8 Categorial Grammar

Categorial Grammar is the second oldest of the approaches discussed in this book. It was developed in the 30s by the Polish logician Kazimierz Ajdukiewicz (Ajdukiewicz 1935). Since syntactic and semantic descriptions are tightly connected and all syntactic combinations correspond to semantic ones, Categorial Grammar is popular amongst logicians and semanticists. Some stellar works in the field of semantics are those of Richard Montague (1974).

Other important works come from David Dowty in Columbus, Ohio (1979), Michael Moortgat in Utrecht (1989), Glyn Morrill in Barcelona (1994), Bob Carpenter in New York (1998) and Mark Steedman in Edinburgh (1991; 1997; 2000). A large fragment for German using Montague Grammar has been developed by von Stechow (1979). The 2569-page grammar of the *Institut für Deutsche Sprache* in Mannheim (Eroms, Stickel & Zifonun 1997) contains Categorial Grammar analyses in the relevant chapters. Fanselow (1981) worked on morphology in the framework of Montague Grammar. Uszkoreit (1986a), Karttunen (1986, 1989) and Calder, Klein & Zeevat (1988) developed combinations of unification-based approaches and Categorial Grammar.

The basic operations for combining linguistic objects are rather simple and well-understood, so that it is no surprise that there are many systems for the development and processing of Categorial Grammars (Yampol & Karttunen 1990; Carpenter 1994; Bouma & van Noord 1994; Lloré 1995; König 1999; Moot 2002; White & Baldridge 2003; Baldridge, Chatterjee, Palmer & Wing 2007; Morrill 2012). An important contribution has been made by Mark Steedman's group (see for instance Clark, Hockenmaier & Steedman 2002; Clark & Curran 2007).

Implemented fragments exist for the following languages:

- German (Uszkoreit 1986a; König 1999; Vierhuff, Hildebrandt & Eikmeyer 2003; Vancoppenolle, Tabbert, Bouma & Stede 2011)
- English (Villavicencio 2002; Baldridge 2002; Beavers 2003, 2004)
- Finish (Karttunen 1989)
- French (Baschung, Bes, Corluy & Guillotin 1987)
- Dutch (Bouma & van Noord 1994; Baldridge 2002)
- Tagalog (Baldridge 2002)
- Turkish (Hoffman 1995; Baldridge 2002)

In addition, Baldridge, Chatterjee, Palmer & Wing (2007: 15) mention an implementation for Classical Arabic.

Some of the systems for the processing of Categorial Grammars have been augmented by probabilistic components, so that the processing is robust (Osborne & Briscoe 1997; Clark, Hockenmaier & Steedman 2002). Some systems can derive lexical items from corpora and Briscoe (2000) and Villavicencio (2002) use statistical information in their UG-based language acquisition models.

8.1 General remarks on the representational format

8.1.1 Representation of valence information

In Categorial Grammar, complex categories replace the SUBCAT feature that is used in GPSG to ensure that a head can only be used with suitable grammatical rules. Simple phrase structure rules can be replaced with complex categories as follows:

(1)

Rule	Category in the lexicon
vp → v(ditrans) np np	(vp/np)/np
vp → v(trans) np	vp/np
vp → v(np_and_pp) np pp(to)	(vp/pp)/np

vp/np stands for something that needs an np in order for it to form a vp.

In Categorial Grammar, there are only a few very abstract rules. One of these is forward application, also referred to as the multiplication rule:

(2) forward application:
X/Y ∗ Y = X

This rule combines an X looking for a Y with a Y and requires that Y occurs to the right of X/Y. The result of this combination is an X that no longer requires a Y. X/Y is called the *functor* and Y is the *argument* of the functor.

Valence is encoded only once in Categorial Grammar, as in GB theory, in the lexicon. In GPSG, valence information was present in grammatical rules and in the SUBCAT feature of the lexical entry.

Figure 8.1 shows how a lexical entry for a transitive verb is combined with its object. A derivation in CG is basically a binary branching tree, it is however mostly represented

$$\dfrac{\dfrac{chased}{vp/np} \quad \dfrac{Mary}{np}}{vp}>$$

Figure 8.1: Combination of a verb and its object (preliminary)

as follows: an arrow under a pair of categories indicates that these have been combined via a combinatorial rule. The direction of this arrow indicates the direction of this combination. The result is given beneath the arrow. Figure 8.2 shows the tree corresponding to Figure 8.1. One usually assumes left associativity for '/', that is, (vp/pp)/np = vp/pp/np.

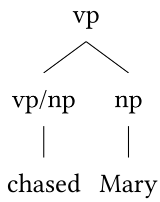

Figure 8.2: Derivation in Figure 8.1 as a tree diagram

If we look at the lexical entries in (1), it becomes apparent that the category v does not appear. The lexicon only determines what the product of combination of a lexical entry with its arguments is. The symbol for vp can also be eliminated: an (English) vp is something that requires an NP to its left in order to form a complete sentence. This can be represented as s\np. Using the rule for backward application, it is possible to compute derivations such as the one in Figure 8.3.

(3) Backward application:
$Y * X\backslash Y = X$

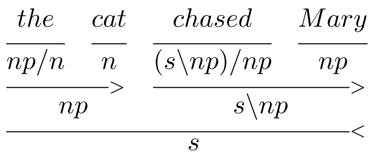

Figure 8.3: Analysis of a sentence with a transitive verb

In Categorial Grammar, there is no explicit difference made between phrases and words: an intransitive verb is described in the same way as a verb phrase with an object: s\np. In the same way, proper nouns are complete noun phrases, which are assigned the symbol np.

8.1.2 Semantics

As already mentioned, Categorial Grammar is particularly popular among semanticists as syntactic combinations always result in parallel semantic combinations and even for complex combinations such as those we will discuss in more detail in the following sections, there is a precise definition of meaning composition. In the following, we will take a closer look at the representational format discussed in Steedman (1997: Section 2.1.2).

Steedman proposes the following lexical entry for the verb *eats*:[1]

(4) eats := (s: *eat′*(x, y)\np$_{3S}$:x)/np:y

In (4), the meaning of each category is given after the colon. Since nothing is known about the meaning of the arguments in the lexical entry of *eat*, the meaning is represented by the variables x and y. When the verb combines with an NP, the denotation of the NP is inserted. An example is given in (5):[2]

(5) $$\frac{(s : eat'(x,y)\backslash np_{3S} : x)/np : y \quad np : apples'}{s : eat'(x, apples')\backslash np_{3S} : x}>$$

When combining a functor with an argument, it must be ensured that the argument fits the functor, that is, it must be unifiable with it (for more on unification see Section 6.6). The unification of np:y with np: *apples′* results in np: *apples′* since *apples′* is more specific than the variable y. Apart from its occurrence in the term np:y, y occurs in the description of the verb in another position (s: *eat′*(x, y)\np$_{3S}$:x) and therefore also receives the value *apples′* there. Thus, the result of this combination is s: *eat′*(x, *apples′*)\np$_{3S}$:x as shown in (5).

Steedman notes that this notation becomes less readable with more complex derivations and instead uses the more standard λ-notation:

(6) eats := (s\np$_{3S}$)/np: $\lambda y.\lambda x.eat'(x,y)$

Lambdas are used to allow access to open positions in complex semantic representations (see Section 2.3). A semantic representation such as $\lambda y.\lambda x.eat'(x,y)$ can be combined with the representation of *apples* by removing the first lambda expression and inserting the denotation of *apples* in all the positions where the corresponding variable (in this case, y) appears (see Section 2.3 for more on this point):

(7) $\lambda y.\lambda x.eat'(x,y)$ *apples′*
$\lambda x.eat'(x, apples')$

This removal of lambda expressions is called β-reduction.

If we use the notation in (6), the combinatorial rules must be modified as follows:

(8) X/Y:f * Y:a = X: f a
Y:a * X\Y:f = X: f a

In such rules, the semantic contribution of the argument (a) is written after the semantic denotation of the functor (f). The open positions in the denotation of the functor are represented using lambdas. The argument can be combined with the first lambda expression using β-reduction.

Figure 8.4 on the next page shows the derivation of a simple sentence with a transitive verb. After forward and backward application, β-reduction is immediately applied.

[1] I have adapted his notation to correspond to the one used in this book.
[2] The assumption that *apples* means *apples′* and not *apples′*(z) minus the quantifier contribution is a simplification here.

$$
\begin{array}{ccc}
\dfrac{Jacob}{np : jacob'} & \dfrac{eats}{(s\backslash np)/np : \lambda y.\lambda x.eat'(x,y)} & \dfrac{apples}{np : apples'}
\end{array}
$$

$$
\dfrac{(s\backslash np)/np : \lambda y.\lambda x.eat'(x,y) \quad np : apples'}{\text{s\np: } \lambda y.\lambda x.eat'(x,y)\ apples' = \lambda x.eat'(x, apples')} >
$$

$$
\dfrac{np : jacob' \quad \text{s\np: } \lambda x.eat'(x, apples')}{\text{s: } \lambda x.eat'(x, apples')\ jacob' = eat'(jacob', apples')} <
$$

Figure 8.4: Meaning composition in Categorial Grammar

8.1.3 Adjuncts

As noted in Section 1.6, adjuncts are optional. In phrase structure grammars, this can be captured for example by rules that have a certain element (for instance a VP) on the left-hand side of the rule and the same element and an adjunct on the right-hand side of the rule. Since the symbol on the left is the same as the one on the right, this rule can be applied arbitrarily many times. (9) shows some examples of this:

(9) a. VP $\rightarrow$ VP PP
 b. Noun $\rightarrow$ Noun PP

One can analyze an arbitrary amount of PPs following a VP or noun using these rules.

In Categorial Grammar, adjuncts have the following general form: X\X or X/X. Adjectives are modifiers, which must occur before the noun. They have the category n/n. Modifiers occurring after nouns (prepositional phrases and relative clauses) have the category n\n instead.[3] For VP-modifiers, X is replaced by the symbol for the VP (s\np) and this yields the relatively complex expression (s\np)\(s\np). Adverbials in English are VP-modifiers and have this category. Prepositions that can be used in a PP modifying a verb require an NP in order to form a complete PP and therefore have the category ((s\np)\(s\np))/np. Figure 8.5 on the following page gives an example of an adverb (*quickly*) and a preposition (*round*). Note that the result of the combination of *round* and *the garden* corresponds to the category of the adverb ((s\np)\(s\np)). In GB theory, adverbs and prepositions were also placed into a single class (see page 96). This overarching class was then divided into subclasses based on the valence of the elements in question.

8.2 Passive

In Categorial Grammar, the passive is analyzed by means of lexical rule (Dowty 1978: 412; Dowty 2003: Section 3.4). (10) shows the rule in Dowty (2003: 49).

(10) Syntax: $\alpha \in$ (s\np)/np $\rightarrow$ PST-PART(α) $\in$ PstP/np$_{by}$
 Semantics: α' $\rightarrow \lambda y \lambda x \alpha'(y)(x)$

[3] In Categorial Grammar, there is no category symbol like $\overline{\text{X}}$ for intermediate projections of $\overline{\text{X}}$ theory. So rather than assuming $\overline{\text{N}}/\overline{\text{N}}$, CG uses n/n. See Exercise 2.

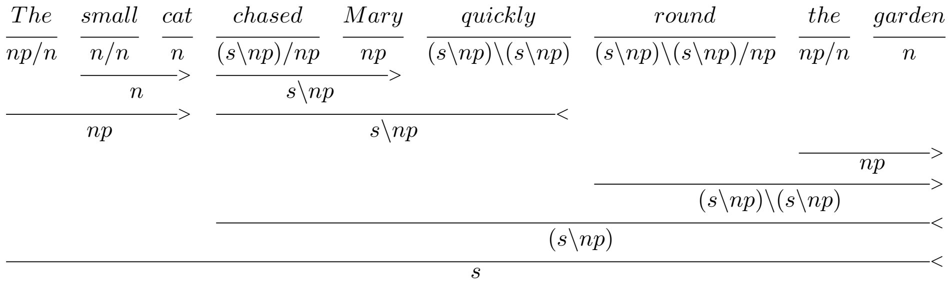

Figure 8.5: Example of an analysis with adjuncts in Categorial Grammar

Here, PstP stands for past participle and np_{by} is an abbreviation for a verb phrase modifier of the form vp\vp or rather (s\np)\(s\np). The rule says the following: if a word belongs to the set of words with the category (s\np)/np, then the word with past participle morphology also belongs in the set of words with the category PstP/np_{by}.

(11a) shows the lexical entry for the transitive verb *touch* and (11b) the result of rule application:

(11) a. touch: (s\np)/np
     b. touched: PstP/np_{by}

The auxiliary *was* has the category (s\np)/PstP and the preposition *by* has the category np_{by}/np, or its unabbreviated form ((s\np)\(s\np))/np. In this way, (12) can be analyzed as in Figure 8.6.

(12) John was touched by Mary.

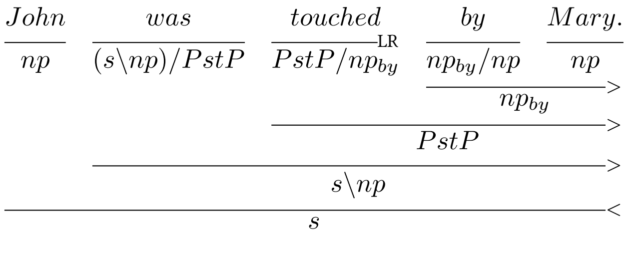

Figure 8.6: Analysis of the passive using a lexical rule

The question as to how to analyze the pair of sentences in (13) still remains unanswered.[4]

(13) a. He gave the book to Mary.
     b. The book was given to Mary.

[4] Thanks to Roland Schäfer (p. m., 2009) for pointing out these data to me.

gave has the category ((s\np)/pp)/np, that is, the verb must first combine with an NP (*the book*) and a PP (*to Mary*) before it can be combined with the subject. The problem is that the rule in (10) cannot be applied to *gave* with a *to*-PP since the pp argument is sandwiched between both np arguments in ((s\np)/pp)/np. One would have to generalize the rule in (10) somehow by introducing new technical means[5] or assume additional rules for cases such as (13b).

8.3 Verb position

Steedman (2000: 159) proposed an analysis with variable branching for Dutch, that is, there are two lexical entries for *at* 'eat': an initial one with its arguments to the right, and another occupying final position with its arguments to its left.

(14) a. *at* 'eat' in verb-final position: $(s_{+\mathrm{SUB}}\backslash np)\backslash np$
b. *at* 'eat' in verb-initial position: $(s_{-\mathrm{SUB}}/np)/np$

Steedman uses the feature SUB to differentiate between subordinate and non-subordinate sentences. Both lexical items are related via lexical rules.

One should note here that the NPs are combined with the verb in different orders. The normal order is:

(15) a. in verb-final position: $(s_{+\mathrm{SUB}}\backslash np[nom])\backslash np[acc]$
b. in verb-initial position: $(s_{-\mathrm{SUB}}/np[acc])/np[nom]$

The corresponding derivations for German sentences with a bivalent verb are shown in Figures 8.7 and 8.8.

$$
\begin{array}{ccc}
er & ihn & isst \\
\hline
np[nom] & np[acc] & (s_{+\mathrm{SUB}}\backslash np[nom])\backslash np[acc] \\
\cline{2-3}
& \multicolumn{2}{c}{s_{+\mathrm{SUB}}\backslash np[nom] \quad <} \\
\hline
\multicolumn{3}{c}{s_{+\mathrm{SUB}} \quad <}
\end{array}
$$

Figure 8.7: Analysis of verb-final sentences following Steedman

$$
\begin{array}{ccc}
isst & er & ihn \\
\hline
((s_{-\mathrm{SUB}}/np[acc])/np[nom] & np[nom] & np[acc] \\
\cline{1-2}
\multicolumn{2}{c}{s_{-\mathrm{SUB}}/np[acc] \quad >} & \\
\hline
\multicolumn{3}{c}{s_{-\mathrm{SUB}} \quad >}
\end{array}
$$

Figure 8.8: Analysis of verb-initial sentences following Steedman

[5] Baldridge (p. M. 2010) suggests using regular expressions in a general lexical rule for passive.

In Figure 8.7 the verb is first combined with an accusative object, whereas in Figure 8.8, the verb is first combined with the subject. For criticism of these kinds of analyses with variable branching, see Netter (1992) and Müller (2005b, 2015b).

Jacobs (1991) developed an analysis which corresponds to the verb movement analysis in GB. He assumes verb-final structures, that is, there is a lexical entry for verbs where arguments are selected to the left of the verb. A transitive verb would therefore have the entry in (16a). Additionally, there is a trace in verb-final position that requires the arguments of the verb and the verb itself in initial position. (16b) shows what the verb trace looks like for a transitive verb in initial position:

(16) a. Verb in final position:
(s\np[nom])\np[acc]
b. Verb trace for the analysis of verb-first:
((s\((s\np[nom])\np[acc]))\np[nom])\np[acc]

The entry for the verb trace is very complex. It is probably simpler to examine the analysis in Figure 8.9.

$$
\begin{array}{cccc}
isst & er & ihn & \_ \\
\hline
(s\backslash np[nom])\backslash np[acc] & np[nom] & np[acc] & (((s\backslash(s\backslash np[nom])\backslash np[acc])\backslash np[nom])\backslash np[acc] \\
\cline{3-4}
 & & \multicolumn{2}{c}{(s\backslash(s\backslash np[nom])\backslash np[acc])\backslash np[nom]} \; < \\
\cline{2-4}
 & \multicolumn{3}{c}{s\backslash((s\backslash np[nom])\backslash np[acc])} \; < \\
\hline
\multicolumn{4}{c}{s} \; <
\end{array}
$$

Figure 8.9: Analysis of verb-initial sentences following Jacobs (1991)

The trace is the head in the entire analysis: it is first combined with the accusative object and then with the subject. In a final step, it is combined with the transitive verb in initial-position.[6] A problem with this kind of analysis is that the verb *isst* 'eats', as well as *er* 'he' and *ihn* 'him'/'it', are arguments of the verb trace in (17).

(17) Morgen [isst [er [ihn _]]]
tomorrow eats he him
'He will eat it/him tomorrow.'

Since adjuncts can occur before, after or between arguments of the verb in German, one would expect that *morgen* 'tomorrow' can occur before the verb *isst*, since *isst* is just a normal argument of the verbal trace in final position. As adjuncts do not change the categorial status of a projection, the phrase *morgen isst er ihn* 'tomorrow he eats him' should be able to occur in the same positions as *isst er ihn*. This is not the case, however. If we replace *isst er ihn* by *morgen isst er ihn* in (18a) the result is (18b), which is ungrammatical.

[6] See Netter (1992) for a similar analysis in HPSG.

(18) a. Deshalb isst er ihn.
therefore eats he him
'Therefore he eats it/him.'

b. * Deshalb morgen isst er ihn.
therefore tomorrow eats he him

An approach which avoids this problem comes from Kiss & Wesche (1991) (see Section 9.3). Here, the authors assume that there is a verb in initial position which selects a projection of the verb trace. If adverbials are only combined with verbs in final-position, then a direct combination of *morgen* 'tomorrow' and *isst er ihn* 'eats he it' is ruled out. If one assumes that the verb in first-position is the functor, then it is possible to capture the parallels between complementizers and verbs in initial position (Höhle 1997): finite verbs in initial position differ from complementizers only in requiring a projection of a verb trace, whereas complementizers require projections of overt verbs:

(19) a. dass [er ihn isst]
that he it eats

b. Isst [er ihn _]
eats he it

This description of verb position in German captures the central insights of the GB analysis in Section 3.2.

8.4 Local reordering

Up to now, we have seen combinations of functors and arguments where the arguments were either to the left or to the right of the functor. The saturation of arguments always took place in a fixed order: the argument furthest to the right was combined first with the functor, e. g. (s\np)/pp first combined with the PP, and the result of this combination was combined with the NP.

There are a number of possibilities to analyze ordering variants in German: Uszkoreit (1986b) suggests accounting for possible orders lexically, that is, that each possible order corresponds to a lexical item. One would therefore have at least six lexical items for a ditransitive verb. Briscoe (2000: 257) and Villavicencio (2002: 96–98) propose a variant of this analysis where the order of arguments is modified in the syntax: a syntactic rule can, for example, change the order (S/PRT)/NP into (S/NP)/PRT.

A different approach is suggested by Steedman & Baldridge (2006). They discuss various options for ordering arguments attested in the languages of the world. This includes languages in which the order of combination is free, as well as languages where the direction of combination is free. Steedman and Baldridge introduce the following convention for representing categories: elements in curly brackets can be discharged in any order. '|' in place of '\' or '/' serves to indicate that the direction of combination is free. Some prototypical examples are shown in (20):

(20)

English	$(S\backslash NP)/NP$	S(VO)
Latin	$S\{\|NP[nom], \|NP[acc]\}$	free order
Tagalog	$S\{/NP[nom], /NP[acc]\}$	free order, verb-initial
Japanese	$S\{\backslash NP[nom], \backslash NP[acc]\}$	free order, verb-final

Hoffman (1995: Section 3.1) has proposed an analysis analogous to that of Japanese for Turkish and this could also be used in conjunction with an analysis of verb position for German. This would correspond to the GB/MP analysis of Fanselow (2001) or the HPSG analysis presented in Section 9.4.

8.5 Long-distance dependencies

Steedman (1989: Section 1.2.4) proposes an analysis of long-distance dependencies without movement or empty elements. For examples such as (21), he assumes that the category of *Harry must have been eating* or *Harry devours* is s/np.

(21) a. These apples, Harry must have been eating.
  b. apples which Harry devours

The fronted NP *these apples* and the relative pronoun *which* are both functors in the analysis of (21) which take s/np as their argument. Using the machinery introduced up to now, we cannot assign the category s/np to the strings in (21) although it is intuitively the case that *Harry devours* is a sentence missing an NP. We still require two further extensions of Categorial Grammar: type raising and forward and backward composition. Both of these operations will be introduced in the following sections.

8.5.1 Type Raising

The category np can be transformed into the category (s/(s\np)) by *type raising*. If we combine this category with (s\np), then we get the same result as if we had combined np and (s\np) with the forward application rule in (2). (22a) shows the combination of an NP with a VP (a sentence missing an NP to its left). The combination of the type-raised NP with the VP is given in (22b).

(22) a. $np * s\backslash np = s$
  b. $s/(s\backslash np) * s\backslash np = s$

In (22a), a verb or verb phrase selects an NP to its left (s\np). In (22b), an NP having undergone type raising selects a verb or verb phrase to its right which requires an NP to its left (s\np).

Type raising simply reverses the direction of selection: the VP in (22a) is the functor and the NP is the argument, whereas in (22b), it is the type raised NP, which acts as the functor, and the VP is the argument. In each case, the result of the combination is the same. This change of selectional direction may just seem like a trick at first glance, but as we will see, this trick can be extremely useful. First, however, we will introduce forward and backward composition.

8.5.2 Forward and backward composition

(23) shows the rules for forward and backward composition.

(23) a. Forward composition (> B)
X/Y ∗ Y/Z = X/Z

b. Backward composition (< B)
Y\Z ∗ X\Y = X\Z

These rules will be explained using forward composition as an example. (23a) can be understood as follows: X/Y more or less means; if I find a Y, then I am a complete X. In the combinatorial rule, X/Y is combined with Y/Z. Y/Z stands for a Y that is not yet complete and is still missing a Z. The requirement that Y must find a Z in order to be complete is postponed: we pretend that Y is complete and use it anyway, but we still bear in mind that something is actually still missing. Hence, if we combine X/Y with Y/Z, we get something which becomes an X when combined with a Z.

8.5.3 Analysis of long-distance dependencies

By using forward composition, we can assign *Harry must have been eating* the category s/np. Figure 8.10 shows how this works. *must* is a verb which requires an unmarked

$$
\begin{array}{cccccc}
\textit{These apples} & \textit{Harry} & \textit{must} & \textit{have} & \textit{been} & \textit{eating} \\
\hline
np & s/(s\backslash np)\ {>}\mathbf{T} & (s\backslash np)/vp & vp/\textit{vp-en} & \textit{vp-en}/\textit{vp-ing} & \textit{vp-ing}/np \\
\cline{2-3}
 & \multicolumn{2}{c}{s/vp \quad {>}\mathbf{B}} & & & \\
\cline{2-4}
 & \multicolumn{3}{c}{s/\textit{vp-en} \quad {>}\mathbf{B}} & & \\
\cline{2-5}
 & \multicolumn{4}{c}{s/\textit{vp-ing} \quad {>}\mathbf{B}} & \\
\cline{2-6}
 & \multicolumn{5}{c}{s/np \quad {>}\mathbf{B}}
\end{array}
$$

Figure 8.10: Application of forward composition to VP-chains

infinitive form, *have* requires a participle and *been* must combine with a gerund. In the above figure, the arrow with a small 'T' stands for type raising, whereas the arrows with a 'B' indicate composition. The direction of composition is shown by the direction of the arrow.

For the analysis of (21a), we are still missing one small detail, a rule that turns the NP at the beginning of the sentence into a functor which can be combined with s/np. Normal type raising cannot handle this because it would produce s/(s\np) when s/(s/np) is required.

Steedman (1989: 217) suggests the rule in (24):

(24) Topicalization (↑):
X ⇒ st/(s/X)
where X ∈ { np, pp, vp, ap, s′ }

st stands for a particular type of sentence, namely one with topicalization.

If we replace X with np, we can turn *these apples* into st/(s/np) and complete the analysis of (21a) as shown in Figure 8.11.

$$
\begin{array}{llllll}
These\ apples & Harry & must & have & been & eating \\
\xrightarrow{>\uparrow} st/(s/np) & \xrightarrow{>\mathbf{T}} s/(s\backslash np) & (s\backslash np)/vp & vp/vp\text{-}en & vp\text{-}en/vp\text{-}ing & vp\text{-}ing/np
\end{array}
$$

$$
\begin{array}{l}
s/(s\backslash np)\ \ (s\backslash np)/vp \xrightarrow{>\mathbf{B}} s/vp \\
s/vp\ \ vp/vp\text{-}en \xrightarrow{>\mathbf{B}} s/vp\text{-}en \\
s/vp\text{-}en\ \ vp\text{-}en/vp\text{-}ing \xrightarrow{>\mathbf{B}} s/vp\text{-}ing \\
s/vp\text{-}ing\ \ vp\text{-}ing/np \xrightarrow{>\mathbf{B}} s/np \\
st/(s/np)\ \ s/np \xrightarrow{>} st
\end{array}
$$

Figure 8.11: Analysis of long-distance dependencies in Categorial Grammar

The mechanism presented here will of course also work for dependencies that cross sentence boundaries. Figure 8.12 shows the analysis for (25):

(25) Apples, I believe that Harry eats.

$$
\begin{array}{llllll}
Apples & I & believe & that & Harry & eats \\
\xrightarrow{>\uparrow} st/(s/np) & \xrightarrow{>\mathbf{T}} s/(s\backslash np) & (s\backslash np)/s' & s'/s & s/(s\backslash np) & (s\backslash np)/np
\end{array}
$$

$$
\begin{array}{l}
s/(s\backslash np)\ \ (s\backslash np)/s' \xrightarrow{>\mathbf{B}} s/s' \\
s/(s\backslash np)\ \ (s\backslash np)/np \xrightarrow{>\mathbf{B}} s/np \\
s/s'\ \ s'/s \xrightarrow{>\mathbf{B}} s/s \\
s/s\ \ s/np \xrightarrow{>\mathbf{B}} s/np \\
st/(s/np)\ \ s/np \xrightarrow{>} st
\end{array}
$$

Figure 8.12: Analysis of long-distance dependencies across sentence boundaries

Using the previously described tools, it is, however, only possible to describe extractions where the fronted element in the sentence would have occurred at the right edge of the phrase without fronting. This means it is not possible to analyze sentences where the middle argument of a ditransitive verb has been extracted (Steedman 1985: 532). Pollard (1988: 406) provides the derivation in Figure 8.13 on the facing page for (26).

(26) Fido we put downstairs.

In this analysis, it is not possible to combine *we* and *put* using the rule in (23a) since (s\np) is not directly accessible: breaking down ((s\np)/pp)/np into functor and argument gives us ((s\np)/pp) and np. In order to deal with such cases, we need another variant of composition:

$$
\begin{array}{cccc}
\textit{Fido} & \textit{we} & \textit{put} & \textit{downstairs}\\
\overline{(st/pp)/((s/pp)/np)}^{>\uparrow\uparrow} & \overline{s/(s\backslash np)}^{>T} & \overline{((s\backslash np)/pp)/np} & \overline{pp}\\
 & \multicolumn{2}{c}{\overline{\qquad (s/pp)/np \qquad}^{>BB}} & \\
\multicolumn{3}{c}{\overline{\qquad\qquad st/pp \qquad\qquad}^{>}} & \\
\multicolumn{4}{c}{\overline{\qquad\qquad\qquad st \qquad\qquad\qquad}^{>}}
\end{array}
$$

Figure 8.13: Analysis of long-distance dependencies across sentence boundaries

(27) Forward composition for n=2 (> BB)
X/Y ∗ (Y/Z1)/Z2 = (X/Z1)/Z2

With this addition, it is now possible to combine the type-raised *we* with *put*. The result is (s/pp)/np. The topicalization rule in (24), however, requires an element to the right of st with the form (s/X). This is not the case in Figure 8.13. For the NP *Fido*, we need a functor category which allows that the argument itself is complex. The rule which is needed for the case in (26) is given in (28).

(28) Topicalization for n=2 (↑↑):
X2 ⇒ (st/X1)/((s/X1)/X2)
where X1 and X2 ∈ { NP, PP, VP, AP, S′ }

If we assume that verbs can have up to four arguments (z. B. *buy*: buyer, seller, goods, price), then it would be necessary to assume a further rule for composition as well as another topicalization rule. Furthermore, one requires a topicalization rule for subject extraction (Pollard 1988: 405). Steedman has developed a notation which provides a compact notation of the previously discussed rules, but if one considers what exactly these representations stand for, one still arrives at the same number of rules that have been discussed here.

8.6 Summary and classification

The operations of Combinatory Categorial Grammar, which go beyond those of standard Categorial Grammar, allow for so much flexibility that it is even possible to assign a category to sequences of words that would not normally be treated as a constituent. This is an advantage for the analysis of coordination (see Section 21.6.2) and furthermore, Steedman (1991) has argued that intonation data support the constituent status of these strings. See also Section 15.2 for a direct model of incremental language processing in Categorial Grammar. In phrase structure grammars, it is possible to use GPSG mechanisms to pass information about relative pronouns contained in a phrase up the tree. These techniques are not used in CG and this leads to a large number of recategorization rules for topicalization and furthermore leads to inadequate analyses of pied-piping constructions in relative clauses. As the topicalization analysis was already discussed in Section 8.5, I will briefly elaborate on relative clauses here.

Steedman & Baldridge (2006: 614) present an analysis of long-distance dependencies using the following relative clause in (29):

(29) the man that Manny says Anna married

The relative pronoun is the object of *married* but occurs outside the clause *Anna married*.

Steedman assumes the lexical entry in (30) for relative pronouns:

(30) (n\n)/(s/np)

This means the following: if there is a sentence missing an NP to the right of a relative pronoun, then the relative pronoun can form an N-modifier (n\n) with this sentence. The relative pronoun is the head (functor) in this analysis.

Utilizing both additional operations of type raising and composition, the examples with relative clauses can be analyzed as shown in Figure 8.14. The lexical entry for the

$$
\begin{array}{ccccc}
that & Manny & says & Anna & married \\
\hline
(n\backslash n)/(s/np) & s/(s\backslash np)\ {>}\mathbf{T} & (s\backslash np)/s & s/(s\backslash np)\ {>}\mathbf{T} & (s\backslash np)/np \\
 & \multicolumn{2}{c}{\overline{\qquad s/s \qquad}{>}\mathbf{B}} & \multicolumn{2}{c}{\overline{\qquad s/np \qquad}{>}\mathbf{B}} \\
 & \multicolumn{4}{c}{\overline{\qquad\qquad s/np \qquad\qquad}{>}\mathbf{B}} \\
\multicolumn{5}{c}{\overline{\qquad\qquad\qquad n\backslash n \qquad\qquad\qquad}{>}}
\end{array}
$$

Figure 8.14: Categorial Grammar analysis of a relative clause with long-distance dependency

verbs corresponds to what was discussed in the preceding sections: *married* is a normal transitive verb and *says* is a verb that requires a sentential complement and forms a VP (s\np) with it. This VP yields a sentence when combined with an NP. The noun phrases in Figure 8.14 have been type raised. Using forward composition, it is possible to combine *Anna* and *married* to yield s/np. This is the desired result: a sentence missing an NP to its right. *Manny* and *says* and then *Manny says* and *Anna married* can also be combined via forward composition and we then have the category s/np for *Manny says Anna married*. This category can be combined with the relative pronoun using forward application and we then arrive at n\n, which is exactly the category for postnominal modifiers.

However, the assumption that the relative pronoun constitutes the head is problematic since one has to then go to some lengths to explain pied-piping constructions such as those in (31).

(31) a. Here's the minister [[in [the middle [of [whose sermon]]]] the dog barked].[7]

b. Reports [the height of the lettering on the covers of which] the government prescribes should be abolished.[8]

[7] Pollard & Sag (1994: 212).

[8] Ross (1967: 109).

In (31), the relative pronoun is embedded in a phrase that has been extracted from the rest of the relative clause. The relative pronoun in (31a) is the determiner of *sermon*. Depending on the analysis, *whose* is the head of the phrase *whose sermon*. The NP is embedded under *of* and the phrase *of whose sermon* depends on *middle*. The entire NP *the middle of the sermon* is a complement of the preposition *in*. It would be quite a stretch to claim that *whose* is the head of the relative clause in (31a). The relative pronoun in (31b) is even more deeply embedded. Steedman (1997: 50) gives the following lexical entries for *who*, *whom* and *which*:

(32) a. ((n\n)/(s\np))\(np/np) (complex subject-relative phrase)
b. ((n\n)/(s/pp))\(pp/np) (complex extracted PP-relative phrase)
c. ((n\n)/(s/np))\(np/np) (complex extracted NP-relative phrase)

Using (32b) and (32c), it is possible to analyze (33a) and (33b):

(33) a. a report the cover of which Keats (expects that Chapman) will design
b. a subject on which Keats (expects that Chapman) will speak

In the analysis of (33b), *which* requires a preposition to its left (pp/np) so it can form the category (n\n)/(s/pp). This category needs a sentence lacking a PP to its right in order to form a post-nominal modifier (n\n). In the analysis of (33a), *the cover of* becomes np/np by means of composition and *which* with the lexical entry (32c) can combine with *the cover of* to its left. The result is the category (n\n)/(s/np), that is, something that requires a sentence missing an NP.

Ross' examples (31b) can also be analyzed as follows (32c):

(34) reports [the height of the lettering on the covers of]$_{\mathrm{np/np}}$ which]$_{\mathrm{(n\backslash n)/(s/np)}}$ the government prescribes

The complex expression *the height of the lettering on the covers of* becomes np/np after composition and the rest of the analysis proceeds as that of (33a).

In addition to entries such as those in (32), we also need further entries to analyze sentences such as (35), where the relative phrase has been extracted from the middle of the clause (see Pollard 1988: 410):

(35) Fido is the dog which we put downstairs.

The problem here is similar to what we saw with topicalization: *we put* does not have the cateory s/np but rather (s/pp)/np and as such, cannot be directly combined with the relative pronoun in (30).

Morrill (1995: 204) discusses the lexical entry in (32b) for the relative pronoun in (36):

(36) about which John talked

In the lexical entry (32b), *which* requires something to the left of it, which requires a noun phrase in order to form a complete prepositional phrase, that is, *which* selects a preposition. Morrill noted that there is a need to postulate further lexical items for cases like (37) in which the relative pronoun occurs in the middle of the fronted phrase.

(37) the contract [the loss of which after so much wrangling] John would finally have to pay for

These and other cases could be handled by additional lexical stipulations. Morrill instead proposes additional types of the combination of functors and arguments, which allow a functor $B \uparrow A$ to enclose its argument A and produce B, or a functor $A \downarrow B$ to enclose its argument to then yield B (p. 190). Even with these additional operations, he still needs the two lexical items in (38) for the derivation of a pied-piping construction with an argument NP or a PP:

(38) a. $(NP \uparrow NP) \downarrow (N\backslash N)/(S/NP)$
b. $(PP \uparrow NP) \downarrow (N\backslash N)/(S/PP)$

These lexical items are still not enough, however, as (38b) contains a PP but this PP corresponds to an argument PP, which is required for (36). To analyze (31a), which involves a PP adjunct, we need to assume the category (s\np)/(s\np) for the prepositional phrase *in the middle of whose seremon*. We therefore also require at least three additional items for relative pronouns.

By introducing new operations, Morrill manages to reduce the number of lexical entries for *which*, however, the fact remains that he has to mention the categories which can occur in pied-piping constructions in the lexical entry of the relative pronoun.

Furthermore, the observation that relative clauses consist of a phrase with a relative pronoun plus a sentence missing a relative phrase is lost. This insight can be kept if one assumes a GPSG-style analysis where information about whether there is a relative pronoun in the relative phrase can be passed up to the highest node of the relative phrase. The relative clause can then be analyzed as the combination of a sentence with a gap and an appropriately marked relative phrase. For the discussion of such analyses in the framework of GB theory and HPSG/CxG, see Section 21.10.3.

Comprehension questions

1. Identify the functors and arguments in Figures 8.1 and 8.3.
2. Which combination operations do you know?
3. What is composition used for?

Exercises

1. Analyze the following sentence:

   (39) The children in the room laugh loudly.

2. Analyze the noun phrase in (40):

(40) the picture of Mary

Compare the resulting analysis with the structure given in Figure 2.4 on page 69 and think about which categories of $\overline{\mathrm{X}}$ syntax the categories in Categorial Grammar correspond to.

Further reading

Mark Steedman discusses a variant of Categorial Grammar, *Combinatory Categorial Grammar*, in a series of books and articles: Steedman (1991, 2000); Steedman & Baldridge (2006).

Lobin (2003) compares Categorial Grammar with Dependency Grammar and Pickering & Barry (1993) suggest a combination of Dependency Grammar and Categorial Grammar, which they call Dependency Categorial Grammar.

Briscoe (2000) and Villavicencio (2002) discuss UG-based acquisition models in the framework of Categorial Grammar.