THE FORMAL CONSEQUENCES OF USING VARIABLES IN CCG CATEGORIES

Beryl Hoffman *
Dept. of Computer and Information Sciences
University of Pennsylvania
Philadelphia, PA 19104
(hoffman@linc.cis.upenn.edu)

Abstract

Combinatory Categorial Grammars, CCGs, (Steedman 1985) have been shown by Weir and Joshi (1988) to generate the same class of languages as Tree-Adjoining Grammars (TAG), Head Grammars (HG), and Linear Indexed Grammars (LIG). In this paper, I will discuss the effect of using variables in lexical category assignments in CCGs. It will be shown that using variables in lexical categories can increase the weak generative capacity of CCGs beyond the class of grammars listed above.

A Formal Definition for CCGs

In categorial grammars, grammatical entities are of two types: *basic categories* and *functions*. A basic category such as NP serves as a shorthand for a set of syntactic and semantic features. A category such as $S\backslash NP$ is a function representing an intransitive verb; the function looks for an argument of type NP on its left and results in the category S. A small set of combinatory rules serve to combine these categories while preserving a transparent relation between syntax and semantics. Application rules allow functions to combine with their arguments, while composition rules allow two functions to combine together.

Based on the formal definition of CCGs in (Weir-Joshi 1988), a CCG, G, is denoted by (V_T, V_N, S, f, R), where

- V_T is a finite set of terminals,
- V_N is a finite set of nonterminals,
- S is a distinguished member of V_N,
- f is a function that maps elements of $V_T \cup \{\epsilon\}$ to finite subsets of $C(V_N)$, the set of categories, where,

  - $V_N \subseteq C(V_N)$ and
  - if c_1 and $c_2 \in C(V_N)$, then $(c_1\backslash c_2)$ and $(c_1/c_2) \in C(V_N)$.

- R is a finite set of combinatory rules where $X, Y, Z_1, \ldots, Z_n$ are variables over the set of categories $C(V_N)$, and the slash variable $|_i$ can bind to $\backslash$ or $/$. Certain restrictions may be placed on the possible instantiations of the variables in the rules.
  - **Forward Application** (>): $X/Y \;\; Y \rightarrow X$
  - **Backward Application** (<): $Y \;\; X\backslash Y \rightarrow X$
  - **Generalized Forward Composition** (>B(n) or >Bx(n)): For some $n \geq 1$, $X/Y \;\; Y|_1 Z_1|_2 \ldots|_n Z_n \rightarrow X|_1 Z_1|_2 \ldots|_n Z_n$
  - **Generalized Backward Composition** (<B(n) or <Bx(n)): For some $n \geq 1$, $Y|_1 Z_1|_2 \ldots|_n Z_n \;\; X\backslash Y \rightarrow X|_1 Z_1|_2 \ldots|_n Z_n$

The derives relation in a CCG is defined as $\alpha c\beta \Rightarrow \alpha c_1 c_2 \beta$ if R contains the rule $c_1 c_2 \rightarrow c$. The language generated by this grammar is defined as

$$L(G) = \{a_1, \ldots, a_n \mid S \stackrel{*}{\Rightarrow} c_1, \ldots, c_n,$$
$$c_i \in f(a_i), a_i \in V_T \cup \{\epsilon\}, 1 \leq i \geq n\}$$

Under these assumptions, Weir and Joshi (1988) prove that CCGs are weakly equivalent to TAGs, HGs, and LIGs. Their conversion of a CCG to a LIG[1] relies on the fact that the combinatory rules in the CCG are linear. To preserve linearity in CCGs, only the category X in the combinatory rules can be unbounded in size; the variables Y and Z must be bounded in their possible instantiations. In other words, only a finite number of categories can fill the secondary constituent of each combinatory rule. The secondary constituent is the second of the pair of categories being combined in the forward rules and the first of the pair in the backward rules (e.g. $Y|Z_1 \ldots|Z_n$).

Weir and Joshi do not restrict the size of the secondary constituents in the formal definition of the CCG rules, but they prove that the following lemma holds of the grammar.

*I would like to thank Mark Steedman, Libby Levison, Owen Rambow, and the anonymous referees for their valuable advice. This work was partially supported by DARPA N00014-90-J-1863, ARO DAAL03-89-C-0031, NSF IRI 90-16592, Ben Franklin 91S.3078C-1.

[1]Linear Indexed Grammars are a restricted version of Indexed Grammars in which no rule can copy a stack of unbounded size to more than one daughter (Gazdar 1985).

Lemma: *There is a bound (determined by the grammar G) on the number of useful categories that can match the secondary constituent of a rule.*

There are an infinite number of derivable categories in CCGs, however Weir and Joshi show that the number of components that derivable categories have is bounded. The components of a category $c = (c_0|_1 c_1|_2 ...|_n c_n)$ are its immediate components $c_0, ..., c_n$ and the components of these immediate components. A finite set $D_C(G)$ can be defined that contains all derivable components of every *useful* category where a category c is *useful* if $c \stackrel{*}{\Rightarrow} w$ for some w in $V_T{}^*$:

$$c \in D_C(G) \text{ if } c \text{ is a component of } c'$$
$$\text{where } c' \in f(a) \text{ for some } a \in V_T \cup \{\epsilon\}.$$

Given that every useful category matching the secondary constituents Y and $Y|Z_1...|Z_n$ in the combinatory rules has components which are in $D_C(G)$, the lemma given above holds.

However, this lemma does not hold if there are variables in the lexical categories in V_T. Variables can bind a category of any size, and thus useful categories containing variables do not necessarily have all of their derivable components in the finite set $D_C(G)$.

The Use of Variables

Linguistic Use

In CCGs, a type-raising rule can be used in the lexicon to convert basic elements into functions; for example, an NP category can be type-raised to the category $S/(S\backslash NP)$ representing a function looking for an intransitive verb on its right. Steedman uses type-raising of NPs to capture syntactic coordination and extraction facts. In Steedman's Dutch grammar (1985), variables are used in the lexical category for type-raised NPs, i.e. the variable v in the category $v/(v\backslash NP)$ generalizes across all possible verbal categories. The use of variables allows the type-raised NPs in the following coordinated sentence to easily combine together, using the forward composition rule, even though they are arguments of different verbs.

(1) ...dat [Jan Piet] en [Cecilia Henk] zag zwemmen.
...that [Jan Piet] and [Cecilia Henk] saw swim.
...that Jan saw Piet and Cecilia saw Henk swim.

$$\frac{\begin{array}{cc}\text{Jan} & \text{Piet} \\ v/(v\backslash NP) & v'/(v'\backslash NP)\end{array}}{v/(v\backslash NP\backslash NP)} >\text{B}\,(v' = (v\backslash NP))$$

Formal Power

I will show that the use of variables in assigned lexical categories increases the weak generative capacity of CCGs. VAR-CCGs, CCGs using variables, can generate languages that are known not to be Tree-Adjoining Languages; therefore VAR-CCGs are more powerful than the weakly equivalent TAG and CCG formalisms.

The following language is known not to be a TAL:

$$L = \{a^n b^n c^n d^n e^n | n \geq 0\}$$

The following VAR-CCG, G', generates a language L' which is very similar to L:
$f(\epsilon) = S$,
$f(a) = A$,
$f(b) = v\backslash A/(v\backslash B)$,
$f(c) = v\backslash B/(v\backslash C)$,
$f(d) = v\backslash C/(v\backslash D)$,
$f(e) = S\backslash D/S$.

The rules allowed in this grammar are forward and backward application and forward crossing composition with $n \leq 2$. The variable v can bind an arbitrarily large category in the infinite set of categories $C(V_N)$ defined for the grammar.

In the language generated by this grammar, two characters of the same type can combine together using the forward crossing composition rule >Bx(2). The composition of the types for the character **e** is shown below. A string of e's can be constructed by allowing the result of this composition to combine with another e category.

$$\frac{\begin{array}{cc}\text{e} & \text{e} \\ S\backslash D/S & S\backslash D/S\end{array}}{S\backslash D\backslash D/S} >\text{Bx(2)}$$

The types for the characters b, c, and d can combine using the same composition rule; these types contain variables (e.g. v and v' below) which can bind to a category of unbounded size.

$$\frac{\begin{array}{cc}\text{b} & \text{b} \\ v\backslash A/(v\backslash B) & v'\backslash A/(v'\backslash B)\end{array}}{v\backslash A\backslash A/(v\backslash B\backslash B)} >\text{Bx(2)}\,(v' = (v\backslash B))$$

By applying the forward crossing composition rule to a string of n b's, we can form the complex category $v\backslash A_1 \ldots A_n/(v\backslash B_1 \ldots B_n)$ representing this string.

Thus, during the derivation of $a^n b^n c^n d^n e^n$ for $n \geq 0$, the following complex categories are created:

$v\backslash A_1 \ldots A_l/(v\backslash B_1 \ldots B_l)$
$v\backslash B_1 \ldots B_k/(v\backslash C_1 \ldots C_k)$
$v\backslash C_1 \ldots C_j/(v\backslash D_1 \ldots D_j)$
$s\backslash D_1 \ldots D_i$

Once the complex categories for a string of b's, a string of c's, a string of d's, and a string of e's are constructed, we can link one string of a particular character to another using the forward application rule. This rule can only apply to these categories if $i = j, j = k, k = l$, and $l = m$ where m is the number of A's generated and i, j, k, l are as in the complex categories listed above. For example,

$$\frac{v\backslash C_1 \ldots C_j/(v\backslash D_1 \ldots D_j) \quad s\backslash D_1 \ldots D_i}{s\backslash C_1 \ldots C_j} >\,(j = i)$$

With each succesful forward application, we ensure that there are equal numbers of two characters: the E's are linked to the D's, the D's are linked to the C's, etc., so that we have the exact same number of all five characters. In fact, the grammar can be easily extended to generate a language such as $\{a_1^n a_2^n \ldots a_k^n | n \geq 0\}$ for any k.

The language L' generated by G' intersected with the regular language $a^*b^*c^*d^*e^*$ gives the language L above. If we assume that L' is a Tree-Adjoining Language (TAL), then L would be a TAL as well since TALs are closed under intersection with Regular languages. However, since we know that L is not a TAL, L' cannot be a TAL either. Thus, G' generates a language that TAGs and CCGs cannot.

Conclusions

We have seen that using variables in the lexical categories of a CCG can increase its weak generative capacity. However, there is some linguistic motivation for looking at the more powerful formalism of VAR-CCGs. As argued by Gazdar (1985), this extra power may be necessary in order to capture coordination in natural languages. We have seen that type-raised categories with variables in CCGs can be used to capture syntactic coordination and extraction facts in Dutch (Steedman 1985). Further research is needed to decide whether this linguistic motivation warrants the move to a more powerful formalism.

Although VAR-CCGs have a greater weak generative capacity than the class including TAGs, HGs, CCGs, and LIGs, we conjecture that it is still a mildly context-sensitive grammar as defined by Joshi (1985). The language discussed above is a mildly context-sensitive language since it observes the constant growth and semilinearity properties. It is an open question whether VAR-CCGs can generate languages which are beyond mildly context-sensitive. Note that MC-TAGs, which are a more powerful extension of TAGs, can also generate languages like L, and they are known to be mildly context-sensitive formalisms (Weir 1988). In future research, we will investigate exactly what the resulting generative capacity of VAR-CCGs is.

Future Research on Word Order

My current research also involves extending the CCG formalism to handle free word order languages. By representing NPs as type-raised categories, we can derive a scrambled sentence in which the NPs do not occur in the order that the verb specifies:

$$\dfrac{v/(v\backslash NP_2) \qquad \dfrac{v/(v\backslash NP_1) \qquad S\backslash NP_1\backslash NP_2}{S\backslash NP_2}>\mathrm{B}}{S}>$$

In many free word order languages, an NP can be scrambled an unbounded distance away from its verb, i.e. *long distance scrambling*. If we allow unrestricted composition rules for any n arguments as well as the use of variables in type-raised categories in a CCG, a string of any number of scrambled NPs followed by a string of verbs can be derived. We first combine any number of verbs together, using backward composition, to get a complex verb category looking for all of the NPs; next, we combine each NPs with this complex verb category. Any type-raised Np_i can combine with the complex verb regardless of the order specified by the complex verb. The variable in the type-raised category can bind a verbal category of unbounded size, e.g. $(v = S\backslash Np_1\backslash \ldots \backslash Np_{i-1})$.

$$\dfrac{v/(v\backslash Np_i) \qquad S\backslash Np_1\backslash Np_2 \ldots \backslash Np_i \ldots \backslash Np_n}{S\backslash Np_1\backslash \ldots \backslash Np_{i-1}\backslash Np_{i+1}\ldots\backslash Np_n}>\mathrm{Bx(n)}$$

Although we can capture scrambling by using variables in type-raised categories, this analysis is not consistent with incremental processing and cannot account for coordination in scrambled sentences; for instance, in the first example given above, NP_2 and NP_1 cannot combine together before combining with the verb. In future research, I will investigate whether VAR-CCGs is an adequate linguistic formalism in capturing all aspects of free word order languages or whether a formalism such as {}-CCGs (Hoffman 1992), which allows sets of arguments in function categories, is better suited.

References

[1] Gazdar, G. 1985. *Applicability of Indexed Grammars to Natural Languages.* Technical Report CSLI-85-34, Center for Study of Language and Information.

[2] Hoffman, Beryl. 1992. A CCG Approach to Free Word Order Languages. *Proceedings of the 30th Annual Meeting of ACL*, Student Session.

[3] Joshi, A.K., 1985. How much context-sensitivity is required to provide reasonable structural descriptions: Tree adjoining grammars. in D. Dowty and L. Karttunen and A. Zwicky, editors, *Natural Language Parsing: Psycholinguistic, Computational and Theoretical Perspectives*, Cambridge University Press.

[4] Steedman, Mark. 1985. Dependency and Coordination in the Grammar of Dutch and English. *Language*, 61, 523-568.

[5] Weir, David. 1988. *Characterising Mildly Context-sensitive Grammar Formalisms.* Ph.D dissertation. University of Pennsylvania.

[6] Weir, David and Aravind Joshi. 1988. Combinatory Categorial Grammars: Generative Power and Relationship to Linear Context-Free Rewriting Systems. *Proceedings of the 26th Annual Meeting of ACL.*