Interfaces and the Grammar

Mark Steedman
University of Edinburgh

1. Introduction

Among a number of common-language terms that are used by theoretical linguists in ways that are sometimes puzzling, the terms "interface" and "computation" have recently become salient.

The linguistic mapping between a signal, such as speech, and meaning is itself sometimes referred to as an "interface". However, most people think that both the speech wave and meaning are translated into discrete symbolic representations, such as strings of lexical items on the one hand, and either logical forms or model theoretic-representations on the other. Thus it is natural to think of these levels of representation as interfaces of the linguistic system as a whole to other extra-linguistic systems such as articulation and perception on the one hand, and the process of reasoning about the world on the other.

We then need a different word to refer to the mapping between the two interface levels. It is natural to think of this as a "computation." One of the principal contributions of the generative linguistic approach to the problem of characterizing the computation in this sense has been the identification of (competence) grammar as a component. There are obvious evolutionary and developmental reasons to expect this competence grammar-based computation to be rather direct, and in particular to eschew any further intermediate "hidden" interface levels of representation, particularly those of any kind not monotonically related to the two primary interface levels.

The theories presented in Chomsky 1957 and 1965 can be viewed as adhering to this parsimonious view of the grammar in spirit, once it is realized that most transformations are "structure-preserving" (Emonds 1976), and can therefore be "base-generated" (Bresnan 1978; Brame 1978), hence eliminating the need for intermediate levels of representation such as "Deep Structure". (The Augmented Transition Network grammar (ATN) of Woods 1973 was a clever contemporary realization of this idea in computational terms.) A great deal of subsequent work by linguists and non-linguists has clung to a view of grammar as a unitary monotonic syntax-semantics interface, including Gazdar's (1981) pioneering demonstration that the vast majority of syntactic phenomena could be captured with nothing more powerful than Context Free Grammar (CFG), and Montague's (1974) demonstration that meaning representation could be perspicuously captured in terms of intensional logic, using the λ-calculus as a "glue language" to build representations in lock-step with syntactic derivation.

Montague famously observed that in principle the level of logical form could be dispensed with in favor of direct computation via surface derivation on the model theory (although this only seems to work if the model theory itself is regarded as an interface level, distinct from the physical world). He urged that, even if logical form was in practice needed by mortals to keep track of the semantics, it should always be transparent to model theory—no interpretation without denotation. However, in term of both linguistic explanation, psychological plausibility (particularly in terms of the manifest possibility of evolution and acquisition of language), and practical computation, the use of logical forms appears to be essential. In that case, their eliminability in principle in favor of direct representation of the model theory does not seem a particularly helpful observation. In fact, we shall see that it is traditional surface syntactic structure, rather than logical form, that should be eliminated as a representational level.

In psychological and computational terms, the objection to eliminating logical form is that, while unsupervised probabilistic induction of grammars from unanalyzed text to any desired level of confidence is in principle possible, for grammars of the size and ambiguity characteristic of human language, such "unsupervised" grammar induction is of a computational complexity that makes it quite impracticable. The only successful wide coverage parsers, such as those of Collins (1997) and Charniak (2000), are based on "supervised" grammar induction from corpora annotated by human linguists with syntactic structures. Such structures contain all the information that is in the corresponding logical form, suggesting that children learn the same way on the basis of their (doubtless very different) meaning representations and data.

In linguistic terms, the point is strengthened by the existence of various "interface conditions," or ways in which the mapping from sound and meaning to PF and LF are non-straightforward. For example, in the former case, transducing sound into PF is complicated by processes of coarticulation including liason and lenition. In the latter case, logical forms that would otherwise be meaningful are excluded under mysterious generalizations like Condition C of the Binding Theory.

It is important to realize that all such interface conditions are anomalous. If they were a part of the grammar, they would merely make life more difficult than it need be for the child language learner. As such, they ought in evolutionary or developmental terms to have been eliminated, whether by changing the language to suit the representation, or by changing the representation to correspond more directly to the language. Instead, conditions like the above should probably be interpreted as evidence of something wrong in our assumptions, such as that phonemes are not in fact the phonologically primitive elements of the surface representation, or that logical form should be specified dynamically, rather than declaratively, as Kempson et al. (2001) have proposed.

However, a number of very salient linguistic phenomena make it very hard in practice to formulate such a minimalist theory of grammar. In particular, while most constructions obey a "Constituent Condition on Rules" that makes them readily compatible with direct mapping, others, like long-distance dependency, coordination, quantifier scope alternation, and (in English and other European languages) intonation, appear to require rules which either apply to or create non-constituents, in violation of that principle, as in the following examples:

(1) a. I introduced Frankie to Johnny, and Anna to Manny.
 b. Q: I know Anna dated Johnny. But who did she marry?
  A: (Anna MARRIED)(MANNY)
 c. Everyone loves someone.

(In (1a), coordination appears to apply to a non-constituent *Anna to Manny*. In (1b), in which capitals indicate stress and parentheses indicate intonational phrase boundaries, the intonational phrase *Anna married* is also not a traditional constituent. In (1c), there are two readings, only one of which reflects the dominance relations of traditional constituent structure, according to which the universal has scope over the existential.)

Later versions of transformational grammar (partly in response to the vagaries of such phenomena) allowed representational levels to proliferate (Chomsky 1981; Selkirk 1984), including besides the interface levels of phonological form (PF) and logical form (LF), such further levels of representation as D-Structure, S-structure, Intonation Structure, and Information Structure. However, Chomsky (1995b) has recently called for a "Minimalist" return to the two-interface-level model, suggesting that a convergence with the theories that never abandoned it is possible—a welcome move, as far as it goes.

However, it would be premature to announce a Grand Unified Theory. One recent presentation of the Minimalist Program offered the visualization in Figure 1, in which the generative process starts with a numeration or unordered multiset of lexical items. A number of tree-combining and tree-modifying operations including Merge and Move operate subject to criteria like "Shortest Move" (move to the structurally closest permitted position) and "Greed" (don't move unless you have to), which have effects much like the old-style transformational cycle, to recursively generate derivation(s) pairing PF and LF, starting with the elements of the numeration (Chomsky 1995b:225 *et seq.*) until the resources of the numeration are used up. At that point, PF undergoes "Spell-out" after which no further processes of movement can apply to it. However, further processes of "covert" movement can be applied to LF, to achieve well-formed logical forms for language with *in situ wh*-elements and processes like quantifier scope inversion. (In more recent versions of the theory, a notion of "phase" roughly corresponding to CP and NP does the work of criteria like Shortest Move, and Spell-out can take place on multiple occasions.)

It is perhaps worth pausing to ask what it means to call a system of this kind a "computation". The kind of computation it most closely resembles is the variety known as "constraint programming", usually thought of as a variety of logic programming. The relation is explicitly evoked in the work of

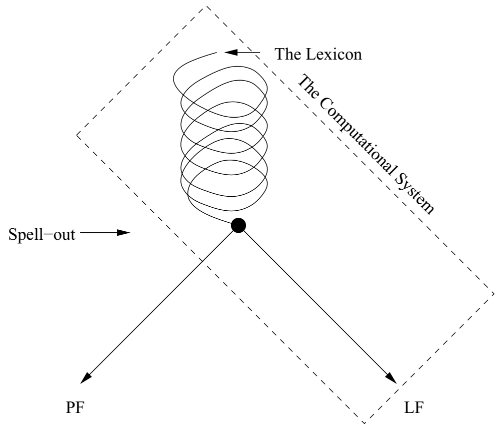


Figure 1: The Computational System according to the Minimalist Program (adapted from Marantz 1995:357).

Stabler (1992, and subsequent work). Constraint programming can produce very efficient solutions for certain classes of problem, such as the eight-queens problem and cryptarithmetic. However, there is no general algorithm for the efficient solution of constraint problems, so this computation is distinctly underspecified. Moreover, it is unclear what this computation characterizes. For example, it seems odd to think that an individual speaker of a given language would map sound onto meaning (or vice versa) by computing all possible moves available to all possible languages and then select among them according to the constraints. Nor is it clear that constraint programming is particularly well adapted to the task, as Stabler points out.

Constraint satisfaction in fact seems better addapted to the problem of specifying the notion “possible natural grammar”, and it is arguable that that is what the MP actually intends. However, even under this interpretation, the constraints seem to represent rather arbitrary computations over derivations.

Be that as it may, many of the most puzzling generalizations of the Government-Binding Theory appear unchanged under this program, as well-formedness conditions on interface representations. For example, the Empty Category Principle (ECP) has been proposed both as an output condition on LF (Lasnik and Uriagereka (2005)) and on PF (Merchant 2002). However descriptively informative such generalizations are, they remain explanatoily anomalous under the assumptions of both the “grammar as interface” approach and the Minimalist Program itself.

The present paper attempts to compare progress toward this common goal of the Minimalist Program (MP) and Combinatory Categorial Grammar (CCG), a “nearly context-free” grammar formalism representative of a group including Lexicalized Tree Adjoining Grammars (LTAG), and Linear Indexed Grammars (LIG). Though originally developed as a linguistic theory of grammar, CCG is becoming widely used in computational applications in which recovery of interpretable structure is needed, including efficient wide coverage parsing, generation, and question-answering (Hockenmaier and Steedman 2002a,b; Hockenmaier 2003; Clark and Curran 2004; Clark et al. 2004; White and Baldridge 2003; Bos et al. 2004.)

The paper reports some recent developments in CCG, drawing on work by Baldridge, whose 2002 PhD thesis showed how categorial functional types can be elegantly restricted via a type-hierarchy whose values correspond to sets of applicable combinatory rules. In this form, CCG is a theory in which the sole locus of language-specific interfaces is the lexicon, which pairs (ordered) phonological forms with (unordered) logical forms, and assigns each such pair a syntactic type. A universal set of purely type-driven syntactic combinatory rules then combines the lexical elements into sentences by concatenating phonological forms and "projecting" logical relations (to both of which the combinatory apparatus is entirely blind) onto pairs of sentential phonological and logical forms. (The derivations themselves are not distinguished in the theory as a level of representation.) In the terms of the Minimalist Program, CCG provides a formal basis for eliminating rules of movement, both covert and overt, in favor of compositional merger.

This version of the theory strengthens earlier results in CCG concerning coordinate and prosodic structure and their interaction with quantifier scope, and addresses some recent criticisms hinging on the reality of the Across the Board Condition and the Strict Competence Hypothesis concerning incremental interpretation by the parser. It stands in contrast to MP grammars in emphasizing the role of lexical heads in specifying language specific constructions. Many principles of the Minimalist Program—for example, Full Interpretation, the definition of Phase, and the domain of (Multiple) Spell-out—can be seen as universal constraints on the constructions that can be lexically headed in CCG. To that extent they should not be seen in MP terms as output constraints at all, but rather as constraints on the process of (multiple) Spell-Out that relates the interface levels. Other constraints—in particular, many of those that relate to movement, including ECP—can simply be eliminated because of the reduction of movement to merger. The conditions on interface levels that remain, such as articulatory conditions, Condition C, and the island constraints, are essentially extra-linguistic.

2. Combinatory Categorial Grammar

The idea that grammars should be specifiable as directly mapping from meaning representation to spoken or written form is pretty much how most people have always thought of language. The belief that the main problem in specifying that mapping—say, when learning a language—lies in acquiring the lexicon is equally commonplace. CCG has at its heart a lexicon associating words with (syntactic and semantic) *Categories*.

2.1. Categories

Categories are syntactic *types* corresponding to functions or arguments, in the former case specifying domain, range, and directionality–e.g. the English transitive verb

(2) $(S\backslash NP)/NP$

They are associated with phonological forms (usually, words), and interpretations or logical forms, written as triples $\Phi := \Sigma : \Lambda$:

(3) married $:= (S\backslash NP)/NP : \lambda x\lambda y.marry'xy$

The simplest CCG derivations are purely applicative and equivalent to CF derivations:

(4)
$$\begin{array}{ccc}
\text{Anna} & \text{married} & \text{Manny} \\
\overline{NP : anna'} & \overline{(S\backslash NP)/NP : \lambda x\lambda y.marry'xy} & \overline{NP : manny'} \\
 & \multicolumn{2}{c}{\underline{\hspace{12em}}{>}} \\
 & \multicolumn{2}{c}{S\backslash NP : \lambda y.marry'manny'y} \\
\multicolumn{3}{c}{\underline{\hspace{20em}}{<}} \\
\multicolumn{3}{c}{S : marry'manny'anna'}
\end{array}$$

For other languages we need a slightly more general notation for categories, proposed in its present form by Baldridge (2002), of which English is a special case:

(5) a. English: $(S\backslash NP)/NP : \lambda x\lambda y.\mathsf{pred}xy$
b. Latin: $S\{|NP_{nom}, |NP_{acc}\} : \lambda\{y,x\}.\mathsf{pred}xy$
c. Tagalog: $S\{/NP_{nom}, /NP_{acc}\} : \lambda\{y,x\}.\mathsf{pred}xy$
d. Japanese: $S\{\backslash NP_{nom}, \backslash NP_{acc}\} : \lambda\{y,x\}.\mathsf{pred}xy$

The set brackets around syntactic argument types indicate that the arguments can combine in any order, and those round syntactic arguments indicate the same with the convention that the variables correspond to the syntactic arguments in left-to-right order. Such categories exemplify a principle of all CCG grammars called the Principle of Categorial Type Transparency (Steedman 2000:36):

(6) *The Principle of Categorial Type Transparency*:
For a given language, the semantic type of the interpretation together with language-specific directionality parameter settings uniquely determines the syntactic type of a category.

This principle is related to the MP principle Full Interpretation.

We follow Jacobson, Hepple, and Baldridge and Kruijff in assuming that rules and function categories are "modalized", as indicated by a subscript on slashes. Baldridge further assumes that slash modalities are features in a type hierarchy, drawn from some finite set $\mathcal{M}$ (the modalities used here are $\mathcal{M} = \{\star, \diamond, \times, \cdot\}$).

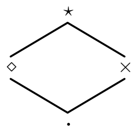


Figure 2: CCG type hierarchy for slash modalities (from Baldridge and Kruijff).

2.2. Combinatory Rules: Application

The function application rules can now be written as follows:

(7) *The functional application rules*

a. $X/_{\star}Y : f \quad Y : a \quad \Rightarrow \quad X : fa$ (>)

b. $Y : a \quad X\backslash_{\star}Y : f \quad \Rightarrow \quad X : fa$ (<)

Because $\star$ is the supertype of all other modalities, the $/_{\star}$ and $\backslash_{\star}$ slashes in the rules are interpreted as "$\preceq\star$"—that is, *all* functional categories can combine by these most basic rules, allowing the derivation (4) as before.

The Japanese transitive verb category (5d), repeated here, is clause-final, with $\cdot$ modality:

(8) tazuneta := $S\{\backslash NP_{nom}, \backslash NP_{acc}\} : \lambda\{y,x\}.visit'xy$

It supports multiple derivations, which are guaranteed to yield identical semantic representations:

(9)

Anna−ga	Manny−o	tazuneta
Anna−NOM	Manny−ACC	visited
$NP_{nom} : anna'$	$NP_{acc} : manny'$	$S\{\backslash NP_{nom}, \backslash NP_{acc}\} : \lambda\{y,x\}.visit'xy$

$$S\{\backslash NP_{nom}\} : \lambda y.visit'manny'y) \quad <$$
$$S : visit'manny'anna' \quad <$$

(10)

Manny−o	Anna−ga	tazuneta
Manny−ACC	Anna−NOM	visited
$NP_{acc} : manny'$	$NP_{nom} : anna'$	$S\{\backslash NP_{nom}, \backslash NP_{acc}\} : \lambda\{y,x\}.visit'xy$

$$S\{\backslash NP_{acc}\} : \lambda x.visit'xanna') \quad <$$
$$S : visit'manny'anna' \quad <$$

2.3. Combinatory Rules: Type-Raising **T**

Type-raising turns argument categories such as *NP* into functions over the functions that take them as arguments, such as the verbs above, into the results of such functions. Thus NPs like *Anna* can take on such categories as the following:

(11) a. Anna := $S/(S\backslash NP) : \lambda p.p\ anna'$

b. Anna := $S\backslash(S/NP) : \lambda p.p\ anna'$

c. Anna := $(S\backslash NP)\backslash((S\backslash NP)/NP) : \lambda p.p\ anna'$

d. etc.

This operation must be limited to ensure decidability, and in practice can be strictly limited to argument categories *NP*, *AP*, *PP*, *VP* and *S*. One way to do this is to specify it in the morpho-lexicon, in the categories for proper names, determiners, and the like. It therefore resembles the traditional operation of *case*.

In Japanese the relation of type-raising and case is completely transparent:

(12) a. Anna-ga := $S/(S\{\backslash NP_{nom}\}) : \lambda p.p\ anna'$
b. Anna-o := $S/(S\{\backslash NP_{acc}\}) : \lambda p.p\ anna'$
c. Anna-o := $(S\{\backslash NP_{nom}\})/(S\{\backslash NP_{nom}, \backslash NP_{acc}\}) : \lambda p.p\ anna'$
d. etc.

2.4. Combinatory Rules: Composition **B**

(13) *Forward composition* (>**B**)
$X/_{\diamond}Y : f \quad Y/_{\diamond}Z : g \quad \Rightarrow_{\mathbf{B}} \quad X/_{\diamond}Z : \lambda x.f(gx)$

The modalities on $X/_{\diamond}Y$ and $Y/_{\diamond}Z$ in the rule are interpreted as "$\preceq\diamond$"—that is, $\cdot$ or $\diamond$. The two modalities on these inputs need not be the same, but the modality on $X/_{\diamond}Z$ in the output has to be the same as that on $Y/_{\diamond}Z$, in accordance with the Principle of Inheritance—see Steedman 2000.

In interaction with simple functional application and lexicalized type-raising, composition engenders a potentially very freely "reordering and rebracketing" calculus, and generalizes the notion of surface or derivational constituency.

For example it allows strings like *Anna Married* to be fully interpreted derivational *constituents*, complete with compositional semantic interpretations, to support relativization without movement or empty categories, as in (15), via the following category for the relative pronoun:

(14) that := $(N\backslash N)/(S/NP) : \lambda p\lambda n\lambda x.(nx) \wedge (px)$

(15) (The man)

$$\begin{array}{ccc}
\text{that} & \text{Anna} & \text{married}\\
\overline{(N\backslash N)/(S/NP) :} & \overline{S/(S\backslash NP) :}^{>\mathbf{T}} & \overline{(S\backslash NP)/NP :}\\
\lambda p\lambda n\lambda x.(nx)\wedge(px) & \lambda f.f\ anna' & \lambda x\lambda y.marry'xy\\
 & \multicolumn{2}{c}{\overline{S/NP : \lambda x.marry'x\ anna'}^{>\mathbf{B}}}\\
\multicolumn{3}{c}{\overline{N\backslash N : \lambda n\lambda x.(nx)\wedge(marry'x\ anna')}^{>}}
\end{array}$$

Such extractions are correctly predicted to be unbounded, since composition can operate across clause boundaries:

(16) (The man)

$$\begin{array}{ccccc}
\text{that} & \text{Manny} & \text{says} & \text{Anna} & \text{married}\\
\overline{(N\backslash N)/(S/NP) :} & \overline{S/(S\backslash NP) :}^{>\mathbf{T}} & \overline{(S\backslash NP)/_{\diamond}S :} & \overline{S/(S\backslash NP) :}^{>\mathbf{T}} & \overline{(S\backslash NP)/NP :}\\
\lambda p\lambda n\lambda x.(nx)\wedge(px) & \lambda f.f\ manny' & \lambda x\lambda y.say'xy & \lambda f.f\ anna' & \lambda x\lambda y.marry'xy\\
 & \multicolumn{2}{c}{\overline{S/_{\diamond}S : \lambda x.say'x\ manny'}^{>\mathbf{B}}} & \multicolumn{2}{c}{\overline{S/NP : \lambda x.marry'x\ anna'}^{>\mathbf{B}}}\\
 & \multicolumn{4}{c}{\overline{S/NP : \lambda x.say'(marry'x\ anna')manny'}^{>\mathbf{B}}}\\
\multicolumn{5}{l}{\overline{N\backslash N : \lambda n\lambda x.(nx)\wedge(say'(marry'x\ anna')manny')}^{>}}
\end{array}$$

It is the lexical category (14) of the relative pronoun that establishes the long-range dependency between noun and verb (via the non-essential use of the variable x in the present notation). This relation too is established in the lexicon: syntactic derivation merely projects it onto the logical form.

Treating unbounded dependency in this way is analogous to lexicalizing movement to SPEC-of-IP/CP. but there is no movement, only merge. Notice that *in situ* object relative pronouns for languages like Mandarin can be handled with a related order-preserving category $(N\backslash N)\backslash(S/NP)$. So this mechanism subsumes both overt and covert movement to merge.

Unifying movement and *in situ* merger in this way is in fact required by the following principle that applies to all CCG lexicons (Steedman 2000:32):

(17) *The Principle of Lexical Head Government*:
Both bounded and unbounded dependencies are specified by the lexical syntactic type of their head.

We shall return to this principle and its consequences later.

It is important to notice also that the category for *says* assumed above restricts composition of the category to the uncrossed composition rule (13) and forbids composition by the crossed version (24), which we will see below is universally available and crucial to the grammar of Dutch. This is a forced move in English in order to prevent overgeneration of Dutch-like word orders. However, it means that, with the categories assumed so far, there is no derivation analogous to the above for unbounded subject extraction, which is nevertheless possible in English in cases like (18a):

(18) a. A man who Manny says likes Anna
b. *A man who Manny says that likes Anna

This is in fact a desirable result, because in general subject extraction is disallowed in English, as (18b) shows. It is *only* the bare-complement-taking verbs like *says* that allow it. We capture this fact with special lexical categories for just those verbs, such as the following, in addition to obvious categories like VP/S' and VP/S (see Steedman 1996 for discussion— the feature *agr* mediates agreement and prevents **the men that Manny says likes Anna*):

(19) says := $((S\backslash NP)/NP_{+ANT,agr})/(S\backslash NP_{agr})$

This category allows the extraction as follows:

(20) (The man)

$$
\begin{array}{cccc}
\text{that} & \text{Manny} & \text{says} & \text{likes Anna} \\
\hline
(N_{agr}\backslash N_{agr})/(S/NP_{agr}) : & S/(S\backslash NP) : \;{>}\mathbf{T} & ((S\backslash NP)/NP_{+ANT,agr})/(S\backslash NP_{agr}) : & S\backslash NP_{sng} : \;{>}\mathbf{T} \\
\lambda p\lambda n\lambda x.(nx)\wedge(px) & \lambda f.f\ manny' & \lambda p\lambda x\lambda y.say'(px)y & \lambda y.marry'anna'y
\end{array}
$$

$$
\begin{array}{l}
\xrightarrow{\ >\ } (S\backslash NP)/NP_{+ANT,sng} : \lambda x\lambda y.say'(married'anna'x)y \\
\xrightarrow{\ >\mathbf{B}^2\ } S/NP_{+ANT,sng} \\
\qquad \lambda x.say'(married'anna'x)manny' \\
\xrightarrow{\ >\ } N_{sng}\backslash N_{sng} : \lambda n\lambda x.(nx)\wedge(say'(marry'anna'x)manny')
\end{array}
$$

The feature $+ANT$ is incompatible with any lexical in situ material and prevents overgeneration of **Manny says likes Anna the man*. It is important to notice that the relative pronoun category is not restricted to complements requiring a $+ANT$ argument. This is in fact a requirement of the Principle of Lexical Head Government (17). In other words, there is no such thing as "antecedent government" in the strict sense of that term (Chomsky 1981).

The unbounded tough-movement construction can be handled analogously to the relative pronoun (14). Adjectives like *easy* have the following category (again, the unbounded dependency is not restricted to $+ANT$ arguments:

(21) easy := $(S_{pred}\backslash NP_{agr})/(S_{to}\backslash NP)/NP_{agr}) : \lambda p\lambda x.easily'(px\ one')$

This category supports the following unbounded dependencies:

(22) a. Manny is easy to please.
b. Manny is easy to believe (that) we can please.
c. Manny is easy to believe (*that) pleases Anna.

For example:

(23)

$$
\begin{array}{cccc}
\text{Manny} & \text{is} & \text{easy} & \text{to please} \\
\hline
S/(S\backslash NP_{sng}) & (S\backslash NP)_{sng}/(S_{pred}\backslash NP_{sng}) & (S_{pred}\backslash NP_{agr})/(S_{to}\backslash NP)/NP_{agr}) & ((S_{to}\backslash NP)/NP) \\
\lambda p.p\ manny' & \lambda p\lambda x.px & \lambda p\lambda x.easily'(px\ one') & \lambda x\lambda y.please'xy
\end{array}
$$

$$
\begin{array}{cc}
\xrightarrow{\ >\mathbf{B}\ } S/(S_{pred}\backslash NP_{sng}) & \xrightarrow{\ >\mathbf{B}\ } S_{pred}\backslash NP \\
\lambda p.p\ manny' & \lambda x.easily'(please'x\ one')
\end{array}
$$

$$
\xrightarrow{\ >\ } S : easily'(please'manny'one')
$$

Unlike the pure associative Lambek Calculus, CCG allows rules like the following:

(24) *Forward Crossed Composition* ($>\mathbf{B}_{\times}$)
$X/_{\times}Y : f \quad Y\backslash_{\times}Z : g \quad \Rightarrow_{\mathbf{B}} \quad X\backslash_{\times}Z : \lambda x.f(gx)$

It also allows the following kind of generalization of all kinds of composition:

(25) *Forward Crossed Composition* ($>\mathbf{B}^2_{\times}$)
$X/_{\times}Y : f \quad (Y\backslash_{\times}Z)/W : g \quad \Rightarrow_{\mathbf{B}} \quad (X\backslash_{\times}Z)/W : \lambda y\lambda x.f(gyx)$

These rules are restricted by the $\times$ modality, because they have a re-ordering effect.

Crossed composition is used in English for constructions like Heavy NP Shift:

(26) I met yesterday a tall man in a raincoat

$S/(S\backslash NP)$ $(S\backslash NP)/NP$ $(S\backslash NP)\backslash(S\backslash NP)$ $(S\backslash NP)\backslash((S\backslash NP)/NP)$

$(S\backslash NP)/NP$ $<\mathbf{B}_{\times}$

$S\backslash NP$ $<$

S $<$

English nominal functor categories such as determiners $NP/_{\diamond}N$ must bear the $\diamond$ modality, preventing examples like the following:

(27) *A tall in a raincoat man

$NP/_{\diamond}N$ $N/_{\diamond}N$ $N\backslash_{\diamond}N$ N

$*N/N$ $*<\mathbf{B}_{\times}$

3. CCG is “Nearly Context Free”

The inclusion of such rules increases the expressive power of the formalism beyond context-free languages. (Vijay-Shanker and Weir show that CCG is weakly equivalent to TAG and Linear Indexed Grammar (LIG). This is the least powerful known member of Joshi’s “Mildly Context Sensitive Languages.” This equivalence gives rise to a polynomial time worst-case parsing complexity result, and (more importantly) means that standard parsing algorithms can apply. It allows crossed dependencies, as in the following (from Shieber), in which type-raised NP categories are abbreviated as $NP^{\uparrow}{}_{case}$:

(28) das mer em Hans es huus hälfed aastriiche

that we−NOM Hans−DAT the house−ACC helped paint

$NP^{\uparrow}_{nom}$ $NP^{\uparrow}_{dat}$ $NP^{\uparrow}_{acc}$ $((S_{+SUB}\backslash NP_{nom})\backslash NP_{dat})/VP$ $VP\backslash NP_{acc}$

$((S_{+SUB}\backslash NP_{nom})\backslash NP_{dat})\backslash NP_{acc}$ $>\mathbf{B}_{\times}$

$(S_{+SUB}\backslash NP_{nom})\backslash NP_{dat}$ $>$

$S_{+SUB}\backslash NP_{nom}$ $>$

S_{+SUB} $>$

“that we helped Hans paint the house”

It is correctly predicted that the following word orders are also allowed in at least some dialects (Shieber 1985:338-9):

(29) a. a. das mer em Hans hälfed es huus aastriiche.
b. b. das em Hans mer es huus hälfed aastriiche.

The construction is completely productive, so the dependencies are not only intersective, but unbounded. For example, we have the following (also from Shieber):

(30) das mer d′chind em Hans es huus lönd hälfe aastriiche

that we−NOM the children−ACC Hans−DAT the house−ACC let help paint

$NP^{\uparrow}_{nom}$ $NP^{\uparrow}_{acc}$ $NP^{\uparrow}_{dat}$ $NP^{\uparrow}_{acc}$ $((S_{+SUB}\backslash NP_{nom})\backslash NP_{acc})/VP$ $(VP\backslash NP_{dat})/VP$ $VP\backslash NP_{acc}$

$(((S_{+SUB}\backslash NP_{nom})\backslash NP_{acc})\backslash NP_{dat})/VP$ $>\mathbf{B}^{2}_{\times}$

$(((S_{+SUB}\backslash NP_{nom})\backslash NP_{acc})\backslash NP_{dat})\backslash NP_{acc}$ $>\mathbf{B}_{\times}$

$((S_{+SUB}\backslash NP_{nom})\backslash NP_{acc})\backslash NP_{dat}$ $>$

$(S_{+SUB}\backslash NP_{nom})\backslash NP_{acc}$ $>$

$S_{+SUB}\backslash NP_{nom}$ $>$

S_{+SUB} $>$

“that we let the children help Hans paint the house”

Again the unbounded dependencies are projected from what are, apart from the more permissive modality, the same lexical verb frames as those for the corresponding English verbs, by (crossed) composition rules, without syntactic movement.

4. Derivational Ambiguity

Since fragments like *Anna married* can act as constituents for purposes of relativization, as in (15), they can also act as constituents of canonical clauses, as in (31)

(31)

Anna	married	Manny
$\overline{S/(S\backslash NP)}^{>T}$	$(S\backslash NP)/NP$	$\overline{S\backslash(S/NP)}^{<T}$
$: \lambda f.f\ anna'$	$: \lambda x\lambda y.marry'xy$	$: \lambda p.p\ manny'$

$$\overline{S/NP : \lambda x.marry'x\ anna'}^{>B}$$

$$\overline{S : marry'manny'anna'}^{<}$$

Of course, the traditional derivation is allowed as well:

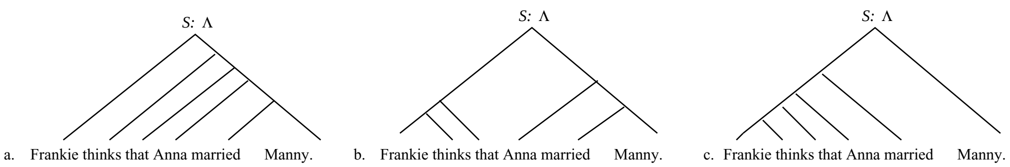
(32)

Anna	married	Manny
$\overline{S/(S\backslash NP)}^{>T}$	$(S\backslash NP)/NP$	$\overline{(S\backslash NP)\backslash((S\backslash NP)/NP)}^{<T}$
$: \lambda f.f\ anna'$	$: \lambda x\lambda y.marry'xy$	$: \lambda p.p\ manny'$

$$\overline{S\backslash NP : \lambda y.marry'manny'y}^{<}$$

$$\overline{S : marry'manny'anna'}^{>}$$

More complex sentences are multiply ambiguous:

(33)


This phenomenon has been referred to (misleadingly) as "Spurious" ambiguity, since all the derivations have the same interpretation Λ. Interestingly, the so called "spurious" constituents include most **left prefixes**, a fact with implications for incremental processing.

The point of allowing non-standard constituents like *Anna married*$_{S/NP}$ is that they occur as the residue of relativization and coordination, as in *the man that Anna married* and *Frankie divorced, and Anna married, Manny*.

It immediately follows that properties dependent on traditional command relations, such as binding, must be defined over logical forms:

(34) a. Manny likes himself.
b. *Himself likes Manny.

Once binding is seen as a property of logical form, it further follows that surface structure can be entirely dispensed with. Surface derivations do not constitute a representational level at all in CCG. They are merely an uninterpreted record of different ways in which the same typed logical form can be put together. Phonological Form and Logical form are the sole grammatical levels of representation.

The benefits of this move will be immediately apparent when we turn to the phenomena of coordination, intonation, and quantifier scope that seem most problematic for structurally monotonic theories of grammar.

5. Coordination

English conjunctions like *and* bear the following category, in which 'X' is S or any function category into S, and '$\wedge$' schematises over the usual pointwise recursion over logical conjunction (Partee and Rooth):

(35) and := $(X\backslash_{\star}X)/_{\star}X : \lambda p\lambda q.p\wedge q$

This category allows a movement- and deletion- free account of right node raising, as in (36):

(36) [Manny dated] and [Anna says he married] a saxophonist

$$\frac{[\text{Manny dated}]}{S/NP}{>}\mathbf{B} \quad \frac{\text{and}}{(X\backslash_{\star}X)/_{\star}X} \quad \frac{[\text{Anna says he married}]}{S/NP}{>}\mathbf{B} \quad \frac{\text{a saxophonist}}{S\backslash(S/NP)}{<}\mathbf{T}$$

$$\frac{(X\backslash_{\star}X)/_{\star}X \quad S/NP}{(S/NP)\backslash_{\star}(S/NP)}{>}$$

$$\frac{S/NP \quad (S/NP)\backslash_{\star}(S/NP)}{(S/NP)}{<}$$

$$\frac{(S/NP) \quad S\backslash(S/NP)}{S}{<}$$

The $\star$ modality on the conjunction category (35) means that it can *only* combine like types by the application rules (7). Hence, the across-the-board condition (ATB) on extractions from coordinate structures (including the "same case" condition)is a prediction or theorem, rather than a stipulation, axiom, or interface condition..

(37) a. A saxophonist [that$_{(N\backslash N)/(S/NP)}$ [[Anna married]$_{S/NP}$ and [Manny detests]$_{S/NP}$]$_{S/NP}$]$_{N\backslash N}$
b. A saxophonist [that$_{(N\backslash N)/(S/NP)}$ *[[Anna married]$_{S/NP}$ and [detests Manny]$_{S\backslash NP}$]]
c. A saxophonist that$_{(N\backslash N)/(S/NP)}$ *[[Anna married]$_{S/NP}$ and [Manny detests him]$_{S}$]]
d. A saxophonist that$_{(N\backslash N)/(S/NP)}$ *[[Anna married him]$_{S}$ and [Manny detests]$_{S/NP}$]

In Japanese, the interaction of type-raising (which we have seen is specified by morphological case) and composition similarly allows multiple derivations. In particular, we have:

(38) Anna-ga Manny-o tazuneta
Anna-NOM Manny-ACC visited

$$\frac{\text{Anna-ga}}{S/(S\backslash NP_{nom}) : \lambda p.p\ anna'} \quad \frac{\text{Manny-o}}{(S\backslash NP_{nom})/(S\{\backslash NP_{nom},\backslash NP_{acc}\}) : \lambda p.\lambda y.p\ manny' y} \quad \frac{\text{tazuneta}}{S\{\backslash NP_{nom},\backslash NP_{acc}\} : \lambda\{y,x\}.visit' xy}$$

$$\frac{}{S/(S\{\backslash NP_{nom},\backslash NP_{acc}\}) : \lambda p.p\ manny' anna'}{>}\mathbf{B}$$

$$\frac{}{S : visit' manny' anna'}{>}$$

'Anna visited Manny.'

(The conjunction category in Japanese is enclitic, unlike English proclitic (35).)

The possibility of non-standard "argument cluster" constituents such as *Anna-ga Manny-o* in derivation (38) correctly predicts that such clusters can coordinate:

(39) [Anna-ga Manny-o] , [Erika-ga Sara-o] tazuneta.
Anna-NOM *Manny*-ACC and Erika-NOM *Sara*-ACC visit-PAST.CONCL

$$\frac{}{S/(S\{\backslash NP_{nom},\backslash NP_{acc}\})} \quad \frac{}{(X/_{\star}X)\backslash_{\star}X} \quad \frac{}{S/(S\{\backslash NP_{nom},\backslash NP_{acc}\})} \quad \frac{}{S\{\backslash NP_{nom},\backslash NP_{acc}\}}$$

$$\frac{}{(S/(S\{\backslash NP_{nom},\backslash NP_{acc}\}))/(S/(S\{\backslash NP_{nom},\backslash NP_{acc}\}))}{<}$$

$$\frac{}{S/(S\{\backslash NP_{nom},\backslash NP_{acc}\})}{>}$$

$$\frac{}{S}{>}$$

'Anna visited Manny and Erika Sara.'

The prediction of similar argument cluster coordinations in the Dutch/Swiss German verb-raising construction (30) and in English in examples like (41) is similarly immediate (cf.Steedman and Dowty)

(40) give := $(VP/NP)/_{\diamond}NP : \lambda x\lambda y.give' yx$

(41) Give the piano-player a drink and the singer a cigar

$$\frac{\text{Give}}{(VP/NP)/_{\diamond}NP} \quad \frac{\text{the piano-player}}{(VP/NP)\backslash_{\diamond}((VP/NP)/_{\diamond}NP)}{<}\mathbf{T} \quad \frac{\text{a drink}}{VP\backslash(VP/NP)}{<}\mathbf{T} \quad \frac{\text{and}}{(X\backslash_{\star}X)/_{\star}X} \quad \frac{\text{the singer}}{(VP/NP)\backslash_{\diamond}((VP/NP)/_{\diamond}NP)}{<}\mathbf{T} \quad \frac{\text{a cigar}}{VP\backslash(VP/NP)}{<}\mathbf{T}$$

$$\frac{}{VP\backslash_{\diamond}((VP/NP)/_{\diamond}NP)}{<}\mathbf{B} \qquad \frac{}{VP\backslash_{\diamond}((VP/NP)/_{\diamond}NP)}{<}\mathbf{B}$$

$$\frac{}{(VP\backslash_{\diamond}((VP/NP)/_{\diamond}NP))\backslash(VP\backslash_{\diamond}((VP/NP)/_{\diamond}NP))}{>}$$

$$\frac{}{VP\backslash_{\diamond}((VP/NP)/_{\diamond}NP)}{<}$$

$$\frac{}{VP}{<}$$

However, the category (40) makes the order of arguments for the syntactic category of ditransitives the reverse of that of the predicate in the corresponding logical form, to allow a standard account of binding asymmetries at the level of logical form:

(42) a. I shall introduce the participants to each other.
b. *I shall introduce each other to the participants.

We have already seen that the non-standard derivations of CCG force us to distinguish logical form from derivation. However, this is a departure from other categorial accounts that obey Montague's stricture more narrowly.

6. English Intonation and Information Structure

The fact that substrings like *Anna married* and *Manny says that Anna married* are accorded the full status of derivational constituents in CCG means that Intonation Structure and Surface structure can be reunited in a single derivation rather than being consigned to different "tiers", or giving rise to separate syntax/intonation structure and syntax/information-structure interfaces.

(43) Q: I know who married DANNY. But who married MANNY?
A: (ANNA) (married MANNY).
H* L L+H* LH%

(44) Q: I know which man Anna DATED. But which one did she MARRY?
A: (Anna MARRIED) (MANNY).
L+H*LH% H* LL%

7. Quantifier Raising or "Covert Movement"

The existence of conjoinable constituents like *Every boy admires* explains why quantifier "movement" appears to be subject to the "Across-the-Board" condition which applies to *wh*-movement, as in examples like the following, adapted from Geach (1972):

(45) Every man likes, and every woman dislikes, some boy.

(There are only two readings, not four—cf. Park 1995). However, we must further assume that non universals translate as Skolem terms, in contrast to universals, which translate as Montagovian Generalized Quantifiers (Steedman 2004).

The universals *every* and *each* are true generalized quantifier determiners:

(46) every, each := $NP^{\uparrow}_{3SG}/_{\diamond}N_{3SG} : \lambda p\lambda q\lambda \ldots \forall x[px \rightarrow qx \ldots]^{\{x\}}$

$NP^{\uparrow}$ schematizes over all *NP* types raised over functions of the form $T|NP$, which we can spell out for the determiner case as:

(47) a. every, each := $(S/(S\backslash NP_{3SG}))/_{\diamond}N_{3SG} : \lambda p\lambda q.\forall x[px \rightarrow qx]^{\{x\}}$
b. every, each := $(S\backslash(S/NP))/_{\diamond}N_{3SG} : \lambda p\lambda q.\forall x[px \rightarrow qx]^{\{x\}}$
c. every, each := $((S\backslash NP)\backslash((S\backslash NP)/NP))/_{\diamond}N_{3SG} : \lambda p\lambda q.\lambda y.\forall x[px \rightarrow qxy]^{\{x\}}$
d. every, each := $(((S\backslash NP)/NP)\backslash(((S\backslash NP)/NP)/NP))/_{\diamond}N_{3SG} : \lambda p\lambda q.\lambda y\lambda z.\forall x[px \rightarrow qyxz]^{\{x\}}$ etc.

This is analogous to lexicalizing covert movement to SPEC-of-IP/CP. However, again, there is no movement, only merger. Cf. earlier remark about syntactically analogous *in situ wh*.

All other "quantifiers" are in fact referential expressions (cf. Woods 1975; VanLehn 1978; Webber 1978; Fodor and Sag 1982; Park 1996).

(48) a, an, some := $NP^{\uparrow}_{agr}/_{\diamond}N_{agr} : \lambda p\lambda q.q(skolem'p)$

Terms like $skolem'boy$ can be "specified" at any time, to become Skolem terms in all variables bound by quantifiers into whose scope the term has been brought at the point in the derivation where specification occurs. If specification is done as soon as the nounphrase is complete, then the set of variables in question is empty and specification yields a Skolem constant, which behaves like a proper name, with "scope everywhere", giving the appearance of wide scope. If specification is delayed until the derivation is complete, then a Skolem term dependent on all scoping quantifiers results, gicing the narrow scope reading.

These categories therefore allow the Narrow-scope Boy Reading for *Every man likes some boy* to be derived as follows (specification is expressed by a dotted underline):

(49)

$$\begin{array}{ccc}
\text{Every man} & \text{likes} & \text{some boy} \\
\hline
S/(S\backslash NP_{3SG}) & (S\backslash NP_{3SG})/NP & S\backslash(S\backslash NP) \\
:\lambda p.\forall y[man'y \rightarrow py]^{\{y\}} & \lambda x.\lambda y.admire'xy & :\lambda q.q(skolem'boy') \\
\hline
\multicolumn{2}{c}{S/NP \quad >\mathbf{B}} & \\
\multicolumn{2}{c}{:\lambda x.\forall y[man'y \rightarrow likes'xy]^{\{y\}}} & \\
\hline
\multicolumn{3}{c}{S : \forall y[man'y \rightarrow likes'(skolem'boy')y]^{\{y\}} \quad <} \\
\multicolumn{3}{c}{\cdots\cdots\cdots\cdots\cdots\cdots} \\
\multicolumn{3}{c}{S : \forall y[man'y \rightarrow likes'sk^{(y)}_{boy'}y]^{\{y\}}}
\end{array}$$

The right-branching derivation allows the same logical form.

The Wide-scope Boy Reading is obtained from the same derivation as follows:

(50)

$$\begin{array}{ccc}
\text{Every man} & \text{likes} & \text{some boy} \\
\hline
S/(S\backslash NP_{3SG}) & (S\backslash NP_{3SG})/NP & S\backslash(S\backslash NP) \\
:\lambda p.\forall y[man'y \rightarrow py]^{\{y\}} & \lambda x.\lambda y.admire'xy & :\lambda q.q(skolem'boy') \\
\hline
\multicolumn{2}{c}{S/NP \quad >\mathbf{B}} & \cdots\cdots \\
\multicolumn{2}{c}{:\lambda x.\forall y[man'y \rightarrow likes'xy]^{\{y\}}} & :\lambda q.q(sk_{boy'}) \\
\hline
\multicolumn{3}{c}{S : \forall y[man'y \rightarrow likes'(sk_{boy'})y]^{\{y\}} \quad <}
\end{array}$$

Again, a right-branching derivation allows the same logical form. The fact that (45) has two readings not four follows immediately (Steedman 2000):

(51) [Every man likes, and every woman dislikes]$_{S/NP}$, some boy.

8. Conclusions

CCG achieves the following simplifications of interest to any Minimalist Program for the theory of Grammar:

1. All covert and overt "movement" reduces to strictly type-driven (not structure-dependent) merger of string-adjacent syntactic types and logical forms projected from the lexicon by combinatory derivation.

2. The notions "phase" and "transformational cycle" reduce to the notion "lexically headed domain" at the level of logical form.

3. The notion of "spell-out" or "occasion when information is sent from syntax to phonology," whether single or multiple, is similarly redundant. Since there is no movement, there is no point in the derivation after which overt movement is prohibited. The relation between syntax and phonology is completely determined by the lexicon and combinatory projection.

4. Remaining "conditions on interface levels", such as articulatory conditions, Binding Condition C, and the island conditions are essentially external to grammar proper.

5. The availability of predominantly left branching derivations like (33c) allows assembly of phonological and logical forms to be incremental or "on-line" rather than at a single stage in the computation, without any additional imposition of interface conditions, supporting dynamic approaches to conditions B and C.

Seen in this light, most remaining open questions in both CCG and MP concern the notion "possible lexical category". Many of them are addressed by MP principles like Full Interpretation and Shortest Move. Since the move from $\overline{X}$ theory to "Bare Phrase Structure" theory (Chomsky 1995b,a) looks very much like a move to a categorial, rather than phrase-structure, base grammar, it is natural to look for convergence.

Such principles look rather different when viewed from this perspective. For example, it is Shortest Move that in MP terms allows bounded raising in (a) below, and also (via A-chain formation on the controlled subjects) in (b), whilst disallowing unbounded raising in (c) to deliver a reading where it is likely that Anna seems to be happy:

(52) a. Anna seems to be happy.
b. Anna is likely to seem to be happy
c. *Anna is likely that it seems to be happy.

The raising predicates *seems* and *likely* have the following categories (as well as categories (54), supporting the expletive predications):

(53) a. seems/seem := $(S\backslash NP)/(S_{to}\backslash NP) : \lambda p\lambda x.seemingly'(px)$
b. likely := $(S_{pred}\backslash NP)/(S_{to}\backslash NP) : \lambda p\lambda x.probably'(px)$

(54) a. seems := $(S\backslash NP_{it})/S_{CP} : \lambda s\lambda x.seemingly's$
b. likely := $(S_{pred}\backslash NP_{it})/S_{CP} : \lambda s\lambda x.probably's$

Thus *likely to seem to be happy* in (52b) is derived as follows (combination of the copula and *to* is suppressed to aid readability):

(55)
likely — to seem — to be happy

$$(S_{pred}\backslash NP)/(S_{to}\backslash NP) : \lambda p\lambda x.probably'(px) \quad (S_{to}\backslash NP)/(S_{to}\backslash NP) : \lambda p\lambda x.seemingly'(px) \quad S_{to}\backslash NP : happy'$$

$$\overline{(S_{pred}\backslash NP)/(S_{to}\backslash NP) : \lambda p\lambda x.probably'(seemingly'(px))}^{>\mathbf{B}}$$

$$\overline{S_{pred}\backslash NP : \lambda x.probably'(seemingly'(happy'x))}^{>}$$

However, these categories do not allow any analysis for (52c). In fact the only possibility for obtaining the intended reading is to give *likely* a completely unrelated relative pronoun-like category analogous to a tough-movement predicate like *easy*, (21), and to give *seems* a category analogous to that of a subject-extracting bare complement verb like (19). One way one might think of doing this is as follows:

(56) *likely := $(S_{pred}\backslash NP)/(S_{CP}/NP) : \lambda p\lambda x.probably'(px)$

(57) *seems := $((S\backslash NP_{it})/NP_{+ANT})/(S_{to}\backslash NP) : \lambda p\lambda x\lambda y.seemingly'p(x)$

Derivation of (52c) could then proceed analogously to the subject extraction in (20).

The expletive feature *it* and the antecedent-governed feature $+ANT$ are needed, in order to prevent overgeneration of the following:

(58) a. *Anna seems Manny to be happy.
b. *It seems Anna to be happy.
c. *Anna is likely that Manny seems to be happy.
d. *Anna is likely that it seems Manny to be happy.

However, these categories will immediately overgeneralize to other unbounded dependencies, allowing relativization, tough movement, and other absurdities:

(59) a. *A woman who it seems to be happy.
b. *Anna is easy to believe that it seems to be happy.
c. *Anna is likely that Manny likes
d. *Anna is likely that Manny thinks loves him

It might seem that we could rewrite the above categories as follows, using categories with a feature unique to the Shortest Move-violating categories—call it *FIXIT*—and introducing a parallel restriction $-FIXIT$ on the relative pronoun and tough-movement categories to prevent examples like (59):

(60) *likely := $(S_{pred}\backslash NP)/(S_{CP}/NP_{+FIXIT}) : \lambda p\lambda x.probably'(px)$

(61) *seems := $((S\backslash NP_{it})/NP_{+FIXIT})/(S_{to}\backslash NP) : \lambda p\lambda x\lambda y.seemingly'p(x)$

Not only would such a step amount to introducing a new species of lexically specified unbounded dependency into the grammar just for this construction. Both (60) and the revised relative pronoun and tough-movement categories would be in violation of the Principle of Lexical Head Government (17), for these categories are not the head of the dependency that they mediate. Such categories have no parallel elsewhere in the grammar of English or any other language.

Thus Shortest Move as it applies to raising and control appears to be a consequence of radically lexicalizing the grammars, rather than an active principle of the theory of grammar.

Acknowledgements

This paper draws on work with many colleagues over a long period, most recently, Jason Baldridge, Gann Bierner, Cem Bozsahin, Ruken Cakici, Stephen Clark, Mark Hepple, Julia Hockenmaier, Beryl Hoffman, and Nobo Komagata. I am grateful to Cem Bozsahin for reading the draft and making a number of suggestions which have improved the paper.

References

Baldridge, J. (2002). *Lexically Specified Derivational Control in Combinatory Categorial Grammar*. PhD thesis, University of Edinburgh.

Baldridge, J. and Kruijff, G.-J. (2003). Multi-modal combinatory categorial grammar. In *Proceedings of 11th Annual Meeting of the European Association for Computational Linguistics*, pages 211–218, Budapest.

Bos, J., Clark, S., Steedman, M., Curran, J. R., and Hockenmaier, J. (2004). Wide-coverage semantic representations from a ccg parser. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04), Geneva*, pages 1240–1246. ACL.

Brame, M. (1978). *Base Generated Syntax*. Noit Amrofer, Seattle, WA.

Bresnan, J. (1978). A realistic transformational grammar. In Halle, M., Bresnan, J., and Miller, G., editors, *Linguistic Structure and Psychological Reality*, pages 1–59. MIT Press, Cambridge, MA.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 132–139, Seattle, WA.

Chomsky, N. (1957). *Syntactic Structures*. Mouton, The Hague.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Chomsky, N. (1981). *Lectures on Government and Binding*. Foris, Dordrecht.

Chomsky, N. (1995a). Bare phrase structure. In Webelhuth, G., editor, *Government and Binding Theory and the Minimalist Program*, pages 383–439. Blackwell, Oxford.

Chomsky, N. (1995b). *The Minimalist Program*. MIT Press, Cambridge, MA.

Clark, S. and Curran, J. R. (2004). Parsing the WSJ using CCG and log-linear models. In *Proceedings of the 42nd Meeting of the ACL*, pages 104–111, Barcelona, Spain.

Clark, S., Steedman, M., and Curran, J. R. (2004). Object-extraction and question-parsing using CCG. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 111–118, Barcelona, Spain.

Collins, M. (1997). Three generative lexicalized models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, Madrid*, pages 16–23, San Francisco, CA. Morgan Kaufmann.

Dowty, D. (1988). Type-raising, functional composition, and nonconstituent coordination. In Oehrle, R. T., Bach, E., and Wheeler, D., editors, *Categorial Grammars and Natural Language Structures*, pages 153–198. Reidel, Dordrecht.

Emonds, J. (1976). *A Transformational Approach to English Syntax*. Academic Press, New York.

Gazdar, G. (1981). Unbounded dependencies and coordinate structure. *Linguistic Inquiry*, 12:155–184.

Hepple, M. (1990). *The Grammar and Processing of Order and Dependency: A Categorial Approach*. PhD thesis, University of Edinburgh.

Hockenmaier, J. (2003). *Data and models for statistical parsing with CCG*. PhD thesis, School of Informatics, University of Edinburgh.

Hockenmaier, J. and Steedman, M. (2002a). Acquiring compact lexicalized grammars from a cleaner treebank. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, pages 1974–1981, Las Palmas, Spain.

Hockenmaier, J. and Steedman, M. (2002b). Generative models for statistical parsing with Combinatory Categorial Grammar. In *Proceedings of the 40th Meeting of the ACL*, pages 335–342, Philadelphia, PA.

Jacobson, P. (1990). Raising as function composition. *Linguistics and Philosophy*, 13:423–476.

Joshi, A. (1988). Tree adjoining grammars. In Dowty, D., Karttunen, L., and Zwicky, A., editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge.

Kempson, R., Meyer-Viol, W., and Gabbay, D. (2001). *Dynamic Syntax: The Flow of Language Understanding*. Blackwell, Oxford.

Lasnik, H. and Uriagereka, J. (2005). *A Course in Minimalist Syntax*. Blackwell, Oxford.

Merchant, J. (2002). Pf output constraints and elliptical repair in sai comparatives. In *Proceedings of the 21st West Coast Conference on Formal Linguistics*, pages 292–305, Somerville, MA. Cascadilla Press.

Montague, R. (1974). *Formal Philosophy: Papers of Richard Montague*. Yale University Press, New Haven, CT. Richmond H. Thomason, ed.

Park, J. (1995). Quantifi er scope and constituency. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, Cambridge MA*, pages 205–212, San Francisco, CA. Morgan Kaufmann.

Partee, B. and Rooth, M. (1983). Generalised conjunction and type ambiguity. In Baüerle, R., Schwarze, C., and von Stechow, A., editors, *Meaning, Use, and Interpretation of Language*, pages 361–383. de Gruyter, Berlin.

Selkirk, E. (1984). *Phonology and Syntax*. MIT Press, Cambridge, MA.

Shieber, S. (1985). Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–343.

Stabler, E. (1992). *The Logical Approach to Syntax*. MIT Press, Cambridge MA.

Steedman, M. (1985). Dependency and coordination in the grammar of dutch and english. *Language*, 61:523–568.

Steedman, M. (1996). *Surface Structure and Interpretation*. MIT Press, Cambridge, MA.

Steedman, M. (2000). *The Syntactic Process*. MIT Press, Cambridge, MA.

Steedman, M. (2004). Surface-compositional scope-alternation without existential quantifi ers. Draft 5.1. University of Edinburgh.

Vijay-Shanker, K. and Weir, D. (1994). The equivalence of four extensions of context-free grammar. *Mathematical Systems Theory*, 27:511–546.

White, M. and Baldridge, J. (2003). Adapting chart realization to ccg. In *Proceedings of the 9th European Workshop on Natural Language Generation, held in conjunction with the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Budapest, April 2003*, San Francisco, CA. Morgan Kaufmann. to appear.

Woods, W. (1973). An experimental parsing system for transition network grammars. In Rustin, R., editor, *Natural Language Processing*, pages 111–154. Algorithmics Press, New York.

Proceedings of the 24th West Coast Conference on Formal Linguistics

edited by John Alderete, Chung-hye Han, and Alexei Kochetov

Cascadilla Proceedings Project Somerville, MA 2005

Copyright information

ISBN 1-57473-407-5 library binding

Ordering information

Orders for the library binding edition are handled by Cascadilla Press.
To place an order, go to www.lingref.com or contact:

Cascadilla Press, P.O. Box 440355, Somerville, MA 02144, USA
phone: 1-617-776-2370, fax: 1-617-776-2271, e-mail: sales@cascadilla.com

Web access and citation information

This entire proceedings can also be viewed on the web at www.lingref.com. Each paper has a unique document # which can be added to citations to facilitate access. The document # should not replace the full citation.

This paper can be cited as:

Steedman, Mark. 2005. Interfaces and the Grammar. In *Proceedings of the 24th West Coast Conference on Formal Linguistics*, ed. John Alderete et al., 19-33. Somerville, MA: Cascadilla Proceedings Project.

or:

Steedman, Mark. 2005. Interfaces and the Grammar. In *Proceedings of the 24th West Coast Conference on Formal Linguistics*, ed. John Alderete et al., 19-33. Somerville, MA: Cascadilla Proceedings Project. www.lingref.com, document #1203.