A CCG APPROACH TO FREE WORD ORDER LANGUAGES

Beryl Hoffman *

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

INTRODUCTION

In this paper, I present work in progress on an extension of Combinatory Categorical Grammars, CCGs, (Steedman 1985) to handle languages with freer word order than English, specifically Turkish. The approach I develop takes advantage of CCGs' ability to combine the syntactic as well as the semantic representations of adjacent elements in a sentence in an incremental manner. The linguistic claim behind my approach is that free word order in Turkish is a direct result of its grammar and lexical categories; this approach is not compatible with a linguistic theory involving movement operations and traces.

A rich system of case markings identifies the predicate-argument structure of a Turkish sentence, while the word order serves a pragmatic function. The pragmatic functions of certain positions in the sentence roughly consist of a sentence-initial position for the topic, an immediately pre-verbal position for the focus, and post-verbal positions for backgrounded information (Erguvanli 1984). The most common word order in simple transitive sentences is SOV (Subject-Object-Verb). However, all of the permutations of the sentence seen below are grammatical in the proper discourse situations.

- (1) a. Ayşe gazeteyi okuyor.
Ayşe newspaper-acc read-present.
Ayşe is reading the newspaper.
b. Gazeteyi Ayşe okuyor.
c. Ayşe okuyor gazeteyi.
d. Gazeteyi okuyor Ayşe.
e. Okuyor gazeteyi Ayşe.
f. Okuyor Ayşe gazeteyi.

Elements with overt case marking generally can scramble freely, even out of embedded clauses. This suggests a CCG approach where case-marked elements are functions which can combine with one another and with verbs in any order.

*I thank Young-Suk Lee, Michael Niv, Jong Park, Mark Steedman, and Michael White for their valuable advice. This work was partially supported by ARO DAAL03-89-C-0031, DARPA N00014-90-J-1863, NSF IRI 90-16592, Ben Franklin 91S.3078C-1.

Karttunen (1986) has proposed a Categorical Grammar formalism to handle free word order in Finnish, in which noun phrases are functors that apply to the verbal basic elements. Our approach treats case-marked noun phrases as functors as well; however, we allow verbs to maintain their status as functors in order to handle object-incorporation and the combining of nested verbs. In addition, CCGs, unlike Karttunen's grammar, allow the operations of composition and type raising which have been useful in handling a variety of linguistic phenomena including long distance dependencies and nonconstituent coordination (Steedman 1985) and will play an essential role in this analysis.

AN OVERVIEW OF CCGs

In CCGs, grammatical categories are of two types: curried functors and basic categories to which the functors can apply. A category such as X/Y represents a function looking for an argument of category Y on its right and resulting in the category X . A basic category such as X serves as a shorthand for a set of syntactic and semantic features.

A short set of combinatory rules serve to combine these categories while preserving a transparent relation between syntax and semantics. The application rules allow functors to combine with their arguments.

Forward Application ($>$):

$$X/Y \quad Y \Rightarrow X$$

Backward Application ($<$):

$$Y \quad X \backslash Y \Rightarrow X$$

In addition, CCGs include composition rules to combine together two functors syntactically and semantically. If these two functors have the semantic interpretation F and G , the result of their composition has the interpretation $\lambda x F(Gx)$.

Forward Composition ($> B$):

$$X/Y \quad Y/Z \Rightarrow X/Z$$

Backward Composition ($< B$):

$$Y \backslash Z \quad X \backslash Y \Rightarrow X \backslash Z$$

Forward Crossing Composition ($> Bx$):

$$X/Y \quad Y \backslash Z \Rightarrow X \backslash Z$$

Backward Crossing Composition ($< Bx$):

$$Y/Z \quad X \backslash Y \Rightarrow X/Z$$

FREE WORD ORDER IN CCGs

Representing Verbs:

In this analysis, we represent both verbs and case-marked noun phrases as functors. In Karttunen's analysis (1986), although a verb is a basic element rather than a functor, its arguments are specified as subcategorization features of its basic element category. We choose to directly represent a verb's subcategorization in its functor category. An advantage of this approach is that at the end of a parse, we do not need an extra process to check if all the arguments of a verb have been found; this falls out of the combination rules. Also, certain verbs need to act as active functors in order to combine with objects without case marking.

Following a suggestion of Mark Steedman, I define the verb to be an uncurried function which specifies a *set* of arguments that it can combine with in any order. For instance, a transitive verb looking for a nominative case noun phrase and an accusative case noun phrase has the category $S|\{Nn, Na\}$. The slash $|$ in this function is undetermined in direction; *direction* is a feature which can be specified for each of the arguments, notated as an arrow above the argument, e.g. $S|\{\overrightarrow{n}\}$. Since Turkish is not strictly verb final, most verbs will not specify the direction features of their arguments.

The use of uncurried notation allows great freedom in word order among the arguments of a verb. However, we will want to use the curried notation for some functors to enforce a certain ordering among the functors' arguments. For example, object nouns or clauses without case-marking cannot scramble at all and must remain in the immediately pre-verbal position. Thus, verbs which can take a so called *incorporated* object will also have a curried functor category such as $S|\{Nn, Nd\}|\{\overleftarrow{n}\}$ forcing the verb to first apply to a noun without case-marking to its immediate left before combining with the rest of its arguments.

Representing Nouns:

The interaction between case-marking and the ability to scramble in Turkish supports the theory that case-marked nouns act as functors. Following Steedman (1985), order-preserving type-raising rules are used to convert nouns in the grammar into functors over the verbs. The following rules are obligatorily activated in the lexicon when case-marking morphemes attach to the noun stems.

Type Raising Rules:

$$N + \text{case} \Rightarrow (v|\{\dots\}) | \overleftarrow{\{v|\{N\text{case}, \dots\}\}}$$

$$N + \text{case} \Rightarrow (v|\{\dots\}) | \overleftarrow{\{v|\{N\text{case}, \dots\}\}}$$

The first rule indicates that a noun in the presence of a case morpheme becomes a functor looking for a verb on its right; this verb is also a functor looking

for the original noun with the appropriate case on its left. After the noun functor combines with the appropriate verb, the result is a functor which is looking for the remaining arguments of the verb. v is actually a variable for a verb phrase at any level, e.g. the verb of the matrix clause or the verb of an embedded clause. The notation $\dots$ is also a variable which can unify with one or more elements of a set.

The second type-raising rule indicates that a case-marked noun is looking for a verb on its left. Our CCG formalism can model a strictly verb-final language by restricting the noun phrases of that language to the first type-raising rule. Since most, but not all, case-marked nouns in Turkish can occur behind the verb, certain pragmatic and semantic properties of a Turkish noun determine whether it can type-raise using either rule or is restricted to only the first rule.

The Extended Rules:

We can extend the combinatory rules for uncurried functions as follows. The sets indicated by braces in these rules are order-free, i.e. Y in the following rules can be any element in the set.¹

Forward Application' ($>$):

$$X|\{\overrightarrow{Y}, \dots\} \quad Y \Rightarrow X|\{\dots\}$$

Backward Application' ($<$):

$$Y \quad X|\{\overleftarrow{Y}, \dots\} \Rightarrow X|\{\dots\}$$

Using these new rules, a verb can apply to its arguments in any order, or as in most cases, the case-marked noun phrases which are type-raised functors can apply to the appropriate verbs.

Certain coordination constructions (such as SO and SOV, SOV and SO) force us to allow two type-raised noun phrases which are looking for the same verb to combine together. Since both noun phrases are functors, the application rules above do not apply. The following composition rules are proposed to allow the combining of two functors.

Forward Composition' ($> B$):

$$X|\{\overrightarrow{Y}, \dots_1\} \quad Y|\{\dots_2\} \Rightarrow X|\{\dots_1, \dots_2\}$$

Backward Composition' ($< B$):

$$Y|\{\dots_1\} \quad X|\{\overleftarrow{Y}, \dots_2\} \Rightarrow X|\{\dots_1, \dots_2\}$$

The following example demonstrates these rules in analyzing sentence (1)b in the scrambled word order Object-Subject-Verb:²

¹We assume that a category $X|\{\}$ where $\{\}$ is the empty set rewrites by some clean-up rule to just X .

²The bindings of the first composition are $v_2 = v_1$, $\{\dots_2\} = \{\overrightarrow{Na}, \dots_1\}$.