Multi-Modal Combinatory Categorial Grammar

Jason Baldridge
ICCS
School of Informatics
2 Buccleuch Place
University of Edinburgh
Edinburgh EH8 9LW, UK
jmb@cogsci.ed.ac.uk

Geert-Jan M. Kruijff
Universität des Saarlandes
Computational Linguistics
Lehrstuhl Uszkoreit
Building 17, Postfach 15 11 50
66041 Saarbrücken, Germany
gj@coli.uni-sb.de

Abstract

The paper shows how Combinatory Categorial Grammar (CCG) can be adapted to take advantage of the extra resource-sensitivity provided by the Categorial Type Logic framework. The resulting reformulation, Multi-Modal CCG, supports lexically specified control over the applicability of combinatory rules, permitting a universal rule component and shedding the need for language-specific restrictions on rules. We discuss some of the linguistic motivation for these changes, define the Multi-Modal CCG system and demonstrate how it works on some basic examples. We furthermore outline some possible extensions and address computational aspects of Multi-Modal CCG.

1 Introduction

The family of categorial grammar frameworks contains a diverse set of formalisms committed to many of the same core principles, such as compositionality, a strong degree of lexicalism, and semantic transparency. Nonetheless, each formalism has its own characteristics with respect to the category constructors and rules of combination it employs and the differing kinds of linguistic explanations which arise from its unique assumptions.

A formalism's category constructors and combination rules together define its *resource-sensitivity*: given the resources we have (i.e. expressions and their associated categories) we can, or can *not*, apply particular rules to form more complex expressions. It is particularly this idea of resource-sensitivity that is realized differently across different formalisms.

In this paper, we explain how two prominent categorial grammar frameworks, namely Combinatory Categorial Grammar (CCG, Steedman (2000)) and Categorial Type Logic (CTL, Morrill (1994); Moortgat (1997); Oehrle (to appear)), can be integrated to create a hybrid categorial framework, Multi-Modal Combinatory Categorial Grammar (Baldridge, 2002), that inherits the attractive properties of both. Specifically, Multi-Modal CCG retains the favorable computional aspects of CCG whilst incorporating the approach to resource-sensitivity taken in CTL. At the theoretical level, we discuss how this form of resource-sensitivity enables us to replace the rather *ad hoc* rule-specific constraints found in CCG with a clean resource-management regime, leading to more parsimonious linguistic analyses. We also discuss some of the benefits of modalized CCG for computational processing.

Although Multi-Modal CCG has the same set of rules as standard CCG, its improved resource-sensitivity enables it to have a *universal* rule component. It places *all* cross-linguistic variation in the lexicon, leading to a typological perspective on grammar that not only describes, but can also make predictions about, syntactic structure: Multi-Modal CCG provides a new view on how a typological perspective can be incorporated in a lexicalized, non-transformational setting.

Overview. We introduce CCG in §2, and then motivate the need for a revised notion of resource-sensitivity in CCG in §3. In §4, we introduce Multi-Modal CCG and present possible extensions in §5. Finally, we discuss the computational advantages of Multi-Modal CCG in §6 and then close with conclusions.

2 Combinatory Categorial Grammar

CCG's grammatical objects are categories which may be either atomic elements or (curried) functions which specify the canonical linear direction in which they seek their arguments. Lexical entries are specified by pairing words with categories via the $\vdash$ operator. Some simplified example entries are given below:[1]

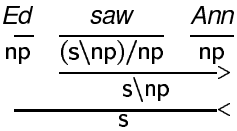
(1) (a) *Ed* $\vdash$ np, (b) *Ann* $\vdash$ np, (c) *saw* $\vdash$ (s\np)/np

The basic rules for combining categories are forward (>) and backward (<) application:

(>) $X/Y \;\; Y \Rightarrow X$

(<) $Y \;\; X\backslash Y \Rightarrow X$

With these rules and the categories given in (1), we can provide the derivation (2) for a simple sentence such as *Ed saw Ann*. Because of CCG's semantic transparency, a logical form for the sentence can be built compositionally and in parallel to the syntactic derivation. We will, however, suppress semantics in this paper.

(2)


CCG also utilizes further rules based on the composition (**B**), type-raising (**T**), and substitution (**S**) combinators of combinatory logic, each of which gives rise to several directionally-distinct rules. For example, there are forward and backward rules for both composition and type-raising:

(>**B**) $X/Y \;\; Y/Z \Rightarrow X/Z$

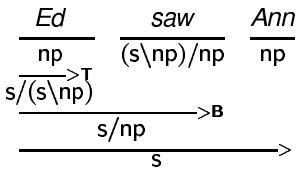
(<**B**) $Y\backslash Z \;\; X\backslash Y \Rightarrow X\backslash Z$

(>**T**) $X \Rightarrow Y/(Y\backslash X)$

(<**T**) $X \Rightarrow Y\backslash(Y/X)$

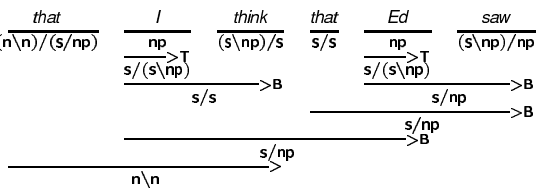
[1]We use Steedman's argument rightmost notation for categories. It is common in some traditions to use an alternative notation in which arguments sought to the left are placed on the left of the functor, e.g. *saw* $\vdash$ (np\s)/np.

These additional rules induce *associativity* in derivations, which is crucial for building the non-standard constituents for which categorial grammars are well-known. For example, we can now provide the following alternative derivation to (2), in which the verb combines with its subject before applying to its object:

(3)


Note that the combinatory rules have an invariant type-driven semantics, so this derivation produces precisely the same logical form as (2).

Arguably, one of CCG's greatest successes has been in demonstrating that the behavior of unbounded dependencies in syntax can be explained in a purely type-driven fashion in which the same lexical types that are responsible for bounded dependencies interact with CCG's combinatory rules to generalize to unbounded phenomena. For example, the same category (1c) given for *saw* is implicated in the derivation of not only *Ed saw Ann*, but also in the relative clause object extraction *that I think that Ed saw*.

(4)


Other frameworks typically rely on empty elements or extra lexical assignments to handle such relative clauses. Such strategies are often specific to the construction in question, whereas CCG supports a uniform treatment of phenomena as diverse as relativization, coordination, topicalization, intonational phrasing, and incremental processing (Steedman, 2000).

The rules introduced above are all *harmonic*, or *order preserving*. Thus, they are unhelpful when we consider sentences in which certain elements have "moved" with respect to their canonical position, such as in English heavy-NP shift, where an adverb comes between a verb and its object:

(5) Ed saw briefly his old friend from Skye.

To handle such sentences without relying on categorial ambiguity, rules are needed to combine the adverb with the verb before the latter consumes its object argument. CCG makes available two permutation-inducing composition rules that provide exactly the required functionality:

($>\mathbf{B}_\times$) $X/Y \;\; Y\backslash Z \;\Rightarrow\; X\backslash Z$

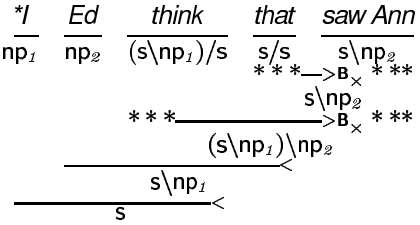
($<\mathbf{B}_\times$) $Y/Z \;\; X\backslash Y \;\Rightarrow\; X/Z$

As can be seen from the form of these rules, the primary functor composes with the secondary one in a manner that puts it between the secondary functor and its argument Z. This is precisely what is needed for (5): $<\mathbf{B}_\times$ permits a derivation using the same categories necessary for capturing the non-shifted version. The non-order-preserving nature of these rules is what boosts CCG's generative strength beyond context-free.

3 Rule restrictions

CCG uses a limited number of rules which work in conjunction with complex lexical categories to provide grammatical analyses, leaving the bulk of cross-linguistic variation in the lexicon. Even so, any given rule can be banned or restricted in any given grammar, so variation can thus arise in the rule component as well. For example, Steedman (2000) argues that the permutation-inducing rule $>\mathbf{B}_\times$ must be banned from the grammar of English to avoid derivations such as (6), where $>\mathbf{B}_\times$ would allow the subject of the embedded verb *saw* to scramble into the higher clause.

(6)

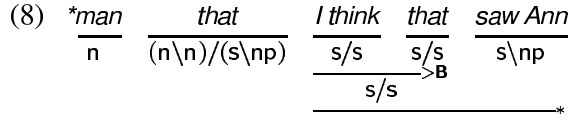


By banning $>\mathbf{B}_\times$ from the grammar of English, Steedman not only ensures that such scrambled orders do not arise, but also correctly *predicts* that embedded subjects cannot be extracted:

(7) *man$_i$ that I think that t_i saw Ann

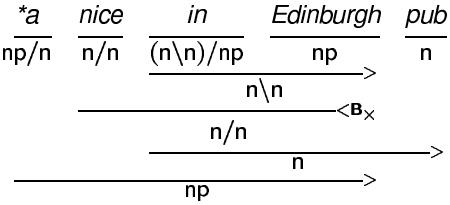
The extraction of an object is performed with only the harmonic rules, as shown in (4). Providing a derivation for (7), however, relies on $>\mathbf{B}_\times$ and thus fails under the assumption that $>\mathbf{B}_\times$ is banned, as shown in (8).

(8)



The rules of a given grammar can also be restricted to apply only to particular types. This is necessary in English for $<\mathbf{B}_\times$, which is needed for phenomena such as heavy-NP shift, as in (5), but which also can lead to lead to ungrammatical scrambled orders inside noun phrases. For example, consider (9), in which the category of a post-nominal prepositional phrase composes into that of a pre-nominal adjective.

(9)



To block such derivations, Steedman (2000) restricts the rule $<\mathbf{B}_\times$ for English as follows:[2]

($<\mathbf{B}_\times$) $Y/Z \;\; X\backslash Y \;\Rightarrow\; X/Z$

where $X = Y = s\$$

Furthermore, multiple versions of a given combinatory rule, each having its own restrictions, can be employed. For example, Steedman's analysis of Dutch utilizes two restricted versions of *each* of the rules $>\mathbf{B}$, $>\mathbf{B}_\times$, and $<\mathbf{B}_\times$.

The use of rule restrictions is unappealing for a number of reasons. First and foremost, the restrictions themselves are often *ad hoc* and can lead to the abuse of grammatical features. For example, Steedman's restricted version of $>\mathbf{B}_\times$ for Dutch can only apply when the secondary functor is a category that will ultimately produce a matrix clause (i.e., $s\_{SUB}\$$), whereas we wish to avoid using such features to enforce combinatory *control*. That is, the formal system should not have to refer to grammar specific properties. Another *ad hoc* use of features with rule restrictions is that some analyses utilize features on categories that have the sole function of controlling the applicability of a single rule, such as [±FORWARD COMPOSITION]

[2] The $ notation used in the restriction schematizes over all functions into the target category s.

(Trechsel, 2000). A further unappealing aspect of rule restrictions is that they are at times not simple declarations of categories with particular features, but can also involve modified forms of the rule schemata, such as one of Steedman's versions of $>\mathbf{B}$ for Dutch:

$$(>\mathbf{B}) \quad X/Y \;\; Y/(Y\backslash Z) \quad \Rightarrow \quad X/(Y\backslash Z) \qquad \text{where } Y = s\$$$

None of these ways of restricting rules is cross-linguistically motivated — they are employed to obtain control over very specific aspects of a particular analysis of a particular language. Also, they can be arbitrarily sensitive to particular categories, such that in the most extreme case, we could imagine writing a combinatory rule for every possible combination we want our grammar to handle. This would amount to using the categorial grammar system as a kind of specialized phrase structure grammar — at which point we lose the appeal of the purely type-driven nature of categorial grammar.

These problems can be solved by incorporating the more fine-grained slash types of the Categorial Type Logic (CTL) tradition into CCG to produce a system with just as much control, but which has a rule component that is universal to all grammars. Even though CTL already has a clean resource-management regime that can deal with the problems mentioned in this section, we choose to adapt CCG because of its far more attractive computational properties. No efficient algorithms exist that solve the NP-hard search problem in parsing CTL grammars, whereas realistic CCG grammars can be parsed with much greater efficiency. As we show in the next section, CCG's resource-sensitivity can be straightforwardly enhanced, and §6 shows that we do not give up CCG's computational attractiveness in doing so.

4 Improved resource-management

The main reason why CCG has needed rule restrictions is that it has an impoverished notion of the categorial slash. CCG uses a single pair of slashes $\{\backslash, /\}$, which allows it only to distinguish between arguments found to the left or the right, respectively. The rules make reference to these slashes, which in the case of the composition rules gives rise to a spectrum of four distinct rules.[3] There is, however, no way to discriminate between the primary functors $X\backslash Y$ of the backward rules:

$$(<) \quad Y \;\; X\backslash Y \quad \Rightarrow \quad X$$
$$(<\mathbf{B}) \quad Y\backslash Z \;\; X\backslash Y \quad \Rightarrow \quad X\backslash Z$$
$$(<\mathbf{B}_\times) \quad Y/Z \;\; X\backslash Y \quad \Rightarrow \quad X/Z$$

This fact means that there is no *lexical* way of specifying that a particular functor can only be used with a particular one of these rules. For example, if we give the category (10) to the coordinator, it can act as the primary functor in the $<\mathbf{B}$ rule (after consuming its first argument), and thereby produce derivations for ungrammatical strings as shown in (11).

(10) *and* $\vdash (s\backslash s)/s$

(11) **man*

that	*sleeps*	*and*	*he*	*talks*
$(n\backslash n)/(s\backslash np)$	$s\backslash np$	$(s\backslash s)/s$	np	$s\backslash np$

s (<)

$s\backslash s$ (>)

$s\backslash np$ (<B)

$n\backslash n$ (>)

To avoid such derivations, Steedman (2000) employs a ternary rule of coordination that ensures that coordinating particles cannot behave in this manner. However, this strategy threatens to increase the size of the rule base as the semantics and usage of different coordinators are accommodated. In the context of providing a universal rule base, it would thus be preferable to be able to deal with different kinds of coordination in a lexical manner.

The Categorial Type Logic tradition (CTL, Morrill (1994); Moortgat (1997); Oehrle (to appear)) provides a very clean solution to the lack of discrimination noted above. The fundamental idea is that not all slashes are the same; that is, we should be able to distinguish any number of slash types, each of which exhibits its own particular behavior. Rather than having just $\{\backslash, /\}$ like in CCG, CTL distinguishes multiple modes for combining categories and each mode i is associated with its own leftward and rightward slashes $\{\backslash_i, /_i\}$.[4] As with CCG, there are basic rules for string-adjacent, non-associative combination which allow a functor category to consume its arguments respecting the directionality of the slash, but unlike CCG,

[3] The Principles of Consistency and Inheritance filter out other possible rules based on the composition combinator.

[4] We ignore the product $\bullet_i$ and unary modes here.

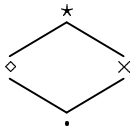
these rules build logical structures that record the modes through which constituents have combined. These structures can then be subsequently reconfigured by more powerful *structural rules* that allow specific ways of rebracketing (associativity), reordering (permutativity), or both.

Unlike CCG's rules, the structural rules of CTL cannot be restricted by imposing extra-logical constraints. Instead, they are sensitive to the modes by which sub-structures have been built and therefore cannot apply in all contexts. The typed slashes of CTL project their modes into structures and thereby limit or enable the application of structural rules to their local context. Thus, contrary to CCG, there are no rule-constraints acting as absolute and global choices; instead, parametric options regarding the way in which expressions can be combined are selectively invoked via the appropriate category assignments in the lexicon.

Despite the overall architectures of CCG and CTL being quite different, the use of modalized slashes to control combinatory behavior can be incorporated in CCG to enable cleaner resource-management and render rule restrictions unnecessary. The basic intuition is that each of the different combinatory rules will be sensitive to particular modes and thus apply only to input categories which have the appropriate slash types. To start, we assume four modes governing different levels of associativity and permutativity, as follows:

	non-permutative	permutative
non-associative	$\star$	$\times$
associative	$\diamond$	$\cdot$

Furthermore, we organize these modes into the type hierarchy shown on the right. The most limited mode $\star$ is thus the top of the hierarchy, whereas the most permissive one $\cdot$ inherits the properties of all the others.



With these modes, we can make use of their corresponding slash types, such as $/_\star$ and $\backslash_\times$, in the categories of the combinatory rules. For example, we can now state the application rules as follows:

(>) $X/_\star Y \quad Y \Rightarrow X$

(<) $Y \quad X\backslash_\star Y \Rightarrow X$

Because the mode $\star$ is the root of the hierarchy, these rules are thus available to categories with any slash type. However, now consider the modalized composition rules:

(>**B**) $X/_\diamond Y \quad Y/_\diamond Z \Rightarrow X/_\diamond Z$

(<**B**) $Y\backslash_\diamond Z \quad X\backslash_\diamond Y \Rightarrow X\backslash_\diamond Z$

(>**B**$_\times$) $X/_\times Y \quad Y\backslash_\times Z \Rightarrow X\backslash_\times Z$

(<**B**$_\times$) $Y/_\times Z \quad X\backslash_\times Y \Rightarrow X/_\times Z$

The first ramification of these formulations is that categories defined with the slashes $\backslash_\star$ and $/_\star$ will not be able to serve as input to these rules since the mode $\star$ is not a subtype of either $\diamond$ or $\times$. This is precisely what is needed to provide lexical categories for coordinating particles without enabling illicit derivations such as (11). We now can simply assign the category (12) to *and*, with the result that the category $s\backslash_\star s$ of *and he talks* cannot compose with that of *sleeps*, as shown in (13).

(12) *and* $\vdash (s\backslash_\star s)/_\star s$

(13)

sleeps	*and*	*he*	*talks*	
$s\backslash.np$	$(s\backslash_\star s)/_\star s$	np	$s\backslash.np$	
			s	$<$
	$s\backslash_\star s$			$>$
$s\backslash_\star s$				$*$

Note that the fact that the category of intransitive verbs has the $\backslash.$ slash allows it to serve as the primary functor in all the backward rules, since the mode $\cdot$ is a subtype of all the other modes.

A second effect of the modalized composition rules is that the modes $\diamond$ and $\times$ can now be used to discriminate between the harmonic and the crossed rules, giving Multi-Modal CCG a lexical handle on permutation. This means that it is no longer necessary to ban >**B**$_\times$ in English since we can instead assign the complementizer the category (14). This category can only compose associatively, thereby allowing extraction of embedded objects while blocking that of subjects.

(14) *that* $\vdash s/_\diamond s$

Similarly, we get lexical control over <**B**$_\times$ and the permutation of elements within noun phrases (9). Whereas standard CCG requires a rule restriction to avoid such derivations, the applicability of the <**B**$_\times$ rule can be blocked by the use of the modes $\star$ and $\diamond$ on the categories of pre-nominal and post-nominal modifiers:

(15)

**a*	*nice*	*in*	*Edinburgh*	*pub*	
$np/_\diamond n$	$n/_\diamond n$	$(n\backslash_\star n)/_\star np$	np	n	
		$n\backslash_\star n$			$>$
	n				$*$

Note that the categories $n/_{\diamond}n$ and $n\backslash_{\star}n$ cannot combine through either $>\mathbf{B}_{\times}$ or $<\mathbf{B}_{\times}$.

Although the mode $\times$ (non-associative & permutative) does not appear necessary for English, it is needed in the grammar of Dutch. The CCG account given by Steedman (2000) for crossing dependencies in Dutch subordinate clauses relies crucially on $>\mathbf{B}_{\times}$. However, Steedman must restrict the harmonic rule $>\mathbf{B}$ in order to block some ungrammatical orders. With the multi-modal setting, Baldridge (2002) shows that such restrictions are unnecessary because the same slash $/_{\times}$ that allows two subordinate verbs to combine through $>\mathbf{B}_{\times}$ also correctly blocks $>\mathbf{B}$ from applying. The encoding of Steedman's analysis into Multi-Modal CCG thus *predicts* that certain word orders are ungrammatical, whereas this must be stipulated in standard CCG. The encoding also has no need of the restrictions Steedman places on several other rules — instead it uses exactly the same rules that are used for English.

The input to the type-raising rules does not make reference to any slashes, but there are two slashes in the output category. Type-raising is actually provable in the most basic CTL system, and any mode can decorate the output slashes; however, it must be the same mode on both slashes. We use a variable mode i for the modalized versions of the type-raising rules:

$$(>\mathbf{T}) \quad X \Rightarrow Y/_i(Y\backslash_i X)$$
$$(<\mathbf{T}) \quad X \Rightarrow Y\backslash_i(Y/_i X)$$

There is no modal control over the applicability of type-raising, but the combinatory potential of the output category will be subject to constraints made on the category it applies to.

The core aspect of the multi-modal extension of CCG is thus the simple, but powerful ability to stratify the rules so that lexical items can be declared suitable or unsuitable as inputs to different rules. We have outlined here just a few of the motivating examples that demonstrate the utility of this ability and the manner in which it allows us to cast aside rule restrictions for controlling the grammar. This is a principled move which replaces arbitrary, globally declared restrictions with a small set of cross-linguistically motivated distinctions encoded in terms of the multiple slash types utilized in CTL. Baldridge (2002) supports this formulation with multi-modal analyses for a wide range of phenomena in English, Dutch, Turkish, Tagalog, and Toba Batak.

5 Extensions

Modes lead not only to a very clean formalization of resource-sensitivity, but also give rise to an interesting linguistic perspective. As e.g. Hepple (1995) notes, each pair of decorated slashes $\{\backslash_i, /_i\}$ can correspond to a particular grammatical phenomenon. Rules then model how different phenomena can be combined to form larger grammatical structures.

For example, one way we can use modal decoration is to model *dependency*, the asymmetry between heads and dependents, which has been used in CTL to give accounts of coordination and word order, e.g. (Moortgat and Morrill, 1991; Kruijff, 2001). Observe that we really need modes here: function-argument structure does not correspond to dependency. For example, a sentential adjunct can have the category $s\backslash s$ where the adjunct acts as the function, taking a verb (s) as its argument. However, the adjunct itself is the dependent of the verbal head, and not vice versa.

Adopting notation common in dependency grammar, we use as our basic modes a pair of arrows $\leftarrow, \rightarrow$ that point from head (h) to dependent (d): $(d \leftarrow h)$ or $(h \rightarrow d)$. Additionally, we use $\sqcap$ to handle headless constructions (e.g. coordination), and $\leftrightarrow$ as analogous to $\cdot$. The issue that now arises is how to integrate these modes with $\{\star, \diamond, \times, \cdot\}$: the latter modes control adjacency, and we would like to keep adjacency and dependency as separate (orthogonal) dimensions.

Therefore, we propose to keep these dimensions as separate decorations on slashes as well, as follows: given dependency mode d, and adjacency mode a, we can form slashes $\backslash_a^d$ versus $/_a^d$. Combinatory rules can operate on either dimension or on both, e.g. consider $(<\mathbf{B}_{\times})$ with dependency:

$$(<\mathbf{B}_{\times}) \quad Y/_{\times}^{\rightarrow}Z \;\; X\backslash_{\times}^{\rightarrow}Y \Rightarrow X/_{\times}^{\rightarrow}Z$$

Another modal dimension is one which distinguishes a slash as being suitable or unsuitable as the primary functor in a combinatory rule. This can be used to implement the notion of antecedent government discussed by Steedman (2000) for allowing some arguments to be extractable but not

movable. Encoding this modally avoids the need to require all lexical noun phrases to be declared as not antecedent governed and maintains the separation we desire between features that control combination and features for agreement, tense, etc.

Though we have not made use of CTL's unary modes in the presentation of Multi-Modal CCG given here, they could prove useful in limiting the applicability of type-raising rules. Following CTL, we introduce $\Diamond_i$ and its residuated counterpart $\Box_i^{\downarrow}$ such that we have, for an arbitrary category A, $\Box_i^{\downarrow}\Diamond_i A \Rightarrow A$ and $A \Rightarrow \Diamond_i\Box_i^{\downarrow}A$. Having unary modes would remove the need under some analyses to make type-raising of all argument categories obligatory. Thus, $>\mathbf{T}$ would appear as follows:

$$(>\mathbf{T}) \quad \Box_{tr}^{\downarrow}X \quad \Rightarrow \quad Y/_i(Y\backslash_i X)$$

As such, $>\mathbf{T}$ can only apply to $\Box_{tr}^{\downarrow}$'d categories: the categories for determiners and noun phrases would then be $(\Box_{tr}^{\downarrow}\Diamond_{tr}\mathsf{np})/_{\Diamond}\mathsf{n}$ and $\Box_{tr}^{\downarrow}\Diamond_{tr}\mathsf{np}$, respectively. Because of residuation, we can drop $\Box_{tr}^{\downarrow}\Diamond_{tr}$ if we do not need to type-raise.

Finally, the multi-modal setting also enables us to introduce more powerful combinators into the grammar, possibly taking it beyond mild context-sensitivity. Precisely because of the tight resource-sensitive control over the applicability of combinatory rules, we can avoid a collapse to a situation where "anything goes".

6 Computational aspects and implementation

CCG has mildly context-sensitive generative power and CCG grammars can be parsed in worst-case polynomial time by using a structure sharing algorithm (Vijay-Shanker and Weir, 1990). This algorithm does incur some computational overhead, and Komagata (1999) shows that the performance of a worst-case exponential CKY parser with a semantic equivalency check is cubic in the average case (tested on Japanese sentences averaging 20 words in length). This a major attraction of CCG over CTL, for which no reasonably efficient parsers have been constructed that can handle realistic grammars.

Multi-Modal CCG inherits CCG's attractive computational properties and adds the possibility to take advantage of some new strategies. Most importantly, it remains mildly context-sensitive. We have not added any new rules of combination; instead, we employ the standardly assumed rules and make them sensitive to particular kinds of slashes. A Multi-Modal CCG grammar can be simulated with standard CCG by adding a mode feature to the ultimate targets of (possibly complex) categories and then formulating the rules with restrictions that reference those features. For example, the multi-modal category (16) would be converted into (17).

(16) $(\mathsf{s}\backslash.\mathsf{np})/_{\times}(\mathsf{s}\backslash.\mathsf{np})$

(17) $(\mathsf{s}\backslash\mathsf{np}_{mode=.})/(\mathsf{s}_{mode=\times}\backslash\mathsf{np}_{mode=.})$

The rule $>\mathbf{B}_{\times}$ would then be formulated as follows, where a and b are variables standing for any atomic category.

$$(<\mathbf{B}_{\times}) \quad Y/Z \;\; X\backslash Y \quad \Rightarrow \quad X/Z$$
$$\text{where } Y = \mathsf{a}_{mode=\times}\$1 \text{ and } Z = \mathsf{b}_{mode=\times}\$2$$

Using restrictions in this manner in standard CCG would technically provide the same advantages as Multi-Modal CCG, but is a less clean formulation. Furthermore, by using modally decorated slashes to enforce these effects, we have a clear separation between specifications that control category combining operations and more standard features which encode distinctions such as number, gender, verbal voice, tense, etc.

It is straightforward to adapt an existing CCG parser to deal with modally decorated slashes as they are a simple, non-recursive feature added to the slash specification. Though there are no reductions in overall parsing complexity, we nonetheless obtain several advantages by using Multi-Modal CCG. One is that the invariant rule component makes it possible to implement the combinatory rules as hard-coded procedures that are the same for all grammars (Baldridge, 2002). With standard CCG rules, it may be necessary to unify the input categories against several restrictions in order to verify that a rule can even be applied, whereas a multi-modal rule needs only to inspect the simple slash data structures of the input categories to do the same. Also, a standard CCG rule may need to check that the X portion of a combinatory rule satisfies some restriction (e.g., the restricted $<\mathbf{B}_{\times}$ rule for English). A multi-

modal rule can instead ignore the X and just ensure that the Y portion of both inputs unify. Finally, CCG grammars typically need multiple versions of the same rule, whereas only one version of each is necessary in Multi-Modal CCG. The extra resource-sensitivity of multi-modal rules thus allows us to pack the functionality of several standard CCG rules into a single procedure, again cutting down on the number of unifications which are necessary in applying a grammar's rules.

The fact that slashes are more refined could make it seem that writing grammars would become more difficult since we must make choices about the modes. Our experience, however, is that the task becomes easier since different constructions can be dealt with on their own terms without worrying about complex interactions with rule restrictions that are already in place for other aspects of the grammar. We can further reduce the "burden" of specifying modes by defining the grammar as a lexical inheritance hierarchy along the lines of Villavicencio (2002), thereby exploiting redundancy between different classes of lexical items.

7 Conclusions

We have presented an adaptation of CCG in which the applicability of combinatory rules is controlled directly through lexically assigned categories. Using modally decorated slashes as in CTL, we obtain a fine-grained form of derivational control in a purely lexicalized fashion. Consequently, we can assume all combinatory rule schemas to be active *universally*. Grammars need to differ in the types of lexical categories only, i.e. what types of modalized slashes they use, and thus what rules they make applicable. We motivated the basic theory on data from English and Dutch, and discussed several possible extensions. Further linguistic evidence and more in-depth Multi-Modal CCG analyses for English, Dutch, Turkish, Tagalog, and Toba Batak are provided in Baldridge (2002).

In this paper, we have omitted semantics. See Steedman (2000) or Baldridge and Kruijff (2002) for different ways of constructing logical forms compositionally with CCG.

Acknowledgements. We would like to thank Cem Bozsahin, Mark McConville, Mark Steedman and the EACL reviewers for comments. Jason Baldridge's work is supported by Edinburgh-Stanford Link R36763, ROSIE project. Geert-Jan Kruijff's work is supported by the DFG SFB 378 *Resource-Sensitive Cognitive Processes*, Project NEGRA EM6.

References

Jason Baldridge and Geert-Jan Kruijff. 2002. Coupling CCG and hybrid logic dependency semantics. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 319–326, Philadelphia, Pennsylvania.

Jason Baldridge. 2002. *Lexically Specified Derivational Control in Combinatorial Categorial Grammar*. Ph.D. thesis, University of Edinburgh.

Mark Hepple. 1995. Mixing modes of linguistic description in categorial grammar. In *Proc. EACL-7, Dublin Ireland*.

Nobo Komagata. 1999. *Information Structure in Texts:A Computational Analysis of Contextual Appropriateness in English and Japanese*. Ph.D. thesis, University of Pennsylvania.

Geert-Jan M. Kruijff. 2001. *A Categorial Modal Architecture of Informativity: Dependency Grammar Logic & Information Structure*. Ph.D. thesis, Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic.

Michael Moortgat and Glyn Morrill. 1991. Heads and phrases: Type calculus for dependency and constituent structure. Unpublished manuscript. Available from http://www-lsi.upc.es/~glyn/.

Michael Moortgat. 1997. Categorial type logics. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Language*. Elsevier Science B.V.

Glyn V. Morrill. 1994. *Type Logical Grammar: Categorial Logic of Signs*. Kluwer Academic Publishers, Dordrecht, Boston, London.

Richard T. Oehrle. to appear. Multi-modal type-logical grammar. In Robert D. Borsley & Kersti Borjars, editor, *Non-transformational Syntax: A Guide to Current Debate*. Basil Blackwell, Oxford, United Kingdom.

Mark Steedman. 2000. *The Syntactic Process*. The MIT Press, Cambridge Mass.

Frank Trechsel. 2000. A CCG account of Tzotzil pied piping. *Natural Language and Linguistic Theory*, 18:611–663.

K. Vijay-Shanker and David Weir. 1990. Polynomial time parsing of combinatory categorial grammars. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics, Pittsburgh*, pages 1–8, June.

Aline Villavicencio. 2002. *The Acquisition of a Unification-Based Generalised Categorial Grammar*. Ph.D. thesis, University of Cambridge.