash e}\,\mathrm{Ax}}{(e)^{\mathrm{SUBJ}} \vdash \mathrm{SUBJ}\ e}\,\Diamond\mathrm{R} \quad \dfrac{}{t \vdash t}\,\mathrm{Ax}}{(e)^{\mathrm{SUBJ}}, \mathrm{SUBJ}\ e \multimap t \vdash t}\,{\multimap}\mathrm{L}}{(e)^{\mathrm{SUBJ}}, \mathrm{XCOMP}\ (\mathrm{SUBJ}\ e \multimap t) \multimap (\mathrm{SUBJ}\ e \multimap t), (\mathrm{SUBJ}\ e \multimap t)^{\mathrm{XCOMP}} \vdash t}\,{\multimap}\mathrm{L}$$

A simple calculation shows that the lambda term labelling the consequent reduces to:

$$promise'(kim', \lambda x.talk'(x))$$

which, together with the meaning postulate (14), implies the desired semantic interpretation.

4. Functionality and Functional Identification

As noted above, f-terms have a tree structure. This section discusses how some of the properties of the graph structure of LFG's f-structures can be obtained in R-LFG. These extensions are needed to express the LFG analysis of Raising and related constructions; see Bresnan (1982) for a justification of such linguistic analyses. Two separate components are added to R-LFG to achieve this. First, we add an additional proof rule which imposes a weak functionality on modal contexts. Second, we extend the f-term language to allow functional identifications, and add a proof rule schema to interpret them.

In Raising constructions, such as *Kim seems to talk*, there is syntactic and semantic evidence that the matrix subject in such constructions is an argument of only the embedded clause. That is, the semantic interpretation of this example is

$$seem'_*(talk'(kim')). \tag{16}$$

Now, by assigning *seem* a lexical entry similar to the one assigned to *promised* above, we can provide the following lambda term as a semantic analysis for *Kim seems to talk*:

$$seem'(kim', \lambda x.talk'(x)). \tag{17}$$

If we require that our semantic interpretations always satisfy the following meaning postulate

$$\forall P.\forall x.\, seem'(x, P) \leftrightarrow seem'_*(P(x))$$

then we would get the desired semantic interpretation. This approach is standardly used in categorial grammar analyses of Raising.

This treatment of Raising suffers from several well-known difficulties (Dowty, 1985; Jacobson, 1990), none of which are fatal, but which some linguists take to indicate that this treatment is linguistically misguided.

– Raising verbs typically take scope over intensional subjects, while Control verbs do not. Thus (18b) entails the existence of a unicorn, while (18a) does not.

$$\begin{tikzcd}
\lambda Q.\mathit{seem}'_*(Q) : \mathrm{XCOMP}\ t \multimap t \arrow[dr, "\mathrm{XCOMP}"] \arrow[dd, "\mathrm{SUBJ}"'] & \\
& \lambda x.\mathit{talk}'(x) : \mathrm{SUBJ}\ e \multimap t \arrow[dl, "\mathrm{SUBJ}"] \\
\mathit{kim}' : e &
\end{tikzcd}$$

Figure 8. Identification of modal contexts in an informal analysis of the Raising construction *Kim seems to talk.*

(18) a. A unicorn seemed to be in the garden.

b. A unicorn promised to be in the garden.

– Dummy subjects can appear in matrix subject position of Raising verbs, but not of Control verbs. Thus (19a) is grammatical, while (19b) is not.

(19) a. There seemed to be a riot in the garden.

b. *There wanted/promised to be a riot in the garden.

For these and other reasons, Lexical Functional Grammarians typically eschew the meaning-postulate treatment of Raising just described. In R-LFG we can provide an alternative account of Raising which avoids these problems by following the original Bresnan (1982) LFG analysis more closely.

One of the insights of that paper is that Raising constructions can be described in terms of a f-structure *reentrancy* which identifies the matrix subject and the embedded subject. In R-LFG terms, this would mean that in the f-term associated with this expression the modal contexts $(\cdot)^{\mathrm{SUBJ}}$ and $((\cdot)^{\mathrm{SUBJ}})^{\mathrm{XCOMP}}$ are equivalent.

Purely informally at this stage, suppose that this identification of modal contexts has been achieved somehow, and that *seems* is assigned the lexical entry (20) (we will modify this below).

$$\text{seems} : \mathrm{V} : (\lambda Q.\mathit{seem}'_*(Q) : \mathrm{XCOMP}\ t \multimap t) \qquad (20)$$

Then the LFG analysis expressed in terms of "identification" of contexts would involve the proof of the antecedent depicted in Figure 8, and might be reasonably expected to produce the lambda term in (16).

Now we provide the two extensions to R-LFG to enable us to express such analyses in R-LFG.

First, we add a new proof rule 'F', which permits two modal contexts tagged with the same grammatical function $g \in \mathcal{G}$ to be merged.[9]

$$\frac{\Gamma[((\Delta_1, \Delta_2)^{\otimes})^g] \vdash \alpha : A}{\Gamma[((\Delta_1)^g, (\Delta_2)^g)^{\otimes}] \vdash \alpha : A} \text{ F}$$

Rule 'F' corresponds approximately to the functionality requirement in f-structures. It differs in that it merely permits, but does not require, two modal contexts corresponding to the same grammatical relation to be identified.

Second, we extend the term language $\mathcal{F}_t$ to include functional identifications. A *functional identification* is a new type of atomic term of the form $g_1 \cdots g_m = h_1 \cdots h_n$, where each $g_i, h_j \in \mathcal{G}$. That is, the set of labelled f-terms is extended as in (21).

$$\mathcal{F}_t \;\to\; \mathcal{F}_l \mid (\mathcal{F}_t, \mathcal{F}_t)^{\otimes} \mid (\mathcal{F}_t)^g \mid (\mathcal{G}^* = \mathcal{G}^*) \tag{21}$$

Functional identifications are used by instances of the structural rule schema '=' below. Informally, a functional identification is a license to move the contents of one modal context into another modal context. They play the same role here as do the path equivalences of feature logics (Rounds, 1997), which justifies their notation. They are semantically impotent, as they do not affect the labels of f-formulae. Note that while functional identifications co-exist within an f-term with labelled f-formulae, they are not labelled themselves.

$$\frac{\Gamma[(\cdots(\Delta)^{h_n}\cdots)^{h_1}] \vdash \alpha : A}{\Gamma[((\cdots(\Delta)^{g_m}\cdots)^{g_1}, g_1 \cdots g_m = h_1 \cdots h_n)^{\otimes}] \vdash \alpha : A} =$$

Because functional path equivalences in LFG play the same role as functional identification in R-LFG, functional identifications in R-LFG use the same notation as functional path equivalence in LFG. But functional identification in R-LFG is actually weaker than functional

[9] The proof rule 'F' is the inverse of the celebrated axiom 'K' of modal logic. Axiom 'K' is $\Diamond(A \otimes B) \to \Diamond A \otimes \Diamond B$, while rule 'F' is $\Diamond A \otimes \Diamond B \to \Diamond(A \otimes B)$. Note that if grammatical functions were interpreted by '$[\cdot]^{\downarrow}$' rather than $\langle\cdot\rangle$, the standard rule 'K' would suffice.

In this paper I assume that g ranges over $\mathcal{G}$ in rule 'F', the set of grammatical functions, but in fact it would suffice if g were restricted to the *open grammatical functions* (Bresnan, 1982), i.e., those functions which may be grammatically controlled. Indeed, one might characterize the open grammatical functions as precisely those $g \in \mathcal{G}$ for which 'F' holds.

$$\begin{tikzcd}
 & \lambda Q.seem'_*(Q) : \textsc{xcomp}\ t \multimap t \arrow[dl, "\textsc{subj}"'] \arrow[dr, "\textsc{xcomp}"] & \\
kim' : e \arrow[dr, dashed] & & \lambda x.talk'(x) : \textsc{subj}\ e \multimap t \arrow[dl, "\textsc{subj}"] \\
 & {} &
\end{tikzcd}$$

Figure 9. A depiction of the f-term (23). The dotted line indicates functional identification.

path equivalences in LFG, as it only permits a single asymmetrical rewriting of one modal context into another.[10]

These extensions are used in the analysis of Raising as follows. The lexical entry for the verb *seems* is modified as shown in (22) to include a functional identification between its matrix subject and its complement's subject.

$$\text{seems} : \text{V} : (\lambda Q.seem'_*(Q) : \textsc{xcomp}\ t \multimap t,\ \textsc{subj} = \textsc{xcomp subj})^{\otimes} \tag{22}$$

With this modified lexical entry, there is a c-structure proof of category S associating the phonological form *Kim seems to talk* with the labelled f-term:

$$\begin{pmatrix} (kim' : e)^{\textsc{subj}}, \\ \lambda Q.seem'_*(Q) : \textsc{xcomp}\ t \multimap t,\ \textsc{subj} = \textsc{xcomp subj}, \\ (\lambda x.talk'(x) : \textsc{subj}\ e \multimap t)^{\textsc{xcomp}} \end{pmatrix}^{\otimes} \tag{23}$$

The structure of this f-term is depicted in Figure 9. The following proof shows that this f-term can reduce to the saturated type t. The proof begins by first using the functional identification in the f-term labelling *seems* to "lower" the matrix subject into an XCOMP context, and proof rule 'F' to incorporate it into the XCOMP context that contains the embedded verb phrase. The orientation of this functional identification is important; the proof would not go through if its

[10] One could make functional identifications behave more like the path equivalences of feature logics by adding additional rules, such as the following:

$$\frac{\Gamma[h_1 \cdots h_n = g_1 \cdots g_m] \vdash \alpha : A}{\Gamma[g_1 \cdots g_m = h_1 \cdots h_n] \vdash \alpha : A}\ \text{S}.$$

This rule has the effect of permitting a functional identification to be used in either direction. However, I know of no linguistic motivation for such symmetry, and hence I have not included it here. In fact, all linguistic constructions analyzed using LFG functional identification I am aware of are *asymmetric*; e.g., there is no variant of the Raising construction with free variation in the location in which the shared argument is realized.

arguments were reversed. Notice also that the proof depends on the ability to apply proof rules inside f-terms (e.g., the first application of the the '$\multimap$ L' rule from the bottom).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{e \vdash e}\,\text{Ax}}{(e)^{\textsc{subj}} \vdash \textsc{subj}\; e}\,\Diamond\text{R} \qquad \dfrac{\dfrac{\dfrac{}{t \vdash t}\,\text{Ax}}{(t)^{\textsc{xcomp}} \vdash \textsc{xcomp}\; t}\,\Diamond\text{R} \quad \dfrac{}{t \vdash t}\,\text{Ax}}{(t)^{\textsc{xcomp}}, \textsc{xcomp}\; t \multimap t \vdash t}\,\multimap\text{L}}{((e)^{\textsc{subj}}, \textsc{subj}\; e \multimap t)^{\textsc{xcomp}}, \textsc{xcomp}\; t \multimap t \vdash t}\,\multimap\text{L}}{((e)^{\textsc{subj}})^{\textsc{xcomp}}, \textsc{xcomp}\; t \multimap t, (\textsc{subj}\; e \multimap t)^{\textsc{xcomp}} \vdash t}\,\text{F}}{(e)^{\textsc{subj}}, \textsc{xcomp}\; t \multimap t,\; \textsc{subj} = \textsc{xcomp}\;\textsc{subj}, (\textsc{subj}\; e \multimap t)^{\textsc{xcomp}} \vdash t}\,=$$

It is straightforward to check that this proof is labelled with the desired lambda term:

$$\mathit{seem}'_*(\mathit{talk}'(\mathit{kim}')).$$

This analysis avoids the intensionality problem mentioned above in connection with example (18a) because the matrix subject *Kim* is semantically an argument of the embedded verb *talk*.

Dummy subject NPs can be accounted for by associating them with f-formulae in the lexicon that differ from those associated with normal (non-dummy) NPs. For example, the dummy *there* might be associated with the atomic f-formula d. Then a verb that subcategorizes for a dummy NP subject cannot appear with a non-dummy NP subject, and vice versa. Because the functional identification merely lowers the matrix subject to the embedded subject the dummy/non-dummy distinction is preserved under Raising, as required.

To conclude this section, note that for this particular example an alternative analysis using the lexical entry for *seems* in (24) would derive the correct lambda term labelling, and would not require the proof schema '='.

$$\text{seems} : \text{V} : \begin{pmatrix} \lambda Q.\mathit{seem}'_*(Q) : \textsc{xcomp}\; t \multimap t, \\ \lambda x.x : \textsc{subj}\; e \multimap \textsc{xcomp}\;\textsc{subj}\; e \end{pmatrix}^{\otimes} \tag{24}$$

In (24) the f-formula $\textsc{subj}\; e \multimap \textsc{xcomp}\;\textsc{subj}\; e$ moves an entity of type e from the subject into the complement's subject, in effect "hard-wiring" the particular context modification performed by the functional identification schema into the lexical entry for *seems*.

However, this analysis would not extend to the treatment of dummy subjects just sketched, or the treatment of Quirky case marking in Raising discussed below. In both of these cases an object of arbitrary type must be "lowered" into the embedded context. This would require a polymorphic f-formula, such as $\forall X.\textsc{subj}\; X \multimap \textsc{xcomp}\;\textsc{subj}\; X$. This f-formula would be labelled with the polymorphic lambda term $\lambda x.x$, where x must range over elements of arbitrary type.

Such an extension would take us out of a purely propositional system, and is tantamount to reintroducing unification into the type machinery. Because functional identifications are unlabelled in the system presented in this paper, we do not have to label them with polymorphic lambda terms and we can remain within a strongly typed lambda calculus. Whether it will prove necessary on linguistic grounds to move to a polymorphic lambda calculus will require further empirical linguistic investigation.

5. Completeness and Contraction

This section discusses how LFG's completeness and coherence constraints (Kaplan and Bresnan, 1982) are reflected in R-LFG. The completeness constraint requires that an argument be associated with every subcategorized argument position of every predicate. The coherence constraint requires that every predicate except the root predicate appear as the argument of some other predicate. Together, they ensure that every predicate's argument requirements are met in any syntactic structure in which it appears, and that every predicate appears somewhere in the semantic structure of the utterance.

As Dalrymple et al. (1995a) observe, something like the completeness and coherence constraints are implicit in a linear resource sensitive framework, and do not have to be stipulated as independent constraints as in LFG. In fact, the resource sensitive frameworks seem to have a slight advantage, as the *type well-foundedness* they enforce automatically rules out cyclic semantic structures in which a predicate is one of its own arguments (and hence may be disconnected from the semantic structure of the rest of the utterance, but never the less the disconnected structure would satisfy LFG's completeness and coherence constraints).

However, while the LFG completeness and coherence constraints impose a certain resource sensitivity, they do not require linearity. That is, they merely require that a predicate must be used as the argument of *one or more* predicates, rather than exactly one predicate. Thus LFG completeness and coherence constraints seem to correspond to a system obtained by adding the structural rule 'C' (Contraction) to our Gentzen system for labelled f-terms, yielding a modalized version of LPC (van Benthem, 1995).

$$\frac{\Gamma[(\Delta,\Delta)^{\otimes}] \vdash \alpha : A}{\Gamma[\Delta] \vdash \alpha : A}\ \mathrm{C}$$

In fact, it is unclear whether rule ‘C’ should be incorporated into R-LFG. In any case, it seems that if rule ‘C’ is incorporated into the grammar it will need to be restricted to certain Δ, as discussed below.

The structural rule ‘C’ permits us to duplicate resources as needed. When the proof rule ‘C’ is added, every resource on the left must be used at least once (rather than exactly once) in the proof. With only the structural rules ‘P’ and ‘A’ modal contexts contain multisets of resources; the addition of rule ‘C’ reduces structural sensitivity to *sets* of resources. Thus in a framework with ‘C’ the number of times a resource appears is immaterial; perhaps this could be viewed as embodying the slogan: “grammars do not count”.

For example, in the standard LFG analysis of Control constructions such as *Kim promised to talk*, the subject *Kim* is analysed as an argument of both the matrix verb *promised* and the embedded verb *to talk* (Bresnan, 1982). We can formulate the LFG analysis in R-LFG using proof rule ‘C’ as follows.

The lexical item *promised* is given the lexical entry in (25). Notice that it includes the same functional identification as the lexical entry for *seems* in (22).

$$\text{promised} : \text{V} : \begin{pmatrix} \lambda P.\lambda x.\mathit{promise}'_*(x, P) : \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t), \\ \textsc{subj} = \textsc{xcomp subj} \end{pmatrix}^{\otimes} \tag{25}$$

Then using the other lexical entries and the syntactic rules (11) and (13) there is a c-structure proof labelled with the phonological form *Kim promised to talk* and the f-term below:

$$\begin{pmatrix} (\mathit{kim}' : e)^{\textsc{subj}}, \\ \lambda P.\lambda x.\mathit{promise}'_*(x, P) : \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t), \\ \textsc{subj} = \textsc{xcomp subj}, \\ (\lambda x.\mathit{talk}'(x) : \textsc{subj}\ e \multimap t)^{\textsc{xcomp}} \end{pmatrix}^{\otimes} \tag{26}$$

This f-term is depicted in Figure 10. The f-term proof uses rule ‘C’ to duplicate the subject, and then proceeds much as in the Raising analysis shown above.

$$\begin{tikzcd}
& \lambda P.\lambda x.\mathit{promise}'_*(x,P) : \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t) \arrow[dl, "\textsc{subj}"'] \arrow[dr, "\textsc{xcomp}"] & \\
\mathit{kim}' : e \arrow[dr, dashed] & & \lambda x.\mathit{talk}'(x) : \textsc{subj}\ e \multimap t \arrow[dl, "\textsc{subj}"] \\
& {} &
\end{tikzcd}$$

Figure 10. A depiction of the f-term (26). The dotted line indicates functional identification. Note that both the matrix and the embedded predicates consume the subject, which is duplicated using Contraction.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{e \vdash e}\,\text{Ax}}{(e)^{\textsc{subj}} \vdash \textsc{subj}\ e}\,\Diamond\text{R} \quad \dfrac{\dfrac{\dfrac{}{t \vdash t}\,\text{Ax}}{(t)^{\textsc{xcomp}} \vdash \textsc{xcomp}\ t}\,\Diamond\text{R} \quad \dfrac{\dfrac{\dfrac{}{e \vdash e}\,\text{Ax}}{(e)^{\textsc{subj}} \vdash \textsc{subj}\ e}\,\Diamond\text{R} \quad \dfrac{}{t \vdash t}\,\text{Ax}}{(e)^{\textsc{subj}}, \textsc{subj}\ e \multimap t \vdash t}\,\multimap\text{L}}{(e)^{\textsc{subj}}, (t)^{\textsc{xcomp}}, \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t) \vdash t}\,\multimap\text{L}}{(e)^{\textsc{subj}}, ((e)^{\textsc{subj}}, \textsc{subj}\ e \multimap t)^{\textsc{xcomp}}, \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t) \vdash t}\,\multimap\text{L}}{(e)^{\textsc{subj}}, ((e)^{\textsc{subj}})^{\textsc{xcomp}}, \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t), (\textsc{subj}\ e \multimap t)^{\textsc{xcomp}} \vdash t}\,\text{F}}{(e)^{\textsc{subj}}, (e)^{\textsc{subj}}, \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t), \textsc{subj} = \textsc{xcomp}\ \textsc{subj}, (\textsc{subj}\ e \multimap t)^{\textsc{xcomp}} \vdash t}\,=}{(e)^{\textsc{subj}}, \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t), \textsc{subj} = \textsc{xcomp}\ \textsc{subj}, (\textsc{subj}\ e \multimap t)^{\textsc{xcomp}} \vdash t}\,\text{C}$$

The lambda term labelling this proof is:

$$\mathit{promise}'_*(\mathit{kim}', \mathit{talk}'(\mathit{kim}')).$$

The duplication of the subject's semantics *kim′* in this lambda term reflects non-linearity introduced into this analysis by the use of rule 'C'.

Note that instead of using the general Contraction rule 'C', the specific instance of the schema needed here could have been added to the lexical entry for *promise* as in (27), (just as the functional identification could have been hardwired into the lexical entry for *seems*, as in (24)).

$$\text{promised} : \text{V} : \begin{pmatrix} \lambda P.\lambda x.\mathit{promise}'_*(x,P) : \textsc{xcomp}\ t \multimap (\textsc{subj}\ e \multimap t), \\ \lambda x.x \otimes x : \textsc{subj}\ e \multimap (\textsc{xcomp}\ \textsc{subj}\ e \otimes \textsc{subj}\ e) \end{pmatrix}^{\otimes} \tag{27}$$

The lexical entry (27) duplicates the subject, leaving one copy in the subject context and embedding the other into the complement's subject.

It is not easy to determine whether it is better to use general structural rules or whether hardwiring into particular lexical entries and syntactic rules any special properties, such as the non-linearity of Control constructions, is to be preferred. In fact, there is no reason why both approaches could not be used in one grammar. The issue is complicated, and requires a detailed empirical linguistic investigation.

The existence of constructions involving non-linearity but without obvious lexical sources of this non-linearity might be construed as an argument for a general rule of Contraction. For example, constructions such as (28) seem to involve non-linearity, but without any obvious lexical source.

(28) Kim ate the fish raw.

In (28) either *Kim* or *the fish* is an argument of the adjunct *raw*, as well as being an argument of the main verb *ate*.

However, in R-LFG there is an obvious place to locate such a non-linearity: namely in the f-term annotation on the syntactic rule that licenses this construction. Some sort of f-term annotation will be needed on these rules anyway to determine the relationship between the adjunct and the main clause. Further, in related constructions, such as the resultative in (29), it seems reasonable that the syntactic rule introduces non-trivial semantic material.[11]

(29) Kim hammered the metal flat.

In any case, it seems that an unrestricted Contraction rule will overgenerate, so Contraction will have to severely restricted in some way, e.g., to apply to just certain types.[12] Although this paper does not include a treatment of quantifier scope, some facts about the interaction of quantifier scope, Raising and Control suggest that Contraction, if it appears in the grammar at all, must be restricted.

In Raising constructions subject quantifiers can take scope over just the embedded clause; i.e., (30a) can have the reading in (30c) in addition to the reading in (30b).

(30) a. Everyone seemed to talk.

[11] In categorial grammar such constructions would be handled by lexical rules which transform either the basic verb or the adjunct. These lexical rules "compile" into the lexical entries they produce both the syntactic configurations and the specialized semantics of these constructions. It seems to me that the resulting grammars are *over-lexicalized*, in that generalizations about the relationship between syntactic configuration and semantic interpretation can only be expressed lexically.

[12] An anonymous reviewer pointed out that if non-intersective adjectives are assigned the f-formula $(e \multimap t) \multimap (e \multimap t)$ (which is the translation of the type standardly associated with them in model-theoretic semantic accounts) and if Contraction is permitted to apply freely, then we would incorrectly predict that such adjectives could apply to an argument an arbitrary number of times.

b. $\forall x.seem'_*(talk'(x))$.

c. $seem'_*(\forall x.talk'(x))$.

This suggests that functional identification can move quantifiers (whatever type they may turn out to be) from the matrix subject to the embedded subject. Since quantifiers will presumably be of higher type than e, a polymorphic treatment of functional identification may be desirable.[13]

In contrast, in Control constructions subject quantifiers must take scope over the whole construction, i.e., (31a) only has reading (31b), a reading of the form (31c) is unavailable.[14]

(31) a. Everyone promised to talk.

b. $\forall x.promise'_*(x, talk'(x))$.

c. $^\star\ \forall x.promise'(x, \forall y.talk'(y))$.

Yet reading (31c) might plausibly be generated by duplicating the subject quantifier using Contraction, and then lowering one of the resulting quantifiers into embedded subject position using functional identification.

Thus it seems necessary to restrict the Contraction rule 'C' in some way. In order to capture the Control phenomena described here it suffices if Contraction is restricted to f-formulae of type e. The actual formulation of an appropriate restriction on 'C' involves empirical questions about linguistic phenomena (for example, can the quirky case marking phenomenon discussed below appear with Control constructions as well as Raising?), and is not attempted here.[15]

6. Agreement in a Resource Sensitive Setting

The previous sections have sketched how the LFG analyses of Raising and Control can be re-expressed in a resource-conscious setting.

[13] This is not strictly necessary, however. Carpenter (1996) proposes an analysis of Raising which, in R-LFG terms, involves the functional identification of objects of only quantifier type. General type-raising mechanisms are used to lift subjects of type e to this quantifier type if necessary.

[14] I thank Mary Dalrymple for drawing my attention to this contrast.

[15] Kehler et al. (1995) discuss a related problem that arises in the Linear Logic "glue language" approach of Dalrymple and colleagues. The primary problem they discuss is the duplication of resources in coordination constructions. While coordination is not discussed in the present paper, it seems that the treatment of coordination and feature dependencies offered in Bayer and Johnson (1995) generalizes unproblematically to R-LFG.

This section explores whether the treatment of features can also be recast in such a resource-conscious manner. Bayer and Johnson (1995) argue that such an approach is empirically superior to standard feature structure "unification-based" treatments of agreement in its account of certain coordination constructions, and here we argue that it also provides a straightforward account of the Icelandic quirky case marking construction.

This way of treating agreement is conceptually independent from the previous ones, in that one could combine the resource-based account of grammatical functions described above with a standard "unification-based" account of feature structure constraints. Carpenter (1996) and Morrill (1994) discuss polymorphic extensions to the type system of a categorial grammar that encode features in much the way a DCG does, and Dörre, König, and Gabbay (1996) show how to fibre a feature-structure system into a Lambek Categorial Grammar. It seems that either of these approaches could be adapted to R-LFG fairly easily.

With a resource logic we describe agreement phenomena in terms of assignment and discharge of features.[16] Informally, the idea is that an argument may need to be assigned certain features in order to "unlock" its value. There are many ways to formalize such an intuition, but one of the simplest is to assign such arguments an f-formula that is an implication from the agreement features they need to their "natural" semantic type. For example, an accusative noun phrase could be assigned the f-formula $acc \multimap e$, i.e., an object which when it receives an acc resource yields an object of type e. (Here acc is an atomic f-formula in $\mathcal{A}$).

Leaving aside the lambda term labelling for the moment, we can formalize in R-LFG the standard intuition that subject noun phrases are assigned "structural" nominative case, i.e., nominative case by virtue of their syntactic configuration, while object noun phrases are assigned case "lexically" by the verb, using the following lexical entries and syntactic rules.

The general idea is as follows. A case resource is consumed by a noun phrase marked with that case. These case resources are introduced by the f-terms associated with syntactic rules and lexical entries. A "structural" case is a case resource introduced by a syntactic rule while a "lexical" case is a case resource introduced by a lexical entry.

[16] This is in fact a standard way of speaking of agreement among linguists.

$$\begin{tikzcd}
 & \text{OBJ}\ (acc \multimap e) \multimap (\text{SUBJ}\ e \multimap t) \arrow[dl, "\text{SUBJ}"'] \arrow[dr, "\text{OBJ}"] & \\
nom, nom \multimap e & & acc \multimap e
\end{tikzcd}$$

Figure 11. A depiction of the f-term (34) labelling the c-structure for *she saw him*

she : NP : $nom \multimap e$
he : NP : $nom \multimap e$
her : NP : $acc \multimap e$
him : NP : $acc \multimap e$
likes : V : $\text{OBJ}\ (acc \multimap e) \multimap (\text{SUBJ}\ e \multimap t)$

$$\text{S} \to \underset{\text{SUBJ}\ (nom \otimes \downarrow)}{\text{NP}}\ \text{VP} \tag{32}$$

$$\text{VP} \to \text{V} \left(\underset{\text{OBJ}\ \downarrow}{\text{NP}}\right) \left(\underset{\text{XCOMP}\ \downarrow}{\text{VP}}\right) \tag{33}$$

These lexical entries and phrase structure rules can be used in a c-structure proof labelled with the phonological form *she likes him* and the f-term (34).

$$\begin{pmatrix} (nom, nom \multimap e)^{\text{SUBJ}}, \\ \text{OBJ}\ (acc \multimap e) \multimap (\text{SUBJ}\ e \multimap t), \\ (acc \multimap e)^{\text{OBJ}} \end{pmatrix}^{\otimes} \tag{34}$$

(Recall that under the "syntactic sugaring" convention of subsection 3.2 $(nom, nom \multimap e)^{\text{SUBJ}}$ abbreviates $((nom, nom \multimap e)^{\otimes})^{\text{SUBJ}})$. Figure 11 depicts the f-term (34), and the following proof shows that this term derives the saturated type t.

$$\dfrac{\dfrac{\dfrac{}{acc \multimap e \vdash acc \multimap e}\,\text{Ax}}{(acc \multimap e)^{\text{OBJ}} \vdash \text{OBJ}\ (acc \multimap e)}\,\Diamond\text{R} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{nom \vdash nom}\,\text{Ax} \quad \dfrac{}{e \vdash e}\,\text{Ax}}{nom, nom \multimap e \vdash e}\,\multimap\text{L}}{(nom, nom \multimap e)^{\text{SUBJ}} \vdash \text{SUBJ}\ e}\,\Diamond\text{R} \quad \dfrac{}{t \vdash t}\,\text{Ax}}{(nom, nom \multimap e)^{\text{SUBJ}}, \text{SUBJ}\ e \multimap t \vdash t}\,\multimap\text{L}}{(nom, nom \multimap e)^{\text{SUBJ}}, \text{OBJ}\ (acc \multimap e) \multimap (\text{SUBJ}\ e \multimap t), (acc \multimap e)^{\text{OBJ}} \vdash t}\,\multimap\text{L}$$

By contrast, consider the ungrammatical utterance *him likes her.* There is a c-structure proof associating this phonological form with the f-term (35). This f-term is the same as (34) except that underlined atom *acc* replaces an atom *nom*.

$$\begin{pmatrix} (nom, \underline{acc} \multimap e)^{\text{SUBJ}}, \\ \text{OBJ}\ (acc \multimap e) \multimap (\text{SUBJ}\ e \multimap t), \\ (acc \multimap e)^{\text{OBJ}} \end{pmatrix}^{\otimes} \tag{35}$$

Unlike (34), there is no proof of (35); the following partial proof shows where a proof corresponding to the one for (34) fails.

$$\dfrac{\dfrac{\dfrac{}{acc \multimap e \vdash acc \multimap e}\,\text{Ax}}{(acc \multimap e)^{\text{OBJ}} \vdash \text{OBJ}\ (acc \multimap e)}\,\Diamond\text{R} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\text{FAIL}}{nom \vdash \underline{acc}} \quad \dfrac{}{e \vdash e}\,\text{Ax}}{nom, \underline{acc} \multimap e \vdash e}\,\multimap\text{L}}{(nom, \underline{acc} \multimap e)^{\text{SUBJ}} \vdash \text{SUBJ}\ e}\,\Diamond\text{R} \quad \dfrac{}{t \vdash t}\,\text{Ax}}{(nom, \underline{acc} \multimap e)^{\text{SUBJ}}, \text{SUBJ}\ e \multimap t \vdash t}\,\multimap\text{L}}{(nom, \underline{acc} \multimap e)^{\text{SUBJ}}, \text{OBJ}\ (acc \multimap e) \multimap (\text{SUBJ}\ e \multimap t), (acc \multimap e)^{\text{OBJ}} \vdash t}\,\multimap\text{L}$$

Now consider the semantic labelling of such f-formulae. The problem is that this method of treating agreement assigns a higher type that encodes agreement features as well as the type of the semantic interpretation. It may be plausible to treat some agreement features (e.g., number features) as having a semantic dimension that is reflected in the type system, but it seems that some features (e.g., case features in English) are essentially semantically neutral, and should be semantically impotent.[17]

There are several ways in which semantically impotent features such as *nom*, *acc*, etc., might be handled. One straightforward but perhaps unsatisfying treatment interprets these types as ranging over singelton sets, so that the domain of objects of type $nom \multimap e$ is isomorphic to the domain of objects of type e.

Alternatively, the type system could be extended to include *semantically impotent* connectives in the manner described by Morrill (1994). (The modal operators are already treated as semantically impotent in R-LFG, and we could add a semantically impotent implication '$\multimap\!\!\bullet$' for which the f-formula to type mapping is given by $(A \multimap\!\!\bullet B)^{\natural} = B^{\natural}$).

Finally, inspired by recent work on a Morphological projection in LFG (Butt, Fortmann, and Rohrer, 1996), we might decide to provide agreement with its own proof system, possibly with the formula generated by an additional labelling on either the c-structure or the f-term proof system.

[17] This is not to say that such features are without semantic importance. Such features can certainly influence the structure, or even the existence, of a proof, and hence determine the readings available for a sentence.

Thus it seems that there are ways for resolving the tension between an f-formula assignment that encodes agreement features and the types of the lambda terms labelling the f-formulae.

We close this section with a discussion of quirky case marking in Icelandic. This example shows how the treatment of agreement interacts with the analysis of Raising constructions discussed earlier, and shows how a resource sensitive treatment of agreement can describe phenomena that seem to require extensions to the basic feature structure machinery.

This discussion is based on the LFG analysis presented in Andrews (1982), who describes this phenomenon using LFG's non-monotonic constraint equations '$\cdot =_c \cdot$', and the HPSG analysis suggested in Sag (1995), who use a multiple inheritance hierarchy to capture the same phenomena. Neither of these analyses can be stated in a pure "unification" framework, and indeed seem to require very substantial extensions to that basic unification feature machinery. By contrast, the analysis presented here is essentially a direct translation into R-LFG of the LFG analysis developed by Andrews (1982), and uses the basic R-LFG machinery already presented.

In Icelandic, most subject noun phrases are marked with nominative case, as in (36a). However, the subjects of so-called "quirky case marking" verbs are marked with a non-nominative case determined by the verb. In (36b) the subject *drengina* appears in accusative case because it is the subject of the quirky case marking verb *vanta*.

(36) a. *drengurinn kyssti stúlkuna*
the-boy.nom kissed the-girl.acc
'The boy kissed the girl'

b. *drengina vantar mat*
the-boys.acc lacks food.acc
'The boys lack food'

Interestingly, the same kind of alternation appears in related Raising constructions. If the complement verb is a quirky case marker, then the matrix subject is marked with the case assigned by that verb (37b), otherwise if the complement verb is non-quirky the matrix subject appears in nominative case (37a).

(37) a. *hann virðist elska hana*
he.nom seems love her.acc
'He seems to love her'

b. *hana virðist vanta peninga*
her.acc seems lack money.acc
'She seems to lack money'

Andrews (1982) captures this in LFG by making the structural nominative case assignment *optional* in the Icelandic version of the syntactic rule (32). Quirky case marking verbs lexically assign case to their subjects, in much the same way that transitive verbs do to their objects. He then uses a constraint equation to ensure that all noun phrases are assigned the particular case they exhibit. For instance, the lexical entry for the accusative subject noun phrase in (36b) would include the constraint equation $(\uparrow \textsc{case}) =_c \textsc{acc}$, which requires that some other lexical entry or syntactic rule set the value of the CASE feature to ACC. (Such constraint equations are non-monotonic since adding an additional constraint to a set of constraints may satisfy an otherwise unsatisfied constraint equation in that set).

Note that in this LFG analysis it is crucial that noun phrases are associated with such constraint equations. If these constraint equations were replaced with ordinary attribute-value constraints then subjects of non-quirky verbs could appear with any case whatsoever (as structural nominative case is only optionally assigned).

The behaviour of quirky case marking verbs in Raising constructions falls out immediately in this analysis. Because the complement's subject is functionally identified with the matrix subject, a quirky case marking complement verb can assign case directly to the matrix subject. If the complement verb is non-quirky, the matrix subject receives structural nominative case as before.

As it stands, R-LFG has no way to directly express optional features. Strictly speaking no extension is necessary, as we could always interpret lexical entries or syntactic rules containing such features as abbreviations for a set of lexical entries or syntactic rules that contain no optional features, as Kaplan and Bresnan (1982) do. For example, we could implement the optional structural nominative case assignment in Icelandic by adding another phrase structure rule to our grammar just like (32) except that it lacks the nominative case atom *nom*.

However, it is straightforward to enrich the f-formula language with connectives that allow us to express such optionality directly. One way of doing this is to introduce the additive connective '&' and the identity 1 from Linear Logic (Girard, 1995).[18] We extend the f-formula language defined in (1) as follows:

[18] While not discussed in this paper, it seems that the dual additive connective '$\oplus$' has linguistic applications as well. For example, an NP which can appear in either nominative or accusative case, such as the English proper name *Kim*, could be assigned the lexical entry $(nom \oplus acc) \multimap e$. Indeed, the connectives '$\vee$' and

$$\mathcal{F} \;\rightarrow\; \mathcal{A} \mid (\mathcal{F} \multimap \mathcal{F}) \mid (\mathcal{F} \otimes \mathcal{F}) \mid (\mathcal{G}\,\mathcal{F}) \mid (\mathcal{F} \,\&\, \mathcal{F}) \mid 1 \qquad (38)$$

The corresponding sequent calculus proof rules for these connectives are as follows.

$$\frac{\Gamma[B] \vdash A}{\Gamma[B \,\&\, C] \vdash A}\,\&\mathrm{L}_1 \qquad \frac{\Gamma[C] \vdash A}{\Gamma[B \,\&\, C] \vdash A}\,\&\mathrm{L}_2 \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \,\&\, B}\,\&\mathrm{R}$$

$$\frac{\Gamma[\Delta] \vdash A}{\Gamma[(1, \Delta)^{\otimes}] \vdash A}\,1\mathrm{L} \qquad \frac{}{\vdash 1}\,1\mathrm{R}$$

With this extension we can express an *optional* resource of type A as the f-formula $A \,\&\, 1$. The rule $\&\mathrm{L}_1$ simplifies $A \,\&\, 1$ to A, while rule $\&\mathrm{L}_2$ simplifies $A \& 1$ to 1, which in turn can be deleted by rule 1L. (The right rules are not used in the linguistic applications here).

The syntactic rule in (32) modified to make structural nominative case assignment optional is shown in (39). Sample lexical entries for the nominative noun phrase *drengurinn* 'the boy' and its accusative plural form *drengina*, as well as the "normal" transitive verb *kyssti* 'kissed', the quirky case marking transitive verb *vantar* 'lacks' and the Raising verb *virðist* 'seems', are given below.

drengurinn : NP : $nom \multimap e$

drengina : NP : $acc \multimap e$

kyssti : V : $\textsc{obj}\ (acc \multimap e) \multimap (\textsc{subj}\ e \multimap t)$

vantar : V : $\textsc{obj}\ (acc \multimap e) \multimap (\textsc{subj}\ (acc \multimap e) \multimap t)$

virðist : V : $(\textsc{xcomp}\ t \multimap t, \textsc{subj} = \textsc{xcomp}\ \textsc{subj})^{\otimes}$

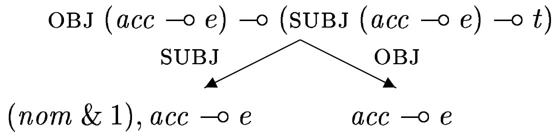
$$\mathrm{S} \rightarrow \underset{\textsc{subj}\ ((nom\ \ \&\ 1) \otimes \downarrow)}{\mathrm{NP}} \ \mathrm{VP} \qquad (39)$$

Using rule (39) with the VP rule (33) above and appropriate lexical entries, there is a c-structure proof labelled with the phonological form (36b) *drengina vantar mat* 'the boys lack food' and the f-term (40).

$$\begin{pmatrix} ((nom \,\&\, 1), acc \multimap e)^{\textsc{subj}}, \\ \textsc{obj}\ (acc \multimap e) \multimap (\textsc{subj}\ (acc \multimap e) \multimap t), \\ (acc \multimap e)^{\textsc{obj}} \end{pmatrix}^{\otimes} \qquad (40)$$

The f-term (40) is depicted in Figure 12. The subject has been assigned an optional nominative case structurally by the syntactic rule

'∧' used in Bayer and Johnson (1995) and Johnson and Bayer (1995) correspond precisely to the additives '⊕' and '&' respectively.



$$\begin{tikzcd}
& \text{OBJ}\,(acc \multimap e) \multimap (\text{SUBJ}\,(acc \multimap e) \multimap t) \arrow[dl, "\text{SUBJ}"'] \arrow[dr, "\text{OBJ}"] & \\
(nom \,\&\, 1), acc \multimap e & & acc \multimap e
\end{tikzcd}$$

Figure 12. A depiction of the f-term (40) associated with the quirky accusative Icelandic example (36b).

(39), but the quirky verb subcategorizes for a subject that it assigns accusative case to. A proof that (40) can derive the saturated type t is given below. In that proof the optional nominative case is deleted, and the subject receives accusative case from the verb. In this proof, QTV abbreviates $\text{OBJ}\ (acc \multimap e) \multimap (\text{SUBJ}\ (acc \multimap e) \multimap t)$ (the f-formula labelling the quirky transitive verb *vantar* 'lacks') and Γ abbreviates the f-term (40).

$$\dfrac{\dfrac{\dfrac{\dfrac{}{acc \multimap e \vdash acc \multimap e}\,\text{Ax}}{(acc \multimap e)^{\text{OBJ}} \vdash \text{OBJ}\ (acc \multimap e)}\,\Diamond\text{R} \qquad \dfrac{\dfrac{\dfrac{}{acc \multimap e \vdash acc \multimap e}\,\text{Ax}}{(acc \multimap e)^{\text{SUBJ}} \vdash \text{SUBJ}\ (acc \multimap e)}\,\Diamond\text{R} \quad \dfrac{}{t \vdash t}\,\text{Ax}}{(acc \multimap e)^{\text{SUBJ}}, \text{SUBJ}\ (acc \multimap e) \multimap t \vdash t}\,\multimap\text{L}}{\dfrac{(acc \multimap e)^{\text{SUBJ}}, QTV, (acc \multimap e)^{\text{OBJ}} \vdash t}{(1, acc \multimap e)^{\text{SUBJ}}, QTV, (acc \multimap e)^{\text{OBJ}} \vdash t}\,1\text{L}}\,\multimap\text{L}}{\Gamma \vdash t}\,\&\text{L}_2$$

Suppose the subject noun phrase in (36b) had been nominative case marked instead of accusative case marked. Then it would have been labelled with the f-formula $nom \multimap e$. The optional nominative case available structurally could satisfy its need for a nominative case, but the resulting object would be of type e and hence incompatible with the quirky verb, which requires a subject of type $acc \multimap e$. Thus an accusative case marked subject is required with a quirk accusative case assigning verb.

Similarly, suppose that the verb were a normal transitive verb as in (36a), rather than a quirky case assigner. Then the optional nominative case introduced by the syntactic rule (32) is the only possible case resource available to the subject. Since every noun phrase consumes a case resource, the subject must be marked with nominative case.

Now we turn to the interaction of Raising and quirky case marking. It is straightforward to show that there is a c-structure proof labelled with the phonological form (37b) *hana virðist vanta peninga* 'she seems to lack money' and the f-term in (41).

$$\begin{tikzcd}
& \mathrm{XCOMP}\ t \multimap t \arrow[dl, "\mathrm{SUBJ}"'] \arrow[dr, "\mathrm{XCOMP}"] & & \\
(nom \,\&\, 1), acc \multimap e \arrow[dr, dashed] & & \mathrm{OBJ}\ (acc \multimap e) \multimap (\mathrm{SUBJ}\ (acc \multimap e) \multimap t) \arrow[dl, "\mathrm{SUBJ}"] \arrow[dr, "\mathrm{OBJ}"] & \\
& {} & & acc \multimap e
\end{tikzcd}$$

Figure 13. A depiction of the f-term (41) associated with the quirky accusative Icelandic example (37b).

$$\left(\begin{array}{l} ((nom \,\&\, 1), acc \multimap e)^{\mathrm{SUBJ}}, \\ \mathrm{XCOMP}\ t \multimap t,\ \mathrm{SUBJ} = \mathrm{XCOMP\ SUBJ}, \\ \left(\begin{array}{l} \mathrm{OBJ}\ (acc \multimap e) \multimap (\mathrm{SUBJ}\ (acc \multimap e) \multimap t), \\ (acc \multimap e)^{\mathrm{OBJ}} \end{array}\right)^{\mathrm{XCOMP}} \end{array}\right)^{\otimes} \tag{41}$$

The f-term (41) is depicted in Figure 13. The proof that this term derives the saturated type t is given below. In that proof QVT abbreviates the f-formula labelling the quirky transitive verb as above, RV abbreviates the f-formula $\mathrm{XCOMP}\ t \multimap t$, $\mathrm{SUBJ} = \mathrm{XCOMP\ SUBJ}$ labelling the Raising verb *virðist* 'seems', and Δ abbreviates the f-term (41).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{acc \multimap e \vdash acc \multimap e}\,\mathrm{Ax}}{(acc \multimap e)^{\mathrm{OBJ}} \vdash \mathrm{OBJ}\ (acc \multimap e)}\,\Diamond\mathrm{R} \quad \dfrac{\dfrac{\dfrac{}{acc \multimap e \vdash acc \multimap e}\,\mathrm{Ax}}{(acc \multimap e)^{\mathrm{SUBJ}} \vdash \mathrm{SUBJ}\ (acc \multimap e)}\,\Diamond\mathrm{R} \quad \dfrac{\dfrac{\dfrac{}{t \vdash t}\,\mathrm{Ax}}{(t)^{\mathrm{XCOMP}} \vdash \mathrm{XCOMP}\ t}\,\Diamond\mathrm{R} \quad \dfrac{}{t \vdash t}\,\mathrm{Ax}}{(t)^{\mathrm{XCOMP}}, \mathrm{XCOMP}\ t \multimap t \vdash t}\multimap\mathrm{L}}{((acc \multimap e)^{\mathrm{SUBJ}}, \mathrm{SUBJ}\ (acc \multimap e) \multimap t)^{\mathrm{XCOMP}}, \mathrm{XCOMP}\ t \multimap t \vdash t}\multimap\mathrm{L}}{((acc \multimap e)^{\mathrm{SUBJ}}, QTV, (acc \multimap e)^{\mathrm{OBJ}})^{\mathrm{XCOMP}}, \mathrm{XCOMP}\ t \multimap t \vdash t}\multimap\mathrm{L}}{((acc \multimap e)^{\mathrm{SUBJ}})^{\mathrm{XCOMP}}, (QTV, (acc \multimap e)^{\mathrm{OBJ}})^{\mathrm{XCOMP}}, \mathrm{XCOMP}\ t \multimap t \vdash t}\,\mathrm{F}}{(acc \multimap e)^{\mathrm{SUBJ}}, RV, (QTV, (acc \multimap e)^{\mathrm{OBJ}})^{\mathrm{XCOMP}} \vdash t}\,=}{\dfrac{(1, acc \multimap e)^{\mathrm{SUBJ}}, RV, (QTV, (acc \multimap e)^{\mathrm{OBJ}})^{\mathrm{XCOMP}} \vdash t}{\Delta \vdash t}\,\&\mathrm{L}_2}\,1\mathrm{L}$$

In this proof, the rules '&L$_2$' and '1L' are used to delete the optional nominative case assigned structurally to the subject. Then the functional identification associated with the Raising verb and rule 'F' are used to lower subject, still with its case marking requirement unfullfilled, to the complement's subject. From then on the proof resembles the one for the monoclausal quirky case example discussed above. By a similar argument to the one given for the monoclausal examples, one can see that if the complement contains a quirky case marking verb, then the subject must consume the case that it assigns, but that if the complement contains a normal verb then the subject will receive the nominative case marking optionally assigned structurally by syntactic rule (32).

Notice that this analysis depends crucially on the ability of functional identification to "move" f-formulae other than e from one modal context to another. This means that we could not describe the required functional identification in Icelandic Raising constructions by a single non-polymorphic f-formula the way that we could for the corresponding English constructions in using a lexical entry (24) for *seems* that "hardwired" a single type of functional identification.

7. Conclusion

We have seen that some of the basic linguistic analyses standardly formulated using feature structure constraints in LFG and other linguistic theories can be stated in a resource sensitive substructural logic. At least some examples of linguistic phenomena such as agreement can be expressed in terms of resource production and consumption in a way that corresponds to linguists' intuitions about feature assignment and checking, and this treatment extends straightforwardly to examples, such as Icelandic quirky case marking, that are difficult to account for using basic "unification" constraints.

The resulting system is simpler than the standard LFG model because the f-structure component and the Linear Logic "glue language" component have been merged. Perhaps just as interestingly, this has forced us to develop a novel treatment of features as resources to be produced and consumed, rather than as objects determined by the simultaneous interaction of constraints. Informally, a purely syntactic feature such as *acc* is "produced" by a f-formula of the shape *acc*, its presence is "checked" by a formula of the shape $acc \multimap acc$, and the feature is ultimately "checked off" or "consumed" by a formula of the shape $acc \multimap \ldots$.

In a resource based system such as the present one, the identification of the appropriate modes of resource control or sensitivity becomes perhaps the central question. We raised this issue in the discussion of Contraction above, and exactly the same questions arise with respect to agreement features as well.

For example, in standard LFG analyses a purely syntactic feature such as CASE is typically "set" with one or more "defining" equations, such as $(\uparrow \text{CASE}) = acc$, and "checked" by zero or more "constraint" equations, such as $(\uparrow \text{CASE}) =_c acc$. This "set once or more times, check zero or more times" mode of structural sensitivity is captured by Intuitionistic Logic, and can be obtained in R-LFG by adding the Contraction rule discussed above and an additional rule of Weakening (Girard, Lafont, and Taylor, 1989).

$$\frac{\Gamma[\Delta] \vdash \alpha : A}{\Gamma[(\beta : B, \Delta)^{\otimes}] \vdash \alpha : A}\ \mathrm{W}$$

However, in the context of R-LFG the addition of a Weakening rule seems to be going too far, even if the rule were restricted to semantically vacuous f-formulae such as *acc*.

Standard LFG uses feature structure inconsistency to prevent two different feature values being assigned to the same element, but R-LFG does not contain a directly comparable consistency requirement. (Indeed, some of the analyses presented in Bayer and Johnson (1995) require that a single constituent bear two distinct feature values). Instead, the resource accounting mechanisms of R-LFG require that every feature assigned to a constituent be consumed, so "unwanted" features are never consumed, and hence lead to ungrammaticality. Thus in a sense, the feature consistency requirements of standard unification grammars are supplanted by resource accounting in R-LFG. But the addition of Contraction and Weakening sufficiently reduces the structural sensitivity of the logic to the extent that such a system would permit analyses that we intend to rule out. For example, it is straightforward to show that with unrestricted Contraction and Weakening the Icelandic fragment presented above could derive the ungrammatical example (42).

(42)	*	*hann*	*virðist*	*vanta*	*peninga*
		he.nom	seems	lack	money.acc

'He seems to lack money

Thus it seems as if we should ban Weakening from R-LFG. On the other hand, perhaps one could make a stronger case for Contraction to apply to semantically vacuous f-formulae such as *acc*. On the face of it, it seems unreasonable to be able to speak of a noun phrase having "two" accusative case features. Still, the issues are tricky, and will have to await further empirical linguistic investigation.

While it is perhaps surprising that we can formulate such a wide range of LFG analyses in resource sensitive terms, it is important to recognize that we have only accounted for a fraction of LFG's linguistic analyses and the corresponding formal devices used in them. Major linguistic topics covered in LFG but not discussed here are the following:

– modification and adjunction, analysed using set-values in LFG (Kaplan and Bresnan, 1982),

– coordination, analysed using set-values in LFG (Kaplan and Maxwell III, 1988),

– long-distance dependencies, analysed using functional uncertainty in LFG (Kaplan and Zaenen, 1995),

– anaphoric dependencies, analysed using "inside-out" functional uncertainty in LFG (Dalrymple, 1993), and

– interactions between quantifier scope and anaphora, analysed using quantification over f-structure elements in a linear logic framework (Dalrymple et al., 1995a).

As it stands, R-LFG has no account for these phenomena and it seems that R-LFG will need to be extended in order to formulate these analyses. Specifically, LFG's functional uncertainty, expressed in terms of regular expressions over grammatical functions, seems to involve the same kind of generalized modal operators found in Dynamic Logic. The ability to move "backwards" out of a modal context also seems to be important in order to express the Dalrymple (1993) account of anaphoric dependencies and possibly also adjunction.[19] But devising an appropriate resource sensitive dynamic logic is research that still remains to be done.

Obtaining an adequate treatment of coordination, especially non-consitituent coordination, will probably require a tighter integration of the c-structure construction process with the f-formula proofs. Note that in non-constituent coordination examples the uniformity seems to exist at the c-structure, rather than at the f-term level. For example, in (43) the coordinated partial constituents have dissimilar f-terms (the left one is $\textsc{obj}\ e \multimap (\textsc{subj}\ e \multimap t)$, while the right one is $\textsc{xcomp}\ \textsc{obj}\ e \multimap (\textsc{subj}\ e \multimap t)$, yet both have the same c-structure type, namely $\textsc{vp}/\textsc{np}$.

(43) Kim [liked] and [wanted to eat] the tasty new cheese.

I close this paper with some remarks of a highly speculative nature concerning the computational properties of R-LFG.

In most "unification-based" grammars, including most LFGs, it seems that often the feature structure constraints can be divided into two classes.

First, some feature structure constraints are used to *check* the consistency of the values assigned to features ranging over a small, finite sets of constants (e.g., person, number and case features). Such constraint

[19] If grammatical functions were formalized as '$[\cdot]^{\downarrow}$' modalities as in footnote 9, then the '$\langle\cdot\rangle$' modality would behave as an "inside-out" grammatical relation.

are "meant" to fail in some cases: if there were no possible inputs that yielded unsatisfiable constraints over such a feature, then there would be no reason to include this feature in the grammar.

Second, some feature constraints are used to *construct* a structure that will serve as the input to semantic interpretation. Unlike the previous kind of constraint, such constraints need never be inconsistent with any other constraints in the grammar. The feature values subject to such constraints tend to be unboundedly large, recursive feature structures.

It seems to me that the inability to distinguish these two kinds of constraints is the source of much of the computational complexity in unification grammar parsing. Specifically, the copying needed in unification parsing is required because of the possibility that the resulting feature structure will be used in "checking" mode: if we could guarantee that further unifications would only incorporate this feature structure into larger structures and never "unify into" this structure then we would not need to copy.

Efficient implementations of unification grammar parsers, such as the one described by Maxwell III and Kaplan (1995), seem to detect and "special-case" these two roles, and they run into problems when faced with "zipper unification" (as in the Bresnan et al. (1982) analysis of Dutch Cross Serial Dependencies) where a single unification is used to perform both roles at once.

Now, in resource sensitive frameworks such as R-LFG these two roles are separated. The lambda term labelling the f-term proofs records the semantically relevant combinatorial structure of the utterance, and no copying is required during its construction. The f-term records the requirements of lexical items and syntactic rules that still have to be fullfilled; as soon as they have been satisfied they are deleted from the f-term. Thus grammars formulated in resource sensitive frameworks in which semantic interpretation is provided by a Curry-Howard mapping might have simpler efficient parsing algorithms than corresponding grammars formulated in standard unification systems.

However, a significant amount of work still remains to be done here. The Gentzen system provided here does provide a decision procedure for f-terms, but it is not very efficient. The validity problem for propositional linear logic is undecidable, and is NP-complete even for the purely multiplicative fragment. Thus it seems that efficient parsers for resource sensitive grammars will have to exploit the properties of the particular systems that arise in natural language.

References

Andrews, Avery D. 1982. The representation of Case in modern Icelandic. In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations.* The MIT Press, Cambridge, Massachusetts, pages 427–502.

Bayer, Samuel and Mark Johnson. 1995. Features and agreement. In *The Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 70–76, San Francisco. Morgan Kaufmann.

Bresnan, Joan. 1982. Control and complementation. In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations.* The MIT Press, Cambridge, Massachusetts, pages 282–390.

Bresnan, Joan, Ronald M. Kaplan, Stanley Peters, and Annie Zaenen. 1982. Cross-serial dependencies in dutch. *Linguistic Inquiry*, 13:613–635.

Butt, Miriam, Christian Fortmann, and Christian Rohrer. 1996. Syntactic analyses for parallel grammars: Auxiliaries and genitive NPs. In *Proceedings of COLING 1996.*

Carpenter, Bob. 1996. *Lectures on Type-Logical Semantics.* The MIT Press, Cambridge, Massachusetts.

Chierchia, Gennaro. 1984. *Topics in the Syntax and Semantics of Infinitives and Gerunds.* Ph.D. thesis, University of Massachusetts, Ahmerst.

Dalrymple, Mary. 1993. *The Syntax of Anaphoric Binding.* CSLI Publications, Stanford, CA. CSLI Lecture Notes, number 36.

Dalrymple, Mary, John Lamping, Fernando Pereira, and Vijay Saraswat. 1995a. Linear logic for meaning assembly. In Glyn V. Morrill and Richard T. Oehrle, editors, *Formal Grammar: Proceedings of the Conference of the European Summer School in Logic, Language and Information*, pages 75–93, Barcelona, Spain.

Dalrymple, Mary, John Lamping, Fernando C. N. Pereira, and Vijay Saraswat. 1995b. Linear logic for meaning assembly. In *Proceedings of CLNLP*, Edinburgh.

Dalrymple, Mary, John Lamping, Fernando C. N. Pereira, and Vijay Saraswat. 1996a. A deductive account of quantification in LFG. In Makoto Kanazawa, Christopher J. Piñón, and Henriette de Swart, editors, *Quantifiers, Deduction, and Context.* CSLI Publications, Stanford, CA.

Dalrymple, Mary, John Lamping, Fernando C. N. Pereira, and Vijay Saraswat. 1996b. Intensional verbs without type-raising or lexical ambiguity. In Jerry Seligman and Dag Westerståhl, editors, *Logic, Language and Computation.* Center for the Study of Language and Information, Stanford, California, pages 167–182. Also in Proceedings of the Conference on Information-Oriented Approaches to Logic, Language and Computation/Fourth Conference on Situation Theory and its Applications, Saint Mary's College of California, Moraga, California. June 1994.

Dalrymple, Mary, John Lamping, Fernando C. N. Pereira, and Vijay Saraswat. 1996c. Quantifiers, anaphora, and intensionality. *Journal of Logic, Language, and Information*, to appear.

Dörre, Jochen, Esther König, and Dov Gabbay. 1996 Fibred semantics for feature-based grammar logic. *Journal of Logic, Language and Information.* 5:3–4.

Dowty, David R. 1985. On recent analyses of the semantics of Control. *Linguistics and Philosophy*, 8(3):291–332.

Gabbay, Dov M. 1994. Classical vs non-classical logics (the universality of classical logic). In Dov M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 2: Deduction Methodologies. Oxford University Press, Oxford, pages 359–500.

Girard, Jean-Yves. 1995. Linear Logic: Its syntax and semantics. In Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors, *Advances in Linear Logic.* Cambridge University Press, Cambridge, England, pages 1–42.

Girard, Jean-Yves, Yves Lafont, and Paul Taylor. 1989. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge, England.

Jacobson, Pauline. 1990. Raising as function composition. *Linguistics and Philosophy*, 13(4):423–457.

Johnson, Mark. 1988. *Attribute Value Logic and The Theory of Grammar*. Number 16 in CSLI Lecture Notes Series. Chicago University Press.

Johnson, Mark. 1991. Features and formulae. *Computational Linguistics*, 17(2):131–152.

Johnson, Mark. 1995. Logic and feature structures. In Mary Dalrymple and Ronald M. Kaplan, editors, *Formal Properties of Lexical-Functional Grammar*. CSLI Lecture Notes Series.

Johnson, Mark. 1997. Features as Resources in R-LFG. In *Proceedings of the 1997 LFG Conference*. CSLI Publications.

Johnson, Mark. to appear. Type-driven semantic interpretation and feature dependencies in R-LFG. In Mary Dalrymple, editor, *Working with Glue: Resource Accounting and Semantic Interpretation in Lexical Functional Grammar*. The MIT Press.

Johnson, Mark and Samuel Bayer. 1995. Features and agreement in lambek categorial grammar. In Gyln V. Morrill and Richard T. Oehrle, editors, *Formal Grammar: Proceedings of the Conference of the European Summer School in Logic, Language and Information*, pages 123–137, Barcelona, Spain. Universitat Politècnica de Catalunya.

Kaplan, Ronald and John T. Maxwell III. 1988. Constituent coordination in Lexical-Functional Grammar. In *The Proceedings of the 12th International Conference on Computational Linguistics*, page 297-302.

Kaplan, Ronald M. and Joan Bresnan. 1982. Lexical-Functional grammar: A formal system for grammatical representation. In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations*. The MIT Press, chapter 4, pages 173–281.

Kaplan, Ronald M. and Annie Zaenen. 1995. Long-distance dependencies, constituent structure and functional uncertainty. In Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell III, and Annie Zaenen, editors, *Formal Issues in Lexical-Functional Grammar*, number 47 in CSLI Lecture Notes Series. CSLI Publications, chapter 3, pages 137–165.

Kasper, Robert T. and William C. Rounds. 1990. The logic of unification in grammar. *Linguistics and Philosophy*, 13(1):35–58.

Kehler, Andrew, Mary Dalrymple, John Lamping, and Vijay Saraswat. 1995. The semantics of resource-sharing in Lexical-Functional Grammar. In *Proceedings of the 7th Meeting of the European Association for Computational Linguistics*, University College Dublin, March.

König, Esther. 1995. Lexical Functional Grammars and Lexical Grammars. Technical report, Institute for Computational Linguistics, Universität Stuttgart.

Maxwell III, John T. and Ronald M. Kaplan. 1995. The interface between phrasal and functional constraints. In Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell III, and Annie Zaenen, editors, *Formal Issues in Lexical-Functional Grammar*, number 47 in CSLI Lecture Notes Series. CSLI Publications, chapter 15, pages 403–430.

Moortgat, Michael. 1997. Categorial type logics. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Language*. The MIT Press, Cambridge, Massachusetts, pages 93–178.

Morrill, Glyn V. 1994. *Type-logical Grammar: Categorial Logic of Signs*. Kluwer Academic Publishers, Dordrecht.

Rounds, William C. 1997. Feature logics. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Language*. The MIT Press, Cambridge, Massachusetts, pages 475–533.

Sag, Ivan A. 1995. HPSG problem set 4: Icelandic case. Technical report, The Center for the Study of Language and Information. Available as http://hpsg.stanford.edu/hpsg/lecture-materials/pset4-icelandic.ps.

Shieber, Stuart M. 1986. *An Introduction to Unification-based Approaches to Grammar*. CSLI Lecture Notes Series. Chicago University Press, Chicago.

van Benthem, Johan. 1995. *Language in Action: Categories, Lambdas and Dynamic Logic*. The MIT Press, Cambridge, Massachusetts.

Address for correspondence:
Department of Cognitive and Linguistic Sciences, Box 1978
Brown University
Providence, RI 02912, USA
Email: mj@lx.cog.brown.edu