Extraction and Coordination
in
Phrase Structure Grammar and Categorial Grammar

Glyn Verden Morrill

PhD
University of Edinburgh
1988

To Hilda Verden and Katie Morrill

Acknowledgements

First of all I would like to thank my parents. They gave me the freedom to find my vocation and they have always supported me in my decision to do research; I hope the completion of this thesis brings them the happiness it brings me.

I have been very lucky to find myself at the Centre for Cognitive Science: I am constantly impressed by those around me. I owe a debt of thanks to many people, but I would specifically like to mention Bob Carpenter and Einar Jowsey. They have both provided me with extensive assistence and advice, stimulating discussion, and heartening encouragement. I am grateful to my supervisors, Ewan Klein and Mark Steedman; if my argument holds water it is because they have insisted that it be made to do so; thanks also to Mark Hepple who commented on a late draft of this thesis.

Finally, and most importantly, my thanks to A, for everything.

Abstract

A large proportion of computationally-oriented theories of grammar operate within the confines of *monostratality* (i.e. there is only one level of syntactic analysis), *compositionality* (i.e. the meaning of an expression is determined by the meanings of its syntactic parts, plus their manner of combination), and *adjacency* (i.e. the only operation on terminal strings is concatenation). This thesis looks at two major approaches falling within these bounds: that based on phrase structure grammar (e.g. Gazdar), and that based on categorial grammar (e.g. Steedman).

The theories are examined with reference to extraction and coordination constructions; crucially a range of 'compound' extraction and coordination phenomena are brought to bear. It is argued that the early phrase structure grammar metarules can characterise operations generating compound phenomena, but in so doing require a categorial-like category system. It is also argued that while categorial grammar contains an adequate category apparatus, Steedman's primitives such as composition do not extend to cover the full range of data. A theory is therefore presented integrating the approaches of Gazdar and Steedman.

The central issue as regards processing is derivational equivalence: the grammars under consideration typically generate many semantically equivalent derivations of an expression. This problem is addressed by showing how to axiomatise derivational equivalence, and a parser is presented which employs the axiomatisation to avoid following equivalent paths.

Contents

Chapter I: Introduction *1*

Chapter II: Phrase Structure Grammar Extended with Metarules *19*

Chapter I

Introduction

In accounts of natural language grammar, a distinction is usually drawn between expressions like (1a) and (2a), and their counterparts (1b) and (2b):

(1) a. I liked London
    b. London, I liked

(2) a. I liked London but Suzy hated London
    b. I liked but Suzy hated, London

The 'a' examples are typically considered to be more 'basic' than the 'b' examples. For instance, in classical transformational grammar the former might be base-generated while the latter are only derived via transformation. I will refer to the former as *canonical* and the latter as *non-canonical*. Non-canonicality such as the extraction in (1b) and the coordination in (2b) constitutes a major problem area in natural grammar, and will be the central concern here.

The general approach in this thesis is to characterise canonical English using 'pure' *phrase structure grammar* (PSG) and 'pure' *categorial grammar* (CG), and to augment these basic systems to capture non-canonicality. The augmentation retains the *monostratal* character of the basic formalisms, so that there is a single level of syntactic analysis. It also retains the property of *adjacency* whereby terminal strings are built up by concatenation only. I shall be concerned throughout with *compositionality*, i.e. the manner in which the meanings of expressions are determined by the meanings of their (syntactic) parts, and the rules by which they are formed.

In this chapter I describe equivalent PSG and CG grammars for canonical English. In Chapter II I discuss the metarule augmentation of PSG that originated with Gazdar (1981), and develop a particular grammar for topicalisation, relativisation, right extraposition, heavy shift, parasitic extraction, right node raising, left node raising (coordination reduction or non-constituent conjunction), and across-the-board extraction generally. The complex noun phrase constraint, subject condition, NP constraint, A-over-A constraint, fixed subject constraint and left branch condition are discussed in relation to the grammar. It is noted that English exhibits a whole range of 'compound' instances of extraction and coordination phenomena in which more that one element is displaced, and it is shown that the PSG with metarules does not undergenerate with respect to this data. However I argue that in characterising the data, the account of non-canonicality adopts the category apparatus of categorial grammar, and I also argue that the categorial account of complementation enables a simpler account of feature distribution. In this way a categorial approach is motivated.

In Chapter III therefore I describe the characterisation of non-canonicality by augmentation of CG that originated with Ades and Steedman (1982). I argue that in this case the account does not generalise to compound non-canonicality, and in Chapter IV I present a CG-based metarule account which is a synthesis of the earlier phrase structure and categorial approaches.

In Chapter V I consider various issues relating to universal grammar that arise from the inquiry, and Chapter VI discusses parsing and meaning representation. The thesis is concluded in Chapter VII with some suggestions for further research.

1. Pure Phrase Structure Grammar

A phrase structure grammar contains rules like the following:

(3) a. S → NP VP
    b. VP → TV NP

The interpretation of such rules is that expressions of the categories on the right hand side can be concatenated to form expressions of the category on the left hand side. In addition to rules such as these, a phrase structure grammar will contain a lexical assignment of basic expressions ('words') to categories. If *Bill* and *Mary* are lexically assigned to the category *NP*, and *met* is lexically assigned to the category *TV*, then *Bill met Mary* will belong to the category *S* and will have the analysis shown in Figure 1.

Under an alternative formulation lexical assignments are expressed by phrase structure rules like (4a), and basic expressions may also be introduced *syncategorematically* by rules like (4b).

(4) a. TV → met
    b. REL → who VP

However here lexical assignment will be distinguished from phrase structure rules, and defined to be a function from the set of words into sets of categories. Also, attention will be restricted to rules with exactly two daughter categories. Thus, for example, a transitive

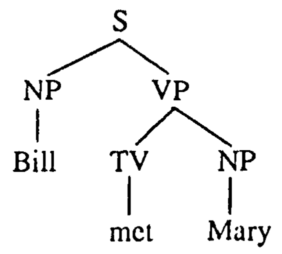


Figure 1

prepositional verb *TPV* will combine first with its direct object to form a prepositional verb phrase *PV*, and then with a prepositional phrase; the rules in (5) assign a nested structure as shown in Figure 2 rather than the more usual flat structure. This requirement is in anticipation of comparison with categorial grammar where the binary structure is standard.

(5) a. PV → TPV NP
b. VP → PV PP
c. PP → TP NP

Although the grammar forming the focus of discussion will be binary, various analytic possibilities are offered by a phrase structure grammar allowing arbitrary degree branching, particularly in conjunction with immediate dominance/linear precedence factoring (Gazdar and Pullum 1981).[1] On occasion the discussion will include references to these possibilities.

Rules will be assigned a simple semantics which will be a function that applies to the meanings of daughter expressions in left-to-right order to give the meaning of the mother

---

[1] The factoring involves definition of phrase structure by means of immediate dominance (ID) rules of the form (i) stating dominance relations that may hold, and linear precedence (LP) rules of the form (ii) stating sister ordering relations that must hold.

(i) X → Y, Z, ...
(ii) X < Y

A local tree is generated an ID/LP grammar if and only if it matches some ID rule and all LP rules.

[2] Alternatively, the semantics of a rule could have been construed as a function that applies simultaneously to an ordered tuple of daughter meanings to give the mother meaning

Although I will talk about meaning throughout, concern will not be with what *meaning* is, but with how meanings are built up, i.e. we will be concerned with *compositionality* itself. The discussion is abstracted over whether the domain of 'meanings' is taken to be built out of Montagovian individuals, truth values, and possible worlds, or if it consists of structures like Lexical-Functional Grammar's f-structures (Kaplan and Bresnan 1982), Kamp's Discourse Representation Structures (Kamp 1981), Webber's Level-1 representations (Webber 1979), or other semantic objects. All that is important here is that meanings can be regarded as set-theoretic objects and functions, and are built up compositionally in the manner prescribed.

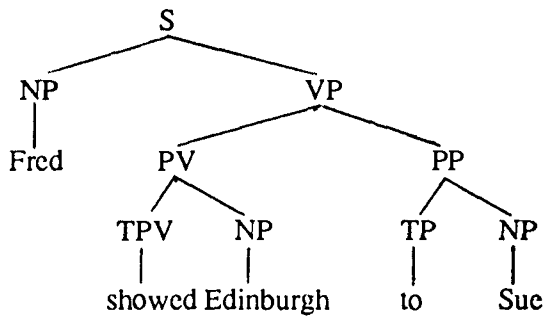


Figure 2

expression.[2] For example:[3]

(6) a. S → NP VP $\lambda x \lambda y[y\ x]$
b. VP → TV NP $\lambda x \lambda y[x\ y]$

The rule (6a) states that in assembling a subject noun phrase and a verb phrase into a sentence, the meaning of the sentence is given by applying the meaning of the verb phrase to that of the noun phrase; rule (6b) states that transitive verb meanings apply to object noun phrase meanings to give verb phrase meanings. Then *Bill met Fred,* as derived earlier, will have the meaning given by the λ-term (7a) which has the reduced form (7b); *met′*, *Fred′*, and *Bill′* denote the meanings of the corresponding words.

(7) a. $\lambda z \lambda w[w\ z]\ \mathrm{Bill}'\ (\lambda x \lambda y[x\ y]\ \mathrm{met}'\ \mathrm{Fred}')$
b. $\mathrm{met}'\ \mathrm{Fred}'\ \mathrm{Bill}'$

Similarly, with rules assigned the semantics in (8), *Fred showed Edinburgh to Sue* has meaning (9).

(8) a. PV → TPV NP $\lambda x \lambda y[x\ y]$
b. VP → PV PP $\lambda x \lambda y[x\ y]$
c. PP → TP NP $\lambda x \lambda y[x\ y]$

(9) $\mathrm{showed}'\ \mathrm{Edinburgh}'\ (\mathrm{to}'\ \mathrm{Sue}')\ \mathrm{Fred}'$

---

[3] In λ-terms application is indicated by juxtaposition and is left-associative.

Here, rules are being assigned semantics directly in a rule-to-rule fashion (cf. Bach 1976). Klein and Sag (1985) show how the semantics of rules can be inferred from the types corresponding to the participating categories, in a process called ***type-driven translation***. For example, if an *NP* is of type $(e \to t) \to t$ and a *VP* is of type $((e \to t) \to t) \to t$ it can be inferred that the semantics of the sentence expansion rule that combines them is to apply the latter to the former, rather than vice-versa. On a type-driven translation approach it would not be necessary to explicitly list a semantics for each rule; rather, this would be inferred on the basis of a category-to-type map. In categorial grammar category symbols encode types directly.

2. *Pure Categorial Grammar*

The characteristic feature of categorial grammar is its category system, but there are many variants. I will first describe the conception and notation assumed here, and then relate this to other versions at the end of the section.

Given a set of basic categories, the full set of categories is recursively defined thus:[4]

(10) a. If X is a basic category
then X is a category
b. If X and Y are categories
then X/Y and X\Y are categories

An expression of category *X/Y* is one which combines with an expression of category *Y* on its right to form an expression of category *X*; an expression of category *X\Y* is one which combines with an expression of category *Y* on its left to form an expression of category *X*. For example a transitive verb may have a category *(S\NP)/NP* whereby it combines with an object on its right and then a subject on its left to form a sentence; similarly a transitive prepositional verb may have a category *((S\NP)/((S\NP)\(S\NP)))/NP* whereby it combines with an object on its right, then a prepositional adverbial, *(S\NP)\(S\NP)* further to its right, and then a subject on its left, to form a sentence. A left-associativity convention will be adopted for slashes so that *(S\NP)/NP* and *((S\NP)/((S\NP)\(S\NP)))/NP* may be written *S\NP/NP* and *S\NP/(S\NP\(S\NP))/NP* respectively. Analyses of ***Bill met Mary*** and ***Fred showed Edinburgh to Sue*** are as follows:

---

[4] The set defined is the smallest one satisfying the specified conditions.

```
(11)   Bill    met     Mary
       ----  -------   ----
        NP   S\NP/NP    NP
             -----------
                 S\NP
       ------------
             S

(12)   Fred          showed              Edinburgh        to           Sue
       ----  ---------------------------  ---------  -------------------  ---
        NP   S\NP/(S\NP\(S\NP))/NP           NP      S\NP\(S\NP)/NP       NP
             -----------------------------------     -------------------
                   S\NP/(S\NP\(S\NP))                     S\NP\(S\NP)
                    -----------------------------------------------------
                                        S\NP
       ---------------------------------------
                       S
```

This notation for analyses is due to Steedman and amounts to an inversion of the usual down-growing tree; the derivation might be more conventionally represented as shown in Figure 3 (cf. Figure 1).

What I will call a *pure categorial grammar* simply consists of a lexicon which is an assignment of basic expressions to directional categories. The distributional behaviour of words and the phrases they form is implicit in their lexical categories. As such, categorial grammar is highly 'lexicalist', with syntactic structure projected directly from lexical categories.

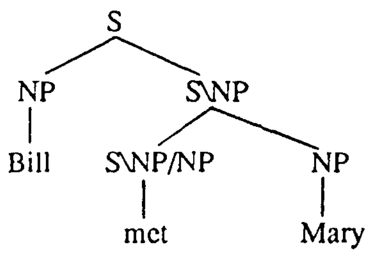


Figure 3

Accompanying the syntactic rule that an expression of category *X/Y* (or *X\Y*) combines with an expression of category *Y* to its right (or left) to form an expression of category *X*, there is a semantic rule that the meaning of the resulting expression is given by applying the meaning of the *X/Y* (or *X\Y*) *functor* subexpression to the meaning of the *Y argument* subexpression. In view of the semantics, combination to the right is referred to as *forward application* and combination to the left is referred to as *backward application*:

(13) a. *Forward Application* (>)
X/Y: x + Y: y => X: x y
b. *Backward Application* (<)
Y: y + X\Y: x => X: x y

The rules show how the meanings of the expressions, after the colons, are to be applied. The meaning of the verb phrase *met Mary* in (11) is (14).

(14) met′ Mary′

The meaning of the sentence *Bill met Mary* is (15), the same as that assigned by the PSG.

(15) met′ Mary′ Bill′

The meaning of *Fred showed Edinburgh to Sue* is likewise the same as that assigned in the PSG:

(16) showed′ Edinburgh′ (to′ Sue′) Fred′

The semantics in the rules of application demands a certain relation between categories and types, in order that meanings are of the right type to apply to each other. Where $\tau(X)$ is the type associated with category *X*, (17) holds.

(17) $\tau(X/Y) = \tau(X\backslash Y) = \tau(Y)\rightarrow\tau(X)$

It is in this connection that categorial grammar can be traced back past Ajdukiewicz to Lesniewski and Husserl and the theory of types.

The particular categorial category system described above is *directional*, i.e. the slashes indicate direction of combination. Bar-Hillel (1953) introduced directional slashes, and these are used in, for example, Lambek (1958, 1961), Lyons (1968), Bach (1983), Dowty (1988), Moortgat (1988), Steedman (1987a,b). In other versions the slash may be non-directional, allowing combination in either direction, or else direction of combination may be governed by some other component of grammar. Ajdukiewicz (1935) had a non-directional slash. Work originating from a semantic perspective, such as Geach (1972), Montague (1973), Bach (1979, 1980), Szabolsci (1983), and van Bentham (1986), tends not to assume directionality. Amongst the non-directional work originating from a more syntactic point of view, Ades and Steedman (1982) and Steedman (1985) constrain the categories that can participate in forward and backward application; and Flynn (1983) employs a general ordering principle.

There is the following argument against non-directional slash categories. The idea of syntactic categories adopted here is that a category is a class of distributionally equivalent expressions, so that grammaticality is preserved under substitution of expressions of like category. Now in a non-directional categorial category system, expressions which apply backwards to expressions of category *Y* to form expressions of category *X* belong to the same category as ones which apply forwards to expressions of category *Y* to form expressions of category *X*. Yet in general such expressions are not distributionally equivalent. Some further component is required to say whether a category is forward-combining or backward-combining. But if the grammar contains some expressions combining forwards with *Y* to form *X*, and some combining backwards for identical *Y* and *X*, the relevant inference of directionality cannot be made on the basis of the categories because while the categories are the same, the required inferences are different. Thus in the absence of directional slashes, facts such as the following demand some additional means of distinction:

(18) a. the happy man
b. *the man happy

(19) a. the man outside
b. *the outside man

(20) a. John will leave
b. *John leave will

(21) a. John dances well
b. *John well dances

There are several distinct category notations occurring in the categorial literature. That used here is one used, for example, by Steedman and Dowty. The reader is warned against confusion with a notation Moortgat has used, according to which the (Steedman) category *(X\Y)/Z* is written *Z/(Y\X)*, and that used by Lambek, according to which *(X\Y)/Z* is written *(Y\X)/Z*.

Categories as they have been defined are fully *curried*, i.e. arguments are taken 'one-at-a-time'. Proposals have been made to allow *non-curried* categories, ones which take arguments 'several-at-once'. For example _S\NP/NP*NP_ might index a category of expressions which combine with two *NP*s simultaneously to yield a function over a third *NP*. See e.g. Ajdukiewicz (1935, p210) and Bar-Hillel (1953, p49) for original proposals, and Wood (1988), and Oehrle (1987), for recent applications. Such categories are not used here.

For proposals to extend categorial grammar beyond the jurisdiction of adjacency, i.e. to include structural operations over and above concatenation, see e.g. Bach (1984) and Huck (1985).

3. *Grammar for Canonical English*

In this section I present PSG and CG grammars for a fragment of canonical English. The grammars are strongly equivalent, i.e. they generate the same strings and assign them the same structures; they also assign the same meanings. The exact equivalence is intended to facilitate comparison later. These grammars will form the 'base' of the augmented grammars covering non-canonicality.

The categories used in the PSG and CG grammars are illustrated in Figure 4. The categories in the categorial grammar are recursively defined over the basic categories noun *N*, noun phrase *NP*, sentence *S*, and complementized sentence *SP*. Other basic category sets are possible: this one has been chosen largely for notational convenience.

For the time being the issue of features will be largely avoided. If desired it would be possible to add features indicating number, person, case, and the verb forms: finite, infinitival, base-form, passive, present or past participial (cf. Gazdar, Klein, Pullum, and Sag 1985), and the kinds of complementizer a complementized sentence has: *that*, *whether*, *for*, etc. For example in phrase structure grammar, atomic categories could be uniformly extended so that subject-verb agreement is achieved by a Definite Clause Grammar (Pereira and Warren 1980) type of positional encoding of feature values as in (22) and (23) where variables are in upper case and values are in lower case.

(22) a. NP[3,NUM,CASE] → DET[NUM] N[NUM]
b. S[VFORM] → NP[PER,NUM,nom] VP[PER,NUM,VFORM]

PSG Categories	CG Categories	
S	S	sentence
VP	S\NP	verb phrase
ADV	S\NP\(S\NP)	adverbial
PP	S\NP\(S\NP)	intransitive preposition
TP	S\NP\(S\NP)/NP	transitive preposition
XP	S\NP\(S\NP)/(S\NP)	control preposition
AUX	S\NP/(S\NP)	auxiliary
XV	S\NP/(S\NP)	control verb
TXV	S\NP/(S\NP)/NP	transitive control verb
TV	S\NP/NP	transitive verb
TTV	S\NP/NP/NP	ditransitive verb
SV	S\NP/SP	sentential verb
TSV	S\NP/SP/NP	transitive sentential verb
TTSV	S\NP/SP/NP/NP	ditransitive sentential verb
PV	S\NP/(S\NP\(S\NP))	prepositional verb
TPV	S\NP/(S\NP\(S\NP))/NP	transitive prepositional verb
COP	S\NP/(N/N)	copula
SP	SP	complementized sentence
COMP	SP/S	complementizer
N	N	common noun
PP	N\N	intransitive preposition
REL	N\N	relative clause
TP	N\N/NP	intransitive preposition
RELPROs	N\N/(S\NP)	subject relative pronoun
RELPROo	N\N/(S/NP)	object relative pronoun
AP	N/N	adjective
Af	N/N/(S\NP/NP)	*tough*-like adjective
SN	N/SP	sentential noun
PN	N/(N\N)	prepositional noun
NP	NP	proper name, noun phrase
DET	NP/N	determiner

Figure 4: Common Categories

(23) a. walks := VP[3,sg,fin]
b. the := DET[NUM]

Uninstantiated variables indicate ambivalence; bound uninstantiated variables achieve the effects of feature percolation. By way of further example, the sentential complement of *prefer* in *he prefers that John stay* has a *that* complementizer and an uninflected main verb. This may be implemented by the following rule and lexical entry:

(24) VP[PER,NUM,VFORM] →
SV[PER,NUM,VFORM,COMP,SUBORDVFORM] SP[COMP,SUBORDVFORM]

(25) prefers := SV[3,sg,fin,that,bse]

In categorial grammar it may be appropriate to likewise structure categories. For example, the category of *prefer* in *he prefers that John stay* might be written S[fin]\NP[3,sg,nom]/SP[that,bse], and that of the definite article *the* might be NP[3,NUM,CASE]/N[NUM]. One question that arises is whether there should be features on complex categories as a whole, rather than just on basic categories; there will be some discussion of this in Section 1.2 of Chapter V.

I will present the basic PSG and CG grammars in parallel, listing the rules and lexical entries of the PSG, and the corresponding lexical entries of the CG.

The following rules and lexical assignments state that a determiner can combine with a noun on its right to form a noun phrase, and that a verb phrase can combine with a noun phrase on its left to form a sentence:

(26) a. NP → DET N λxλy[x y]
the := DET
b. the := NP/N

(27) a. S → NP VP λxλy[y x]
left := VP
b. left := S\NP

Thus:

(28) [[The students] left]

Adverbials such as *quickly* combine with verb phrases on their left to form new verb phrases:

(29) a. VP → VP ADV λxλy[y x]
quickly := ADV
b. quickly := S\NP\(S\NP)

(30) The students [left quickly]

With and *while* form adverbial phrases when they combine with noun phrases and present participial verb phrases on their right respectively:

(31) a. PP → TP NP $\lambda x \lambda y[x\ y]$
with := TP
VP → VP PP $\lambda x \lambda y[y\ x]$
b. with := S\NP\(S\NP)/NP

(32) The students left [with John]

(33) a. ADV → XP VP $\lambda x \lambda y[x\ y]$
while := XP
b. while := S\NP\(S\NP)/(S\NP)

(34) The students grumbled [while leaving]

Note that the category *PP* in the PSG will cut across both adnominal and adverbial preposition phrases, but that the basic categories assumed for the CG do not allow this because nouns *N* and intransitive verb phrases *S\NP* are distinct.[5] The 'X' in 'XP' is intended to indicate that the argument is controlled, though details of how this control is achieved (presumably lexically), are not discussed.

Auxiliary verbs combine with verb phrases on their right to form new verb phrases:

(35) a. VP → AUX VP
will := AUX
b. will := S\NP/(S\NP)

Auxiliary, infinitival *to*, and modal ordering properties can be directly encoded featurally in categorial grammar. Thus for *will have to leave* there is (36) where *S[fin]*, *S[bse]* and *S[inf]* are written *Sfin*, *Sbse* and *Sinf*.

```
(36)       will                 have                 to                   leave
           -----------------    -----------------    -----------------    -------
           Sfin\NP/(Sbse\NP)    Sbse\NP/(Sinf\NP)    Sinf\NP/(Sbse\NP)    Sbse\NP
                                                     -------------------------f
                                                                Sinf\NP
                                -----------------------------------f
                                            Sbse\NP
           ---------------------------------------f
                           Sfin\NP
```

---

[5] Semantically, adnominal and adverbial prepositional phrases will presumably both be of a type mapping $e \to t$ to $e \to t$.

Unacceptable orderings such as **have will to leave* cannot be derived.[6]

Control verbs like *try* which take infinitival verb phrase complements also have category S\NP/(S\NP) in the categorial grammar:

(37) a. VP → XV VP $\lambda x \lambda y[x\ y]$
try := XV
b. try := S\NP/(S\NP)

(38) I [tried [to leave]]

Transitive verbs take a single object; ditransitives take two:

(39) a. VP → TV NP $\lambda x \lambda y[x\ y]$
referenced := TV
b. referenced := S\NP/NP

(40) I [referenced you]

(41) a. TV → TTV NP $\lambda x \lambda y[x\ y]$
b. lent ↦ S\NP/NP/NP

(42) I [lent John] Faust

Complementizers and verbs subcategorized for complementized sentences are characterised thus:

(43) a. SP → COMP S $\lambda x \lambda y[x\ y]$
that := COMP
b. that := SP/S

(44) [that [John left]]

(45) a. VP → SV SP $\lambda x \lambda y[x\ y]$
thinks := SV
b. thinks := S\NP/SP

(46) He [thinks [that John left]]

---

[6] For discussion of auxiliary ordering in phrase structure grammar and categorial grammar see e.g. Gazdar, Pullum and Sag (1982), and Bach (1983) and Carpenter (forthcoming), respectively.

For transitive sentential verbs, and the somewhat exceptional ditransitive sentential verb *bet*, there is:

(47) a. SV → TSV NP λxλy[x y]
told := TSV
b. told := S\NP/SP/NP

(48) She [told Ralph] that she went

(49) a. TSV → TTSV NP λxλy[x y]
bet := TTSV
b. bet := S\NP/SP/NP/NP

(50) She [bet Ralph] five pounds that she would win

I assume that subcategorized prepositional phrases retain the categories they have as adjuncts, and I will regard particles as intransitive prepositions bearing the full prepositional phrase category. This means that for a transitive particle-taking verb, the ordering *rang John up* as opposed to *rang up John* is regarded as canonical, notwithstanding the transformational tradition whereby the former is derived from the latter by 'particle shift'.

(51) a. VP → PV PP λxλy[x y]
searched, looked := PV
b. searched, looked := S\NP/(S\NP\(S\NP))

(52) a. up := PP
b. up := S\NP\(S\NP)

(53) a. We [searched [for Ralph]]
b. We [looked up]

(54) a. PV → TPV NP λxλy[x y]
put, rang := TPV
b. put, rang := S\NP/(S\NP\(S\NP))/NP

(55) a. I [put Faust] on the table
b. I [rang John] up

The copula can combine with predicative elements in general, and adjectives in particular, on its right:[7]

(56) a. VP → COP AP   λxλy[x y]
is := COP
b. is := S\NP/(N/N)

(57) John [is fat]

While adjectives combine forwards with nouns, post-modifiers (adnominals) such as *outside* combine backwards. Complex adnominals include those consisting of a prepositional phrase, and those consisting of a subject relative clause:

(58) a. N → AP N   λxλy[x y]
fat := AP
b. fat := N/N

(59) the [fat man]

(60) a. N → N PP   λxλy[y x]
outside := PP
b. outside := N\N

(61) the [man outside]

(62) a. PP → TP NP   λxλy[x y
from := TP
b. from := N\N/NP

(63) the man [from Edinburgh]

(64) a. REL → RELPROs VP   λxλy[x y]
who := RELPROs
b. who := N\N/(S\NP)

(65) the woman [who swam]

I have assumed that adnominals modify nouns, as opposed to full noun phrases, for the standard reason that this more directly reflects the semantics. Thus in *every woman who*

---

[7] The copula cannot combine with intensional adjectives:
(i) *John is alleged

swam, quantification is over the class of women who swam, just as in *every woman* quantification is over the class of women. See Bach and Cooper (1978) for the alternative proposal, whereby adnominals modify noun phrases, and Janssen (1983, chapter XIII) for criticism of that proposal.

It is assumed that nouns, like verbs, are subcategorized (cf. Chomsky 1970). This assumption is necessary in the case of sentential nouns, though not in the case of prepositional nouns, since nouns could be modified by prepositional phrases 'adjunctivally' anyway, but if noun subcategorization for complementized sentences is being hypothesised, it seems appropriate to assume noun subcategorization for prepositional phrases also.

(66) a. N → SN SP λxλy[x y]
belief := SN
b. belief := N/SP

(67) the [belief [that John went]]

(68) a. N → PN PP
search, picture := PN
b. search, picture := N/(N\N)

(69) a. the [search [for Ralph]]
b. the [picture [of John]]

The account of object relative clauses and *tough*-like adjectives involves non-canonicality and is considered later.

According to the coordination schema of Dougherty (1970, 1971), expressions of like category conjoin to form coordinate structures of that category:

(70) $[\mathrm{X}\ \ \mathrm{Coord}\ \ \mathrm{X}]_{\mathrm{X}}$

Such a schema immediately characterises a range of facts. Thus for example sentences, adverbials, verb phrases, and noun phrases can coordinate with themselves as in (71), but not with each other as in (72):[8]

---

[8] Throughout '*' and '?' are used to indicate (my) acceptability judgements.

(71) a. [John arrived and Mary left]
b. John left [quickly and without saying goodbye]
c. John [picked up his bag and left]
d. [John and Sue] went home

(72) a. *John arrived and without saying hello
b. *John left quickly and Sue

A principle challenge to such a like-category coordination schema is provided by coordination of 'unlike'categories:

(73) John is [rich and an excellent cook]

One possibility is that the identity requirement be 'loosened' in some sense, e.g. as in Gazdar et al. (1985). Another one is that in such cases the conjuncts do actually share some category; see Partee (1986) and Carpenter (forthcoming) on this point. For instance adjectives and indefinite noun phrases seem to have the same character in examples like (74).

(74) a. John came back rich
b. John came back an excellent cook

At any rate, the aim here will be to see just how far a like category coordination schema can take us. A fuller set of schemata for binary and iterative coordination is shown in (75) where '$^+$' indicates one or more repetitions.

(75) *Coordination*
$[X^+ \text{ and } X]_X$
$[(X \text{ and})^+ X]_X$
$[X^+ \text{ or } X]_X$
$[(X \text{ or})^+ X]_X$
$[\text{neither } (X \text{ nor})^+ X]_X$
$[\text{both } X \text{ and } X]_X$
$[\text{either } X \text{ or } X]_X$

For example:

(76) a. John, Fred and Bill
b. John and Fred and Bill
c. John, Fred or Bill
d. John or Fred or Bill
d. Neither John nor Fred nor Bill
e. both John and Fred
f. either John or Fred

These schemata could be refined along the lines of Gazdar et al. 1985, Chapter 8) to respect the claim of Ross (1967) that the final coordinator and conjunct in a coordinate structure form a constituent. The semantics of coordination will not be discussed here, but see Gazdar (1980), Partee and Rooth (1983), and Keenan and Faltz (1985).

Chapter II

Phrase Structure Grammar Extended with Metarules

In the last chapter a PSG grammar and a CG grammar for canonical English were described. In this chapter I outline how the PSG grammar can be extended with metarules to capture a range of extraction and coordination data. The mode of generalisation is essentially that initiated by Gazdar (1981, 1982) to characterise left extraction, right extraction, right node raising, and across-the-board extraction. Sag (1983) employs the same technique for parasitic extraction, and Schachter and Mordechay (1983) characterise non-constituent coordination, or coordination reduction, by an account which is symmetric with that for right node raising; they accordingly refer to the construction as left node raising, a practice that I will continue here.

I will attempt to explain and motivate some departures from existing accounts, and I will extend discussion from 'simple' non-canonicality (Section 1) to 'compound' non-canonicality (Section 2). Roughly speaking, the former involve cases with one displaced element and the latter cases with more than one displaced element. With a few exceptions (e.g. Abbot 1976; Maling and Zaenen 1982 Section 2.2.1.3; Stucky 1987) English compound non-canonicality has received little attention from linguists; a major aim of this thesis is to draw attention to its bearing on linguistic theory. In this chapter I will argue that the kinds of metarules proposed for simple non-canonicality actually also express appropriate generalisations for compound non-canonicality.

I will be using the terminology of transformational grammar to present the data but it should be clear that this does not indicate a theoretical allegiance to any of the concepts involved; transformational terms are used purely descriptively.

1. Simple Non-Canonicality

Sections 1.1 and 1.2 consider extraction and coordination respectively.

1.1. Extraction

'Extraction' refers to phenomena in which elements are displaced from their usual location; the following sections describe left extraction, right extraction, and parasitic extraction.

1.1.1. Left Extraction

By left extraction or 'fronting', I shall mean primarily topicalisation and relativisation:[1]

(1) a. London$_i$, I liked e_i
    b. the town which$_i$ I liked e_i

Transformationally the fronted element is viewed as having been moved, for example by a transformation of *Wh* movement. I will indicate extractions as in (1) with '*e*' at extraction sites or 'gaps', and with coindexing of gaps and their 'fillers'. Left extraction can pass through arbitrarily many clause boundaries, and is an instance of 'long distance' or 'unbounded' dependency:

(2) a. London$_i$, I think that John argued that Sue likes e_i
    b. the town which$_i$ I think that John argued that Sue likes e_i

Following Gazdar et al. (1985), an account of such extraction can be viewed as coming in three parts: there is the analysis of the extraction site, the analysis of the filler (or landing) site, and the analysis of the mediating material. For the latter, Gazdar (1981) proposed the following metarule:

(3) $$\begin{array}{l} X \rightarrow \ldots Y \ldots \\ \Longrightarrow \\ X/Z \rightarrow \ldots Y/Z \ldots \end{array}$$

A symbol 'X/Y' stands for expressions of category *X* 'missing' a subexpression of category *Y*. This makes the interpretation of the slash similar to the one in CG; the extent of this similarity should emerge in the course of this chapter. The rule (3) states that if expressions of certain categories can combine to form an expression of category *X*, then the corresponding expressions with one lacking a subexpression of category Z can combine to form an expression of category *X/Z*. By way of example, application of (3) to (4a) and

---

[1] The parallelism between these phenomena will be taken to be sufficient motivation to group them together. Although topicalisation and relativisation have not always been collapsed together, the theories considered here all provide parallel treatments. Other left extraction phenomena, such as interrogative formation, will not be explicitly discussed though their treatment will presumably follow much the same pattern as that of topicalisation and relativisation.

(5a) can yield (4b) and (5b) respectively.

(4) a. S → NP VP
b. S/NP → NP VP/NP

(5) a. VP → TV NP
b. VP/NP → TV NP/NP

This achieves transmission, or percolation, of the information that there is a gap. To introduce gaps, Gazdar (1981) chooses to allow the interpretation of *X/X* to extend to the case whereby the empty string is regarded as an expression of category *X* lacking a subexpression of category *X* so that the empty string is of category *X/X* for all categories *X*:

(6) X/X → *e*

I will refer to this as null rule gap introduction. Then for example *I liked* is analysed with an empty node as shown in Figure 1. An alternative method of gap introduction is by metarules such as (7) (cf. Gazdar, Klein, Pullum, and Sag 1982, p49).

(7) X → ... Y ...
==>
X/Y → ... ...

This proposal does not necessitate empty nodes; (7) just states that if a sequence of categories including *Y* can analyse as *X* then the sequence with *Y* missing can analyse as *X/Y*. Under metarule gap introduction *I liked* is analysed as shown in Figure 2. The relative merits of null rule gap introduction and metarule gap introduction will be considered in the

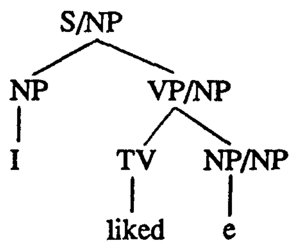


Figure 1

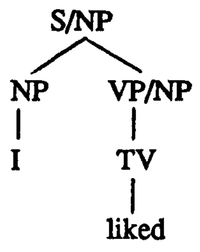


Figure 2

course of this discussion.

A topic may be introduced thus:

(8) *Topic Introduction*
$S \rightarrow X\ S/X$

So the analysis of *London, I liked* is completed like this:

(9) $[\text{London}_{NP,i}\ [\text{I liked}\ e_i]_{S/NP}]_S$

The capacity to characterise unbounded extraction is illustrated in the analysis of *London, I think that John argued that Sue likes*, in Figure 3.

Relativisation can be treated correspondingly, with an object relative pronoun *RELPROo* introduced by:[2]

(10) REL → RELPROo S/NP

Thus *which I think that John argued that Sue likes* has the analysis of Figure 3 except for the filler introduction step.

As well as noun phrases, complementized sentences can topicalise; complex adjectives topicalise better than basic ones:

(11) $[\text{That John will stay}]_i$, I can believe e_i

---

[2] Subject relative pronoun introduction was illustrated in Chapter 1; pied piping is discussed later, in connection with categorial grammar.

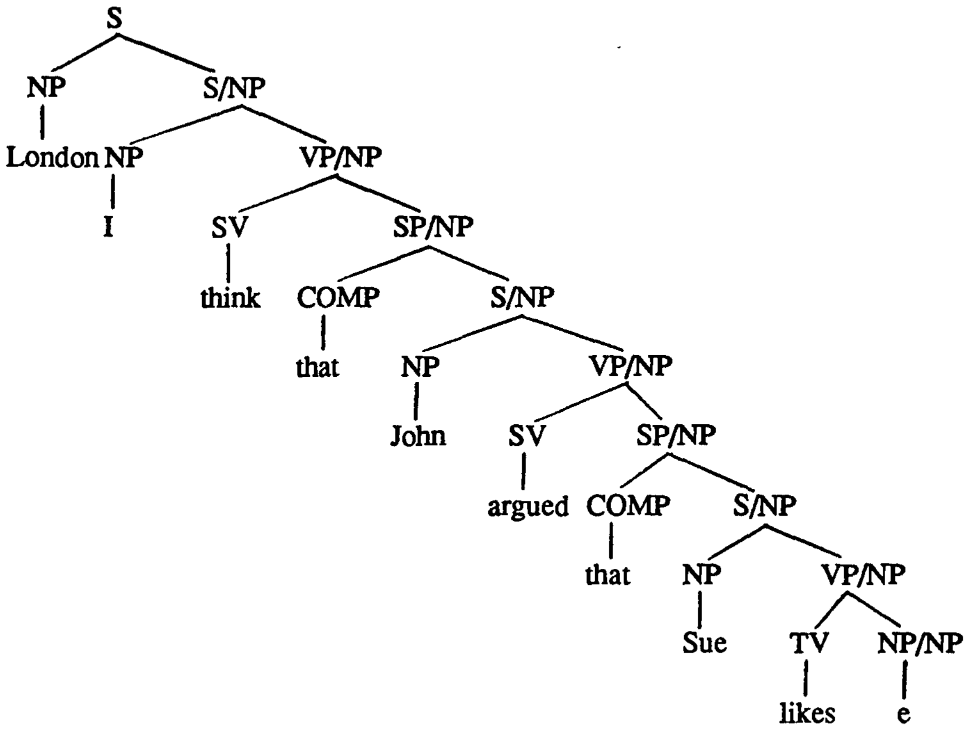


Figure 3

(12) a. ?[Easy to please]$_i$, John is e_i
b. *Angry$_i$, they are e_i

Topicalisation of a finite verb phrase is unacceptable; that of a base form verb phrase and infinitival verb phrase is better:

(13) a. *[Will eat mushrooms]$_i$ I think that John e_i
b. ?[Eat mushrooms]$_i$ I think that John will e_i
c. ?[To go to London]$_i$ Mary wants e_i

Adverbial and adnominal prepositional phrases can appear sentence-initially, but the former can occur with or without the intonational stress characteristic of topicalisation:

(14) a. [On Monday]$_i$ Sue arrived e_i
b. ?[To London]$_i$ we bought five tickets e_i

Overall it will be assumed that noun phrases, complementized sentences, prepositional phrases, and adjective phrases can topicalise, so that Topic Introduction is constrained to introduce {*NP*, *SP*, *PP*, *AP*}.

In a binary grammar, Gazdar's (1981) slash transmission metarule schema given earlier reduces to the following two instances:

(15) $X \rightarrow Y\ Z$
$==>$
$X/W \rightarrow Y\ Z/W$

(16) $X \rightarrow Y\ Z$
$==>$
$X/W \rightarrow Y/W\ Z$

The former will generate extraction from clause-final positions; the latter will generate extraction from non-clause-final positions, such as:

(17) a. Faust$_i$, I put e_i on the table
b. [That John will stay]$_i$, I can believe e_i easily

To a first approximation the meaning of a topicalised sentence is the same as that of the corresponding canonical sentence. This suggests the standard treatment whereby the meaning of a sentence with a gap is the same as that of the corresponding sentence, but abstracted over the meaning of the gap, so that the meaning of a topicalised sentence can be obtained by applying the meaning of the sentence-with-gap to that of the topic. Similarly, a relative pronoun needs to be supplied with the sentence meaning abstracted over the gap meaning, because this expresses the predicate by which the relative clause restricts its head noun; in this case the relative pronoun applies as the functor. Then the various rules and metarules can be supplied with semantics as follows; the implications '==>' carry subscripts identifying the metarules.[3]

(18) *Right Abstraction*
$X \rightarrow Y\ Z$ ϕ
$==>_R$
$X/W \rightarrow Y\ Z/W$ $\lambda x \lambda y \lambda z[\phi\ x\ (y\ z)]$

---

[3] The formulation of the semantics here is different from the usual 'designated variable' method, which Engdahl (1986, pp24-28) notes to be technically problematic.

(19) *Middle Abstraction*
$X \rightarrow Y\ Z$ $\qquad \phi$
$\Longrightarrow_M$
$X/W \rightarrow Y/W\ Z$ $\qquad \lambda x \lambda y \lambda z[\phi\ (x\ z)\ y]$

(20) *Null Rule Gap Introduction*
$X/X \rightarrow e$ $\qquad \lambda x[x]$

(21) *Topic Introduction*
$S \rightarrow X\ S/X$ $\qquad \lambda x \lambda y[y\ x]$

(22) *Relative Pronoun Introduction*
REL → RELPROo S/NP $\qquad \lambda x \lambda y[x\ y]$

Note that the semantics of a metarule is a functional abstraction over the contribution to the mother meaning of the missing element; this corresponds to the intuition that semantically an extracted element 'belongs' at its extraction site. The semantics of empty node expansion is the identity function (as in e.g. Gazdar, Klein, Pullum, and Sag 1985). By way of example of how the semantics works, the semantics of the derived rules in the analysis of *London, I liked* are as follows:

(23) a. $VP \rightarrow TV\ NP$ $\qquad \lambda x \lambda y[x\ y]$
$\Longrightarrow_R$
$VP/NP \rightarrow TV\ NP/NP$ $\qquad \lambda x_1 \lambda y_1 \lambda z_1[\lambda x \lambda y[x\ y]\ x_1\ (y_1\ z_1)] =$
$\lambda x_1 \lambda y_1 \lambda z_1[x_1\ (y_1\ z_1)]$
b. $S \rightarrow NP\ VP$ $\qquad \lambda x \lambda y[y\ x]$
$\Longrightarrow_R$
$S/NP \rightarrow NP\ VP/NP$ $\qquad \lambda x_2 \lambda y_2 \lambda z_2[\lambda x \lambda y[y\ x]\ x_2\ (y_2\ z_2)] =$
$\lambda x_2 \lambda y_2 \lambda z_2[y_2\ z_2\ x_2]$

The full analysis is as shown in Figure 4.

There are a variety of constraints on left extraction apart from the category of the extracted element: 'island' constraints on the nodes through which an extraction can be mediated, and constraints on extraction sites. I will consider island constraints first, and then constraints on extraction sites.

Consider the following:

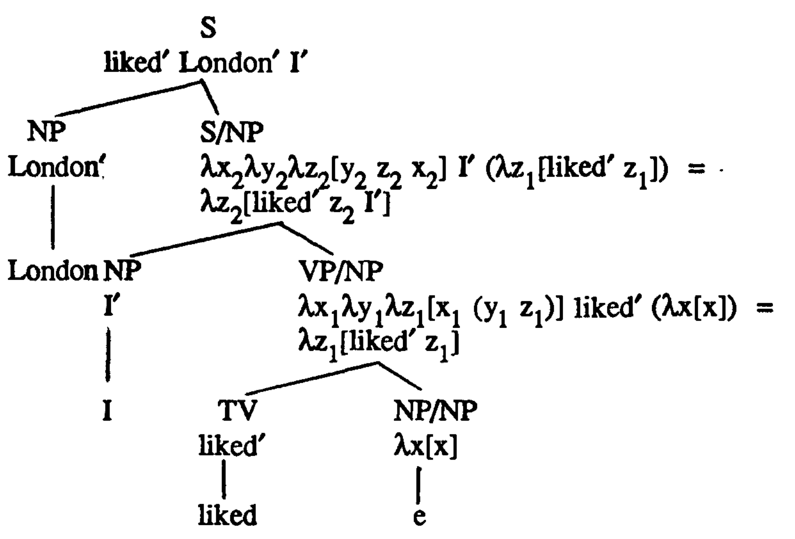


Figure 4

(24) a. *the machine which$_i$ I met [the man who invented $e_i]_{NP}$

b. *the items which$_i$ he explained [the fact that he bought $e_i]_{NP}$

Ross (1967) accounts for such unacceptability in terms of a 'complex noun phrase constraint' which asserts that noun phrases containing relative clauses and noun phrases containing noun complement clauses are islands to extraction. Nothing in the above account leads us to expect such a constraint, but it could be captured by, for example, stipulating that an analysis containing a node *N/NP* is not legitimate. However although the generalisation holds by and large, I find, say (25) semi-acceptable.

(25) ?a colleague whom John acquired [a belief that I disliked $e_i]_{NP}$

This suggests that unacceptability of complex noun phrase constraint violations may not indicate ungrammaticality. Similarly, Kuno (1976) notes that extraction out of relative clauses sounds better when the higher relative clause semantically concerns the antecedent. He provides the following paradigm (acceptability judgements are mine):

(26) a. *the child who$_i$ John married [a girl who dislikes e_i]$_{NP}$
b. ?the child who$_i$ I know [a family which is willing to adopt e_i]$_{NP}$
c. the child who$_i$ there is [nobody who is willing to adopt e_i]$_{NP}$

Chung and McCloskey (1983) claim that extraction from subject relative clauses is more acceptable than extraction from object relative clauses; for example they contrast (26c) with (27) which they mark as unacceptable, though I find this particular example good.

(27) the child that$_i$ there is [no one who the authorities can persuade to accept e_i]$_{NP}$

In view of data such as (28), Chomsky (1973) forwarded the 'subject condition' which asserts that all subjects are islands.[4]

(28) ?a woman whom$_i$ [a picture of e_i]$_{NP}$ used to hang over the fireplace

Again this might be captured by stipulating that *NP[nom]/NP* is somehow ill-formed (though non-nominative and non-NP subjects present complications), but again also the facts are not clear; I find (28) semi-acceptable, and (29) fine.

(29) a woman whom$_i$ [a picture of e_i]$_{NP}$ sold for over seven million pounds

Stipulating **NP/NP* in general would capture the 'NP Constraint' which Bach and Horn (1976) forward to embrace such constraints as the complex noun phrase constraint and subject condition; according to this *all* noun phrases are islands. The constraint captures the examples in (30) but implies that those in (31) have some exceptional structure.

(30) a. *the man who$_i$ John destroyed a book about e_i
b. ?the man who$_i$ I lost a picture of e_i

(31) a. the programme which$_i$ I missed the end of e_i
b. the town which$_i$ I bought a ticket to e_i

According to the 'A-over-A constraint' of Chomsky (1964) it is ungrammatical to extract any constituent out of a superordinate constituent of the same category. The condition has an 'NP-over-NP' instantiation which is close to the NP Constraint. Additionally, the condition characterises the following paradigm:

---

[4] The constraint also covers Ross's sentential subject constraint according to which sentential subjects are islands:
(i) *the subject which$_i$ that John likes e_i is obvious
(ii) *the pleasures which$_i$ for you to give up e_i would be a pity

(32) a. the tunnel [out of which]$_{PP,i}$ John emerged e_i
b. *the tunnel [of which]$_{PP,i}$ John emerged [out e_i]$_{PP}$
c. the tunnel which$_{NP,i}$ John emerged [out of e_i]$_{PP}$

In this case an appropriate condition might be **X/X*.

In general, adverbials (like adnominals) have an island character, suggesting **ADV/NP*:

(33) *a debate which$_i$ John made his vote [without attending e_i]$_{ADV}$

However not all extractions are completely unacceptable:

(34) a. ?the city which$_i$ John met Mary [in e_i]$_{ADV}$
b. ?the people who$_i$ John left the party [without meeting e_i]$_{ADV}$

And some cases are fully acceptable (cf. Chomsky 1982, p72):

(35) a. the papers which$_i$ John went to Paris [without reading e_i]$_{ADV}$
b. the people who$_i$ he arrived [with e_i]$_{ADV}$
c. the path which$_i$ we ran [along e_i]$_{ADV}$

Here too then island constraints on extraction present a complicated picture; compare incidentally the acceptable NP extraction in (35a) with the unacceptable prepositional phrase extraction in (36) (cf. the A-over-A constraint).

(36) *the people [to whom] he went to Paris [without speaking e_i]$_{ADV}$

The general situation with such constraints as these illustrates a recurrent methodological dilemma: examples which are apparently identical syntactically differ in acceptability. Logically, there are three possibilities: the examples are all grammatical but the comprehension of the unacceptable ones is inhibited; the examples are all ungrammatical but the comprehension of the acceptable ones is facilitated; or the acceptable examples are grammatical and the unacceptable ones are ungrammatical (i.e. the examples were not actually identical syntactically).

A claim that there are unacceptable grammatical sentences would have a precedent in examples like the following where the syntactic identity in all significant respects indicates that the 'b' examples are unacceptable despite grammaticality.

(37) a. The woman who John met left
b. *The woman who the man who the dog bit met left

(38) a. I gave to John the most recent version of the paper
b. *I gave to John it

(39) a, You and I ought to go shopping
b. *I and you ought to go shopping

Example (37) illustrates how apparent well-formedness diminishes with centre-embedding of relative clauses. Well-formedness judgements of (38) are susceptible to the 'heaviness' of the object noun phrase, and the ordering preference in (39) seems to be of extra-linguistic origin. A claim that an ungrammatical expression is acceptable would be more unusual (though see e.g. Chomsky 1970 pp193-5; Otero 1972; Langendoen and Bever 1973). I have sketched how it might be possible to realise the various constraints above in the grammar. However it is clear that this would not constitute an explanation of the phenomena, just a description. This fact, together with the uncertain character of the constraints, implies that this is not a very interesting way to proceed here, and I will assume that violations of these island constraints are grammatical, though usually unacceptable. Some unacceptability of extraction from adjuncts may be due to a certain contradiction in fronting, and thereby bringing into prominence, an element which belongs to a subordinate clause and which is presumably semantically peripheral.

Constraints on extraction sites will now be considered. The 'fixed subject constraint' (or '*that*-trace filter' or 'empty subject filter') of Bresnan (1972) and Chomsky and Lasnik (1977) prohibits extraction of a subject immediately following a complementizer:

(40) *the man who$_i$ I think that e_i left

(Related examples such as *the man who I think left* are discussed in Chapter III in the context of categorial grammar.) The fixed subject constraint appears to be rather more robust than some of the earlier constraints (though see Sobin 1987). However the existing grammar generates such extraction, via the rule derived in (41); see Figure 5.

(41)
$$\begin{array}{l} S \rightarrow NP\ \ VP \\ \quad ==>_M \\ S/NP \rightarrow NP/NP\ \ VP \end{array}$$

In some versions of Generalised Phrase Structure Grammar (GPSG, e.g. that in Gazdar et al. 1985) the fixed subject constraint is captured by a lexical head constraint (Flickinger 1983) which restricts the application of metarules to rules introducing lexical heads: the role of metarules is restricted to introduction of gaps, and general slash transmission is

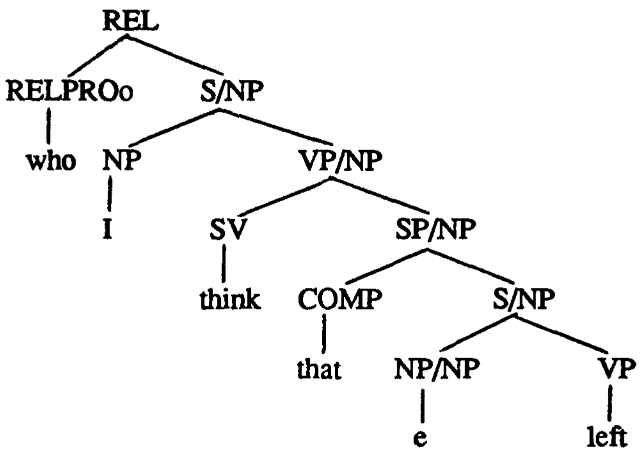


Figure 5

achieved by separate feature percolation conventions. Then the rule in (41) would not be derived because the input would not be a rule with a lexical head.[5] Note however that the gap introduction metarule (42), as an alternative to null rule gap introduction, appears to get the facts right.

(42) X → Y Z
==>
X/Z → Y

If we have this rule, but not the corresponding one introducing a gap on the left-hand daughter, then a gap can only be on a right branch, even though the binary metarule introduced earlier allows percolation *through* a left branch, as is required for extraction from non-clause-final position. This achieves an effect like that of the 'left branch condition' of Ross (1967) and accordingly fits other instances of the left branch condition whereby extraction of the determining noun phrase in a possessive construction is ungrammatical:

(43) a. I saw [John's book]
b. *the man who$_i$ I saw 's book

---

[5] In GPSG intransitive verbs would be introduced by a unary rule mapping an X-bar level 0 intransitive verb to the VP X-bar level.

(44) a. This is John's
b. Whose is this?

(45) a. I borrowed John's book
b. *Whose did you borrow book?

Such a realisation of the left branch condition also seems to correctly characterise the state of affairs whereby fronting of an adjective phrase from pre-nominal position (a left branch) is considerably less acceptable than fronting from post-copula position (a right branch):

(46) a. John is very tall
b. How tall is John?

(47) a. I met a very tall man
b. *How tall did you meet a man?

The implication here then is that metarule gap introduction represents an improvement on null rule gap introduction in that it is capable of realising an appropriate version of the left branch condition. Apparent violations of the left branch condition such as in Polish (Borsley 1983) remain as a topic for further study.

1.1.2. Right Extraction

'Right extraction' phenomena include 'right extraposition' and 'heavy shift'. Right extraposition refers to the appearance of a noun modifier to the right of its normal position:

(48) A man e_i arrived [who swims]$_i$

The rules and metarules introduced in Section 1.1.1 already enable a clause with a missing subexpression of category *X* to be analysed as an expression of category *S/X* with a meaning which is the abstraction of the clause meaning over that of the missing subexpression. Gazdar (1981) proposes the following kind of rule to introduce right extracted elements:

(49) *Rightward Filler Introduction*
$X \rightarrow X/Y\ \ Y \qquad \lambda x \lambda y[x\ y]$

Giving this a semantics in which the left-hand daughter meaning is applied to the right-hand daughter meaning results in right extracted sentences correctly being assigned the same meanings as their canonical counterparts. For example (48) is analysed as shown in

Figure 6; the semantics is as follows:

(50) man ⇒ man′
e ⇒ λx[x]
man *e* ⇒ λx[x man′]
a ⇒ a′
a man *e* ⇒ λx[a′ (x man′)]
arrived ⇒ arrived′
a man *e* arrived ⇒ λx[arrived′ (a′ (x man′))]
who ⇒ who′
swims ⇒ swims′
who swims ⇒ who′ swims′
a man *e* arrived who swims ⇒ arrived′ (a′ (who′ swims′ man′))

In addition to non-subcategorized adnominals, a noun complement can be right extraposed:

(51) A rumour e_i spread [that TTK had gone bust]$_i$

And extraposition can be from object as well as from subject:

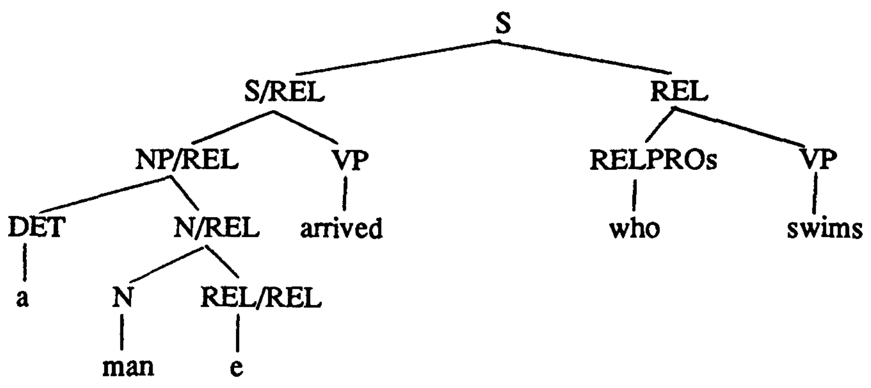


Figure 6

(52) a. I met a man e_i yesterday [who plays hockey]$_i$
b. I met a man e_i yesterday [from London]$_i$
c. I spread a rumour e_i yesterday [that TTK had gone bust]$_i$

Right extraposition need not be to sentence-final position, for example it can occur within a noun phrase (see Akmajian 1975, p123). In the following the complement is right extraposed past the relative clause:

(53) a belief e_i which I do not share [that Mary will come back]$_i$

So noun modifiers generally can undergo right extraposition.

Heavy shift refers to the appearance of a verb complement to the right of its usual position; acceptability is dependent on this element being large (heavy).[6] I will assume that heavy shift of all elements, large and small, is grammatical, but that the acceptability of the latter is for some reason impaired.[7] (One possibility is that this connects with the tendency for new information to come at the end of a sentence: heavy elements are relatively 'likely' to contain new information.) In the following the direct object is heavy shifted past the indirect object:

(54) I gave e_i to John [the most recent version of the paper(/*it)]$_i$

This has the analysis shown in Figure 7 where semantics is as follows:

(55) *e* ⇒ λx[x]
gave ⇒ gave′
gave *e* ⇒ λx[gave′ x]
to ⇒ to′
John ⇒ John′
to John ⇒ to′ John′
gave *e* to John ⇒ λx[gave′ x (to′ John′)]
the most ... ⇒ the-most-...′
gave *e* to John the most ... ⇒ gave′ the-most-...′ (to′ John′)
I gave *e* to John the most ... ⇒ gave′ the-most-...′ (to′ John′) I′

It is also possible to heavy shift a direct object past a second complement which is a verb phrase or a complementized sentence:

---

[6] "Heavy noun phrase shift" refers to the case where the element involved in specifically a noun phrase.

[7] Cf. the discussion of acceptability and grammaticality earlier

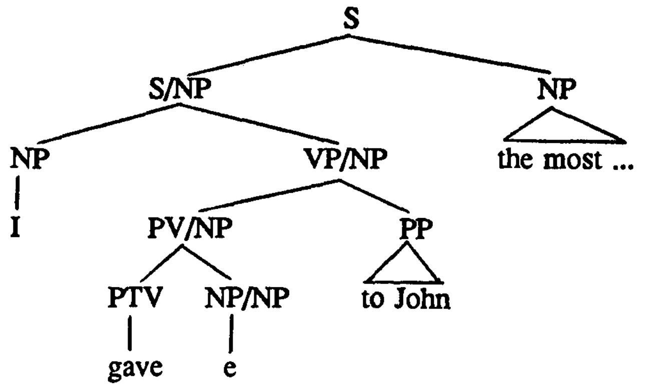


Figure 7

(56) a. I believe e_i to be incompetent [a good number of the members of the board$]_i$

b. I convinced e_i that I was a student [a rather nervous-looking security guard$]_i$

Under the analysis of verb-particle constructions assumed here, an object canonically comes left of the particle, and moves right of it on the pattern of heavy shift. However in this case the object does not need to be particularly heavy:

(57) I rang e_i up [the press$]_i$

If this analysis is correct, it may indicate that it is the relative weights of the elements involved, rather than the absolute weight of the extracted element, that is important; in particular note that when the object is very light, it is unacceptable in post-particle position (under the intended reading):[8]

(58) *I looked up it

Heavy shift of the indirect object of a verb like *gave* in its 'dative-shifted' form is less acceptable:[9]

---

[8] I am grateful to Pete Whitelock for discussion on the construal of particles as intransitive prepositional phrases, and on the analysis of particle shift as heavy shift of the object.

[9] In transformational approaches, [*gave* NPind NPdir], is regarded as having been derived by a rule of 'dative-shift' from [*gave* NPdir *to* NPind].

(59) a. *Mary gave e_i a book [each of the students who seemed genuinely interested]$_i$

It is difficult to capture this in the grammar, particularly because left and right extraction are treated via the same mechanisms, and the corresponding left extraction is of considerably higher acceptability:

(60) the student whom I gave e_i the book

It is possible that in this case processing of the right extraction is confounded by the identity of the categories commuted, these both being noun phrases. Perhaps related is the fact that the two objects of *bet* cannot be commuted:

(61) *I bet e_i \$5 [the man over there]$_i$ that we'd win

Note that the 'landing site' in (61) is not clause-final, but this doesn't seem to be the origin of the unacceptability because in general an element need not heavy shift as far as clause-final position; in the following the right extracted element is introduced under a *VP* node:

(62) I gave e_i to John [the most recent version of the paper]$_i$ without remembering that it criticised his thesis

Elisabet Engdahl (personal communication) has pointed out that the relevant distinction between these examples may be that (61) ends with a complement while (62) ends with an adjunct.

We have seen that an element may be heavy shifted past a complement; it is also possible to heavy shift past an adjunct:

(63) I met e_i yesterday [a student from MIT who thought that most American linguists would reject such an approach out of hand]$_i$

Elements other than noun phrases can undergo heavy shift:

(64) a. We looked e_i everywhere [for some sign of Albert]$_i$
b. He argued e_i passionately [that we should reject the amended motion]$_i$
c. ?John was e_i yesterday [very angry]$_i$

Verb phrases do not heavy shift well:

(65) ?I wanted e_i yesterday [to go shopping]$_i$

In an account employing immediate dominance/linear precedence factoring, (64) need not be regarded as non-canonical, i.e. both word orders may basic; Sag (1987) presents such an account of English word order in which linear precedence rules make reference to a hierarchy of grammatical functions. I continue here however with the assumption that there is a single underlying 'normal' order for this part of English.

In summary, I shall assume that noun phrases, adverbial preposition phrases, complementized sentences, and adjective phrases can undergo heavy shift.

Although various categories can right extract, and be introduced at various locations, not all categories can do so. Thus:

(66) *a e_i left [man from London]$_i$

This could be captured either by restricting the gap categories that can be transmitted rightwards by constraining Middle Abstraction, or else by restricting the categories that can participate in Rightward Filler Introduction. It seems slightly odd to allow gap information to percolate but to forbid it to discharge, and Middle Abstraction will therefore be constrained to transmission of {*NP, SP, PP, AP, REL, ADV*}. Note that this set includes all those categories which can topicalise, as it must to allow topicalisation from clause-non-final positions. Observe that (67) is generated by null rule gap introduction, but prohibited by metarule gap introduction since the latter will not introduce a gap on a left branch.

(67) *$[[e_{NP/NP}\ left_{VP}]_{S/NP}\ Bill_{NP}]_S$

Again then, metarule gap introduction seems preferable to null rule gap introduction.

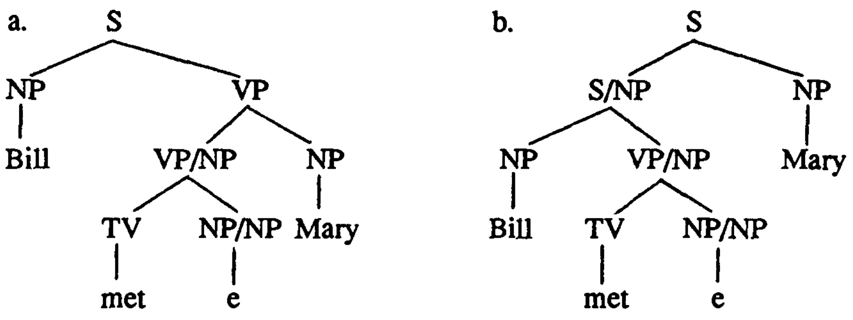


Figure 8

Rightward Filler Introduction can re-introduce a Right Abstracted element in its canonical position, resulting in multiple analyses, but not new meanings. For instance *Bill met Mary* can be analysed as shown in Figure 8; but these analyses both assign the canonical meaning. Thus we saw earlier that the meaning assigned to *Bill met* is *λx[met′ x Bill′]*, and since the semantics of filler introduction is to apply this to the filler, the analysis in Figure 8b will assign the canonical meaning *met′ Bill′ Mary′*. Since Right Abstraction and Rightward Filler Introduction interact in this way, it appears that they *need* not be constrained, and in Section 1.2.1 coordination data is considered which suggests that they *should* not be. However in general expressions will now have many analyses yielding the same meaning. This contravenes the normal assumption of one analysis per meaning, and necessitates knowledge of equivalence classes of analyses if processing is to avoid unnecessary work; this is discussed fully in Chapter VI in relation to categorial grammar.

Ross (1967) originally observed that right extraction appears to have un upward bounded character:

(68) a. *[an argument [about a picture e_i]] started [of Bill's first wife]$_i$
b. *I [believed that John [liked e_i] all my life] [strawberries and cream]$_i$

This led to the postulation by Ross of a 'right roof' constraint. In a similar spirit, Akmajian (1975) and Schachter and Mordechay (1983) suggest that while the sisters of a noun may right extrapose, elements embedded within them cannot. However the following, adapted from Akmajian (1975, p128, n13) contradicts this:

(69) [A number [of reports e_i]] soon appeared [on the Watergate Affair]$_i$

And Stucky's (1987, p391) example (70) involves right extraposition of an embedded modifier.

(70) [The names of [[all the painters e_i]] are unknown [whose work is being exhibited in the Chicago Art Institute next week]$_i$

Similarly, Gazdar (1981) rejects the hypothesis that heavy shift is clause bounded, citing in support Grosu (1972), Witten (1972), Postal (1974) and Andrews (1975). Note for example the following, attributed by Gazdar to Janet Fodor:

(71) I have [wanted [to meet e_i]] for many years [the man who spent so much money planning the assassination of Kennedy]$_i$

Although right extraposition and heavy shift may not be entirely bounded, it is clear that examples like (68), (72) and (73) are of very low acceptability.

(72) *He has believed that [Mary knows a man e_i] for many years [who smuggles]$_i$

(73) *He has believed that [Mary knows e_i] for many years [a man who is widely believed to be involved in smuggling]$_i$

The collapsing of right extraposition and heavy shift suggests that (74) should be grammatical.

(74) *A woman who knows e_i arrived [a man widely believed to be involved in smuggling]$_i$

Furthermore the grammar incorrectly allows right extraction to strand a preposition, whereas this is only acceptable with left extraction:

(75) a. *I talked about e_i to Mary [all the news]$_i$
b. the news which$_i$ I talked about e_i to Mary

Because the current account employs the same machinery for left and right extraction, left extraction and right extraction would be expected to correlate generally. As before, since the account developed does not lead us to expect these constraints and sice they are of uncertain character, I will not try to impose a constraint. One possible factor, suggested by Ewan Klein (personal communication) for asymmetry between left and right extraction, is that while the former (e.g. relativisation) seems to increase the expressive power of the language, the latter only provides alternative ways of saying what could already be said.

The null rule gap introduction incorrectly allows right extraction to violate the left branch condition:

(76) *I think that e_i left [the man who you wanted to meet]$_i$

But for the same reasons as before this would not be the case if there were metarule gap introduction.

1.1.3. Parasitic Extraction

'Parasitic extraction' (Taraldsen 1979; Engdahl 1983) refers to extraction in which one filler corresponds to two extraction sites, with no coordination involved. One of these is often an island; this latter gap is described as being 'parasitic' on the former:[10]

(77) a. ?a paper which$_i$ I filed the records without reading e_i
b. a paper which$_i$ I filed e_i without reading e_i

(78) a. ?a man whom$_i$ the friends of e_i envied Sue
b. a man whom$_i$ the friends of e_i envied e_i

(79) a. ?the man whom$_i$ I expected the picture of e_i to bother Mary
b. ?the man whom$_i$ I expected the picture of e_i to bother e_i

Intuitions as to acceptability of parasitic constructions vary considerably. I find (79b) quite poor, although many speakers find it good.

An account of parasitic extraction must somehow achieve a 'merging' together of gaps so that they are satisfied by a single filler. Within a phrase structure grammar context, Sag (1983) proposes the following:[11]

(80) a. X → ... Y/NP ... Z ...
==>
X → ... Y/NP ... Z/NP ...
b. X → ... Y ... Z/NP ...
==>
X → ... Y/NP ... Z/NP ...

The idea is that application to rules already derived by metarule allows an additional slash to exist under one of the daughters, which is matched with the existing mother slash inherited from another daughter. However although this is what is intended, (80) can also apply to Topic Introduction and the Rightward Filler Introduction, with calamitous results:

---

[10] Recall however that we are regarding neither subjects nor adverbs as complete islands.

[11] Sag is able to express the rule rather more economically using immediate dominance/linear precedence factoring but this is not important here.

(81) a. S → NP S/NP

==>

S → NP/NP S/NP

b. S → S/NP NP

==>

S → S/NP NP/NP

According to these the following should be grammatical sentences:

(82) a. *[A picture of *e*]$_{NP/NP}$ [I liked *e*]$_{S/NP}$

b. *[I liked *e*]$_{S/NP}$ [a picture of *e*]$_{NP/NP}$

The problem in these cases is that the input rule is not actually transmitting a gap, so that the parasitic gap is being merged into thin air. It might be possible to stipulate that the metarule does not apply in these cases, however it is argued in Sections 2.1.1. and 2.3 that in general metarules need to be able to apply to filler introduction rules (in order to transmit gap information out of extracted elements). What we seem to want to say in the parasitic case is that if expressions of certain categories can be combined, then expressions of those categories both containing a gap of the same category can be combined to form an expression regarded as containing a single gap of that category. Thus for our binary grammar, I propose (83).

(83) *Parasitic Abstraction*

X → Y Z ϕ

==>$_P$

X/W → Y/W Z/W $\lambda x \lambda y \lambda z[\phi\ (x\ z)\ (y\ z)]$

Then *filed without reading* is analysed as shown in Figure 9.

Parasitic extraction can be to the right as well as to the left, and the filler-gap distance is unbounded:

(84) I filed e_i without reading e_i (at all) [a paper I was meant to review at once]$_i$

(85) the paper which$_i$ John thinks that Mary said that Sue filed e_i without reading e_i

I will limit attention here to noun phrase--noun phrase parasitic gaps. Note however that example (86) appears to have a prepositional phrase-prepositional phrase parasitic reading, and Tait (1988) offers (87) as an instance of prepositional phrase-noun phrase parasitic

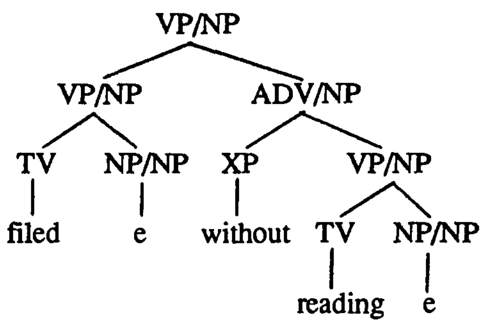


Figure 9

extraction.[12]

(86) a man [to whom]$_i$ I talked e_i without selling a car e_i

(87) [To whom]$_i$ did Mortimer faithfully continue to write e_i after seeing e_i only once?

Consider the following:

(88) *the patient who$_i$ I showed e_i e_i

The current account generates this as shown in Figure 10. However in a grammar with metarule gap introduction, as opposed to null rule gap introduction, the constraint that parasitic gaps cannot be adjacent would be predicted by the parasitic rule given above. This follows because, as mentioned earlier, on this approach a gap cannot be on a left branch, so that a gap is never constituent-initial. Thus of the two subexpressions X/Z and Y/Z concatenated by a parasitic rule, the gap in the first might be rightmost, but the gap in the second cannot be leftmost, so that the gaps will never be adjacent. This is another case where metarule gap introduction seems superior to null rule gap introduction, but either way the unacceptability of (89) remains unaccounted for (cf. Gazdar et al. 1985, p166).

---

[12] Anna Szabolcsi (personal communication) has pointed out that (86) could be a prepositional phrase-noun phrase case if the subordinate verb is in its dative-shifted form.

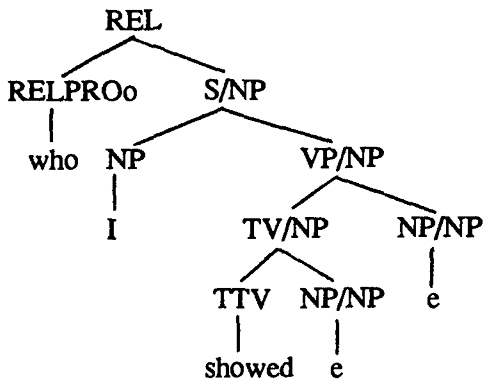


Figure 10

(89) a. ?the models whom$_i$ I sent the pictures of e_i to e_i
b. *the slave whom$_i$ I gave e_i to e_i

The next section considers further cases in which one filler corresponds to two gaps, this time cases involving coordination.

1.2. Coordination of 'Non-Constituents'

"Coordination of 'non-constituents'" refers to a range of coordination phenomena in which the conjuncts are not constituents in the canonical grammar. The scare quotes are used because under the accounts here, the conjuncts actually *are* constituents in the non-canonical grammar. In Sections 1.2.1, 1.2.2, and 1.2.3 I discuss right node raising, left node raising, and across-the-board extraction respectively.

1.2.1. Right Node Raising

Examples such as (90) are described as exhibiting 'right node raising' (Postal 1974 pp125-128; Bresnan 1974).

(90) [I liked e_i but Suzy hated e_i], London$_i$

Transformationally, the right-peripheral object shared by the two verbs in (90) is viewed as having been 'raised' out of the coordinate structure, as illustrated in Figure 11. Right node raising is not a local (i.e. clause-bound) phenomenon; in (91a) it crosses a clause boundary and in (91b) it crosses two clause boundaries; the relation between the filler and the gap is an instance of unbounded dependency.

(91) a. [John said that Sue likes e_i and Fred said that Sue dislikes e_i] [newsletters full of trivia]$_i$

b. [John said that Sue likes e_i and Robert said that Liz thinks that Sue dislikes e_i] [newsletters full of trivia]$_i$

As Gazdar (1981) shows, a coordination schema like that given in Chapter I, together with the devices already introduced to characterise extraction, provides a characterisation of right node raising. Thus (90) is analysed as shown in Figure 12, and the unboundedness exemplified by (91) is captured by iteration of the relevant operations, like the unboundedness of left extraction.

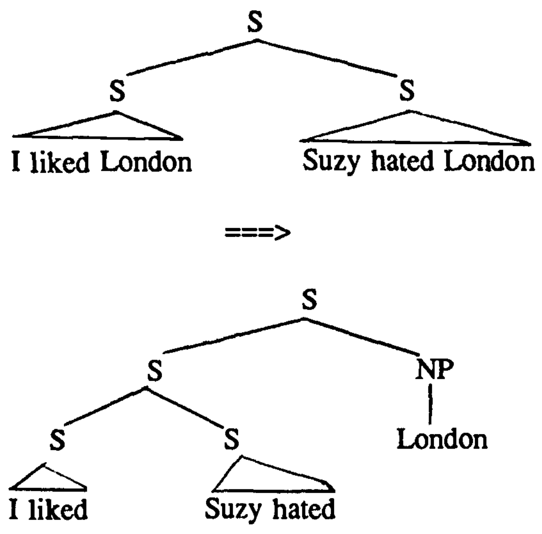


Figure 11: Classical right node raising

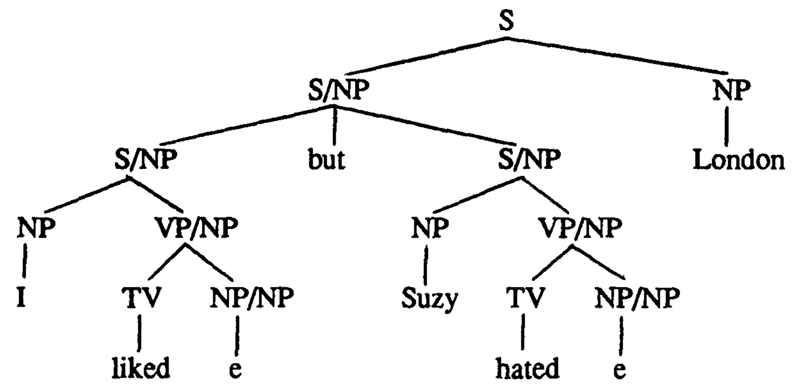


Figure 12

In addition to noun phrases, complementized sentences can be right node raised; note that in (92b) right node raising is out of a noun phrase as opposed to a sentence:

(92) a. [John thinks e_i and Mary knows e_i] [that we haven't been entirely truthful in this matter]$_i$

b. [the belief e_i and the hope e_i] [that they would come back]$_i$

An adjective (phrase) may also right node raise:

(93) [John was e_i and Mary is e_i] [extremely angry]$_i$

Assuming as we are that adnominals and adverbials ordinarily modify common noun phrases and intransitive verb phrases, as opposed to noun phrases and sentences, the following also exhibit right node raising; note that in (95) the node raising is again out of a noun phrase:[13]

(94) a. [John arrived e_i and Mary left e_i] [in the helicopter]$_i$

b. [John arrived e_i and Mary left e_i] hurriedly$_i$

---

[13] Bob Borsley has pointed out that the agreement relations between the singular conjuncts and the plural verb in the relative clause in (95a) are particularly problematic under a right node raising analysis.

(95) a. [a man e_i and a woman e_i] [who like Beethoven]$_i$
b. [a man e_i and a woman e_i] [from London]$_i$
c. [a man e_i and a woman e_i] outside$_i$

Uncomplementized sentences appear to right node raise less readily, but Bresnan (1974) cites (96).

(96) I [have been wondering whether e_i, but wouldn't positively want to state that e_i], [your theory is correct]$_i$

To-infinitival verb phrases seem to right node raise better than finite verb phrases:

(97) a. [John tried e_i and Mary managed e_i] [to finish writing within the six weeks]$_i$
b. ?He thinks [that John e_i or that Mary e_i] [tried to deceive him]$_i$

Right node raising of common noun phrases is also an unclear area; the node raising out of a noun phrase in (98b) is more acceptable than that out of a verb phrase in (98a).

(98) a. ?I [liked this e_i but preferred that e_i] sofa$_i$
b. [a red e_i or a green e_i] tee-shirt$_i$

It is shown above that it is possible to right node raise out of sentences and noun phrases; it is also possible to right node raise out of adverbials as in (99) and adnominals as in (100), notwithstanding their island character as regards left extraction.

(99) a. It was exciting [while landing e_i and while taking off e_i] [in the helicopter]$_i$
b. They left [without waiting e_i and without looking e_i] [for the others]$_i$
c. He implements changes [without consulting e_i and without informing e_i] [the executive]$_i$
d. He said this [before claiming e_i and after denying e_i] [that he was Italian]$_i$

(100) a. the people [who arrived e_i and who left e_i] [in the helicopter]$_i$
b. the people [who agree e_i and who disagree e_i] [about these issues]$_i$
c. the people [who like e_i and who dislike e_i] [the opera]$_i$
d. the people [who believe e_i and who disbelieve e_i] [that we will win]$_i$

Examples (101), (102), and (103) exhibit right node raising out of common noun phrases, verb phrases, and complementized sentences respectively.

(101) a. the [paintings e_i and small sketches e_i] [by Picasso]$_i$
b. the [belief e_i and foolish hope e_i] [that they would come back]$_i$
c. the [arguments for e_i and arguments against e_i] [the second option]$_i$

(102) a. We [will talk e_i and might argue e_i] [about some personal things]$_i$
b. I [read e_i and will reference e_i] [several papers]$_i$
c. He [believes e_i and has proposed e_i] [that we should take more direct action]$_i$
d. We [looked for e_i and found e_i] [a village with a good inn]$_i$
e. He [arrived with e_i but left without e_i] [the girl he used to date at school]$_i$

(103) a. He claims [that he arrived e_i and that he left e_i] yesterday$_i$
b. He claims [that he knew e_i and that he loved e_i] Maria$_i$
c. She wondered [whether he said e_i or whether he implied e_i] [that they laughed]$_i$

By way of summary, noun phrases, complementized sentences, adverbials, prepositional phrases, and adjectives right node raise well; bare sentences, verb phrases, and nouns right node raise slightly less well. The categories out of which it is possible to right node raise include sentences, noun phrases, prepositional phrases, nouns, verb phrases and complementized sentences. The absence of cases in which right node raising is definitely prohibited suggests that Right Abstraction and Rightward Filler Introduction be left unconstrained.

1.2.2. Left Node Raising

Constructions in which verbs appear outside of coordinate structures containing their complements and adjuncts are described as 'left node raising' by Schachter and Mordechay (1983, p267). In the following the conjuncts consist of a complement and an adjunct:

(104) a. I met$_i$ [e_i John on Monday and e_i Sue on Tuesday]
b. He said$_i$ [e_i that he was Italian when we first met him and e_i that he was Spanish when we met him again a week later]
c. John is$_i$ [e_i good natured on Fridays but e_i moody on Mondays]
d. We looked$_i$ [e_i for blackberries on Monday and e_i for strawberries on Tuesday]
e. He wanted$_i$ [e_i to stay on Monday and e_i to go on Tuesday]

In a phrase structure grammar context Sag, Gazdar, Wasow, and Weisler (1985, p161-2) generate left node raising, as well as gapping, by a rule in which the end of a coordinate structure can have the form specified in (105), provided it is interpretable by the informal elliptical interpretation rule (106).

(105) $V^2[\mathrm{CONJ}\ \alpha] \rightarrow \alpha,\ X^{2+}$
where $\alpha \in$ {*and, but, nor, or*}

(106) The interpretation of an elliptical construction is obtained uniformly by substituting its immediate constituents into some immediately preceding structure, and computing the interpretation of the results. [Sag et al., p162]

The *[CONJ α]* specification indicates a constituent (motivated by Ross 1967) formed by the final coordinator and conjunct of a coordinate structure. The comma indicates that this is an immediate dominance rule; linear precedence rules will ensure that the coordinator is constituent-initial. The V^2 stands for verbal X-bar level 2 categories, this includes verb phrases and sentences; the X^2 stands for all bar level 2 categories, i.e. maximal projections. Hudson (1986) points out that this account misplaces conjunct boundaries. In particular the particle ***either*** in (107) makes it clear that the structure is as shown in (108); the Sag et al. account will erroneously attempt to assign a structure such as (109a) or (109b) since the material preceding the end of the coordinate structure must be analysed as a verb phrase or a sentence.

(107) Fred drinks either sherry before dinner or brandy after dinner

(108) Fred drinks [either sherry before dinner or brandy after dinner]

(109) a. Fred [drinks either sherry before dinner] or [brandy after dinner]
b. [Fred drinks either sherry before dinner] or [brandy after dinner]

To characterise left node raising, Schachter and Mordechay (1983) propose what is essentially the following:

(110) $X \rightarrow Y \ldots$
$\Longrightarrow$
$X\backslash Z \rightarrow Y\backslash Z \ldots$

(111) $X \rightarrow Y \ldots$
$\Longrightarrow$
$X\backslash Y \rightarrow \ldots$

Schachter and Mordechay use a different notation (the backward slash used here is deliberately suggestive of categorial grammar). The binary instance of (110) is (112), and although they introduce gaps by the metarule (111), I will continue to illustrate using the null rule gap introduction (113); the filler introduction rule for backward slashes is (114).

(112) *Left Abstraction*
$X \rightarrow Y\ Z$ ϕ
$\Longrightarrow_L$
$X\backslash W \rightarrow Y\backslash W\ Z$ $\lambda x \lambda y \lambda z[\phi\ (x\ z)\ y]$

(113) *Null Rule Gap Introduction*
$X\backslash X \rightarrow e$ $\lambda x[x]$

(114) *Leftward Filler Introduction*
$X \rightarrow Y\ X\backslash Y$ $\lambda x \lambda y[y\ x]$

For example, *(I) met John on Monday and Sue on Tuesday* is analysed as shown in Figure 13.

The following, in which a noun is left node raised from conjuncts containing a complement and an adnominal, is less acceptable for some reason:

(115) ?It's hard to reconcile the arguments$_i$ [e_i that Hamlet is heroic which Mary made and e_i that he is weak which John propounded]

As well as a complement and an adjunct, the conjuncts in a left node raising construction can consist of two complements:

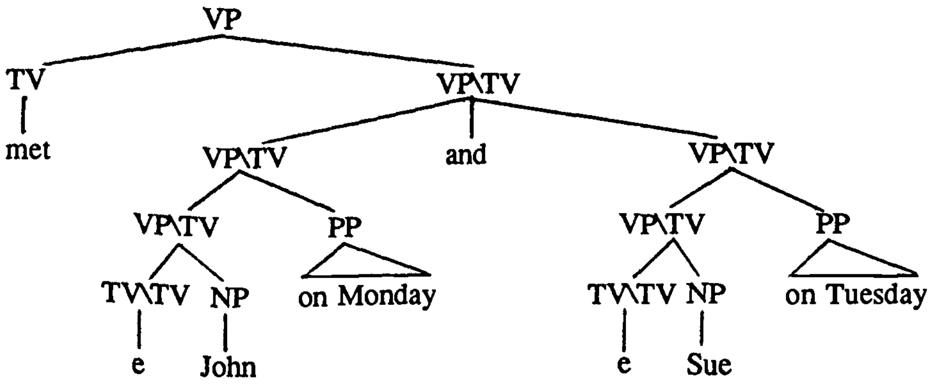


Figure 13

(116) a. He gave$_i$ [e_i a book to John and e_i a record to Mary]

b. He gave$_i$ [e_i John a book and e_i Mary a record]

c. ?He promised$_i$ [e_i John to go and e_i Mary to stay]

d. He told$_i$ [e_i Mary that he was Spanish and e_i Sue that he was Italian]

In the following the subject determiner is left node raised out of the sentences; it is unclear why (117b) is less acceptable than (117a).

(117) a. Each$_i$ [e_i boy dances and e_i girl sings]

b. ?a$_i$ [boy dances and e_i girl sings]

As was the case for right node raising, in the absence of many cases requiring definite prohibition I will leave Left Abstraction and Leftward Filler Introduction unconstrained.

1.2.3. Across-the-Board Extraction

'Across-the-Board' extraction refers to the left or right extraction of elements from each conjunct of a coordinate structure:

(118) a. the newsletter which$_i$ [I gave e_i to John and Sue sent e_i to Mary]
b. [I gave e_i to John and Sue sent e_i to Mary] [several copies of the newsletter]$_i$

In (118) there is extraction from two non-peripheral positions. In (119) one extraction site is right-peripheral and the other is non-peripheral; in (120) both extraction sites are right-peripheral. Note that (120b) can be viewed as right node raising; right node raising is thus a special case of across-the-board right extraction. (Left node raising might also be regarded as across-the-board leftward raising or extraction of left-peripheral elements).

(119) a. the book which$_i$ I [read e_i and sent e_i to John]
b. I [read e_i and sent e_i to John] [a book about horses]$_i$

(120) a. a book which$_i$ [John wrote e_i and I read e_i]
b. [John wrote e_i and I read e_i] [a book which was later to be censored]$_i$

The devices already introduced for extraction and coordination correctly generate these constructions. In particular note that the conjuncts with peripheral and non-peripheral gaps in (119) are both analysed as *VP/NP* and so can coordinate under a like-category schema.

Ross's (1967) 'coordinate structure constraint' characterises coordinate structures as islands, unless extraction is from every conjunct:

(121) a. *the man who$_i$ [John liked e_i and Mary hated Fred]
b. *the man who$_i$ [John liked Fred and Mary hated e_i]

(122) the man who$_i$ [John liked e_i and Mary hated e_i]

This situation fits with the like-category coordination schema: conjuncts of category *S* and *S/NP* cannot coordinate, but those of the same category can. However as is well known the assumption that the empty string is of category *X/X* means that the grammar wrongly allows extraction of a whole conjunct (see e.g. Sag 1982; Gazdar, Pullum, Sag, and Wasow 1982, pp673-4); for example in the following both *a picture of* and the empty string are of category *NP/NP*.

(123) a. *the woman whom$_i$ I saw [e_i and a picture of e_i]
b. *the woman whom$_i$ I saw [a picture of e_i and e_i]

Also, empty node gap introduction allows an intransitive verb to analyse as *S/NP*, incorrectly permitting (124).

(124) *the man who$_i$ [e_i arrived and Mary met e_i]

Again, gap introduction by metarule avoids both these forms of overgeneration: because the empty string then need not belong to any category.

1.3. Summary

The grammar that has been developed is one in which left and right extraction and coordination phenomena are characterised by the same underlying processes; the Abstraction and Filler Introduction augmentations to pure phrase structure grammar are summarized in Figure 14.

Clauses from which there is left extraction leaving a category *X* gap are analysed as being of category *S/X*. There is a rule $X \rightarrow X/Y \;\; Y$ so it is predicted that left extractable elements can also right extract. And by and large this is true:

(125) a. a topic [about which]$_i$ an argument e_i started
b. An argument e_i started [about politics]$_i$

(126) a. the people who$_i$ I believe e_i to be incompetent
b. I believe e_i to be incompetent [a good number of the members of the board]$_i$

However there are some exceptions:

(127) a. the woman who$_i$ I gave e_i a book
b. *I gave e_i a book [a woman I have never seen in my life before]

For similar reasons it is predicted that elements which can left extract can undergo across-the-board right extraction:

(128) a. London$_i$, I liked e_i
b. [I liked e_i but Mary hated e_i], London$_i$

(129) a. London$_i$, Fred said that Sue dislikes e_i]
b. [John said that Sue likes e_i and Fred said that Sue dislikes e_i], London$_i$

(130) a. a topic [about which]$_i$ an argument e_i started
b. [An argument e_i started and a dispute e_i raged], [about politics]$_i$

Right Abstraction
$X \rightarrow Y\ Z \qquad \phi$
$==>_{R}$
$X/W \rightarrow Y\ Z/W \qquad \lambda x \lambda y \lambda z[\phi\ x\ (y\ z)]$

Middle Abstraction
$X \rightarrow Y\ Z \qquad \phi$
$==>_{M}$
$X/W \rightarrow Y/W\ Z \qquad \lambda x \lambda y \lambda z[\phi\ (x\ z)\ y]$
$W \in \{NP, SP, PP, AP, REL, ADV\}$

Parasitic Abstraction
$X \rightarrow Y\ Z \qquad \phi$
$==>_{P}$
$X/NP \rightarrow Y/NP\ Z/NP \qquad \lambda x \lambda y \lambda z[\phi\ (x\ z)\ (y\ z)]$

Left Abstraction
$X \rightarrow Y\ Z \qquad \phi$
$==>_{L}$
$X\backslash W \rightarrow Y\backslash W\ Z \qquad \lambda x \lambda y \lambda z[\phi\ (x\ z)\ y]$

Rightward Filler Introduction
$X \rightarrow X/Y\ Y \qquad \lambda x \lambda y[x\ y]$

Leftward Filler Introduction
$X \rightarrow Y\ X\backslash Y \qquad \lambda x \lambda y[y\ x]$

Topic Introduction
$S \rightarrow X\ S/X \qquad \lambda x \lambda y[y\ x]$
$X \in \{NP, SP, PP, AP\}$

Relative Pronoun Introduction
$REL \rightarrow RELPROo\ S/NP \qquad \lambda x \lambda y[x\ y]$

Figure 14: Augmentations to PSG

---

(131) a. the people who$_i$ I believe *e*$_i$ to be incompetent

b. I believe *e*$_i$ to be incompetent and suspect *e*$_i$ to be apathetic], [a good number of the members of the board]$_i$

Although these various extractions do exhibit some of the symmetry which is expected from a uniform treatment, there are cases where the symmetry does not hold true. Wexler and Culicover (1980, p299) and Levine (1985, p492) provide (132) and (133) respectively, in which there is right node raising from a complex noun phrase; the islandhood of complex noun phrases with respect to left extraction has already been noted.

(132) Mary knows a man who buys, and Bill knows a man who sells, pictures of Fred

(133) John gave a briefcase, and Harry knows someone who had given a set of steak knives, to Bill

Additionally, McCloskey (1986) notes the contrastive acceptability of preposition stranding in right node raising and rightward extraction. We are faced then with a situation in which there are both similarities between and differences amongst various extractions. I will continue with the current completely unified account. However my proposal is to address these discrepancies by making slashes structured so that instead of being atoms (like our present / and \), they receive a componential analysis. Then the similarities and dissimilarities between different kinds of extractions will be characterised through proportionate similarities and dissimilarities in the structure of the associated slashes.

2. *Compound Non-Canonicality*

Section 1 was concerned with cases where there was a single filler or displaced element, although corresponding to this there may have been more than one gap. This section is concerned with cases where there is more than one displaced element. I will use the term "independent extraction" by way of contrast with "parasitic extraction", and I will use "extraction" to embrace both. "Multiple extraction" will not mean "parasitic extraction" or "across-the-board extraction" but the simultaneous existence of more than one instance of (independent or parasitic) extraction.

I will show that allowing the metarules proposed so far to reapply to derived rules can capture a wide range of compound non-canonicality. It has often been proposed that categories should carry at most one slash specification, or that a metarule should not reapply to its own output: constraints usually motivated from a language-theoretic, or a processing perspective (see e.g. Thompson 1982). However the linguistic facts are that there are many cases in which an expression contains more than one gap, and often the appropriate characterisation is one in which a metarule reapplies to its own output. In Section 2.1 I look at multiple extraction, and in Section 2.2 I look at multiple across-the-board extraction.

2.1. Multiple Extraction

Sections 2.1.1 and 2.1.2 consider multiple independent extraction, and independent extraction plus parasitic extraction, respectively.

2.1.1. Multiple Independent Extraction

Two verb complements can be heavy shifted past an adverbial:

(134) I posted e_i e_j yesterday [a copy of the newsletter]$_i$ [to every student]$_j$

In accord with earlier remarks on the relation between left and right extraction, at the same time that either of the complements is heavy shifted, the other may be left extracted:

(135) a. the people [to whom]$_j$ I posted e_i e_j yesterday [copies of the newsletter]$_i$
b. the newsletter which$_i$ I posted e_i e_j yesterday [to every member in the area]$_j$

Consider the following applications of metarules:

(136) PV → TPV NP λxλy[x y]
==>$_R$
PV/NP → TPV NP/NP λxλyλz[x (y z)]

(137) VP → PV PP λxλy[x y]
==>$_R$
VP/PP → PV PP/PP λxλyλz[x (y z)]
==>$_M$
VP/PP/NP → PV/NP PP/PP λxλyλzλw[x z (y w)]

(138) VP → VP ADV λxλy[y x]
==>$_M$
VP/PP → VP/PP ADV λxλyλz[y (x z)]
==>$_M$
VP/PP/NP → VP/PP/NP ADV λxλyλzλw[y (x z w)]

The convention for slashes is left-associative so that, e.g. *VP/PP/NP* is *(VP/PP)/NP*, a verb-phrase-lacking-a-prepositional-phrase lacking a noun phrase, as opposed to *VP/(PP/NP)*, a verb phrase lacking a prepositional-phrase-lacking-a-noun-phrase. The derived rules enable ***posted yesterday a copy of the newsletter to every student*** to be analysed as shown in Figure 15, with correct assignment of the same meaning as that of the canonical ordering:

(139) posted ⇒ posted′
e ⇒ λx[x]
posted *e* ⇒ λx[posted′ x]
e ⇒ λx[x]
posted *e e* ⇒ λxλy[posted′ x y]
yesterday ⇒ yesterday′
posted *e e* yesterday ⇒ λxλy[yesterday′ (posted′ x y)]
a copy ... ⇒ a-copy-...′
posted *e e* yesterday a copy ... ⇒ λy[yesterday′ (posted′ a-copy-...′ y)]
to every .. ⇒ to-every-...′
posted *e e* yesterday a copy ... to every ...
⇒ yesterday′ (posted′ a-copy-...′ to-every′)

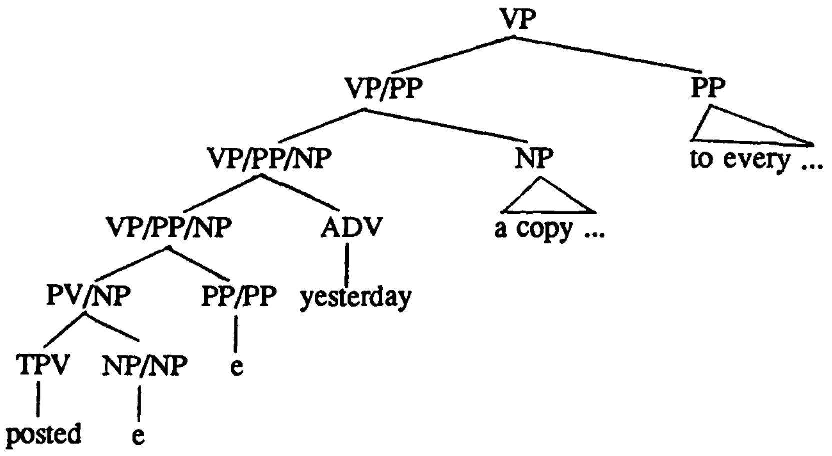


Figure 15

Commutation of two complements heavy shifted beyond the adverbial is semi-acceptable:

(140) ?I posted e_i e_j yesterday [to every member in the area]$_i$ [a copy of the newsletter]$_j$

There are two ways of generating such examples in the current grammar. First, Right Abstraction could apply after Middle Abstraction, in contrast with the earlier case:

(141) VP → PV PP λxλy[x y]
$==>_M$
VP/NP → PV/NP PP λxλyλz[x z y]
$==>_R$
VP/NP/PP → PV/NP PP/PP λxλyλzλw[x w (y z)]

Then Rightward Filler Introduction would locate the prepositional phrase left of the noun phrase. Alternatively, Middle Abstraction could apply to Rightward Filler Introduction:

(142) X → X/Y Y
$==>_M$
X/Z → X/Y/Z Y

The output of (142) will combine *VP/PP/NP*, derived as in (139), with a prepositional phrase first. Thus, if it were necessary to avoid examples like (140) it would apparently be necessary to constrain order of application of metarules, to prevent the first manner of derivation, and also applicability of metarules to filler introduction rules, to prevent the second.

It is possible to right extrapose a subject's relative clause while also left extracting from the predicate verb phrase:

(143) a paper which$_i$ a woman e_j presented e_i [who has been studying computational linguistics for six years]$_j$

In this example, the crucial rule in one analysis is derived thus:

(144) S → NP VP
$==>_R$
S/NP → NP VP/NP
$==>_M$
S/NP/REL → NP/REL VP/NP

Alternatively, as before the metarules could apply in the opposite order, and Middle Abstraction could apply to Rightward Filler Introduction. Right extraction from the verb phrase to a position beyond the extraposed relative clause is of low acceptability:

(145) *a woman e_i presented e_j [who has been studying computational linguistics for six years]$_i$ [a long and involved paper on parsing complexity]$_j$

This would be generated by application of the metarules in (144) in the opposite order, or by application of Middle Abstraction to Rightward Filler Introduction. Again this suggests constraining application of metarules, but evidence against this is provided by (146) in which two subject modifiers are extracted, one to the left and one to the right.[14]

(146) a. a woman [about whom]$_i$ an argument e_i e_j started [which went on all night]$_j$

b. a topic [on which]$_i$ some textbooks e_i e_j appeared [which advocated corpuscular theories of light]

Assuming that the prepositional phrases in (146) canonically occur as complements left of the relative clauses, either Right Abstraction should apply after Middle Abstraction so that the right extracted relative clause is sought first, as in (147), or else Middle Abstraction should apply to Rightward Filler Introduction.

(147) N → N REL
==>$_M$
N/PP → N/PP REL
==>$_R$
N/PP/REL → N/PP REL/REL

Since one or other of these equivalent mechanisms is required, and the grammar already exhibits multiple equivalent analyses, I will not aim to eliminate either device.

The application in (148) of Right Abstraction to Topic Introduction enables unacceptable double topicalisation as in (149).

(148) S → X S/X
==>$_R$
S/Y → X S/X/Y

---

[14] The pied piping in (146) is addressed later.

(149) *[[the book]$_{NP}$ [[on the table]$_{PP}$ [I put *e* *e*]$_{S/PP/NP}$]$_{S/NP}$]$_S$

One condition that could filter such cases is a requirement that topics be main sentence-initial. I find the embedded topicalisation (150), from Baltin (1982), quite acceptable, though I find (151a) and (151b), cited as acceptable by Iwakura (1980) and Gazdar, Klein, Pullum, and Sag (1982), less good.

(150) It's obvious that Mary, he can't stand

(151) a. ?Harry said that Max, Joan would never be willing to marry
b. *The inspector explained that each part he had examined very carefully

The unacceptability of double topicalisation might be interpreted as indicating that metarule application to filler introduction rules should be prohibited. However I find Baltin's (p17) example (152), which involves relativisation out of an embedded topicalisation, acceptable:

(152) He's a man to whom liberty we could never grant

This case necessitates metarule application to Topic Introduction, and is analysed essentially like the double topicalisations above. It therefore remains unclear to me how to prevent double topicalisation in English. One factor of note is the high degree of intonational markedness of topicalisation, (in contrast with e.g. right extraction), and this may be taken as indicating that topicalisation is a somewhat exceptional mechanism of grammar. But whatever the reason for topicalisation constraints in English, the existence of double topicalisation in languages such as Irish suggests that this possibility must be admitted by universal grammar.

Another construction involving extraction is '*tough* movement', so called in view of the adjectives like *tough* which trigger the phenomenon; I will assume that the extraction is unbounded, though I find long distance cases like (153d) less acceptable.

(153) a. [Many divorces]$_i$ are tough (for men) to get over e_i
b. [The exams]$_i$ are easy (for students) to pass e_i
c. [Some exams]$_i$ are hard (for lecturers) to persuade students to take e_i
d. ?[Some theories]$_i$ are hard (for students) to believe that anyone understands e_i

I will avoid the complication of the optionality of the subject in the complement clause.[15] In the case that the subject is absent, the following rule enables (153b) to be analysed as

---

[15] In Gazdar et al. (1985) this alternation is handled by the fact that verb phrases and sentences are of the same category, differing only in a feature *SUBJ*. Borsley (1987) presents several cases where generalistion across verb phrases and sentences is required.

shown in Figure 16 with the semantics shown in (155):

(154) AP → Af VP/NP λxλy[x y]

(155) pass ⇒ pass′
e ⇒ λx[x]
pass *e* ⇒ λx[pass′ x]
to ⇒ to′
to pass *e* ⇒ λx[to′ (pass′ x)]
easy ⇒ easy′
easy to pass *e* ⇒ easy′ (λx[to′ (pass′ x)])
are ⇒ are′
are easy to pass *e* ⇒ are′ (easy′ (λx[to′ (pass′ x)]))
the ⇒ the′
exams ⇒ exams′
the exams ⇒ the′ exams′
the exams are easy to pass *e* ⇒ are′ (easy′ (λx[to′ (pass′ x)])) (the′ exams′)

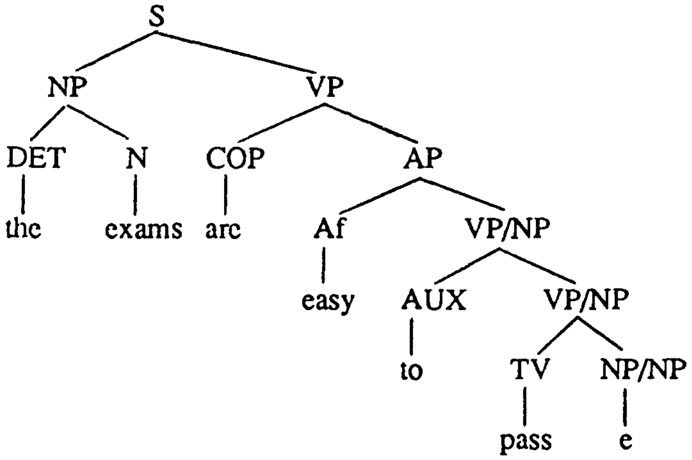


Figure 16

A constraint that is often cited in relation to this construction is the 'nested dependency constraint' of Fodor (1978, Section 3) which states that dependencies must be nested rather than crossed:[16]

(156) a. a violin which$_j$ [the sonatas]$_i$ are hard to play e_i on e_j
b. *some sonatas which$_i$ [the violin]$_j$ is hard to play e_i on e_j

The current grammar does not predict any such constraint; but observe that in the case of a ditransitive verb, as opposed to a transitive prepositional verb, the acceptability ordering is contrary to that suggested by the nested dependency constraint:[17]

(157) a. ?some evidence which$_i$ [the witnesses]$_j$ are hard to show e_j e_i
b. some witnesses whom$_i$ [the evidence]$_j$ is hard to show e_i e_j

2.1.2. *Independent Extraction Plus Parasitic Extraction*

Compound non-canonicality can involve parasitic phenomena. Consider (158), which I regard as grammatical.

(158) ?a paper which$_i$ he showed e_j e_i before submitting e_i [a good number of his colleagues]$_j$

The main verb's second complement and the subordinate verb's object are parasitically left extracted, and the main verb's first complement is heavy shifted. An appropriate characterisation is achieved by (159) as shown in Figure 17.

(159) VP → VP ADV ==>$_P$
VP/NP$_i$ → VP/NP$_i$ ADV/NP$_i$ ==>$_M$
VP/NP$_i$/NP$_j$ → VP/NP$_i$/NP$_j$ ADV/NP$_i$

A sister case is one where the gap in the adverbial is parasitically identified not with the left extracted second complement, but with the right extracted first complement:[18]

(160) a picture which$_i$ he showed e_j e_i without forewarning e_j [the unsuspecting members of the jury]$_j$

---

[16] It is assumed here that the relation between the subject and the gap in the complement of the *tough*-like adjective is induced lexically, though it is indicated by the usual indexing.

[17] Also, Dick Oehrle (personal communication) has pointed out that in the following the contrast is not so sharp:

(i) a topic *which*$_j$ *John*$_i$ is hard to talk to e_i about e_j
(ii) a topic *which*$_i$ *John*$_j$ is hard to talk about e_i to e_j

[18] The relevant reading is one in which *showed* is a ditransitive.

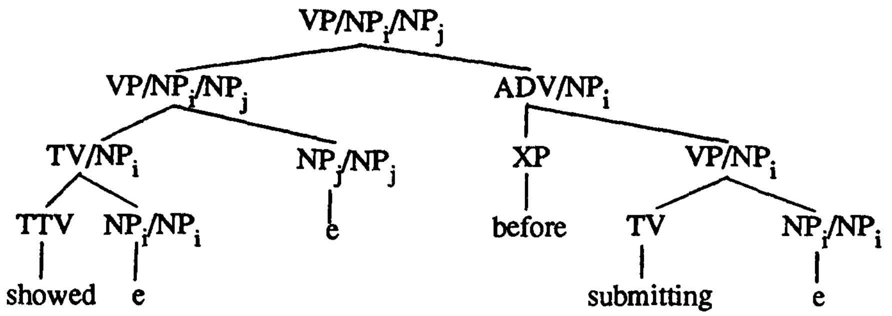


Figure 17

This can be obtained by applying the metarules the other way round:

(161) VP → VP ADV $==>_M$
VP/NP$_i$ → VP/NP$_i$ ADV $==>_P$
VP/NP$_i$/NP$_j$ → VP/NP$_i$/NP$_j$ ADV/NP$_j$

2.2. Multiple Extraction from Coordinate Structure

Sections 2.2.1 and 2.2.2 address multiple across-the-board extraction, and left node raising plus across-the-board extraction, respectively.

2.2.1. Multiple Across-the-Board Extraction

The first examples are cases of double right node raising (see Abbott 1976):

(162) [Mary sent e_i e_j or John gave e_i e_j] [a full report]$_i$ [to every student]$_j$

Note that either of the node raised elements may be left extracted:

(163) a. the students [to whom]$_j$ [Mary sent e_i e_j or John gave e_i e_j] [a full report]$_i$
b. a report which$_i$ [Mary sent e_i e_j or John gave e_i e_j] [to every student]$_j$

Since the direct object is left extracted in (163b) it should also be possible to right extract it, and (164) is indeed acceptable.

(164) [Mary sent e_i e_j or John gave e_i e_j] [to every student]$_j$ [a full and detailed report]$_i$

The cases are facilitated by the rules in (165) and (166) and partial analyses are shown in Figure 18 and Figure 19.

(165) VP → PV PP ==>$_R$
VP/PP → PV PP/PP ==>$_M$
VP/PP/NP → PV/NP PP/PP

(166) VP → PV PP ==>$_M$
VP/NP → PV/NP PP ==>$_R$
VP/NP/PP → PV/NP PP/PP

Such non-canonicality is possible with complements of other categories:

(167) a. [Mary promised e_i e_j and I gave e_i e_j] Ralph$_i$ [a first edition copy of Syntactic Structures]$_j$
b. a book which$_j$ [Mary promised e_i e_j and I gave e_i e_j] Ralph$_i$
c. a student whom$_i$ [Mary promised e_i e_j and I gave e_i e_j] [a first edition copy of Syntactic Structures]$_j$

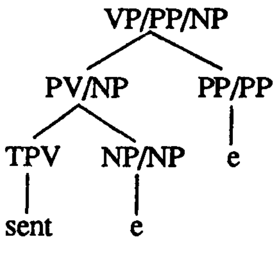


Figure 18

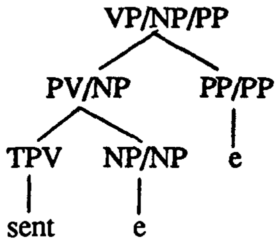


Figure 19

(168) a. [John told e_i e_j and Mary convinced e_i e_j] [all the committee members]$_i$ [that Ralph was a socialist]$_j$

b. [That Ralph was a socialist]$_j$ [John told e_i e_j and Mary convinced e_i e_j] [all the committee members]$_j$

c. ?[The committee members]$_i$ [John told e_i e_j and Mary convinced e_i e_j] [that Ralph was a socialist]$_j$

In these examples there is double right node raising out of sentences. In (169) there is double right node raising out of verb phrases.

(169) a. Mary [has given e_i e_j or will send e_i e_j] [a full report]$_i$ [to every student]$_j$

b. He [has given e_i e_j or will send e_i e_j] [every student]$_i$ [a full report]$_j$

c. He [has told e_i e_j or will notify e_i e_j] [every student]$_i$ [that he's leaving]$_j$

Also, in (170) there is double right node raising from a relative clause and in (171), from an adverbial.

(170) the people [who gave e_i e_j and who sent e_i e_j] [these reports]$_i$ [to the students]$_j$

(171) He lived [without loaning e_i e_j and without donating e_i e_j] [any pictures]$_i$ [to the gallery]$_j$

In the following both the complement and the adverb are right node raised, (unless it is assumed that the adverbials have scope over a whole sentential coordinate structure).

(172) a. [I looked e_i e_j but Mary waited e_i e_j] [for John]$_i$ yesterday$_j$
b. [I saw e_i e_j but Mary missed e_i e_j] Dallas$_i$ yesterday$_j$
c. [I hoped e_i e_j and Mary believed e_i e_j] [that we would finish]$_i$ yesterday$_j$

And in (173) both a noun complement and an adnominal are right node raised from a noun phrase:

(173) [a hope e_i e_j and a belief e_i e_j] [that Mary will come back]$_i$ [which I do not share]$_j$

The right node raising of three complements in (174a) is semi-acceptable. The right node raising of two complements and an adjunct in (174b) seems fine.

(174) a. ?[Sue bet e_i e_j e_k and Mary bet e_i e_j e_k] Bill$_i$ \$5$_j$ [that Fred would win]$_k$
b. [I sent e_i e_j e_k or Mary gave e_i e_j e_k] [each member]$_i$ [a small gift]$_j$ [as a sign of appreciation]$_k$

In (175) there is right node raising *from* the adverbial, and across-the-board extraction *of* the complement:

(175) He [met e_i during e_j and married e_i after e_j] [the great war]$_j$ [a woman whom I've always thought of as my Aunt]$_i$

This is generated via:

(176) VP → VP ADV ==>$_M$
VP/NP$_i$ → VP/NP$_i$ ADV ==>$_R$
VP/NP$_i$/NP$_j$ → VP/NP$_i$ ADV/NP$_j$

The corresponding left extraction is at least as acceptable:

(177) a woman who$_i$ he [met e_i during e_j and married e_i after e_j] [the great war]$_j$

Left extraction of the other complement is also fairly acceptable:

(178) a war which$_j$ he [met e_i during e_j and married e_i after e_j] [a woman whom I've always thought of as my Aunt]$_i$

This would require (179); however the 'crossed' right extraction (180) that this would also facilitate is not acceptable (cf. the general problem of preposition stranding and right extraction).

(179) VP → VP ADV ==>$_R$
VP/NP$_j$ → VP ADV/NP$_j$ ==>$_M$
VP/NP$_j$/NP$_i$ → VP/NP$_i$ ADV/NP$_j$

(180) *He [met e_i during e_j and married e_i after e_j] [a woman]$_i$ [the great war]$_j$

In the following there is across-the-board extraction *of* the first complement and *from* the second complement.

(181) a. the people whom$_i$ he [persuaded e_i to talk e_j and urged e_i to shout e_j] [about their childhood oppressions]$_j$
b. ?the things [about which]$_i$ he [persuaded e_j to talk e_i and urged e_j to shout e_j] [the vast majority of his anxious patients]$_j$

(182) a. the people who$_i$ he [persuaded e_i to leave e_j and urged e_i to oppose e_j] [the political party which they had always supported]$_j$
b. an institution which$_i$ he [persuaded e_j to leave e_i and urged e_i to oppose e_j] [a large number of formerly active supporters]$_j$

(183) a. a scholar who$_i$ [I know e_i to have argued e_j and suspect e_i to believe e_j] [that binding theory has explanatory power]$_j$
b. ?I [know e_i to have argued e_j and suspect e_i to believe e_j] [that binding theory has explanatory power]$_j$ [several of the workers in that research group]$_i$

(184) a. the tokens which$_i$ he [handed e_i to e_j and took e_i from e_j] [each acolyte]$_j$
b. the acolytes whom$_i$ he [handed e_j to e_i and took e_j from e_i] [small tokens of remembrance]$_j$

The crucial step in the analysis of say (184a) is derived thus:

(185) VP → PV PP ==>$_M$
VP/NP$_i$ → PV/NP$_i$ PP ==>$_R$
VP/NP$_i$/NP$_j$ → PV/NP$_i$ PP/NP$_j$

2.2.2. *Left Node Raising Plus Across-the-Board Extraction*

There can be extraction from a predicate verb phrase at the same time that a subject determiner is left node raised. In (186) a complement is extracted; in (187) an adjunct is.

(186) a. the teacher [for whom]$_i$ most$_j$ [e_j boys searched e_i and e_j girls waited e_i]
b. the teacher whom$_i$ most$_j$ [e_j boys like e_i and e_j girls dislike e_i]
c. ?[That they would be caught]$_i$ most$_j$ [e_j boys suspected e_i and e_j girls knew e_i]

(187) ?the coach [in which]$_i$ most$_j$ [e_j girls arrived e_i and e_j boys left e_i]

An example like (186a) will be generated via (188) as illustrated in Figure 20.

(188) S → NP VP ==>$_R$
S/PP → NP VP/PP ==>$_L$
S/PP\DET → NP\DET VP/PP

While a verb is left node raised, it is possible to also extract from the left or right complements or adjuncts comprising the conjuncts; indeed it is possible to have parasitic extraction from two elements in a conjunct. First, extraction from the right-hand element:

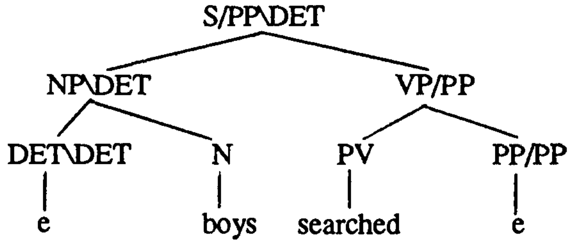


Figure 20

(189) a. a topic [about which]$_i$ I lent$_j$ [e_j John a book e_i and e_j Mary a paper e_i]
b. I lent$_i$ [e_i John a book e_j and e_i Mary a paper e_j] [about subjacency]$_j$

(190) a motion which$_i$ a survey revealed$_j$ [e_j Southerners to support e_i and e_j Northerners to oppose e_i]

(191) the group which$_i$ he wanted$_j$ [e_j Mary to join e_i and e_j John to leave e_i]

(192) [That John left]$_i$ he wants$_j$ [e_j Mary to believe e_i and e_j Sue to disbelieve e_i]

(193) the countries which$_i$ he smuggled$_j$ [e_j alcohol to e_i and e_j drugs from e_i]

Example (189), for instance, will be generated via the rule derived in (194).

(194) VP → TV NP ==>$_L$
VP\TTV → TV\TTV NP ==>$_R$
VP\TTV/PP → TV\TTV NP/PP

The extraction is also possible if the second element is an adjunct as opposed to a complement:

(195) I met$_i$ [e_i John before e_j and e_i Mary during e_j] [the second session]$_j$

Secondly, there is extraction from the left-hand element while left node raising. The examples in (196) are acceptable, but those in (197) are less good.

(196) a. I lent$_i$ [e_i a book e_j to John and e_i a paper e_j to Mary] [about subjacency]$_j$
b. a topic [about which]$_i$ I lent$_j$ [e_j a paper e_i to John and e_j a book e_i to Mary]

(197) a. ?I gave$_i$ [e_i the men e_j the books and e_i the women e_j the records] [who seemed to me to be most deserving]$_j$
b. ?He wanted$_i$ [e_i the men e_j to be imprisoned and e_i the women e_j to be detained] [who had publically opposed his methods of reform]$_j$
c. *I notified$_i$ [e_i the students e_j that I'd attend and e_i the staff e_j that I'd speak] [whom I'd informed earlier that I'd be out of the country]$_j$

(196) will be generated via the rule derived in (198).

(198) VP → PV PP ==>$_L$
VP\PTV → PV\PTV PP ==>$_M$
VP\PTV/PP → PV\PTV/PP PP

A first complement and an adjunct can form a conjunct from which the head verb has been left node raised and the second complement has been extracted:

(199) a. the people [to whom]$_i$ we sent$_j$ [e_j the report e_i on Monday and e_j the newsletter e_i on Tuesday]
b. the newsletter which$_i$ we sent$_j$ [e_j John e_i on Monday and e_j Mary e_i on Tuesday]
c. [That he was Italian]$_i$ he told$_j$ [e_j John e_i on Monday and e_j Mary e_i on Tuesday]

The analysis is along the lines of those above.

Thirdly, there can be parasitic across-the-board extraction from both of a verb's dependents at the same time that the verb is left node raised:

(200) a town which$_i$ I bought$_j$ [e_j a ticket to e_i not wanting to visit e_i and e_j a ticket from e_i not wanting to leave e_i]

This is analysed as shown in Figure 21.

In the following a gap in the adverbial is extracted parasitically with the first complement of the left node raised verb; see Figure 22:

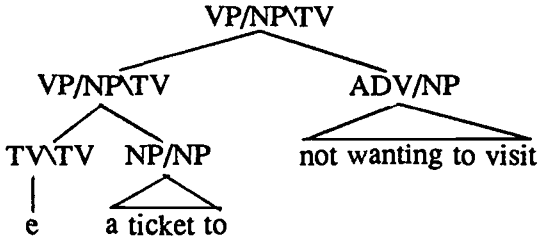


Figure 21

(201) the subjects who$_i$ we gave$_j$ [e_j e_i stimulus A before drugging e_i and e_j e_i stimulus B after drugging e_i]

Alternatively, in a left node raising construction where the conjuncts consist of a verb's first complement and an adverbial, a gap in the adverbial can be extracted parasitically with the second complement of the node raised verb; see Figure 23:

(202) a report which$_i$ he showed$_j$ [e_j John e_i before reading e_i and e_j Mary e_i after reading e_i]

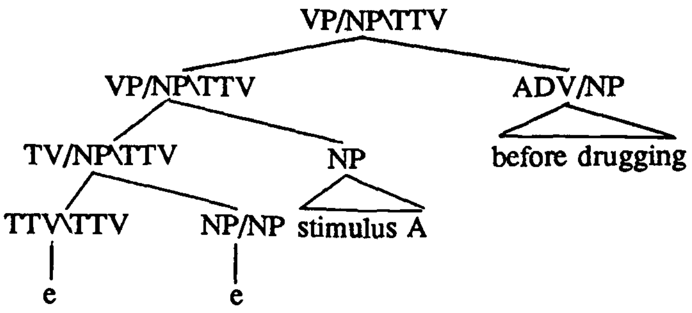


Figure 22

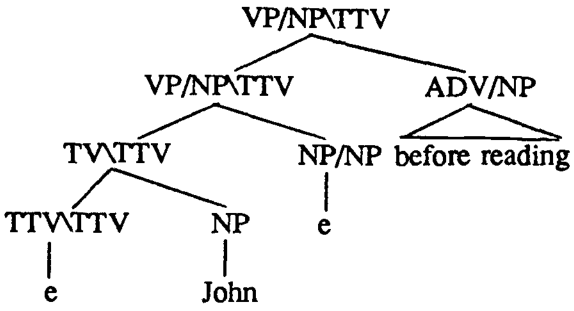


Figure 23

Left node raising in which complements comprising the conjuncts are commuted (i.e. occur in non-canonical order) is of low acceptability:

(203) a. ?He believes$_i$ [e_i e_j to be misguided [the GB-ers]$_j$ and e_i e_k to be mistaken [the GPSG-ers]$_k$]
b. ?I sent$_i$ [e_i e_j to John [a copy of the newsletter]$_j$ and e_i e_k to Mary [a copy of the report]$_k$]
c. *He told$_i$ [e_i e_j that he was Spanish [the girls there]$_i$ and e_i e_k that he was Italian [the people here]$_k$]

And in general it appeared that adjuncts and complements cannot be commuted when there is left node raising:

(204) a. *some friends$_i$ [e_i e_j who play golf [of John]$_j$ and e_i e_k who swim [of Mary]$_k$]
b. *the beliefs$_i$ [e_i e_j which John acquired [that he was ugly]$_j$ and e_i e_k which Mary acquired [that she was beautiful]$_k$]
c. *We met$_i$ [e_i e_j on Monday [the directors]$_j$ and e_i e_k on Tuesday [the union leaders]$_k$]
d. ?We searched$_i$ [e_i e_j this morning [for the girls]$_j$ and e_i e_k this afternoon [for the others]$_k$]
e. He claimed$_i$ [e_i e_j on Monday [that he was Italian]$_j$ and e_i e_k on Tuesday [that he was Spanish]$_k$]

However assuming that right extracted filler-introduction applies freely, according to the existing grammar it should be possible for verbs to be left node raised out of conjuncts containing two commuted elements, as is illustrated in Figure 24 for the case of a transitive prepositional verb; thus the grammar overgenerates.

2.3. Extraction of Incomplete Elements

Observe the following, in which there is extraction from an element which is itself right extracted:

(205) a. a topic [on which]$_i$ I included e_j in the package [several seminal papers e_i]
b. ?a celebrity whom$_i$ several news stories appeared [on e_i]$_j$

Cases like these require metarule application to the Rightward Filler Introduction rule as shown in (206), in order that the gap information can be transmitted out of the right

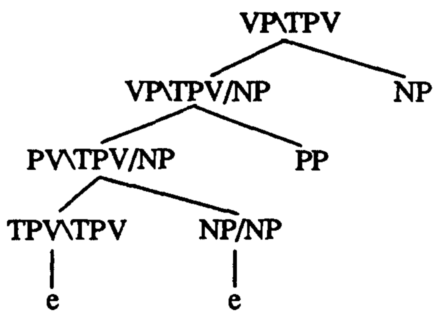


Figure 24

extracted element as in (207).

(206) X → X/Y Y
==>$_R$
X/Z → X/Y Y/Z

(207) [[several news stories *e*] appeared]$_{S/PP}$ [on $e_{PP/NP}$]$_{S/NP}$

Such examples indicate that there must be metarule application to filler introduction rules, and also that the 'value' of a slash must itself be able to carry a slash.

Consider also the following:[19]

(208) I [see each e_i]$_j$ [e_j boy$_i$ on Monday and e_j girl$_i$ on Tuesday]

Here the element which has been left node raised itself contains a gap, which is filled by different nouns in each conjunct. In the analysis of such cases (Figure 25) it is necessary for the slashed category itself to carry a slash specification, and it is again necessary for metarules to apply to filler-introduction rules, because *VP/N* must be instantiated to the gap category *VP/N\(VP/N)* which can be expanded as the empty string.

---

[19] The example cannot be regarded as double left node raising (i) is as follows.

(i) I see$_i$ each$_j$ [e_i e_j boy on Monday and e_i e_j girl on Tuesday]

Such an analysis would require inheritance of a leftward slash *\DET* from a right-hand daughter. This is something which is not possible in the current grammar, and which must not be since it would allow illegitimate word orders.

(209) VP → VP/N N ==>$_L$
VP\(VP/N) → VP/N\(VP/N) N

(210) would require the corresponding treatments.

(210) a. ?He [said that e_i]$_j$ [e_j [he was Italian]$_i$ on Monday and e_j [he was Spanish]$_i$ on Tuesday
b. We [looked for e_i]$_j$ [e_j John$_i$ on Monday and e_j Mary$_i$ on Tuesday]

Dowty's (1988) example (211) is another case in which the value of a slash must bear a slash specification

(211) [Bill gave and Max sold] [a book to Mary and a record to Susan]

In one possible analysis the left-hand and right-hand coordinate structures would be *S/(VP\PTV)* and *VP\PTV* respectively, in another they would be *S/PP/NP* and *S\(S/PP/NP)* respectively.

3. *Discussion*

Section 3.1 discusses category structure in relation to feature distribution in canonical grammar; Section 3.2 discusses various issues relating to categories and compound non-canonicality.

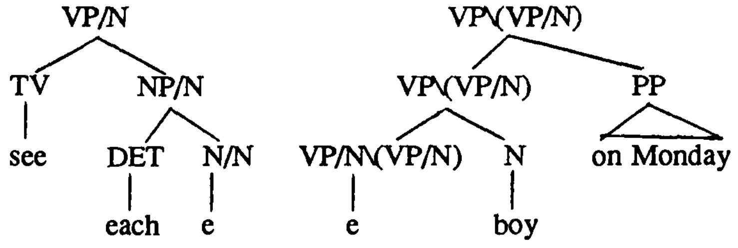


Figure 25

3.1. Category Structure and Feature Distribution

Consider how later versions of generalised phrase structure grammar determine distribution of features as in Figure 26. The relation between head features (e.g. *VFORM*) on a mother and head daughter is governed by a head feature convention such as the following:

(212) The head feature specifications on the head daughter are the same as those on the mother.

The relation between agreement features (e.g. *AGR*) on daughters is governed by a control agreement principle of the following form:

(213) The value of AGR on a controlled daughter is the same as the controlling daughter.

The need for such a daughter-daughter feature distribution principle is removed if we make non-head daughters units of head daughter category structure, as is done in e.g. head-driven phrase structure grammar (Pollard 1985), and unification categorial grammar (Zeevat, Klein, and Calder 1987). On this strategy the category of a transitive verb may be written *S[VFORM FIN, SUBCAT <NP, NP[NUM SG]>]*. A head may combine with the next item on its *SUBCAT*, and the result inherits the remaining *SUBCAT* items. However the grammar still requires a head feature principle specifying identity of head features on mother and head daughter. The earlier analysis takes on the form in Figure 27.

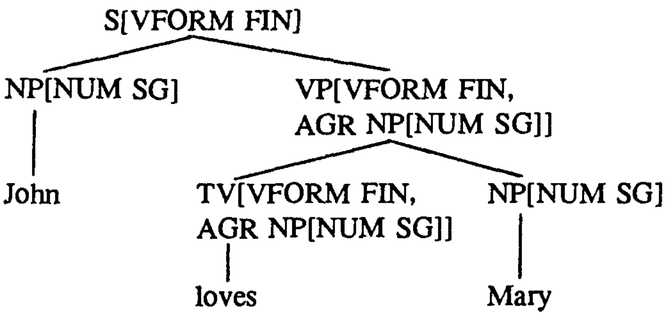


Figure 26: Features in Late Generalised Phrase Structure Grammar

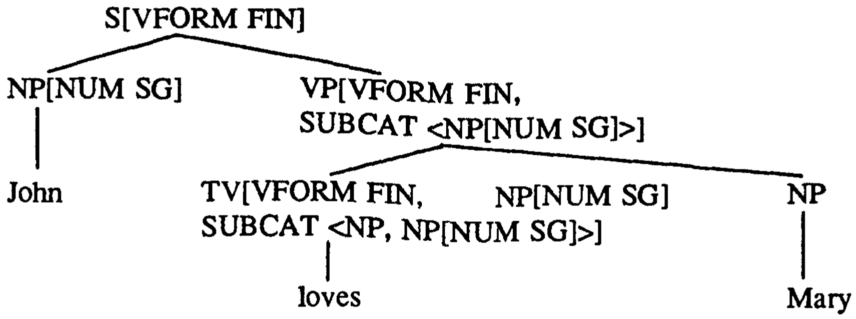


Figure 27: Features in Head-Driven Phrase Structure Grammar

Just as the need for a daughter-daughter feature distribution principle was removed by making non-head daughters a unit of head daughter category structure, head feature distribution between a head daughter and its mother is implemented directly by making the mother a unit of head daughter category structure. This takes us to categorial grammar; with features, a transitive verb category may be written *S[VFORM FIN]\NP[NUM SG]/NP*. The analysis becomes as shown in Figure 28.

The analogue of a head in categorial grammar is a functor. The move to make a non-head daughter a unit of head category structure concurs with Keenan's (1974) observation that functions must agree with their arguments: it follows from the requirement that the

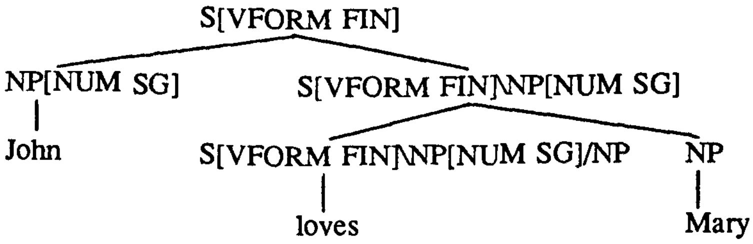


Figure 28: Features in Categorial Grammar

argument categories match. Likewise, the similarity between a mother and its head is rationalised by the fact that a mother is a subunit of head category structure. Through these observations on feature distribution the categorial account of complementation can be motivated.

3.2. Categories and Compound Non-Canonicality

In this chapter PSG has been augmented with a slash apparatus and metarules to characterise non-canonicality. I now consider various issues relating to this approach, loosely structuring discussion around the three components of the analysis: gap introduction, gap transmission, and filler introduction.

I have noted on several occasions that the metarule gap introduction (214) seems to fare better than the null rule gap introduction (215) as regards constraints like the left branch condition and coordinate structure constraint.

(214) a. X → Y Z ==>
X/Z → Y
b. X → Y Z ==>
X\Y → Z

(215) a. X/X → *e*
b. X\X → *e*

Note in this connection that null rule gap introduction and abstraction metarules can together simulate metarule gap introduction. Thus consider application of Right Abstraction as follows:

(216) X → Y Z ==>
X/Z → Y Z/Z

Z/Z in (216) can be expanded as the empty string so that the material dominated by *Y* is analysed as *X/Z*. This is the effect of (214a). Leftwards null rule gap introduction and Left Abstraction can likewise simulate leftwards slash metarule gap introduction. Metarule gap introduction cannot simulate null rule gap introduction: it was noted on several occasions that the latter overgenerates where the former does not. The subsumption of metarule gap introduction by null rule gap introduction, and the overgeneration of the latter, suggests replacement by the former of the latter. In this way empirical considerations lead away from the idea of empty nodes.

Above, gap transmission has been characterised by metarules. However historically this early mechanism was replaced by feature percolation conventions. Accommodation of GPSG to principles of X-bar syntax, and featural analysis of categories, gave rise to the positing of various conventions governing the relation between the features on mother and daughter categories in trees. By interpreting slash as a feature so that, for example, *S/NP* means *S[SLASH NP]*, it became possible for these conventions to govern the relations between slashes on mothers and daughters that were formerly governed by the metarules. Thus the 'foot feature principle' of Gazdar et al. (1985, p82) essentially requires that the slash features appearing on the mother be the union of those appearing on the daughters. Pollard's (1985, p30) 'binding inheritance principle' is a version of such a principle generalised for the case of compound non-canonicality. Note that multiple extraction requires that the value of a slash feature be a list (or set, or partially ordered set), so that say *S/PP/NP* means *S[SLASH <NP,PP>]*. Pollard's inheritance principle is described procedurally as a process of popping the daughter slash stacks, and appending or merging (corresponding to parasitic extraction) to obtain the mother slash stack. The effect is very much like that of the metarules above: they can be regarded as defining such inheritance conventions, the presence or absence of each metarule being a parameter.

However, although one possibility is to interpret the early GPSG slash as a *feature on* categories, there is an alternative possibility, which is to interpret it as an *operator over* them, in the sense that it constructs categories out of categories. There are two rather different views of grammar pivoting around these interpretations of slash. Typically, on the featural side a grammar will have one slash stack-valued feature storing gap categories, and a subcategorization stack-valued feature storing complement categories. Thus categories are feature structures like (217) with various features including *SUBCAT* and *SLASH*.

(217) X[SUBCAT <...>, SLASH <...>, ...]

Gap introduction consists of popping from the subcategorization stack and pushing onto the slash stack. This is the path followed by e.g. Pollard (1985, 1988a,b) in Head-Driven Phrase Structure Grammar, Bouma (1987), and Calder, Klein, Moens, and Reape (1988).

I would like to suggest however that the course we have taken can at least as well be taken as implying an operator interpretation, leading to a convergence of the phrase structure grammar line of inquiry, and the categorial grammar one of Steedman, Dowty, and others.

In Sections 2.1 and 2.2 of this chapter I have shown at length that there are multiple extractions and that the category apparatus must allow stacking of slashes. There was not a substantial body of data involving more than two extractions, but triple extractions do occur in, for example, Scandinavian languages. We might posit an upper bound of the number of extractions in English, e.g. by limiting the depth of recursion of metarules to two. However this seems slightly stipulative.[20] In Section 2.1 of Chapter VII I discuss how a grammar with metarules can be regarded as defining a hierarchy of languages, leading to the idea of degrees of grammaticality. The grammar might characterise the relative unacceptability of multiple extractions gracefully, by reference to a hierarchy of syntactic complexity, rather than by positing a sudden cut-off point. Then the reason for the difference between English and Scandinavian languages may be found in such factors as the means of expression available, and processing of the different grammars, as opposed to in the setting of a grammatical parameter. But in universal grammar at least, it does not seem necessary to propose a bound on the depth of possible stacking of slash categories.

In Section 2.3 I showed that extracted elements may themselves contain gaps, i.e. that the category apparatus must allow nesting of slashes. The coexistence of stacking and nesting of slash categories fits with an interpretation of slash as a freely applying constructive operator over categories. I have advocated also use of two different slashes: one 'leftwards' and one 'rightwards'. The result is precisely the category apparatus of directional categorial grammar.

Further, the filler-introduction rules (218) have the semantics of the application primitives of categorial grammar.

(218) a. X → X/Y Y
b. X → Y X\Y

If we encode complements on slashes, as in the CG of Chapter I, (218) can function as complement introduction rules as well as filler introduction rules. And this move can eliminate entirely the need for a gap introduction device, at least when it is an argument that is extracted, because arguments are 'already on slashes'.[21] Although on the description here the slash operator will signal both gap categories and complement categories, it would be possible to use different operators in the two cases. For example, if '!' were the gap

---

[20] Engdahl (1986, pp22-24, 132-37) also argues against the imposition of an upper bound.

[21] There is nothing actually technically anomalous about having empty nodes in a categorial grammar: the empty string could be regarded as belonging to categories *X/X* and *X\X*, having the identity function as its meaning. Then the semantics is coherent as in the phrase structure grammar. However the left branch constraint and coordinate structure constraint violations mentioned above re-occur, and there is also the dubious possibility of the empty string being supplied as an argument. While

argument operator, we could have a gap introduction rule mapping e.g. *S\NP/NP* to *S\NP!NP*; then a resumptive pronoun may have a category *X!NP\(X/NP)*. In fact we would probably want to characterise similarities and differences between complement and gap slashes by having slashes as structures rather than atoms, as was proposed for gap slashes in Section 1.3.

---

theoretically possible, empty nodes in a categorial grammar seem empirically both unnecessary, and undesirable.

Chapter III

Categorial Grammar Extended with Rules

A variety of rules have been devised by Steedman, Dowty, Moortgat, Szabolcsi, and others, to characterise non-canonicality within a categorial framework. Steedman has referred to this general approach as Combinatory Categorial Grammar in view of the close relation of the semantics of these rules to that of the combinatory logic combinators of Curry and Feys (1958). In Section 1 I discuss a treatment of simple non-canonicality based largely on existing work, and in Section 2 I discuss compound non-canonicality.

1. Simple Non-Canonicality

I consider coordination of 'non-constituents' in Section 1.1, and extraction in Section 1.2.

1.1. Coordination of 'Non-Constituents'

Sections 1.1.1 and 1.1.2 address right node raising and left node raising respectively. I continue to assume the like-category coordination schema of Chapter I; in derivations instances of this will be labeled 'Crd'.

1.1.1. Right Node Raising

Consider the following:

(1) I [read e_i and will reference e_i] [your paper]$_i$

In the CG grammar of Chapter I, *will reference* does not form a constituent. The assumption that conjuncts are constituents of the same category indicates that the CG base grammar should be augmented in such a manner that (where *VP* abbreviates *S\NP*) the sequence consisting of the auxiliary *VP/VP* and the transitive verb *VP/NP* in the right hand conjunct of (1) are able to form a constituent of category *VP/NP*, matching the transitive verb category on the left hand side. The following rule achieves this effect:

(2) *Forward Composition* (>B)

$$X/Y: x \; + \; Y/Z: y \; \Rightarrow \; X/Z: \mathbf{B}\, x\, y$$

(3) $$\mathbf{B}\, x\, y\, z \equiv x\, (y\, z)$$

The rule is called *forward partial combination* in Ades and Steedman (1982, p527) and Steedman (1985, p533), and *Forward Composition* in Steedman (1987a), since its semantics is *functional composition*, denoted by the combinator **B**. The composition of two functions x and y is that function which, applied to an argument z, yields the same result as would be obtained from applying y to z, and then applying x to the result. Thus in λ-terms, the composition of x and y is $\lambda z[x\ (y\ z)]$. A combinatory logic is an applicative system like the λ-calculus, but one in which functional abstraction is not expressed by λ-binding of variables, but by various combinator primitives of which **B** and **W** are defined as in (4).

(4)
a. $((\mathbf{B}\, x)\, y)\, z \equiv x\, (y\, z).$
b. $(\mathbf{W}\, x)\, y \equiv (x\, y)\, y$

The combinator primitives determine certain elementary abstraction operations; application of combinators to each other yields combinators expressing more complex abstractions. For example (**BB**)**W** is equivalent to $\lambda x \lambda y \lambda z[x\ (y\ z)\ (y\ z)]$:

(5)
$$\begin{aligned}
&((((\mathbf{B}\, \mathbf{B})\, \mathbf{W})\, x)\, y)\, z = \\
&((\mathbf{B}\, (\mathbf{W}\, x))\, y)\, z = \\
&(\mathbf{W}\, x)\, (y\, z) = \\
&(x\, (y\, z))\, (y\, z)
\end{aligned}$$

Henceforth I will continue with a left-associativity convention for application in combinatory logic. Other combinators include, e.g. identity **I** and commutation **C**:

(6) $$\mathbf{I}\, x \equiv x$$

(7) $$\mathbf{C}\, x\, y\, z \equiv x\, z\, y$$

Example (1) has the following analysis:

```
(8)    I    read     and    will         reference   your  paper
       --  -------          ----------- ---------   ----  -----
       NP  S\NP/NP          S\NP/(S\NP)  S\NP/NP    NP/N    N
                            ------------------->B   -------->
                                  S\NP/NP              NP
           -------------------------Crd
                     S\NP/NP
                     ------------------------------->
                                  S\NP
       ----------------------------->
                     S
```

The canonical, applicative, derivation of *will reference your paper* assigns the meaning *will′ (reference′ your-paper′)*. As well as allowing generation of the coordination example, the new rule means that e.g. *will reference your paper* has a non-canonical analysis:

```
(9)    will  reference  your paper
       ----- ---------- ----------
       VP/VP   VP/NP       NP
       ------------->B
           VP/NP
           ----------------->
                  VP
```

Under this analysis, the meaning of the subexpression *will reference* is **B** *will′ reference′* (i.e. *λz[will′ (reference′ z)]*): the composition of *will′* and *reference′*. Applying this to *your-paper′* yields **B** *will′ reference′ your-paper′* which reduces to *will′ (reference′ your-paper′)*, the meaning obtained under canonical analysis. So the different analyses yield the same meaning, and this is correct because the expression is unambiguous. As was the case for the phrase structure grammar, the semantic operations of rules are such that there will be many analyses of expressions assigning the same meaning. In general any analysis presented will be one of many possible ones.

As Steedman points out, Forward Composition immediately characterizes a range of right node raising phenomena, thus:

(10) I [desire e_i and want to search for e_i] [the meaning of life]$_i$

```
(11)   want    to       search       for
       ----- -----    ---------- --------
       VP/VP VP/VP    VP/(VP\VP) VP\VP/NP
       ---------->B   ------------------->B
         VP/VP                VP/NP
         ---------------------->B
                   VP/NP
```

However other instances are blocked:

(12) [I liked but Mary disliked] the second play

```
(13)   I   liked
       --  -------
       NP  S\NP/NP
```

Such cases can be captured with the help of a rule called *Forward Type-Raising* (see Steedman 1985, 1987a, and Dowty 1988, and references therein):[1]

(14) *Forward Type-Raising* (>T)
X: x => Y/(Y\X): T x

(15) T x y ≡ y x

The rule can be applied to a subject *NP* to yield *S/(S\NP)* where *Y* is instantiated to *S*. This is now of the right form to compose with a transitive verb *S\NP/NP* to yield *S/NP*, enabling classical right node raising:

(16) [I liked but Mary disliked] the second play

```
(17)     I          liked
      --------    -------
         NP       S\NP/NP
      -------->T
      S/(S\NP)
      ------------------>B
            S/NP
```

In each conjunct, the subject NP is Forward Type-Raised to *S/(S\NP)* and then Forward Composed with the transitive verb, *S\NP/NP*, to form an expression of category *S/NP*. The coordinate structure as a whole, of category *S/NP*, applies to the right node raised *NP* to form a sentence.

While right node raising of complements will proceed along the general pattern illustrated above, type-raising of nouns over adnominals, and verb phrases over adverbials, is needed to obtain right node raising of adjuncts. Right node raising of adnominals can be achieved thus:[2]

---

[1] The rule is related to Montague's (1973) assignment to a proper name like *John* the semantics $\lambda x[x\ john']$ of type $(e \rightarrow t) \rightarrow t$ where *john'* is the type *e* constant denoting the individual "John". Montague's motivation was to bring proper names up to the same type as quantified noun phrases, so that a uniform treatment could be provided.

[2] Note that these operations also provide the means for a [[DET N] REL] analysis of complex noun phrases by semantically abstracting *DET+N* over *REL*.

```
(18)    a      man     and   a     woman    who like Beethoven
       ----    ---         ---- -----    -----------------
       NP/N     N          NP/N   N            N\N
             ------>T          ------>T
             N/(N\N)           N/(N\N)
       ------------->B   ------------>B
        NP/(N\N)           NP/(N\N)
        ----------------------------Crd
                   NP/(N\N)
                   --------------------------->
                              NP
```

Right node raising of adverbials such as that in *John arrived and Mary left yesterday* is achieved similarly, except that in addition the subject must be type-raised to compose.

The unboundedness of right node raising arises because type-raising and composition can run through clause boundaries:

(19) Bill likes and Mary said that Sue adores, cheese soufflé

```
(20)    Mary      said      that   Sue      adores
        ----    -------     ----   ---     -------
        NP      S\NP/SP     SP/S   NP      S\NP/NP
        ------>T ------------>B    ------>T
        S/(S\NP)   S\NP/S          S/(S\NP)
        ------------------>B       ------------------>B
             S/S                        S/NP
             ---------------------------->B
                          S/NP
```

1.1.2. Left Node Raising

Dowty (1988) provides an account of left node raising employing Backward Composition and Backward Type-Raising counterparts to Forward Composition and Forward Type-Raising:[3]

(21) *Backward Composition* (<B)
Y\Z: y + X\Y: x => X\Z: **B** x y

(22) *Backward Type-Raising* (<T)
X: x => Y\(Y/X): **T** x

Note that these rules are exact mirror images of the rules of forward composition and type-raising, in accord with the symmetry of left node raising and right node raising for

---

[3] A non-directional version of (21) is called *backward partial combination* in Steedman (1985, p533).

which the respective rules account.

Consider first complement-adjunct left node raising:

(23) I met [John on Monday and Mary on Tuesday]

A suitable category for the conjuncts is *VP\(VP/NP)*: they form verb phrases once they apply to transitive verbs on their left. The adverbials are *VP\VP*, and the *NP* objects can be Backward Type-Raised to *VP\(VP/NP)*, where *Y* in (22) is instantiated to *VP*. Then an object, *VP\(VP/NP)*, and an adverbial, *VP\VP*, can combine by Backward Composition to form a constituent of category *VP\(VP/NP)*, as desired:

```
(24)        John    on Monday
            ----    ---------
             NP       VP\VP
          ----------<T
          VP\(VP/NP)
          -----------------<B
             VP\(VP/NP)
```

The rest of the analysis is straightforward.

Next, consider the complement-complement case:

(25) I gave [a book to John and a record to Sue]

The first complement, *NP*, can be backward type-raised to *VP/PP\(VP/PP/NP)* (where *PP* abbreviates *S\NP\(S\NP)*): something seeking a prepositional phrase to its right once it combines with a transitive prepositional verb to its left. The second complement can be backward type-raised to *VP\(VP/PP)*, and now these can backward compose to form conjuncts of category *VP\(VP/PP/NP)*:

```
(26)          a book           to John
              ------           -------
                NP               PP
              ------<T         ------<T
          VP/PP\(VP/PP/NP)   VP\(VP/PP)
          ----------------------------<B
                  VP\(VP/PP/NP)
```

Left node raising of a determiner proceeds thus:

(27)
```
Each    boy     dances and    girl      sings
----    ---     ------        ----      -----
NP/N    N       S\NP          N         S\NP
        --------<T            --------<T
       NP\(NP/N)             NP\(NP/N)
        --------------<B      --------------<B
          S\(NP/N)               S\(NP/N)
          ----------------------------Crd
                     S\(NP/N)
--------------------------<
                S
```

Extraction of an incomplete element is obtained by assembling that element with non-canonical operations:

(28) I [see each e_i]$_j$ [e_j boy$_i$ on Monday and e_j girl$_i$ on Tuesday]

(29)
```
 see   each       boy       on Monday ...
-----  ----       ---      -------------
VP/NP  NP/N       N          VP\VP
----------->B ---------<T
  VP/N       VP\(VP/N)
             ------------------<T
                 VP\(VP/N)
```

It will be seen that as before no inapplicable new meanings are assigned by the alternative analyses of expressions made possible by the additional rules. Also there is a sense in which the rules are order-preserving (see Dowty 1988). Thus type-raising X to $Y/(Y\backslash X)$ still only allows X and $Y\backslash X$ to combine in their original order $X+Y\backslash X$ to form Y. Likewise, composing X/Y and Y/Z results in X/Z so that subsequent application results in the original $X/Y+Y/Z+Z$ order. The corresponding situation holds for the backward rules. Since the existing rules are 'safe' in that they preserve order, there is little motivation to constrain them. Yet English does have some order-variation. In particular, right extraction appears to demand a rule which is essentially order-changing.

1.2. Extraction

Sections 1.2.1 and 1.2.2 discuss right extraction and left extraction respectively; Section 1.2.3 deals with various issues involving the fronted elements in relativisation: pied piping, *that*-relatives and *that*-less relatives. Section 1.2.4 considers parasitic extraction.

1.2.1. Right Extraction

Consider the following instance of heavy shift:

(30) I met e_i yesterday [an old school friend who has become a respected film critic]$_i$

We seem to need to combine *met*$_{VP/NP}$ and *yesterday*$_{VP\backslash VP}$ to form an expression of category *VP/NP* so that the result can apply forward to a rightwardly displaced heavy noun phrase. The rule (31) achieves this, so that (30) receives the analysis shown in (32).

(31) *Mixed Backward Composition* (<$\mathbf{B}_x$)

$Y/Z\text{: } y \ + \ X\backslash Y\text{: } x \ \Rightarrow \ X/Z\text{: } \mathbf{B}\ x\ y$

(32)

```
 met   yesterday  an old ...
-----  ---------  ----------
VP/NP    VP\VP        NP
-------------<Bx
    VP/NP
    ------------------>
             VP
```

Mixed Backward Composition appears in Moortgat (1988), Morrill (1987a), and Steedman (1987a). The forward counterpart is (33).

(33) *Mixed Forward Composition* (>$\mathbf{B}_x$)

$X/Y\text{: } x \ + \ Y\backslash Z\text{: } y \ \Rightarrow \ X\backslash Z\text{: } \mathbf{B}\ x\ y$

The use of (31) as opposed to (33) corresponds to the phrase structure grammar inheritance of leftward slashes from left-hand but not right-hand daughters (see Chapter II, footnote 18). The rule (33) must be strictly prohibited in English, otherwise orderings like (34) and (35) are obtained.

(34)

```
*John that left
 ---- ---- ----
  NP  SP/S S\NP
      --------->Bx
        SP\NP
 -----------<
        SP
```

(35) Met$_i$ I e_i John yesterday

Consider now the following, in which there is heavy shift past a complement:

(36) I gave e_i to John [a large red box]$_i$

In a case like this the *PP* complement can be backward type-raised to *VP\(VP/PP)*, and this can combine by mixed backward composition with *VP/PP/NP* on its left to give *VP/NP*, again looking for the *NP* right of the second complement, as shown in (37).

```
(37)    gave      to John     a large red box
      -------- --------- --------------
      VP/PP/NP      PP           NP
                 ---------<T
                 VP\(VP/PP)
      ------------------<Bx
            VP/NP
            ----------------->
                     VP
```

In this way, heavy shift past an adjunct is achieved by mixed backward composition, and heavy shift past a complement is achieved by backward type-raising the complement, and then performing mixed backward composition. As in the phrase structure grammar, 'particle shift' can be treated on the same pattern as heavy shift, and is actually closer conceptually to 'movement' of the object than to that of the particle:

```
(38)   I     rang            up            John
      -- ----------          --            ----
      NP S\NP/PP/NP          PP             NP
                     --------------<T
                     S\NP\(S\NP/PP)
         ------------------------<Bx
               S\NP/NP
               -------------------------->
                         S\NP
      -----------------------<
                  S
```

Consider next right extraposition:

(39) A man e_i arrived [who spoke Russian]$_i$

A noun like *man*$_N$ can be forward type-raised to *N/(N\N)* -- a noun type-raised over an adnominal such as a relative clause. A determiner *a*$_{NP/N}$ can forward compose with this to give *a man*$_{NP/(N\backslash N)}$ and the analysis can proceed as shown in (40). This is the account of right extraposition presented in Morrill (1987a); Moortgat (1988) gives the corresponding treatment for right extraposition in Dutch.

```
(40)    a      man       arrived who spoke Russian
       ----   ---       ------- ---------------
       NP/N    N         S\NP          N\N
              ---->T
             N/(N\N)
       ----------->B
         NP/(N\N)
          ----------------<Bx
               S/(N\N)
               ----------------------->
                          S
```

The meaning of type-raised *man* is T man′ (i.e. λ*x[x man′]*) and the meaning of the expression as a whole, which is the same as that of its canonical counterpart, is derived thus:

(41) a man ⇒ B a′ (T man′)
a man arrived ⇒ B arrived′ (B a′ (T man′))
a man arrived who speaks Russian ⇒
B arrived′ (B a′ (T man′)) who-speaks-Russian′

(42) B arrived′ (B a′ (T man′)) who-speaks-Russian′ =
arrived′ (B a′ (T man′) who-speaks-Russian′) =
arrived′ (a′ (T man′ who-speaks-Russian′)) =
arrived′ (a′ (who-speaks-Russian′ man′))

Mixed Backward Composition performs a similar function to Middle Abstraction in the phrase structure grammar, and accordingly I propose to restrict 'Z' to {*N\N, SP, NP, S\NP\(S\NP), N/N*} to prevent examples like **a left man.*

1.2.2. Left Extraction

The rules that have been presented so far also provide an account of left extraction. Iteration of operations can build an unboundedly long bridge between a gap and a filler, as was first observed in Ades and Steedman (1982):[4]

---

[4] In Ades and Steedman subjects were pre-assigned a higher type so that only composition was required.

```
(43)    who         John    thinks     that   Mary    likes
        ----------  ----    ------     ----   ----    -----
        N\N/(S/NP)  NP      S\NP/SP    SP/S   NP      S\NP/NP
                    ------->T                 ------>T
                    S/(S\NP)                  S/(S\NP)
                    --------------->B         ------------------>B
                         S/SP                       S/NP
                         ----------------->B
                                 S/S
                                 ---------------------->B
                                         S/NP
        ------------------------------------>
                    N\N
```

In (43) the object relative pronoun is lexically assigned a higher type function over *S/NP*, a sentence with a noun phrase gap, so that object relative pronoun filler introduction is keyed to lexical assignment, in the same way that the subject relative pronoun of Chapter I operated by a lexical category *N\N/(S\NP)*.

Consider the topicalised example (44).

(44) [The beginning]$_i$ John thinks that Mary likes e_i

John thinks that Mary likes may be assembled into an expression of category *S/NP* as before, but a number of possibilities suggest themselves for topic introduction, which must involve violation of the directionality specified by the slash. First, there could be a lexical topic rule such that topics have a category *S/(S/X)*. For example if *the* had a topic lexical category *S/(S/NP)/N*, then *the beginning* in (44) would be of category *S/(S/NP)*, which is similar to the *N\N/(S/NP)* category of an object relative pronoun. Second, there could be a unary 'pseudo-type-raising' syntactic topicalisation rule whereby a topic of category *X* can type-raise to *S/(S/X)* (see Steedman 1987a).[5] Third, there could be some syntactic slash-switching rule mapping *[John thinks that Mary likes]*$_{S/NP}$ into *S\NP*.[6] Fourth, there could be a syntactic topicalisation rule combining a topic and an incomplete sentence directly (Morrill 1987b), as in the phrase structure grammar:

(45) *Topic Introduction* (t)

X: x + S/X: y => S: y x

X ∈ {NP, SP, N/N, N\N, S\NP\(S\NP)}

Note that this will assign a topicalised sentence the same meaning as its canonical counterpart. As mentioned earlier, the markedness of the intonation accompanying topicalisation

---

[5] The rule is 'pseudo' in the sense that unlike the other type-raising rules, this is not 'order-preserving'.

[6] The applicability of such a rule would have to be restricted so as to prevent, for example, **Sue [[[John thinks that Mary likes]*$_{S/NP}$]$_{S\backslash NP}$ and left$_{S\backslash NP}$].

can be interpreted as indicating that a rather unusual operation is at work, so almost any topicalisation rule could be a candidate. However I adopt (45) in favour of the topic lexical category solution because it seems unreasonable to make words lexically ambiguous just so that topics can be introduced. And I adopt (45) over pseudo-type-raising and slash-switching because it collapses the unary-operation-plus-syntactic-application of the other proposals into a single step: it seems that pseudo-type-raising should only ever be followed by application to a sentence lacking the topic, and slash-switching should only ever be followed by backward application to the topic; the intermediate categories apparently never participate in processes of composition etc. Evidence to the contrary might motivate the alternative rules.

Left extraction of an adjunct is achieved by type-raising a head over an adjunct, after which derivation proceeds along the pattern of argument extraction. Thus for topicalisation of an adverbial there is the following analysis:

```
(46)    On Monday     John        arrived
        ----------    ----    ----------------
        S\NP\(S\NP)    NP           S\NP
                      ----->T ---------------->T
                     S/(S\NP) S\NP/(S\NP\(S\NP))
                     ---------------------------->B
                             S/(S\NP\(S\NP))
        ----------------------------------t
                              S
```

In the case that extraction is from a clause non-final position, mixed composition is required to achieve the S/X 'clause-with-gap' category. For example the adjunct *yesterday*$_{VP\backslash VP}$ will combine with *met*$_{VP/NP}$ by mixed backward composition to give VP/NP, from which derivation can proceed as above to yield (47).

(47) (the man) who$_i$ I met e_i yesterday

Similarly (48) has the derivation (49).

(48) (the book) which$_i$ I gave e_i to John

```
(49)      which      I          gave        to John
        ----------  --        ----------    -------
        N\N/(S/NP)   NP       S\NP/PP/NP       PP
                   ------>T                 -------------<T
                   S/(S\NP)                 S\NP\(S\NP/PP)
                              --------------------------<Bx
                               S\NP/NP
                   ------------------->B
                           S/NP
        ----------------------->
                  N\N
```

Note the relation of this to the right extraction account. In general a fronted element combines with a clause of category S/X, which could have applied forwards to X, so that as was the case for the PSG based grammar, the theory predicts that elements which can left extract should be able to right extract.

Consider the following extractions of subjects:

(50) a. *the man who$_i$ I believe that e_i left

b. the man who$_i$ I believe e_i left

Steedman (1987a) attributes to Szabolcsi the observation that the current grammar respects the fixed subject constraint:

```
(51)    that  left
        ----  ----
        SP/S  S\NP
        ----------*
```

The combination in (51) could be achieved by mixed forward composition as shown earlier, and this rule is not to be present in English. But under the obvious category assignment, we also fail to generate the extraction in (50b):

```
(52)    believe  left
        -------  ----
        S\NP/S   S\NP
        -------------*
```

The solution adopted here is that *believe* is of the category *S\NP/VP/NP* whereby the subordinate subject and predicate verb phrase are sought separately (Steedman 1987a suggests the same). Ewan Klein (personal communication) notes that this leaves as a puzzle the question of why no verb in English or any other language takes a finite verb phrase complement on its own. However, assuming that *believe* is of this category, it is not necessary to assume that it is also of category *S\NP/S*, so we have the benefit of an account which doesn't rely on extra categories or rules exclusively for the purposes of achieving subject extraction:

```
(53)    who          I         believe        left
        ----------   --        ----------     ----
        N\N/(S/NP)   NP        S\NP/VP/NP     VP
                     ------->T                ------------<T
                     S/(S\NP)                 S\NP\(S\NP/VP)
                               --------------------------<Bx
                                         S\NP/NP
                     --------------------------->B
                                   S/NP
        -------------------------->
                     N\N
```

To-infinitival complement verbs will have the same categorization, enabling extraction like the following:

(54) the people who$_i$ I wanted e_i to go

According to the above account subject gaps are like object gaps in that e.g. *I voted for* and *I believe won* are both *VP/NP*: As Steedman (1987a, p424) points out, this is consistent with the fact that the same relative pronouns appear with both if object relative pronouns are attuned to slash directionality as opposed to case:

(55) a. the man who(m)$_i$ I voted for e_i
b. the man who(m)$_i$ I believe e_i won

And also as would be expected, elements with object and subject gaps can be coordinated:

(56) the man who(m)$_i$ I [voted for e_i and believe e_i won]

The grammar correctly captures the fact that the following is not possible:

(57) *who swims and I know

This follows because the conjuncts have distinct categories, *S\NP* and *S/NP*.

Note that mixed backward composition is needed to achieve across-the-board extraction from a non-right-peripheral position; (58a) has the derivation shown in (59) which illustrates how the conjuncts can be analysed as the transitive verb category as required in (58b) and (58c).

(58) a. I [gave e_i to members and sent e_i to affiliates] [long and detailed reports]$_i$

b. I [read e_i and sent e_i to affiliates] [long and detailed reports]$_i$

c. I [gave e_i to members and published e_i] [long and detailed reports]$_i$

(59)

```
   gave        to members    and       sent      to affiliates
 ----------  ----------            ----------  -------------
 S\NP/PP/NP      PP                S\NP/PP/NP       PP
             ------------<T                    -----------<T
             S\NP\(S\NP/PP)                    S\NP\(S\NP/PP)
 --------------------------<Bx     --------------------------<Bx
         S\NP/NP                            S\NP/NP
         ------------------------------------------Crd
                           S\NP/NP
```

A principle empirical weakness of the phrase structure grammar with null rule gap introduction was certain violations of the coordinate structure constraint. These violations do not occur in the CG grammar, essentially because there are no empty nodes; the grammar predicts that extraction of an entire conjunct is disallowed:

(60) *the man who$_i$ I saw [e_i and a picture of e_i]

This follows simply because the empty string is not a member of any category. The correct predictions of the phrase structure grammar as regards the coordinate structure constraint and across-the-board exceptions carry over to the categorial grammar. Coordinations such as those in (61) are forbidden because the conjuncts are of unlike category.

(61) a. *the people who$_i$ [John likes e_i and Sue likes Mary]

b. *the people who$_i$ [John likes Mary and Sue likes e_i]

As was the case for the phrase structure grammar, the current grammar does not predict that right extraction is bounded at all. It would be possible to achieve heavy shift of one complement past a second by some lexical commutation operation mapping $VP/CMP_2/CMP_1$ to $VP/CMP_1/CMP_2$ but this would not suffice for heavy shift past an adverbial, and right extraposition like (62) exceeds domains 'governed' by lexical categories, and so cannot be so straightforwardly achieved by manipulation of lexical categories.

(62) A number of stories e_i appeared [about the Watergate Affair]$_i$

Also like the phrase structure grammar, the existing grammar does not lead us to expect constraints such as the complex noun phrase constraint or A-over-A constraint. Nor does it predict that extraction from a subject is impossible:

```
(63)    whom        a  picture of   sold  for  seven million pounds
        ---------  -------------  ----------------------------------
        N\N/(S/NP)      NP/NP                        S\NP
                       ------------------------------<Bx
                                          S/NP
        -------------------------------->
                     N\N
```

Extraction from adjuncts is made possible by type-raising over them as shown in (64); it has already been shown that it is necessary to type-raise over adjuncts in order to obtain right node raising, and extraction, of adjuncts.

```
(64)    went to Paris  without finishing
        ------------  ----------------
            VP            VP\VP/NP
          ---------->T
          VP/(VP\VP)
          -------------------------->B
                  VP/NP
```

As before it might be possible to implement constraints by stipulation, but also as before, pursuit of such a direction could of itself only be the beginning of a proper explanation.

1.2.3. Pied Piping, That-Relatives, and That-Less Relatives

Examples of pied piping like (65) can be managed straightforwardly by lexical assignment of categories *N\N/(S\NP)/N* and *N\N/(S/NP)/N* to *whose.*

(65) a. the man whose book arrived

b. the man whose book I read

Cases like (66b) are more challenging.

(66) a. the people whom$_i$ we looked for e_i

b. the people [for whom]$_i$ we looked e_i

The pied piped relative clause in (66b) is synonymous with the relative clause in (66a). Example (66a) has the following analysis:

(67)
```
whom            we      looked     for
----------      --      -------    -----
N\N/(S/NP)      NP      S\NP/PP    PP/NP
               ------>T -------------->B
               S/(S\NP)    S\NP/NP
               ------------------->B
                       S/NP
--------------------->
        N\N
```

We looked for is assigned the meaning of the sentence abstracted over the meaning of the preposition's object, i.e. $\lambda x[looked'\ (for'\ x)\ we']$, so that the relative clause meaning is $whom'(\lambda x[looked'\ (for'\ x)\ we'])$.[7] The semantics of *who(m)* will be $\lambda x\lambda y\lambda z[x\ z\ \&\ y\ z]$ but I will continue to write *who(m)′*. Szabolcsi (1987, section 4) characterises pied piping by lexically assigning a relative pronoun of ordinary category *N\N/(S/NP)*, to an additional pied piper category *N\N/(S/X)\(X/NP)*.[8] Morrill (1987b) presents a binary syntactic schema for pied piping.[9] The accounts are equivalent except for the lexical/syntactic contrast; I shall choose the latter because it collapses the (lexical) type-raising and (syntactic) application of the former into one (syntactic) step. We have then (68) which combines the pied piped material *X/Y* and a relative pronoun *N\N/(S/NP)* or relative pronoun with pied piped material *N\N/(S/Y)*.

(68) *Non-Subject Pied Piping* (p_n)

$X/Y{:}\ x\ +\ N\backslash N/(S/Y){:}\ y \Rightarrow N\backslash N/(S/X){:}\ \lambda z[y\ (\lambda w[z\ (x\ w)])]$

$X, Y \in \{PP, NP\}$

'X' may be *VP\VP*, *N\N*, and possibly also *NP*:

(69) a. the people [for whom]$_{N\backslash N/(S/(VP\backslash VP)),i}$ we looked e_i

b. the subject [about which]$_{N\backslash N/(S/(N\backslash N)),i}$ we had an argument e_i

c. ?the man [a picture of whom]$_{N\backslash N/(S/NP),i}$ John purchased e_i

In (68), *x* and *y* will be the meanings of the pied piped material and the relative pronoun respectively. The fronted constituent formed will apply to a sentence lacking a gap and return the result of applying the relative pronoun meaning to the composition of the meanings of the sentence-with-gap and the pied piped material. Thus in (70) *for whom* has meaning $\lambda z[whom'\ (\lambda w[z\ (for'\ w)])]$. *We looked* has the sentence meaning abstracted over the prepositional phrase, i.e. $\lambda v[looked'\ v\ we']$, so that the overall meaning is

---

[7] In the interests of ease of comprehension, the λ-calculus, rather than combinatory logic, will be used in discussion of semantics in this section.

[8] Szabolcsi's proposals are made primarily with reference to reflexives and reflexive pied piping (*Mary likes herself*; *Mary talks about herself*).

[9] That schema collapses subject and non-subject pied piping at the expense of having redundant instantiations; below I factor out subject and non-subject pied piping.

whom′(λx[looked′ (for′ x) we′]), the same as the non-pied piped relative clause.

```
(70)    for     whom          we      looked
        -----   ----------    --      -------
        PP/NP   N\N/(S/NP)    NP      S\NP/PP
        ---------------pn     ------>T
          N\N/(S/PP)          S/(S\NP)
                              ---------------->B
                                   S/PP
          ------------------------->
                      N\N
```

Consider next the following:

(71) a. the man whom$_i$ a picture of e_i exists
b. the man a picture of whom exists

Mixed composition enables (71a) to be derived as shown in (73). However the existing schema does not capture (71b) because this requires the fronted element to combine with a verb phrase *S\NP* rather than a sentence-with-gap *S/NP*. For subject pied piping I propose:

(72) *Subject Pied Piping* (p$_s$)
NP/Y: x + N\N/(S/Y): y => N\N/(S\NP): λz[y (λw[z (x w)])]

Note that the semantics is the same as that of non-subject pied piping, and that in the same manner as before, the meanings assigned in (73) and (74) are the same.

```
(73)    whom         a      picture   of      exists
        ----------   ----   -------   -----   ------
        N\N/(S/NP)   NP/N   N/PP      PP/NP   S\NP
                     ---------->B
                       NP/PP
                       --------------->B
                            NP/NP
                            ---------------->Bx
                                 S/NP
        ----------------------------->
                  N\N
```

```
(74)    a picture of    whom         exists
        ------------    ----------   ------
           NP/NP        N\N/(S/NP)   S\NP
          -------------------ps
              N\N/(S\NP)
              -------------------->
                     N\N
```

The current grammar does not generate *that*-less relatives such as (75).

(75) the man John met

Ruling out a deletion analysis, several possibilities suggest themselves. First we might try to assign extra lexical categories, but since the gap in the clause can be unboundly deep, it seems unlikely that this is really a lexical phenomenon. Alternatively, we could employ a binary rule combining *N* and *S/NP* to form *N*, or else a unary rule mapping *S/NP* to $N\backslash N$. Under the assumption of like-category coordination, the acceptability of (76) indicates that the sentence-with-gap itself belongs to the relative clause category, indicating the unary mapping of *S/NP: x* to $N\backslash N$: *whom′ x*.

(76) the man [John met and who Mary (eventually) married]

Before we formulate the rule, consider an important feature of *that*-relatives which is that they cannot exhibit pied piping:

(77) a. the man that we looked for
b. *the man for that we looked

If we assign *that* to the object relative pronoun category, it will pied pipe under the existing schemas. In Old English a *wh*-relative pronoun could be followed by *that*, implying that *that* is functioning as a complementizer. If we assume that the semantics of the complementizer is essentially the identity function, we can have a mapping of *SP/NP* to $N\backslash N$ with semantics as before, and because it does not have the relative pronoun category, *that* cannot pied pipe. Gabriel Bes (personal communication) has pointed out that the same device appears to capture *for*-relatives such as (78) where again the relative clause consists of a complementized sentence with an *NP* gap.

(78) the man for you to meet

Overall then we have the following:

(79) *Object That and That-Less Relatives* (r_0)
X/NP: x => $N\backslash N$: $\lambda y \lambda z[x\ z\ \&\ y\ z]$
$X \in \{S, SP\}$

Note that although the category mapping looks arbitrary, this conceals the semantic similarity between *S/NP* and *N* which are both of type $e \rightarrow t$. The semantics of the rule involves just functional abstraction and boolean conjunction: operations required for the semantics of coordination anyway. The rule (79) enables *(the man) John met* and *(the man) that John met* to be analysed as shown in (80) and (81) respectively. The normal relative clause semantics is assigned.

```
(80)    John met
        --------
         S/NP
        --------r_0
         N\N

(81)    that John met
        ---- --------
        SP/S   S/NP
        ----------->B
          SP/NP
          -----r_0
           N\N
```

Finally, note that within this general scheme it still seems necessary to assign *that* to the subject relative pronoun category in order to obtain subject *that*-relatives like *the man that left*.

1.2.4. Parasitic Extraction

Steedman (1987a) proposes the following two rules for parasitic extraction:

(82) a. *Forward Substitution* (>S)

X/Y/Z: x + Y/Z: y => X/Z: S x y

b. *Backward Substitution* (<S)

Y/Z: y + X\Y/Z: x => X/Z: S x y

(83) $S\,x\,y\,z \equiv x\,z\,(y\,z)$

The rules are called "substitution" in view of their semantics, which is *functional substitution*. Recall that the composition of two functions x and y is $\lambda z[x\ (y\ z)]$ in which there is abstraction of y's argument. In the substitution of x and y, an argument of both functions is abstracted: the substitution of x and y is $\lambda z[x\ z\ (y\ z)]$. Szabolcsi (1983) first proposes such a rule for parasitic extraction; she calls it "connection", referring to work by Kayne.

Consider the following:

(84) the student who$_i$ I warned e_i that Mary envied e_i

Here there is parasitic extraction of the object of the main verb and the object in the verb's complement. The incomplete complement can be analysed as *SP/NP* by type-raising and composition; forward substitution can then apply:

```
(85)        who         I       warned     that Mary envied
         ----------    --    ---------- ---------------
         N\N/(S/NP)    NP    S\NP/SP/NP       SP/NP
                      ------->T ---------------------->S
                      S/(S\NP)          S\NP/NP
                      ------------------------->B
                                  S/NP
         ------------------------->
                     N\N
```

In a case like (86) where there is parasitic extraction of the main verb's object and the object in the adjunct, backward substitution combines *filed*$_{VP/NP}$ and *without reading*$_{VP\backslash VP/NP}$ to form an expression of category VP/NP.

(86) which$_i$ I filed e_i without reading e_i

Parasitic extraction from a subject and its predicate verb phrase also proceeds via backward substitution:

```
(87)        whom      the employers of  promoted
         ----------  ---------------- --------
         N\N/(S/NP)       NP/NP        S\NP/NP
                          -------------------<S
                                  S/NP
         ---------------------------->
                     N\N
```

As noted by Steedman (1987a) cases like (88) are not generated.

(88) *the patient who$_i$ I showed e_i e_i

Parasitic gaps cannot be adjacent. Recall that the right-hand daughters in both substitution rules are forward looking functors. A forward slash signifies a missing element to the right, so that for an expression of category *X/Y*, the *Y* 'gap' is never left-peripheral. Since the gap in the right-hand daughter is not left-peripheral, material in this daughter will always separate two parasitic gaps.

2. *Discussion: Rules and Compound Non-Canonicality*

In this section I discuss how the grammar developed so far and shown in Figure 1 would have to be extended to characterize compound non-canonicality.

Forward Application
X/Y: x + Y: y => X: x y

Backward Application
Y: y + X\Y: x => X: x y

Forward Composition
X/Y: x + Y/Z: y => X/Z: **B** x y

Backward Composition
Y\Z: y + X\Y: x => X\Z: **B** x y

Mixed Backward Composition
Y/Z: y + X\Y: x => X/Z: **B** x y
Z ∈ {N\N, SP, NP, S\NP\(S\NP, N/N}

Forward Substitution
X/Y/NP: x + Y/NP: y => X/NP: **S** x y

Backward Substitution
Y/NP: y + X\Y/NP: x => X/NP: **S** x y

Forward Type-Raising
X: x => Y/(Y\X): **T** x

Backward Type-Raising
X: x => Y\(Y/X): **T** x

Topic Introduction
X: x + S\X: y => S: y x
X ∈ {NP, SP, N/N, N\N, S\NP\(S\NP)}

Non-Subject Pied Piping
X/NP: x + N\N/(S/NP): y => N\N/(S/X): **B** y (**C** **B** x)
X ∈ {PP, NP}

Subject Pied Piping
NP/NP: x + N\N/(S/NP): y => N\N/(S\NP): **B** y (**C** **B** x)

Object That and That-Less Relatives
X/NP: x => N\N: **S** (**B** & x) y
X ∈ {S, SP}

Figure 1: Augmentations to CG

In the existing rules, only one slash is ever inherited by the rules of combination. But consider the following:

(89) a. I posted e_i e_j yesterday [a copy of the newsletter]$_i$ [to every student]$_j$
b. the people [to whom]$_j$ I posted e_i e_j yesterday [copies of the newsletter]$_i$
c. the newsletter which$_j$ I posted e_i e_j yesterday [to every member in the area]$_i$

In these examples *two* arguments must be inherited from the verb when it combines with the adverb, thus:

```
(90)    posted     yesterday
       --------   ---------
       VP/PP/NP     VP\VP
       ----------------
          VP/PP/NP
```

The situation is similar in the following:

(91) a. [Mary sent e_i e_j or John gave e_i e_j] [a full report]$_i$ [to every student]$_j$
b. the students [to whom]$_j$ [Mary sent e_i e_j or John gave e_i e_j] [a full report]$_i$
c. a report which$_i$ [Mary sent e_i e_j or John gave e_i e_j] [to every student]$_j$

In such examples it is again necessary for two arguments to be inherited from the verbs in the conjuncts:

```
(92)   Mary   sent
       ---- --------
       S/VP VP/PP/NP
       -------------
         S/PP/NP
```

Steedman (1987c, Appendix B) proposes generalisations like the following, which are equivalent to the '$' generalisation of Ades and Steedman (1982) in that many slashes may be inherited:

(93) a. *Generalised Forward Composition* ($>B^n$)
$X/Y: x \ + \ Y/Z...: y \ => \ X/Z...: B^n\, x\, y$
b. *Generalised Mixed Backward Composition* ($<B^n{}_x$)
$Y/Z...: y \ + \ X\backslash Y: x \ => \ X/Z...: B^n\, x\, y$

(94) $B^0 \equiv B$
$B^n \equiv B\, B^{n-1}\, B$

Once we start expanding the rule set in this way, the question arises as to what constitutes the class of possible rules. Steedman (1987a) proposes two constraints:

(95) *Principle of Directional Consistency* (PDC)
All syntactic combinatory rules must be consistent with the directionality of the principal function. [Steedman (1987a, p407)]

(96) *Principle of Directional Inheritance* (PDI)
If the category that results from the application of a combinatory rule is a function category, then the slash defining directionality for a given argument in that category will be the same as the one defining directionality for the corresponding argument(s) in the input function(s). [Steedman (1987a, p410)]

The principal function is the one whose result category is the same as that of the mother. The PDC states that if this is forward-seeking then its sister must occur to the right, and if it is backward-seeking, to the left. Thus the following are not possible rules:[10]

(97) a. *X\Y + Y => X
b. *X\Y + Y/Z => X/Z

The PDI serves to help interpret the ellipses in (93). The corresponding arguments on mother and daughter must share directionality, thus while (98a) is included in the schema of (93a), (98b) is excluded.

(98) a. X/Y + Y/Z/W => X/Z/W
b. *X/Y + Y/Z/W => X/Z\W

Note however that while the PDI does not exclude the instance (99) of (93a), it was observed that in general the grammar of English should not allow Forward Mixed Composition, in which a backward slash is inherited from a right-hand daughter. Generalised Forward Composition above has the instance (99) of $>B^2$.

(99) X/Y + Y/Z\W => X/Z\W

This involves inheritance of a backward slash from the right hand daughter. In general we want ellipses to be able to range over forward and backward slashes: we don't have a handle by which to block (99). However it causes overgeneration. For example *X/Z\W* in (99) matches the form of a direct object Backward Type-Raised over a prepositional ditransitive verb, thus the verb is erroneously allowed to 'move left':

---

[10] The binary topicalisation rule given earlier violates this principle, but cf. the comments to the effect that topicalisation is a highly marked and therefore presumably atypical mechanism of grammar.

```
(100)   *  gave      John          a book           to Fred
           --------  ----   ---------------       -------
           VP/PP/NP  NP            NP               PP
                     ---->T ---------------<T
                     S/VP   VP/PP\(VP/PP/NP)
                     ----------------------->B^2
                          S/PP\(VP/PP/NP)
           --------------------------<
                        S/PP
                        ------------------------------>
                                     S
```

So the ellipsis generalisation seems to be too strong in that it overgenerates.

But as well as being too strong, the above generalisation appears to be too weak in that it undergenerates. In addition to compound non-canonicality like that above where there is inheritance of multiple slashes from one daughter, there are also cases requiring (non-parasitic) multiple inheritance of slashes from *both* daughters. Recall the following from Section 2.1.1 of Chapter II:

(101) a paper which$_i$ a woman *e*$_j$ presented *e*$_i$ [who has been studying computational linguistics for six years]$_j$

Here there must be inheritance from both the subject and the predicate verb phrase:

```
(102)   a woman   presented
        -------   ---------
        NP/(N\N)  S\NP/NP
        -----------------
           S/NP/(N\N)
```

In (103) there must be inheritance from both the verb and the adverbial.

(103) a. He [met *e*$_i$ during *e*$_j$ and married *e*$_i$ after *e*$_j$] [the great war]$_j$ [a woman whom I've always thought of as my Aunt]$_i$

b. a woman who$_i$ he [met *e*$_i$ during *e*$_j$ and married *e*$_i$ after *e*$_j$] [the great war]$_j$

c. a war which$_j$ he [met *e*$_i$ during *e*$_j$ and married *e*$_i$ after *e*$_j$] [a woman whom I've always thought of as my Aunt]$_i$

Thus:

(104)
$$\begin{array}{cc} \text{met} & \text{during} \\ \hdashline \text{VP/NP} & \text{VP}\backslash\text{VP/NP} \\ \hdashline \multicolumn{2}{c}{\text{VP/NP/NP}} \end{array}$$

And in (105) to (108) there is inheritance *of* the first complement from the verb, and *from* the second complement.

(105) a. the people whom$_i$ he [persuaded e_i to talk e_j and urged e_i to shout e_j] [about their childhood oppressions]$_j$

b. ?the things [about which]$_i$ he [persuaded e_j to talk e_i and urged e_j to shout e_j] [the vast majority of his anxious patients]$_j$

(106) a. the people who$_i$ he [persuaded e_i to leave e_j and urged e_i to oppose e_j] [the political party which they had always supported]$_j$

b. an institution which$_i$ he [persuaded e_j to leave e_i and urged e_i to oppose e_j] [a large number of formerly active supporters]$_j$

(107) a. a scholar who$_i$ [I know e_i to have argued e_j and suspect e_i to believe e_j] [that binding theory has explanatory power]$_j$

b. ?I [know e_i to have argued e_j and suspect e_i to believe e_j] [that binding theory has explanatory power]$_j$ [several of the workers in that research group]$_i$

(108) a. the tokens which$_i$ he [handed e_i to e_j and took e_i from e_j] [each acolyte]$_j$

b. the acolytes whom$_i$ he [handed e_j to e_i and took e_j from e_i] [small tokens of remembrance]$_j$

These require something like (109):

(109)
$$\begin{array}{cc} \text{VP/CMP}_2\text{/CMP}_1 & \text{CMP}_2\text{/X} \\ \hdashline \multicolumn{2}{c}{\text{VP/CMP}_1\text{/X}} \end{array}$$

Such cases indicate a further generalisation along the lines of (110) where there can be (possibly non-parasitic) inheritance from both daughters.

(110) a. Y... + X\Y... => X...

b. X/Y... + Y... => X...

Rules like this embody a general feature percolation convention governed by the PDI and

PDC; but note that the convention must be constrained in English to prevent inheritance of backward slashes from the right-hand daughter, and also extended to allow for parasitic merging of gaps, as is required by (111) and (112) from Section 2.1.2 of Chapter II.

(111) ?a paper which$_i$ he showed e_j e_i before submitting e_i [a good number of his colleagues]$_j$

(112) a picture which$_i$ he showed e_j e_i without forewarning e_j [the unsuspecting members of the jury]$_j$

These need something like (113) and (114) respectively:

(113)
$$\frac{VP/NP_i/NP_j \quad VP\backslash VP/NP_i}{VP/NP_i/NP_j}$$

(114)
$$\frac{VP/NP_i/NP_j \quad VP\backslash VP/NP_j}{VP/NP_i/NP_j}$$

Further apparatus is needed for (115).

(115) a woman [about whom]$_i$ an argument e_i e_j started [which went on all night]$_j$

Since two modifiers of *argument* are extracted, it seems that it must somehow be mapped from $N/(N\backslash N)$ to $N/(N\backslash N)/(N\backslash N)$. In (116), (117), and (118) there are extractions from the right, left, and (parasitically) both elements in the conjuncts during left node raising (cf. Section 2.2.2 of Chapter II):

(116) a topic [about which]$_i$ I lent$_j$ [e_j John a book e_i and e_j Mary a paper e_i]

(117) I lent$_i$ [e_i a book e_j to John and e_i a paper e_j to Mary] [about subjacency]$_j$

(118) a town which$_i$ I bought$_j$ [e_j a ticket to e_i not wanting to visit e_i and e_j a ticket from e_i not wanting to leave e_i]

Examples (116) and (117) appear to require *a book*$_{NP/(N\backslash N)}$ to be mapped to a higher type ultimately forming *VP*, in order for the pattern for left node raising to be followed; the situation is complicated in (118) by the need to associate the parasitic gaps.

Morrill (1987a) shows how it is possible to go about capturing such data by employing extra unary rules to achieve inheritance from both daughters. But overall the account of Section 1 does not generalise straightforwardly to accommodate compound non-canonicality. By contrast, it was shown in Chapter II that PSG augmented with metarules generalises naturally to compound non-canonicality by recursion of metarules. On that approach the metarules began applying to basic phrase structure rules; now interestingly when they are applied to the basic application rules of CG, those same metarules derive the rules employed in Section 1 of this chapter (of course we are now interpreting the PSG slash as the CG operator). For example application of Right Abstraction to Forward Application yields Forward Composition:

(119) X/Y + Y => X ==> X/Y + Y/Z => X/Z

And Application of Middle Abstraction to Backward Application yields Mixed Backward Composition:

(120) Y + X\Y => X ==> Y/Z + X\Y => X/Z

Furthermore, recursive application of metarules achieves the required generalisations; for example applying Right Abstraction to the outputs of (119) and (120) provides the following:

(121) a. X/Y + Y/Z => X/Z ==> X/Y + Y/Z/W => X/Z/W
b. Y/Z + X\Y => X/Z ==> Y/Z + X\Y/W => X/Z/W

These observations suggest a synthesis of the CG and PSG approaches, one augmenting the CG base grammar with the PSG metarules. It is this augmentation of categorial grammar with metarules that is considered in the next chapter.

Chapter IV

Categorial Grammar Extended with Metarules

In Chapter II I showed how PSG extended with metarules can characterise both simple and compound non-canonicality, but it was noted that the latter required a category apparatus like that of CG. This motivated the approach of Chapter III where CG was extended with rules. However it was argued that those augmentation primitives do not successfully capture the generalisations underlying non-canonicality. In this chapter I show how CG can be augmented with the PSG metarules (with slash interpreted as the CG operator) to produce an account of non-canonicality which seems to represent an advance on, and a synthesis of, the earlier accounts.

In presenting the new account it will be convenient to adopt the following notation. Basic (i.e. non-derived) rules will be named by lower case combinators; Forward and Backward Application will be written thus:

(1) a. **f**: $X/Y + Y \Rightarrow X$
f $x\ y \equiv x\ y$
b. **b**: $Y + X\backslash Y \Rightarrow X$
b $y\ x \equiv x\ y$

This notation is intended to make explicit the fact that a rule is a combinatory logic combinator, and that the daughter and mother combination schema after the colon is the type of the combinator. Metarules will be named in upper case:

(2) **R**: $X + Y \Rightarrow Z \Longrightarrow X + Y/W \Rightarrow Z/W$
R $g\ x\ y\ w \equiv g\ x\ (y\ w)$

Metarules such as this are formulated in a categorial context in Geach (1972, p485) and Moortgat (1987, p18).[1] Derived rules will be named by complex combinators, for example the result of applying Right Abstraction **R** to Forward Application **f** is forward composition **Rf**:

(3) **Rf**: $X/Y + Y/Z \Rightarrow X/Z$
R f $x\ y\ z = x\ (y\ z)$

---

[1] Bob Carpenter commended such rules to me in 1985.

Forward Application
f: X/Y + Y => X
f x y ≡ x y

Backward Application
b: Y + X\Y => X
b y x ≡ x y

Right Abstraction
R: X + Y => Z ==>
X + Y/W => Z/W
R g x y w ≡ g x (y w)

Left Abstraction
L: X + Y => Z ==>
X\W + Y => Z\W
L g x y w ≡ g (x w) y

Middle Abstraction
M: X + Y => Z ==>
X/W + Y => Z/W
M g x y w ≡ g (x w) y
Z ∈ {N\N, SP, NP, S\NP\(S\NP), N/N}

Parasitic Abstraction
P: X + Y => Z ==>
X/NP + Y/NP => Z/NP
P g x y w ≡ g (x w) (y w)

Forward Type-Raising
r_f: X => Y/(Y\X)
r_f x y ≡ y x

Backward Type-Raising
r_b: X => Y\(Y/X)
r_b x y ≡ y x

Forward Abstraction
F: X => Y ==> X/Z => Y/Z
F g x z ≡ g (x z)

Backward Abstraction
B: X => Y ==> X\Z => Y\Z
B g x y ≡ g (x z)

Figure 1: CG Extended with Metarules

So far as unary rules are concerned, one interesting possibility is to employ the PSG gap introduction metarules (4).[2]

(4) a. X + Y => Z ==> X => Z/Y
b. X + Y => Z ==> Y => Z\Y

For example, applying (4b) to Forward Application yields backward type-raising:

---

[2] The function of such metarules would clearly be different than that in PSG since a gap introduction device is not needed with arguments already on slashes.

(5) $X/Y + Y \Rightarrow X \Longrightarrow Y \Rightarrow X\backslash(X/Y)$

I am not aware of particularly compelling arguments against this approach, however there is the following point which leads me to adopt a different position. It is widely suspected that type-raising should be a lexical process. For example one problem area in grammar generally is the apparent continuum between complements and adjuncts. Type-raising of a head over an adjunct constitutes a conversion of the adjunct (a functor over the head) into a complement (an argument of the head); in this way type-raising seems to offer a realisation in the grammar of complement-adjunct flexibility. Now consider the type-raising of a transitive verb VP/NP over an adverbial. This can be done by applying the metarule (4a) to mixed backward composition as follows:

(6) $VP/NP + VP\backslash VP \Rightarrow VP/NP \Longrightarrow VP/NP \Rightarrow VP/NP/(VP\backslash VP)$

However the output's mother seeks the adjunct *before* (left of) the complement. This is a strange situation. Intuitively, what is required is to first 'hold off' argument categories, then type-raise the result category, and then restore the argument categories as they were, so that VP/NP becomes $VP/(VP\backslash VP)/NP$. Therefore I adopt the type-raising primitives (7a) along with the unary abstraction metarules (8b).

(7) a. r_f: $X \Rightarrow Y/(Y\backslash X)$
b. r_b: $X \Rightarrow Y\backslash(Y/X)$

(8) a. F: $X \Rightarrow Y \Longrightarrow X/Z \Rightarrow Y/Z$
b. B: $X \Rightarrow Y \Longrightarrow X\backslash Z \Rightarrow Y\backslash Z$

Such unary metarules appear in Zielonka (1981, p220). The 'holding off' unary abstraction can be motivated by examples like (9) mentioned near the end of Chapter III.

(9) a woman [about whom]$_i$ an argument e_i e_j started [which went on all night]$_j$

We need to map $N/(N\backslash N)$ to $N/(N\backslash N)/(N\backslash N)$; this can be done as follows:[3]

(10) a. Fr_f: $X/Z \Rightarrow Y/(Y\backslash X)/Z$
b. $N/(N\backslash N) \Rightarrow N/(N\backslash N)/(N\backslash N)$

Section 2 shows in full how Unary Abstraction Metarules handle this and the other cases that were pointed out to be problematic in Section 2 of Chapter III. An attractive feature is that once such metarules are employed, it seems that the data only requires basic or derived

---

[3] On occasion I will refer to such derived rules as type-raising, though more strictly speaking they are generalisations of type-raising.

unary rules to apply at the point of lexical insertion, relating lexical and preterminal categories rather than applying to their own output, or applying freely in the syntax. This is made possible because the abstraction enables access to the result categories projected from the lexical categories, before arguments are supplied. If this property can be maintained, then type-raising has neither completely lexical nor completely syntactic status, but rather is a rule of lexical insertion. The grammar that will be used in the following sections is shown in Figure 1; I assume the topicalisation, pied piping, and *that*-(less) relative rules of Chapter III. Simple non-canonicality is discussed in Section 1, compound non-canonicality is discussed in Section 2.

1. Simple Non-Canonicality

Sections 1.1 and 1.2 discuss extraction, and coordination of 'non-constituents', respectively.

1.1. Extraction

Left extraction will proceed largely as in Chapter III; but subjects need no longer be type-raised -- they can be combined directly by the rule derived by applying Right Abstraction to Backward Application:

(11) **Rb**: Y + X\Y/Z => X/Z

For example:

```
(12)    who          John  thinks        that  Mary  likes
        ----------   ----  -------       ----  ----  -------
        N\N/(S/NP)   NP    S\NP/SP       SP/S  NP    S\NP/NP
                     -----------Rb             -----------Rb
                           S/SP                      S/NP
                           ----------------Rf
                                 S/S
                                 --------------------Rf
                                             S/NP
        ----------------------------------f
                           N\N
```

The meaning of *Mary likes* derived by **Rb** is **R b** Mary′ likes′ so that *Mary likes Fred*, derived by applying this to *Fred*, has the meaning (13a) which evaluates to (13b); thus the canonical meaning is obtained.

(13) a. **f** (**R** **b** Mary′ likes′) Fred′

b. likes′ Fred′ Mary′

Extraction from clause-non-final position requires Middle Abstraction:

(14)
```
     who        I    met        yesterday
 ----------  --  -------    -----------
 N\N/(S/NP)  NP  S\NP/NP    S\NP\(S\NP)
                 ---------------------Mb
                        S\NP/NP
             ----------------Rb
                     S/NP
 ---------------------f
            N\N
```

As with the Chapter III grammar, fixed subject constraint violations such as (15) are not possible.

(15) *the man who$_i$ I think that e_i left

To see why this is so, note that the category of a clause from which an element of category *X* is left extracted is always *S/X*, which has a forwards leaning slash. Note also that the metarules all preserve slash-directionality. But the subject wanted by the subordinate verb in (15) is sought *backwards*. No operation is capable of switching the directionality (cf. the Principle of Directional Inheritance), so there is no analysis that can relate the backward-sought subject with a forward-sought gap category, and there cannot be fixed subject violations. Similarly, a left branch condition violation like (16) cannot be generated.[4]

(16) the man who$_i$ I met e_i's brother

However as before, constraints like the complex noun phrase constraint, subject condition, noun phrase constraint, and A-over-A constraint are not respected. Nor are adjuncts islands. Unlike in the grammar of Chapter III, it is not necessary to type-raise over adjuncts in order to extract out of them:

(17)
```
 went to Paris   without finishing
 -------------   ----------------
      VP            VP\VP/NP
      --------------------Rb
              VP/NP
```

---

[4] The grammars of Dutch, Hungarian, Norwegian, Polish, Spanish, etc. which do allow left branch extraction sites provide obvious areas for further research.

In Chapter III, heavy shift of a first complement past a second required backward type-raising of the latter, followed by mixed backward composition. In the current grammar type-raising is not required for these cases:

```
(18)        put    on the table  a large red box
          -------- ------------ --------------
          VP/PP/NP       PP              NP
          ---------------Mf
                VP/NP
                ------------------------f
                            VP
```

'Particle shift' will follow the same pattern.

Right extraposition of a relative clause, and extraction of adjuncts in general, requires forward type-raising over the adjuncts:

```
(19)       a    man   arrived who swims
          ----  ---   ------- ---------
          NP/N   N     S\NP      N\N
                -----rf
               N/(N\N)
          -----------Rf
            NP/(N\N)
             ---------------Mb
                S/(N\N)
                --------------------f
                          S
```

Parasitic extraction will proceed as before; Steedman's rules of forward and backward substitution are derived by application of Parasitic Abstraction (20) to Forward and Backward Application to yield (21).

(20) P: X + Y => Z ==> X/W + Y/W => Z/W

(21) a. Pf: X/Y/Z + Y/Z => X/Z

b. Pb: Y/Z + X\Y/Z => X/Z

Also as in Chapter III, extraction like (22) is not possible.

(22) *the man who$_i$ I showed e_i e_i

Recall that this is so because the 'Z' gap category on the right-hand daughter of the output of Parasitic Abstraction is forward-sought, and the metarules preserve directionality. This means that the 'Z' must have been forward-sought on the preterminal category it is ultimately inherited from, so that the gap site -- the canonical location -- is right of that terminal. So the gap is never left-peripheral and there will always be material between parasitic gaps.

1.2. Coordination of 'Non-Constituents'

Right node raising will proceed as in Chapter III except, as in left extraction, it is not necessary to type-raise in order to extract past clause or adjunct boundaries.

However type-raising *is* still needed to invert the functor-argument relation so that functor movement can proceed on the same pattern as argument movement. Thus it is required for right node raising of adjuncts, e.g. *[a man and a woman] who like Beethoven*, and also for left node raising. Complement-Adjunct left node raising such as *I met [John on Monday and Mary on Tuesday]* is achieved by applying Left Abstraction (23) to Backward Application to derive backward composition (24):

(23) $\mathrm{L}: X + Y \Rightarrow Z \Longrightarrow X\backslash W + Y \Rightarrow Z\backslash W$

(24) $\mathrm{Lb}: Y\backslash Z + X\backslash Y \Rightarrow X\backslash Z$

```
(25)      John     on Monday
          ----     ---------
           NP        VP\VP
        ----------r_b
        VP\(VP/NP)
        ------------------Lb
           VP\(VP/NP)
```

Recall that earlier left node raising out of conjuncts consisting of two complements required backward type-raising of both complements. This is not required now: it is only necessary to type-raise the first complement. Note also how the type-raising is pushed down to the point of lexical insertion by applying the derived rule (27) to the determiner:

(26) I gave [a book to John and a record to Mary]

(27) $\mathrm{Fr_b}: X/Y \Rightarrow Z\backslash(Z/X)/Y$

```
(28)              a            book   to John
                 ----          ----   -------
                 NP/N           N       PP
       -----------------Fr_b
       VP/PP\(VP/PP/NP)/N
       --------------------f
         VP/PP\(VP/PP/NP)
         ---------------------------Lf
                VP\(VP/PP/NP)
```

Across-the-board extraction in general is as before, and the impossibility of unbalanced gaps or extraction of a whole conjunct still stand.

2. *Compound Non-Canonicality*

Double heavy shift past an adverbial arises through recursion of Middle Abstraction on itself:

(29) M(Mb): Y/Z/W + X\Y => X/Z/W

```
(30)   posted  yesterday  a copy of the newsletter  to every student
       ------- ---------  ------------------------  ----------------
       VP/PP/NP  VP\VP               NP                     PP
       ---------------M(Mb)
          VP/PP/NP
          --------------------------f
                    VP/PP
                    ---------------------------------------f
                                       VP
```

Similarly, right node raising of two complements arises through recursion of Right Abstraction:

(31) Mary [has given e_i e_j or will send e_i e_j] [a full report]$_i$ [to every student]$_j$

(32) R(Rf): X/Y + Y/Z/W => X/Z/W

```
(33)   has    given
       -----  --------
       VP/VP  VP/PP/NP
       ---------------R(Rf)
          VP/PP/NP
```

(34) [Mary sent e_i e_j or John gave e_i e_j] [a full report]$_i$ [to every student]$_j$

(35) **R(Rb): Y + X\Y/Z/W => X/Z/W**

(36)

```
Mary   sent
----  ----------
 NP   S\NP/PP/NP
 --------------R(Rb)
     S/PP/NP
```

If the direct object is across-the-board left extracted as in (37), **Mf** will combine the *S/PP/NP* coordinate structure with the indirect object:

(37) a report which$_i$ [Mary sent e_i e_j or John gave e_i e_j] [to every student]$_j$

(38)

```
Mary sent or John gave   to every student
----------------------   ----------------
          S/PP/NP                PP
          -------------------------Mf
                     S/NP
```

If one of two elements extracted is an adjunct, generalised type-raising is required:

(39) [I saw e_i e_j but Mary missed e_i e_j] Dallas$_i$ yesterday$_j$

(40)

```
I        saw
--       -----
NP       S\NP/NP
   ------------------Fr_f
   S\NP/(S\NP\(S\NP))/NP
------------------------R(Rb)
   S/(S\NP\(S\NP))/NP
```

And as remarked at the beginning of the chapter, the derived unary rule **Fr$_f$** is also required in the case of (41) where both a complement and an adjunct are extracted:

(41) a woman [about whom]$_i$ an argument e_i e_j started [which went on all night]$_j$

(42) about whom an argument started which ...

N\N/NP N\N/(S/NP) NP/N N/(N\N) S\NP N\N

------------------ p_m

N\N/(S/(N\N))

--------------- Fr_f

N/(N\N)/(N\N)

---------------------- R(Rf)

NP/(N\N)/(N\N)

------------------ M(Mb)

S/(N\N)/(N\N)

---------------------------- Mf

S/(N\N)

-- f

N\N

The meaning assigned by the analysis is expressed by the combinatory logic term (43a) which is equivalent to the λ-term (43b).

(43) a. f (p_m about′ whom′) (M f (M (M b) (R (R f) an′ (F r_f argument′)) started′) which-...′)

b. whom′ (λy[started′ (an′ (which...′ (argument′ (about′ y))))])

Cases of simultaneous independent extraction and parasitic extraction are characterised in a manner basically equivalent to that in the phrase structure grammar:

(44) a paper which$_i$ he showed e_j e_i before submitting e_i [a good number of his colleagues]$_j$

(45) M(Pb): Y/Z/W + X\Y/Z => X/Z/W

(46) showed before submitting

VP/NP$_i$/NP$_j$ VP\VP/NP$_i$

------------------------ M(Pb)

VP/NP$_i$/NP$_j$

(47) a picture which$_i$ he showed e_j e_i without forewarning e_j [the unsuspecting members of the jury]$_j$

(48) P(Mb): Y/Z/W + X\Y/W => X/Z/W

(49) showed without forewarning

VP/NP$_i$/NP$_j$ VP\VP/NP$_j$

------------------------- P(Mb)

VP/NP$_i$/NP$_j$

Extraction both from a verb phrase and an adverbial modifier, and other cases requiring inheritance from two daughters, also proceeds much as it did in the phrase structure grammar:

(50) He [met *e*$_i$ during *e*$_j$ and married *e*$_i$ after *e*$_j$] [the great war]$_j$ [a woman whom I've always thought of as my Aunt]$_i$

(51)

```
 met     during
------  ---------
VP/NP_i  VP\VP/NP_j
-----------------R(Mb)
   VP/NP_i/NP_j
```

(52) the tokens which$_i$ he [handed *e*$_i$ to *e*$_j$ and took *e*$_i$ from *e*$_j$] [each acolyte]$_j$

(53)

```
 handed       to
---------   ------
VP/PP/NP_i  PP/NP_j
-----------------R(Mf)
   VP/NP_i/NP_j
```

Consider next *tough* movement like that in (54).

(54) History$_i$ is hard to understand *e*$_i$

As in the phrase structure grammar it is assumed that predication by the complement of the subject is achieved through the meaning of the copula; the analysis proceeds thus:

(55)

```
     is            hard              to       understand
----------  -------------  -----------  ----------
S\NP/(N/N)  N/N/(S\NP/NP)  S\NP/(S\NP)  S\NP/NP
                           --------------------Rf
                                  S\NP/NP
            ------------------------------f
                         N/N
----------------------------f
          S\NP
```

The potential unboundedness follows in the standard way. According to the nested dependency constraint, filler-gap dependencies must be nested in the case of multiple extractions, but as was pointed out in Chapter II this is not true, even for the *tough* movement constructions with reference to which the constraint was formulated; although the constraint correctly describes (56), the crossed (57b) is at least as acceptable as the nested (57a).

(56) a. a violin which$_j$ [the sonatas]$_i$ are hard to play e_i on e_j
b. *the sonatas which$_i$ [the violin]$_j$ is hard to play e_i on e_j

(57) a. ?some evidence which$_j$ [the witnesses]$_i$ are hard to show e_i e_j
b. some witnesses whom$_i$ [the evidence]$_j$ is hard to show e_i e_j

In fact the current grammar generates both of (56) and also (57b); but (57a) is deemed ungrammatical. This does not shed much light on the phenomena involved:

```
(58)      which        the sonatas are hard    to play      on
       -----------  -------------------   ---------  ------
       N\N/(S/NP_j)        S/(VP/NP_i)        VP/PP/NP_i  PP/NP_j
                                           ---------------R(Mf)
                                               VP/NP_i/NP_j
                           ----------------------------Rf
                                       S/NP_j
       ---------------------------------f
                       N\N
```

```
(59)      which        the violin is hard    to play      on
       -----------  -----------------   ---------  ------
       N\N/(S/NP_i)       S/(VP/NP_j)       VP/PP/NP_i  PP/NP_j
                                         ---------------M(Rf)
                                             VP/NP_j/NP_i
                          ---------------------------Rf
                                      S/NP_i
       --------------------------------f
                       N\N
```

```
(60)      which        the witnesses are hard    to show
       -----------  ----------------------   ---------
       N\N/(S/NP_j)          S/(VP/NP_i)           VP/NP_j/NP_i
                             --------------------------*(i≠j)
       ------------------------*
```

```
(61)      whom         the evidence is hard    to show
       -----------  -------------------   ---------
       N\N/(S/NP_i)        S/(VP/NP_j)          VP/NP_j/NP_i
                           ------------------------Rf
                                     S/NP_i
       --------------------------------f
                       N\N
```

Example (62) exemplifies left node raising with across-the-board extraction from the second element.

(62) a topic [about which]$_i$ I lent$_j$ [e_j John a book e_i and e_j Mary a paper e_i]

This requires the first element to be backward type-raised over the verb, and the second to be forward type-raised (lexically or otherwise) over the adnominal:

```
(63)    John                     a book
        ----                  -------------
        NP                    NP/N  N/(N\N)
        ----------------r_b   -------------Rf
        VP/NP\(VP/NP/NP)        NP/(N\N)
        -------------------------------L(Rf)
             VP/(N/N)\(VP/NP/NP)
```

Example (64) exhibits extraction from the left element with left node raising it has the analysis illustrated in (65).

(64) a topic [about which]$_i$ I lent$_j$ [e_j a paper e_i to John and e_j a book e_i to Mary]

```
(65)          a               book     to John
            ------           -------   -------
             NP/N            N/(N\N)     PP
        -----------------Fr_b
        VP/PP\(VP/PP/NP)/N
        ----------------------------Rf
            VP/PP\(VP/PP/NP)/(N\N)
            ------------------------------M(Lf)
                  VP\(VP/PP/NP)/(N\N)
```

Similarly, the parasitic case (66) has the analysis (67).

(66) a town which$_i$ I bought$_j$ [e_j a ticket to e_i not wanting to visit e_i and e_j a ticket from e_i not wanting to leave e_i]

```
(67)        a      ticket to  not wanting to visit
           ----    ---------  --------------------
           NP/N      N/NP          VP\VP/NP
        -----------Fr_b
        VP\(VP/NP)/N
        ------------------Rf
          VP\(VP/NP)/NP
          --------------------------------P(Lb)
                  VP\(VP/NP)/NP
```

The rules also enable first complement-adjunct left node raising of a verb seeking two complements:

(68) the people [to whom]$_i$ we sent$_j$ [e_j the report e_i on Monday and e_j the newsletter e_i on Tuesday]

```
(69)      the report      on Monday
         ---------------- ---------
         VP/PP\(VP/PP/NP)   VP\VP
         -------------------------L(Mb)
           VP/PP\(VP/PP/NP)
```

However the grammar characterises as ungrammatical commutation of verb dependents with left node raising, thus:

(70) ?He believes$_i$ [e_i e_j to be misguided [the GB-ers]$_j$ and e_i e_k to be mistaken [the GPSG-ers]$_k$]

```
(71)     believes  [to be misguided  the GB-ers ...]
                    ---------------  ----------
         VP/VP/NP  VP/NP\(VP/NP/VP)     NP
                   ----------------------Lf
                       VP\(VP/NP/VP)
         --------------------------*
```

This concludes the examination of the grammar of extraction and coordination from the point of view of phrase structure grammar and categorial grammar, and the exemplification of the grammar that has emerged. In the next chapter I turn to consider issues relating to the notion of 'universal grammar', but first some remarks are due in order to orient the current proposals with respect to some of my earlier work.

The accounts of non-canonicality presented in Morrill (1987a,b) use metarules like the following:

(72) $[X\ [Y\ Z]]_V$ ==> $[X\ Y]_{V/Z}$

Note that the output of (72) is binary. Since the application rules from which rule derivation starts are also binary, it follows that all rules are binary. However a number of considerations have led me to shift to the current proposals. First note that metarules like (72) preserve the binary character of the grammar: there are no basic unary rules in pure CG, and no unary rules are derived. The same rules achieve composition-like effects, and type-raising-like effects, but because all rules are binary this leads to the rather implausible prediction that higher types are only available to non-basic expressions, e.g. a noun phrase consisting of a single word belongs to just its lexical categories, but one consisting of several words is also of higher-type categories such as S/(S\NP). Secondly, the 'universal grammar' suggested by the 'double-barrelled' metarules contains 12 possible metarules when we exclude parasitic phenomena, but to include parasitic constructions we need slightly different kinds of rules (ones with four leaves as opposed to three), and the rule-space expands to 72. This does not seem to constitute a graceful accommodation of parasitic phenomena. A further disadvantage of the earlier metarule proposal is that it used the following metarule for right extraction:

(73) $[[X\ Y]\ Z]_{V} \Rightarrow [X\ Z]_{V/Y}$

This was adopted because the alternative (74a) would allow violation of the fixed subject constraint via (74b).

(74) a. $[X\ [Y\ Z]]_{V} \Rightarrow [X\ Z]_{V/Y}$
b. $[SP/S\ [NP\ S\backslash NP]]_{SP} \Rightarrow [SP/S\ S\backslash NP]_{SP/NP}$

However if right node raising such as that in (75) were to be allowed, *SP/S+NP+S\NP* would have the analysis (76) to which (73) *can* apply as in (77), to nevertheless violate the fixed subject constraint.

(75) ?I think [that John e_i and that Mary e_i] [went to London]$_i$

(76) $[[SP/S\ NP]_{SP/(S\backslash NP)}\ S\backslash NP]_{SP}$

(77) $[[SP/S\ NP]_{SP/(S\backslash NP)}\ S\backslash NP]_{SP} \Rightarrow [SP/S\ S\backslash NP]_{SP/NP}$

For these reasons I have shifted from the earlier model, dubbed *meta-categorial grammar* (MCG) to the one presented in this chapter which (if a name is required) might be called MCG-II.

Chapter V

Universal Grammar

So far I have developed an account of English extraction and coordination synthesising accounts stemming from phrase structure grammar and categorial grammar traditions. This chapter contains various remarks on syntax and semantics in relation to the picture of universal grammar that emerges from this line of inquiry.

1. Syntax

The grammar of English has four binary metarules: one inheriting a forward slash from the right-hand daughter, one inheriting a forward slash from the left-hand daughter, one inheriting a forward slash from both daughters, and one inheriting a backward slash from the left-hand daughter. This family seems to be completed by metarules inheriting a backward slash from the right-hand daughter, and one inheriting a backward slash from both daughters, suggesting that universal grammar should contain the rules shown in Figure 1.[1] In Section 1.1 this model of universal grammar is discussed in relation to Steedman's Principle of Directional Consistency, and in Section 1.2 category structure is discussed. Section 1.3. discusses free word order and Section 1.4 contains some remarks on weak generative capacity.

1.1. Metarules and Directional Consistency

It was shown in Chapter IV how the binary metarules derive the composition and substitution primitives used in Steedman's generalisation of categorial grammar. It has also been remarked that a set of binary metarules can be regarded as defining a percolation convention on slashes of the kind employed in some versions of phrase structure grammar (Gazdar et al. 1985; Pollard 1985).[2] Then selecting a subset of metarules corresponds to fixing the parameters of a percolation convention, each metarule determining a relation that can exist between mother and daughter slash categories.

---

[1] This excludes rules required for topicalisation, pied piping, etc. An interesting possibility which takes us a little far from the theme of this work is that the "Backward Parasitic" rule may be involved in control phenomena.

[2] Such conventions may govern more than just percolation of slash.

Forward Application
f: X/Y + Y => X
f x y ≡ x y

Right Abstraction
R: X + Y => Z ==>
X + Y/W => Z/W
R g x y w ≡ g x (y w)

Forward Middle Abstraction
M_f: X + Y => Z ==>
X/W + Y => Z/W
M_f g x y w ≡ g (x w) y

Forward Parasitic Abstraction
P_f: X + Y => Z ==>
X/W + Y/W => Z/W
P_f g x y w ≡ g (x w) (y w)

Forward Type-Raising
r_f: X => Y/(Y\X)
r_f x y ≡ y x

Forward Abstraction
F: X => Y ==> X/Z => Y/Z
F g x z ≡ g (x z)

Backward Application
b: Y + X\Y => X
b y x ≡ x y

Left Abstraction
L: X + Y => Z ==>
X\W + Y => Z\W
L g x y w ≡ g (x w) y

Backward Middle Abstraction
M_b: X + Y => Z ==>
X + Y\W => Z\W
M_b g x y w ≡ g x (y w)

Backward Parasitic Abstraction
P_b: X + Y => Z ==>
X\W + Y\W => Z\W
P_b g x y w ≡ g (x w) (y w)

Backward Type-Raising
r_b: X => Y\(Y/X)
r_b x y => y x

Backward Abstraction
B: X => Y ==> X\Z => Y\Z
B g x z ≡ g (x z)

Figure 1: Rules in Universal Grammar

As was noted in Chapter III, Steedman has posited general principles governing the rules in universal grammar. Thus the Principle of Directional Consistency (PDC) states:

(1) All syntactic combinatory rules must be consistent with the directionality of the principal function. [Steedman (1987a, p407)]

The *principal function* is the one whose result category is the same as that of the mother. The principle says that if this is forward-seeking then its sister must occur to the right, and if it is backward-seeking, to the left. Now the metarule account provides an explanation for the PDC: the application rules trivially respect the principle (indeed they *define* the directionality of the principle functor), and because other rules are derived by instantiating slashes on these primitives, it follows that all derived rules will respect the PDC.

1.2. Category Structure

Steedman's Principle of Directional Inheritance (PDI) states:

(2) If the category that results from the application of a combinatory rule is a function category, then the slash defining directionality for a given argument in that category will be the same as the one defining directionality for the corresponding argument(s) in the input function(s). [Steedman (1987a, p410)]

Because the slashes instantiated by all the metarules share directionality, the current basis for universal grammar also endorses the PDI. The implication of the slash-harmony is that the directionality of an argument plus the argument itself forms a unit of category structure, so that categories have the binary structure in Figure 2b rather than the tripartite one in Figure 2a. I suspect that the origin of the PDI lies in the directional type system, but I am currently unaware of how it is to be rationalised in the way in which the metarule account rationalises the PDC.

In the CG category system the category resulting from an application is a unit of category structure; for example when *S\NP/NP* applies to form *S\NP*, the result is obtained by simply accessing that subpart of the category structure left of the principle slash. Similarly, to determine whether forward composition (3) could apply, it is necessary to test whether the result *Y* of application of the right hand daughter matches with the argument of the left hand daughter.

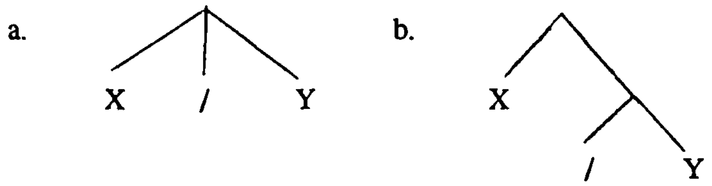


Figure 2: Category Structures

(3) X/Y + Y/Z => X/Z

In fact all rule applications require reference to the results of application, and because the result of application is a unit of category structure in the existing system, this is minimally expensive computationally.

However in category systems which do not employ operators, but features, this state of affairs does not hold. For example, in a featural category like (4) the result is not a unit of category structure but is the unification of *[SLASH ...]*, *[SUBCAT <...>]*, and all the other top-level attribute-value pairs.

(4) X[SUBCAT <C...>, SLASH ..., ...]

So the result cannot be referenced so easily. The same situation holds with less radical augmentation of an operator-based system with features on complex categories. Thus supposing there is a transitive verb category like (5) where *[TNS PRES]* is intended to be a feature on the category as a whole, which is subject to some kind of 'functor feature percolation convention', an analogue of the head feature percolation convention.

(5) S\NP/NP
[TNS PRES]

Then the result of application is again not a unit, but the result of associating *S\NP* and *[TNS PRES]*. The implication is that it is in the interest of processability for augmentation to be limited to constructive operators over categories (which preserve the results of application as units of category structure), and to features on basic categories

There is one more observation I want to make in relation to featural slash category structures, with categories of subcategorized complements encoded in head categories rather than in phrase structure rules. Extraction is effected by shifting elements from the *SUBCAT* stack to the *SLASH* stack. In such a theory it is difficult to obtain extraction of heads since they are not on a SUBCAT stack. Within the categorial grammar, head/functor movement was done by type-raising: turning functors into arguments after which extraction can proceed on the argument pattern. But type-raising, like all other rules, needs to make reference to results of application: in order to type-raise it is necessary to access what a functor category would have formed when it applied to its argument, because the type-raised argument must form the same result when it applies to the functor. This is complicated in featural categories because the results of application are not units of category structure; I believe this is the reason why it is difficult to formulate a simple mechanism for functor movement where categories are feature-based rather than operator-based.

1.3. Free Word Order

The current metarules can characterise word-order variations in languages with ordering more free than that in English. For example Karttunen (1986) reports that in Finnish, the elements of a subordinate clause can be distributed amongst those of a superordinate one; examples below are taken from his paper.[3]

Finnish has a rich inflectional system and exhibits fairly free ordering, though acceptability is highly dependent on intonation and the discourse function of elements. Both SV and VS orders are possible in a simple intransitive sentence. I want to emphasise here the well known possibility that such a states of affairs can be characterised by a slash '|' which is a variable ranging over '/' and '\':

(6) Liisa nukkui
Lisa-nom slept-3sg
'Lisa slept'

(7)

```
Liisa   nukkui
-----  --------
NP_s   Sfin|NP_s
--------------b
     Sfin
```

(8) Nukkui Liisa
slept-3sg Lisa-nom
'Lisa did sleep'

(9)

```
Nukkui     Liisa
--------   -----
Sfin|NP_s  NP_s
--------------f
     Sfin
```

A simple transitive sentence may have all six logically possible orderings. Assigning a transitive verb like *rakasti* a category $Sfin|NP_s|NP_o$ means that the SVO, VOS, SOV and OVS orders, where the object is adjacent to the verb, will be obtained immediately. The other orders can be obtained by order-changing metarules (cf. Stucky 1983):

(10) M_f: X + Y => Z ==> X/W + Y => Z/W

---

[3] I am grateful to Kristiina Jokinen for discussion relating to this section; all errors are of course my own responsibility.

(11) M_b: X + Y => Z ==> X + Y\W => Z\W

(12)
```
  rakasti        Jussi       Liisaa
------------     -----       ------
Sfin|NP_s|NP_o   NP_s        NP_o
------------------------M_f f
     Sfin/NP_o
     --------------------------------f
                  Sfin
```

(13)
```
Liisaa  Jussi    rakasti
------  -----  ------------
 NP_o    NP_s  Sfin|NP_s|NP_o
         ----------------------M_b b
               Sfin\NP_o
 ---------------------------b
           Sfin
```

A negative auxiliary precedes its temporal auxiliary. This can be captured by making their slashes directional and excluding them from the categories which can be 'moved' by the Mixed Abstraction rules:

(14) a. Liisa ei ole nukkunut
Lisa-nom not have-neg slept-pcp
'Lisa hasn't slept'

b. Ei liisa ole nukkunut

c. Ei ole Liisa nukkunut

d. Ei ole nukkunut Liisa

(15) *Liisa ole ei nukkunut

(16)
```
Liisa           ei                         ole              nukkunut
-----  ---------------------   ---------------------   ---------
 NP_s.  Sfin|NP_s/(Sneg|NP_s)   Sneg|NP_s/(Spcp|NP_s)   Spcp|NP_s
                                ---------------------------------f
                                            Sneg|NP_s
        ----------------------------------------------f
                             Sfin|NP_s
 -------------------------------------b
                 Sfin
```

(17)
```
*Liisa           ole                      ei                nukkunut
------  ---------------------   ---------------------   ---------
 NP_s   Sneg|NP_s/(Spcp|NP_s)   Sfin|NP_s/(Sneg|NP_s)   Spcp|NP_s
        ----------------------------------------------*
                                ---------------------------------*
```

Karttunen notes that in (18) the complement and adjunct of *pelaamaan* appear to be able to occur in any of the six positions in the superordinate sequence *En minä ole aikonut ruveta* so that it is not clear any of the 42 possible variants should be excluded.

(18) En minä ole aikonut ruveta pelaamaan näissä tennistä
Not I have intended-pcp start-infl play-inf3 these-in tennis-ptv
'I have not intended to start to play tennis in these (clothes)'

Assuming that an adverbial like *näissä* has a category $S|NP_s|(S|NP_s)$ allowing it to appear either side of a verb, Backward Type-Raising, Backward Abstraction, and Middle Abstraction enable analysis of examples in which the subordinate elements are distributed amongst *the superordinate ones, for example:*

(19) En minä tennistä näissä ole aikonut ruveta pelaamaan
Not I tennis in-these have intended start play

```
(20) a. näissä          ole           aikonut             ruveta          pelaamaan
        ------    ------------     -------------     --------------     ------------
        VP|VP     VPneg/VPpcp      VPpcp/VPinfl      VPinfl/VPinf3      VPinf3|NP_o
                                                                        -----------Br_b
                                                                  VPinf3\(VP/VP)\NP_o
                                                     --------------------------M_b(M_b f)
                                                           VPinfl\(VP/VP)\NP_o
                                   ------------------------------------------M_b(M_b f)
                                              VPpcp\(VP/VP)\NP_o
                  ---------------------------------------------------M_b(M_b f)
                              VPneg\(VP/VP)\NP_o
        ----------------------------------------------M_b b
                     VPneg\NP_o

     b.           En                  minä    tennistä     näissä ...
        ---------------------------   ----    --------     -------------
        Sfin|NP_s/(Sneg|NP_s)         NP_s    NP_o         Sneg|NP_s\NP_o
        ----------------------------M_f f     ----------------------------b
              Sfin/(Sneg|NP_s)                            Sneg|NP_s
              ----------------------------------------------------f
                                        Sfin
```

This brief discussion of free word order illustrates how Mixed Abstraction metarules, and lexical non-specificity for directionality, offer the potential to characterise languages with a free ordering in a manner not unsimilar to that in which a constituent structure language like English is characterised. One weakness which is apparent however is the lack of distinction in the grammar between bounded free ordering, and unbounded extraction (cf. the difficulty in explaining the apparent boundedness of right extractions in English). As mentioned earlier the proposal here, but one which must remain unpursued, is to make slash operators structured.

1.4. Weak Generative Capacity

It is possible to show that the current grammar framework exceeds context-free grammars in weak generative capacity. I will do this by making a very slight adaptation of the corresponding result of Friedman, Dai, and Wang (1986) for another version of categorial grammar.

Consider a grammar with the lexicon (21), the metarules (22), and nothing else except the basic rules of application.

(21) a := A
b := S\A/C\S, S\A/C
c := C

(22) L: X + Y => Z ==> X\W + Y => Z\W
M: X + Y => Z ==> X/W + Y => Z/W

First, note that the language $a^n b^n c^n$ is a (proper) subset of the language generated since the sequence in (23) can always reduce to S as illustrated, for example, in (24).

(23) $A^n \; S\backslash A/C \; S\backslash A/C\backslash S^{n-1} \; C^n$

```
(24)   a a a  b        b          b            c c c
       - - -  -----    -------    -------      - - -
       A A A  S\A/C    S\A/C\S    S\A/C\S      C C C
                       ----------------L(M(Lb))
                          S\A/C\A/C\S
              -------------------M(Lb)
                 S\A/C\A/C\A/C
```

Second, observe that every sentence in the language generated must contain at least one occurrence of 'b' since only its lexical categories contain the distinguished symbol 'S', and since each lexical category for 'b' has one 'A' argument and one 'C' argument, there will be the same number of 'a's and 'c's as 'b's in each sentence. It then follows that the intersection of the language generated with the regular language $a^*b^*c^*$ is exactly $a^n b^n c^n$. There is a law that the intersection of a context-free language with a regular one is itself context-free. Since $a^n b^n c^n$ is non-context-free, it follows that the language generated by the grammar is non-context-free also.

It is also possible to put an upper bound on the generative capacity of the framework with just binary metarules, though I am uncertain of the situation when unary rules are included. Note that although a set of metarules will define an infinite set of rules, successive application of metarules yields mother and daughter categories of higher and higher order. This means that for any two categories of finite order, there are a finite number of rules which can combine them; these can easily be computed. For any finite sequence of categories then, there are a finite number of neighbouring pairs, and for each of these a finite number of mother categories to which they can reduce. Each reduction decrements the sequence length, reducing the problem size; it follows that the language generated by the grammar is decidable. Thus at least so far as the binary metarules are concerned, the languages generated are a subset of the recursive languages. The result of Uszkoreit and Peters (1986) to the effect that in general PSG with metarules can generate any recursively enumerable language does not carry over to the case here where we are dealing with a very small class of metarules. Arbitrary metarules cannot be used; they must be semantically coherent. In the Section 2 I show how the semantic consideration of compositionality alone restricts the languages that can be generated.

2. *Semantics*

In this section I want to turn from consideration of grammars, to consideration of compositionality, a principle embracing the whole of this inquiry into grammar. I will show how a principle of compositionality positing a limit on the class of functions available as the semantics of rules ensures ceilings on what can be generated by grammars respecting the principle, performing the work of such principles as the θ-criterion of Government-Binding theory, and the completeness and coherence conditions of Lexical-Functional Grammar.[4] The discussion includes reference to connections between type systems and implicational logics.[5]

Consider the following ungrammatical sentences:

(25) *John says

---

[4] The material in this section was developed in collaboration with Bob Carpenter. It appears in Morrill and Carpenter (1987) and was presented at the Logic and Linguistics meeting held at Stanford University in July 1987. Various people have contributed to this work in some manner. In particular we would like to thank Inge Bethke, Kit Fine, Ewan Klein, Marcus Kracht, Barry Richards, Henk Zeevat, and two anonymous referees for Linguistics and Philosophy, in which the material is due to appear.

[5] Cf. van Benthem (1987)

(26) *John laughed Mary

Example (25) can be described as a case of 'missing-words' ungrammaticality -- the verb's complement is missing; example (26) can be described as a case of 'redundant-words' ungrammaticality -- the second proper name is superfluous. In this section I discuss how a principle of compositionality, i.e. a regime for building up meanings of expressions out of the meanings of their parts, can rule out such ungrammaticality, independent of a theory of syntax.

The principle of compositionality (see e.g., Janssen 1983 Chapter I; Partee 1984) usually takes the following form:

(27) *Strong Compositionality*

The meaning of an expression is a function of the meanings of its immediate syntactic subexpressions, and their mode of combination.

For this to be contentful it seems necessary to understand "function" in the sense that given submeanings and a mode of combination, there can be only one result meaning. Thus we can associate with each 'mode of combination' (rule) a mathematical function which maps the meanings of subexpressions into the meanings of the expressions formed by the combination. Under strong compositionality, all (non-lexical) ambiguity is formalised as syntactic ambiguity; this is the characteristic feature of Montague semantics (Montague 1970).

Since, according to strong compositionality, the meaning of a sentence is a function of the meanings of its immediate subexpressions, which are in turn functions of the meanings of their immediate subexpressions, and so on, it follows that the meaning of a sentence is ultimately a function of the meanings of its words, this function being the composition of the functions associated with the rules generating the expression. Thus strong compositionality has the following corollary:

(28) *Weak Compositionality*

The meaning of a sentence is a function of the meanings of its words.

Weak compositionality is not committed to the association of meanings with intermediate expressions, or to the keying of semantic analysis on syntactic structure. Under some views of grammar, different readings are associated with a single syntactic analysis, the product of which is a term of a meaning representation language (e.g. a discourse representation structure, Kamp 1981) which is interpretable in different ways. Such grammars can still fall within the jurisdiction of weak compositionality, though not strong compositionality. In what follows we construe compositionality in the weak sense; the claims we make about it have accordingly wider applicability.

We shall propose a refined version of weak compositionality in which the notion of "function" is delimited, and we shall show how the principle ensures effects like those of Lexical-Functional Grammar's completeness and coherence conditions, and Government-Binding's θ-criterion. We do this by exploiting the fact that for the meaning of a sentence to be a function of the meanings of its words, there must be available a function of a type mapping from the types of the meanings of the words into the type of the meanings of sentences. Not all types will be available given our specific formulations of compositionality; by construing types as formulae of implicational logic, we show that typehood is equivalent to theoremhood, and by proving non-validity, we prove that certain kinds of ill-formed sentences could never be generated by grammars respecting what we will call λI-compositionality.

We identify classes of functions by reference to the pure typed λ-calculus, and Combinatory Logic. "Pure" means that we have no constants; functions of "type" $A \rightarrow B$ map from objects of type A into objects of type B. A non-empty set Δ of *basic types* defines a set of *types* as follows (here and throughout the classes defined are the smallest ones satisfying the specified conditions):

(29) a. If $A \in \Delta$
then A is a type
b. If A and B are types
then $A \rightarrow B$ is a type

For example, suppose Δ includes NP, S, and SP, the types of the meanings of noun phrases, sentences, and sentences with complementizers. Then the set of types will include $NP \rightarrow S$, $SP \rightarrow (NP \rightarrow S)$, $NP \rightarrow ((NP \rightarrow S) \rightarrow S)$, and $NP \rightarrow (NP \rightarrow ((NP \rightarrow S) \rightarrow S))$. The arrow '$\rightarrow$' will be used right-associatively so that, for example, this last formula may be written $NP \rightarrow NP \rightarrow (NP \rightarrow S) \rightarrow S$. Given an infinite set Var_A of variables for each type A, the set of *λK-terms* is defined by:

(30) a. If $v_A \in Var_A$
then v_A is a λK-term of type A

b. If ϕ is a λK-term of type $A \rightarrow B$ and ψ is a λK-term of type A
then $\phi\psi$ is a λK-term of type B

c. If $v_A \in Var_A$ and ϕ is a λK-term of type B
then $\lambda v_A \phi$ is a λK-term of type $A \rightarrow B$

A λK-term without any free variables is said to be *closed*. Assuming the standard functional interpretation, we will call the functions definable by closed λK-terms the *λK-functions*. Then one version of weak compositionality is:

(31) *λK-Compositionality*
The meaning of a sentence is a λK-function of the meanings of its words.

The λK-functions are closed under permutation in the sense that if there is a function mapping certain arguments into a certain result, then there is a function mapping any permutation of those arguments into the same result: the different functions are defined by terms in which the λ-bindings appear in different orders. We are free, then, to adopt the convention that the functions mapping the meanings of words into the meanings of the expressions they form apply to the meanings of the words in left-to-right order. Words with meanings of types A_1, A_2, ... can form a sentence, with meaning of type S, *λK-compositionally* only if $A_1 \rightarrow A_2 \rightarrow \ldots \rightarrow S$ is a *λK-type*, i.e. the type of some λK-function: if this is not a λK-type, then the expression as a whole cannot be assigned a meaning of type S by any λK-function of the meanings of its words. For example, for it to be possible for words of type NP and $SP \rightarrow NP \rightarrow S$ to form a sentence, $NP \rightarrow (SP \rightarrow NP \rightarrow S) \rightarrow S$ must be a λK-type. Likewise, for it to be possible for words of type NP, $NP \rightarrow S$, and NP to form a sentence, $NP \rightarrow (NP \rightarrow S) \rightarrow NP \rightarrow S$ must be a λK-type.

In order to determine whether a type is a λK-type, we take advantage of the fact that a function is definable by a closed λK-term if and only if it is definable by a Combinatory Logic (CL) term, as follows:[6]

---

[6] See Curry and Feys (1958) or Barendregt (1981) for proofs of the equivalence. Often, combinatory logic definitions are given using the substitution combinator **S**:

(i) $\mathbf{S}_{(A \rightarrow B \rightarrow C) \rightarrow (A \rightarrow B) \rightarrow A \rightarrow C}\ (\equiv \lambda x_{A \rightarrow B \rightarrow C} \lambda y_{A \rightarrow B} \lambda z_A [x_{A \rightarrow B \rightarrow C} z_A (y_{A \rightarrow B} z_A)])$

{**S**, **K**} is equivalent to {**I**, **B**, **C**, **W**, **K**} as in the main text, which is in fact equivalent to {**B**, **C**, **W**, **K**}. The formulation given is convenient for our purposes.

(32) a. If A, B, and C are types,

$\mathbf{I}_{A\rightarrow A}$ $(\equiv \lambda x_A[x_A])$
is a CL-term of type $A\rightarrow A$

$\mathbf{B}_{(A\rightarrow B)\rightarrow(C\rightarrow A)\rightarrow C\rightarrow B}$ $(\equiv \lambda x_{A\rightarrow B}\lambda y_{C\rightarrow A}\lambda z_C[x_{A\rightarrow B}(y_{C\rightarrow A}z_C)])$
is a CL-term of type $(A\rightarrow B)\rightarrow(C\rightarrow A)\rightarrow C\rightarrow B$

$\mathbf{C}_{(A\rightarrow B\rightarrow C)\rightarrow B\rightarrow A\rightarrow C}$ $(\equiv \lambda x_{A\rightarrow B\rightarrow C}\lambda y_B\lambda z_A[x_{A\rightarrow B\rightarrow C}z_Ay_B])$
is a CL-term of type $(A\rightarrow B\rightarrow C)\rightarrow B\rightarrow A\rightarrow C$

$\mathbf{W}_{(A\rightarrow A\rightarrow B)\rightarrow A\rightarrow B}$ $(\equiv \lambda x_{A\rightarrow A\rightarrow B}\lambda y_A[x_{A\rightarrow A\rightarrow \mathrm{B}}y_Ay_A])$
is a CL-term of type $(A\rightarrow A\rightarrow \mathrm{B})\rightarrow A\rightarrow \mathrm{B}$

$\mathbf{K}_{A\rightarrow B\rightarrow A}$ $(\equiv \lambda x_A\lambda y_B[x_A])$
is a CL-term of type $A\rightarrow B\rightarrow A$

b. If ϕ is a CL-term of type $A\rightarrow B$ and ψ is a CL-term of type A
then $\phi\psi$ is a CL-term of type B

Thus the λK-types are those types which are derivable from the axiom schemata (33a), corresponding to the combinators in (32a), and the *modus ponens* rule (33b), corresponding to the application in (32b).

(33) a. $A\rightarrow A$
$(A\rightarrow B)\rightarrow(C\rightarrow A)\rightarrow C\rightarrow B$
$(A\rightarrow B\rightarrow C)\rightarrow B\rightarrow A\rightarrow C$
$(A\rightarrow A\rightarrow B)\rightarrow A\rightarrow B$
$A\rightarrow B\rightarrow A$

b. $A\rightarrow B, A \vdash B$

Viewing '→' as implication, (33) provides an axiomatisation of Heyting's implicational system, the implicational intuitionistic logic which Anderson and Belnap (1975) call $H_{\rightarrow}$.[7] So we know that a type A is a λK-type if and only if, regarded as an implicational formula, it is a theorem of $H_{\rightarrow}$.[8] For example, assuming that the meaning of *John* is of type NP, and that the meaning of *says* is of type $SP\rightarrow NP\rightarrow S$, the string in (34a) could be generated λK-compositionally as a sentence only if (34b) is a theorem of $H_{\rightarrow}$.

(34) a. *John says

b. $NP\rightarrow(SP\rightarrow NP\rightarrow S)\rightarrow S$

We will prove that (34b) is not a theorem of $H_{\rightarrow}$ by exhibiting a counter-model.

---

[7] Their axiomatisation on p.10 corresponds to {S, K, I} which is equivalent to {S, K}.

[8] Intuitionistic implicational logic differs from classical implicational logic in that Pierce's Law (i) holds in the latter but not the former.

(i) $((A\rightarrow B)\rightarrow A)\rightarrow A$

There are no pure functions with a type of the form (i).

Given a set P of proposition symbols, we define models for implicational logics in the manner prescribed by Urquhart (1972). A model for $H_{\rightarrow}$ is a quadruple $M = \langle L, \cup, \bot, v\rangle$ where $\langle L, \cup, \bot\rangle$ is a join semi-lattice with bottom element $\bot$,[9] and the valuation function v is a function mapping from L into subsets of P, meeting the following condition:

(35) *Hereditary Condition*

For every $p \in P$ and all $i,j \in L$, if $p \in v(i)$ then $p \in v(i \cup j)$.

We refer to the elements of L as indices, and to $\cup$ as the least upper bound operation. Intuitively, the indices are information states, and least upper bound is the operation of combining information. The set of proposition symbols associated with an index by v corresponds to the set of basic propositions which are true at the index. The hereditary condition entails that the set of true propositions increases monotonically as we move up the lattice.

For a model M we define a satisfaction relation $\models_M$ between indices and formulae by:

(36) a. For every $p \in P$ and every $i \in L$, $i \models_M p$ if and only if $p \in v(i)$

b. For all formulae ϕ, ψ, and every $i \in L$, $i \models_M \phi \rightarrow \psi$ if and only if for every $j \in L$, $j \models_M \phi$ only if $i \cup j \models_M \psi$

Thus an implicational formula is determined on the basis of the information at an index i if and only if for every index j which determines the antecedent, the consequent is determined by the information obtained by putting together that at i and j. A formula ϕ is *valid with respect to a model M* if and only if it is satisfied at the bottom index. A formula is *valid* if and only if it is valid in every model.

Given these definitions we can now show that (34b) is not valid by exhibiting a counter-model -- a model which does not satisfy it. Consider the model $\langle\{\bot\}, \cup, \bot, v\rangle$ where $v(\bot) = \{NP\}$. A proof that this is a counter-model runs as follows. We are required to show

(37) $\bot \not\models NP \rightarrow (SP \rightarrow NP \rightarrow S) \rightarrow S$

Since $\bot \models NP$, (37) holds if

---

[9] A join semi-lattice $\langle L, \cup, \bot\rangle$ consists of a set L, with a distinguished element $\bot$, over which a binary operation $\cup$ is defined such that for all $i,j,k \in L$:

i) $i \cup i = i$

ii) $i \cup j = j \cup i$

iii) $i \cup (j \cup k) = (i \cup j) \cup k$

iv) $\bot \cup i = i$

(38) $\perp \not\models (SP \rightarrow NP \rightarrow S) \rightarrow S$

And (38) holds if $\perp \not\models S$ (which is true by assumption) and

(39) $\perp \models SP \rightarrow NP \rightarrow S$

But (39) is true since no member of $\{\perp\}$ satisfies SP. Hence (37) is true and (34b) is not a theorem of $H_{\rightarrow}$ and not the type of any λK-function. So assuming the given assignment of word meanings to types, (34a) could not be generated by any λK-compositional grammar.

Within Lexical-Functional Grammar (LFG), sentences like (34a) in which an argument is missing are excluded by the completeness condition and within Government-Binding (GB), they are excluded by the θ-criterion (in conjunction with the projection principle etc.):

(40) *Completeness*

> An f-structure is *locally complete* if and only if it contains all the governable grammatical functions that its predicate governs. An f-structure is *complete* if and only if it and all its subsidiary f-structures are locally complete. [Kaplan and Bresnan 1982, pp211-2]

(41) *θ-Criterion*

> Each argument bears one and only one θ-role, and each θ-role is assigned to one and only one argument. [Chomsky 1981, p36]

Completeness in LFG requires that f-structures contain the grammatical functions governed by predicates, for example they must contain the grammatical functions fulfilled by complements for which a verb is subcategorized. Completeness excludes sentences like **John says* because the f-structure would not contain the grammatical function SCOMP governed by the predicate 'say'. GB's θ-criterion requires that a verb's θ-roles stand in a one-to-one relation with arguments present. The θ-criterion excludes sentences such as **John says* because the θ-role that should be filled by a complementized sentence would not be assigned to any argument. The θ-criterion also excludes sentences like (42a) which contain a redundant argument. In LFG this is done by the coherence condition (43).

(42) a. *John laughed Mary
b. $NP \rightarrow (NP \rightarrow S) \rightarrow NP \rightarrow S$

(43) *Coherence*

> An f-structure is *locally coherent* if and only if all the governable grammatical functions that it contains are governed by a local predicate. An

> f-structure is *coherent* if and only if it and all its subsidiary f-structures are locally coherent. [Kaplan and Bresnan 1982, p212]

Coherence excludes (42a) because the grammatical function fulfilled by *Mary* will not be governed, and the θ-criterion excludes the sentence because *Mary* will be assigned no θ-role. However λK-compositionality does not exclude such a sentence; for example the following λK-term designates a function of the requisite type $NP \rightarrow (NP \rightarrow S) \rightarrow NP \rightarrow S$:

(44) $\lambda x_{NP} \lambda y_{NP \rightarrow S} \lambda z_{NP} [y_{NP \rightarrow S} x_{NP}]$

What is distinctive about this function is that it engenders vacuous abstraction: z_{NP} does not appear in the body of the λK-term. We suggest that universal grammar does not admit vacuous functional abstraction.[10]

It would be odd for a grammar to afford vacuous abstraction: its significance would be that on occasion the meanings of words do not contribute to the meanings of the sentences in which they appear. Such superfluity would be an unexpected feature in a system which evolved to facilitate communication. Potential counterexamples to our hypothesis include dummy subjects:

(45) a. It seems that Mary left
b. There is a party

However it is not the case that such examples can only be analysed by vacuous abstraction; thus Sag (1982) provides an analysis in which *it* and *there* have the identity function as their lexical semantics.

The suggestion, then, is a refined version of weak compositionality making reference to just the *λI-functions*, the functions definable by closed λK-terms without vacuous abstraction:

(46) *λI-Compositionality*
The meaning of a sentence is a λI-function of the meanings of its words.

The combinator corresponding to vacuous abstraction is K. The functions definable by CL-terms as indicated earlier, except without K, are the λI-functions.[11] The axioms corresponding to these remaining combinators, with *modus ponens*, are those of Church's weak theory of implication, the implicational relevance logic which Anderson and Belnap

---

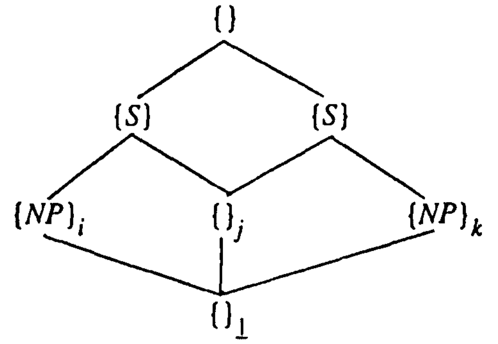
[10] In relation to truth-functional connectives, Gazdar and Pullum (1976) suggest that every conjunct in a coordinate sentence must be potentially relevant to determining the truth value of the whole: a principle which they refer to as "compositionality".

[11] Again, See Curry and Feys (1958) or Barendregt (1981).

call $R_{\rightarrow}$.[12]

We can prove that (42b) is not a theorem of $R_{\rightarrow}$, and hence prove that sentences like (42a) cannot be generated λI-compositionally. A model for $R_{\rightarrow}$ is just like a model for $H_{\rightarrow}$ except that v is not required to meet the hereditary condition, so that the logic is non-monotonic. A counter-model for (42b) is represented by the following diagram:

(47)



Each node corresponds to an index and shows its associated image under v. The least upper bound of any two indices is the lowest index that dominates them both; for convenience of reference some of the nodes have been labeled with subscripts. To prove that (47) is a counter-model we need to show that

(48) $\quad \perp \not\models NP \rightarrow (NP \rightarrow S) \rightarrow NP \rightarrow S$

This is true if for some index x, the antecedent is satisfied at x while the consequent is not satisfied at $x \cup \perp$ $(= x)$. In particular then, (48) is true if

(49) $\quad j \models NP$ and $j \not\models (NP \rightarrow S) \rightarrow NP \rightarrow S$

Since $j \models NP$ (by assumption), the value of (49) coincides with its right hand conjunct, which holds if there is an index which satisfies $NP \rightarrow S$ but whose least upper bound with j fails to satisfy $NP \rightarrow S$, as in

(50) $\quad k \models NP \rightarrow S$ and $k \cup j \not\models NP \rightarrow S$

But the left hand conjunct of (50) is true because it *is* the case that for every index satisfying NP, the least upper bound of that index and k satisfies S. The right-hand side is true because while the index l satisfies NP, the least upper bound of this index and $k \cup j$ does not satisfy S. This completes the proof.

The work reported here was undertaken through a conviction that compositionality provides a correct linguistic methodology. Operating within this methodology, we have shown how an empirical hypothesis, λI-compositionality, captures missing-word and redundant-

---

[12] See especially their axiomatisation $R_{\rightarrow 1}$ of $R_{\rightarrow}$ on p88.

word ungrammaticality. Under type-driven interpretation (such as forms the basis of the Semantic Interpretation Schema of Generalised Phrase Structure Grammar, Gazdar et al. 1985), the interpretation of combination is determined on the basis of the types of the daughter meanings. In its simplest form this is limited to functional application, and certainly falls within the regime of λI-compositionality. In the context of Combinatory Categorial Grammar, Steedman (1988) claims that the interpretation of combination must be extended to include functional composition **B** and substitution **S**, but not **K**, so that these proposals also adhere to λI-compositionality.

In fact λI-compositionality is probably too strong a claim in that it admits only pure functions: the analysis of bare plurals may require the non-lexical introduction of quantification, and *that*-less relative clauses seem to require non-lexical introduction of a conjunction operation to define the restriction of the head noun. λI-compositionality is probably also too generous in that it affords the full power of functional abstraction: it remains an interesting question whether compositionality can be narrowed down to a smaller class of functions. However we take it as encouraging for the methodology of compositionality that even the rather baroque hypothesis λI-compositionality marries up with proposals made in the context of phrase structure grammar and categorial grammar, and brings with it certain desirable effects like those of LFG's completeness and coherence conditions, and GB's θ-criterion.

In the next chapter I return to consider properties of the grammar being advocated. One of the major characteristics is that it assigns multiple equivalent analyses. This contradicts the general assumption that unambiguous expressions have a single derivation, and is perhaps ironic in an approach which from the start adopted a methodology intended to render the grammar computationally manageable. In Chapter VI therefore I discuss processing in general and address the particular issues which arise in relation to the current grammar.

Chapter VI

Processing

This chapter deals with various issues related to parsing and meaning representation. The grammars we have been dealing with are rule-to-rule compositional: the semantic rules assign meanings to expressions in terms of the meanings of the immediate syntactic subparts. The converse situation would be one where the syntactic analysis as a whole is interpreted by semantic rules, so that syntactic analysis is *autonomous*. In a model of grammar incorporating autonomous syntax, the syntactic processor would supply syntactic analyses to be semantically interpreted. But in a rule-to-rule system the syntactic and semantic processing can proceed together (though they don't *have* to do so), and such a regime makes sense in the light of psycholinguistic evidence to the effect that semantic analysis renders available semantic information which can influence further syntactic processing (see e.g. Marslen-Wilson and Tyler 1977, Crain and Steedman 1985, Altmann 1986).

In an ideally efficient processor, no unnecessary work is done; in this chapter I discuss how this ideal might be approached in parsing categorial grammars. In Section 1 I look at the parsing of pure categorial grammar using a unification semantics; in Section 2 I look at the parsing of categorial grammar with metarules using a combinatory logic semantics. Both cases use a Prolog implementation of a chart parsing algorithm described as 'percolation parsing'.

1. Parsing Pure Categorial Grammar with Unification Semantics

In Section 1.1 I discuss pure categorial grammars and charts, in Section 1.2 I describe a chart parsing implementation with unification.

1.1. Pure Categorial Grammar and Charts

The parsing of pure categorial grammar is attractively simple. The only two rules are forward and backward application, which are binary, so that all analyses are binary and are completely specified by a binary tree in which the preterminals are labeled with the lexical categories of the words in the string, and each mother node is labeled with say **f** or **b** according to whether the corresponding reduction was forward or backward application. It was mentioned above that we do not want to do any more work than is necessary. A nice feature of pure categorial grammar is that we can prove that all analyses yield different semantics, so that there is no redundancy amongst analyses.

To see this, note that because the only rules are those of application, all possible meanings of a string $w_1\ w_2\ w_3$ ... are represented by an applicative structure made up of the words' meanings. For example for $a\ b\ c$ the possibilities are $(a'\ (b'\ c'))$, $((a'\ b')\ c')$, $(a'\ (c'\ b'))$ and so on. We can show that all distinct analyses yield different meanings by noting that all distinct applicative structures have different meanings, (ignoring the case where different applicative formulae 'accidently' evaluate to the same result) and showing that all distinct pure categorial grammar analyses correspond to distinct applicative structures. Where the applicative structure corresponding to the left input to forward application is ϕ, and that corresponding the right input is ψ, that of the mother is $(\phi\ \psi)$; where these are inputs to backward application, the applicative structure corresponding to the mother is $(\psi\ \phi)$. Thus the applicative structure corresponding to an analysis bears the same immediate dominance structure as the analysis (but not necessarily the same linear precedence one). So if two analyses differ in their hierarchical structure, their applicative structures also so differ, and are thus distinct. If two derivations share their hierarchical structures, but differ as to whether some nodes are labeled **f** or **b**, then the linear precedence of their applicative structures differ, so that meanings are again distinct. Thus all pure categorial grammar analyses of a given string assign distinct meanings.

A standard technique used for efficient parsing is to use a *chart* (Kay 1967, 1980; Winograd 1983). This is a data structure on which is stored all the constituent analysis that has been performed so far: it is a set of 'edges', each edge spanning a region of the input string. An edge will consist of a position of origin and a landing position, usually encoded by integers indexing the input string, and a syntactic/semantic labeling characterising the constituent. Storing this information means that, for example, when two similar paths of analysis are pursued, work done following one path need not be repeated following the other. For example suppose we were analysing a sentence in which the subject has two analyses, and the predicate has two analyses. Then under one strategy we might find the first subject analysis, and then the first predicate analysis, and then backtrack and find the second predicate analysis, and then backtrack and find the second subject analysis. But at this point we have retraced our steps back past both the predicate analyses, and we have to recompute them. If on the other hand we had stored on a chart all the predicate analyses when they were first computed, the processor would simply need to look up all the analyses which it knows have already been found.

The patterns of control that can be coupled with use of a chart vary considerably. I will describe one particular strategy which is equivalent to that employed in Calder, Moens, and Zeevat (1986) and which can be regarded as an instance of the Cocke-Younger-Kasami algorithm (see e.g. Chapter 4 of Aho and Ullman 1972).

The algorithm is as follows. Starting with an empty chart, read the first word from the input string and record an edge from position zero to position one labeled with the lexical syntax/semantics of the word.[1] Then test to see if there were any edges on the original chart landing at the origin of this new edge (in this first case there will of course be none). Next read the following word, recording an edge from position one to two labeled with its lexical syntax/semantics. Then see whether there are any edges landing at the origin of this new one (in this case there will be just the first word's lexical edge), and if so whether forward or backward application can reduce the two edges. If they can be reduced, add the resulting edge, and check whether there were any edges landing at the origin of *this* one (in this case there will be none). Then read the next word and add its lexical edge, and check through the original edges and for each landing at its origin, test whether forward or backward application can apply and if so add the new edge, and search for edges landing at *its* origin, and so on. Thus at each stage a word is read, its lexical edge added, and a series of other edges are 'precipitated' or 'percolated' leftwards. By way of example, consider parsing the following:

(1)	John	will	leave	tomorrow
	----	------------	-----	-----------
	NP	S\NP/(S\NP)	S\NP	S\NP\(S\NP)

By the time the first two words have been read, the chart will be as follows:

(2) edge(0,NP,1)
edge(1,S\NP/(S\NP),2)

It is not possible to reduce these by application, so the next word is read, its lexical edge *edge(2,S\NP,3)* added, and edges landing at its origin are sought. There is only one of these, *edge(1,S\NP/(S\NP),2)*, and this *can* reduce with the new edge, so that *edge(1,S\NP,3)* is added. Looking for edges landing at this edge's origin, we find *edge(0,NP,1)* which can be applied to, so that *edge(0,S,3)* is added, and this has no edges leading into it. So the state of the chart settles at:

---

[1] If there is no lexical entry then, obviously, fail. The algorithm generalises straightforwardly to handle lexical ambiguity.

(3) edge(0,NP,1)
edge(1,S\NP/(S\NP),2)
edge(2,S\NP,3)
edge(1,S\NP,3)
edge(0,S,3)

Then the next word is read, and *edge(3,S\NP\(S\NP),4)* is added. This has three edges leading into it, two of which it can apply back to. Reducing *edge(2,S\NP,3)* and *edge(3,S\NP\(S\NP),4)* gives *edge(2,S\NP,4)* which reduces with *edge(1,S\NP/(S\NP),2)* to give *edge(1,S\NP,4)* which in turn reduces with *edge(0,NP,1)* to give *edge(0,S,4)*. Reducing *edge(1,S\NP,3)* and *edge(3,S\NP\(S\NP),4)* gives *edge(1,S\NP,4)* which also reduces with the first word's edge to give *edge(0,S,4)*:

(4) edge(0,NP,1)
edge(1,S\NP/(S\NP),2)
edge(2,S\NP,3)
edge(1,S\NP,3)
edge(0,S,3)
edge(3,S\NP\(S\NP),4)
edge(2,S\NP,4)
edge(1,S\NP,4)
edge(0,S,4)
edge(1,S\NP,4)
edge(0,S,4)

The two different analyses will yield the two different scopings of the auxiliary and adverbial.

Notice that the edges added when a word is read all land at the same position: just right of the word. Consequently no two new edges could ever reduce, since they don't lead into one another, and it is only the chart that existed before the word was read that needs to be searched for incoming edges. Thus addition of each edge is entirely independent, and a parallel machine would appropriately allocate attempted reduction of a new edge with each edge leading into it, to separate processors operating in parallel. Thus while the conceptualisation of the parsing process is serial, it lends itself to parallel implementation. The algorithm has the character that all parses are pursued as far as possible (either in parallel or one after the other) at each stage, and can be described as a left-to-right (pseudo-) parallel chart parsing algorithm.

1.2. Implementation with Unification Semantics

In categorial grammar as it has been presented the basic category symbols are *atomic*. There are a variety of ways in which this basic system can be generalised so that these 'basic' symbols are featurally structured (see for instance Pollard 1985, Uszkoreit 1986, Wittenburg 1986, Karttunen 1986, Zeevat, Klein, and Calder 1987, Pareschi and Steedman 1987, Bouma 1987, Whitelock 1988, and Pollard and Sag 1988). Then unifiability rather than identity is the criterion for matching, and semantic representations are built by the process of unification of structures encoding both syntactic and semantic information; these structures are referred to as *signs*. For example, in a move analogous to that from phrase structure grammars to definite clause grammars, syntactic-semantic structures may be Prolog-like terms, thus:

```
(5)     a.  likes   := s(fin,like(X,Y))\np(nom,Y)/np(acc,X)
        b.  John    := np(C,john)
        c.  Mary    := np(C,mary)
```

```
(6)      John                    likes                       Mary
      ----------  ---------------------------------------  ----------
      np(C,john)  s(fin,like(X,Y))\np(nom,Y)/np(acc,X)   np(C,mary)
                  ---------------------------------------------------
                          s(fin,like(mary,Y))\np(nom,Y)
      ---------------------------------------------------------
                     s(fin,like(mary,john))
```

Alternatively, there could be full unification over feature-structures or directed acyclic graphs, see for example Figure 1 and Figure 2 where capitals indicate 're-entrancy'. In the following pages I illustrate such approaches by means of a Prolog implementation of categorial grammar with a unification semantics.

The clauses in (7) declare the left-associative categorial slash operators, the application operator '`', the operator ':' separating semantic representations and basic category symbols, and the lexical assignment operator ':='.

```
(7)         :- op(400,yfx,/).
            :- op(400,yfx,).
            :- op(200,yfx,`).
            :- op(300,xfx,:).
            :- op(500,xfx,:=).
```

The top-level procedure *prs(+Str)* parses anew the string *Str*. It clears the chart, using

$$\text{likes} := \begin{bmatrix} \text{cat} & \text{s} \\ \text{vform} & \text{fin} \\ \text{sem} & \begin{bmatrix} \text{pred} & \text{likes} \\ \text{subj} & \text{X} \\ \text{obj} & \text{Y} \end{bmatrix} \end{bmatrix} \Big\backslash \text{Y}\begin{bmatrix} \text{cat} & \text{np} \\ \text{case} & \text{nom} \end{bmatrix} \Big/ \text{X}\begin{bmatrix} \text{cat} & \text{np} \\ \text{case} & \text{acc} \end{bmatrix}$$

$$\text{john} := \begin{bmatrix} \text{cat} & \text{np} \\ \text{pred} & \text{john} \end{bmatrix}$$

$$\text{mary} := \begin{bmatrix} \text{cat} & \text{np} \\ \text{pred} & \text{mary} \end{bmatrix}$$

Figure 1

$$\text{John likes Mary} := \begin{bmatrix} \text{cat} & \text{s} \\ \text{vform} & \text{fin} \\ \text{sem} & \begin{bmatrix} \text{pred} & \text{likes} \\ \text{subj} & \begin{bmatrix} \text{cat} & \text{np} \\ \text{case} & \text{acc} \\ \text{sem} & \text{mary} \end{bmatrix} \\ \text{obj} & \begin{bmatrix} \text{cat} & \text{np} \\ \text{case} & \text{nom} \\ \text{sem} & \text{john} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 2

retractall(?Cls), and calls *prs1(+Str,+Pos)* instantiating the current position to zero:

(8)
```
prs(String) :-
        retractall(edge(_,_,_)),
        prs1(String,0).
```

The procedure *prs1(+Str,+Pos)* parses the string *Str* starting from the current chart, and position *Pos*. Position is encoded using successor notation so that *0* is zero, *s(0)* is one, *s(s(0))* is two, and so on.

(9)
```
prs1([Word|String],Pos) :-
        Word := LexCat,
        incorp(Pos,LexCat,s(Pos));
        prs1(String,s(Pos)).
```

Sample lexical entries are as follows. Note how the sentential scope of say *will* is achieved by predicating *will'* of the complete sentence that the verb phrase argument would have formed once it obtained a subject, and then equating the subject sought by *will* with the verb's subject by sharing variables. This amounts to a unification implementation of functional composition.

(10)
```
john     :=   j:np.
mary     :=   m:np.
bill     :=   b:np.
you      :=   you:np.
we       :=   we:np.
leave    :=   leave`X:s\X:np.
eating   :=   eating`X:s\X:np.
like     :=   like`X`Y:s\Y:np/X:np.
today    :=   today`X:s\Y:np\(X:s\Y:np).
tomorrow:=    tomorrow`X:s\Y:np\(X:s\Y:np).
will     :=   will`X:s\Y:np/(X:s\Y:np).
to       :=   to`X:s\Y:np/(X:s\Y:np).
seem     :=   seem`X:s\Y:np/(X:s\Y:np).
try      :=   try`X`Y:s\Y:np/(X:s\Y:np).
expect   :=   expect`X`Y:s\Y:np/(X:s\Z:np)/Z:np.
persuade:=    persuade`X`Y`Z:s\Z:np/(Y:s\X:np)/X:np.
promise:=     promise`X`Y`Z:s\Z:np/(Y:s\Z:np)/X:np.
that     :=   that`X:sp/X:s.
think    :=   think`X`Y:s\Y:np/X:sp.
with     :=   with`X`Y:s\Z:np\(Y:s\Z:np)/X:np.
while    :=   while`X`Y:s\Z:np\(Y:s\Z:np)/(X:s\Z:np).
```

The heart of the implementation is *incorp(+MPos,+RCat,+RPos)* which incorporates into the chart between positions *MPos* and *RPos* an edge of category *RCat*, and causes precipitation of all consequential edges.

```
(11)  a. incorp(MPos,RCat,RPos) :-
                assert(edge(MPos,RCat,RPos)), !,
                edge(LPos,LCat,MPos),
                rule(LCat,RCat,MCat),
                incorp(LPos,MCat,RPos).
      b. rule(X/Y,Y,X).
         rule(Y,X\Y,X).
```

The procedure asserts the edge and searches in a failure-driven loop for edges which lead into it and with which it can reduce, calling itself recursively for all such cases.[2] Test clauses are shown in Figure 3. By way of illustration, a call of *prs([john,will,leave,tomorrow])* yields the following edge labels spanning the whole input string, these exhibiting the alternative scopings of the adverbial and auxiliary operators.

(12) a. will‘(tomorrow‘(leave‘)):s
     b. tomorrow‘(will‘(leave‘j)):s

```
str([we,leave]).
str([we,like,mary]).
str([we,like,mary,today]).
str([you,seem,to,like,mary]).
str([you,seem,to,like,mary,today]).
str([we,try,to,like,mary,today]).
str([we,expect,john,to,leave]).
str([we,persuade,john,to,leave]).
str([we,promise,john,to,leave]).
str([we,think,that,you,leave]).
str([we,think,that,you,leave,with,mary]).
str([you,leave,while,sleeping]).
str([you,leave,while,sleeping,today]).
str([we,think,that,you,leave,while,sleeping,today]).
```

Figure 3

---

[2] I am grateful to Bob Carpenter for suggesting implementation of the failure-driven loop by omission of a base clause.

The unification approach raises a number of questions. For example, what criteria can be brought to bear on the issue of choosing amongst the various possible unificiation formalisms, and on the issue of choosing a grammar for a language within a particular formalism? There are many possibilities in both cases: the area is unconstrained. In Appendix A I show that categorial grammar with unification is NP-complete, i.e. in the worst case the problem of determining whether a grammar generates a string is computationally intractable. Also in the Appendix I discuss the relation of a unification semantics approach to ones in which syntactic category symbols are kept separate from semantic representations. It is suggested there that a unification approach paying proper regard to the semantic function of unification may avoid the awkward complexity result. In Section 2 however I consider a rather different approach in which a combinatory logic semantics is used.

2. *Parsing Generalised Categorial Grammars with Combinatory Logic Semantics*

While pure categorial grammar has the attractive property that all derivation paths have a distinct semantics, generalised categorial grammars typically lose this property. Consider for example the paths in (13a) and (13b).

(13) a. $[_{f}\,[_{Rf}\,X/Y\;\; Y/Z]_{X/Z}\;\; Z]_{X}$
f (R f x y) z = x (y z)
b. $[_{f}\,X/Y\;\; [_{f}\,Y/Z\;\; Z]_{Y}]_{X}$
f x (f y z) = x (y z)

The paths are equivalent because the result of applying the composition of two functions to an argument is the same as the result of applying the first function to the result of applying the second to the argument. Since there are many derivation paths with the same semantics, naive pursuit of all possible derivations of a string would be highly redundant, and is in fact computationally unfeasible.

A number of solutions to this problem have been advocated. Pareschi and Steedman (1987) prescribe what they call "lazy chart parsing" in which a special 'revealing' procedure is invoked when failure enforces backtracking; Hepple (1987) argues that the algorithm is incorrect. Wittenburg (1987) suggests compilation of combinators into computationally manageable 'predictive' combinators; thus (14) is replaced by (15).

(14) *Forward Composition*
X/Y + Y/Z => X/Z

(15) *Forward-Predictive Forward Functional Composition*

X/(Y/Z) + Y/W => X/(W/Z)

However the derived grammar is not strictly equivalent to the original one, for example it does not include the following instance of forward composition:

(16) VP/PP + PP/NP => VP/NP

To my knowledge, a proof of equivalence, in some sense, does not exist; it may be appropriate to regard the proposal simply as a different grammar.

Karttunen proposes a 'subsumption' check whereby before an edge is added to the chart, a search is made to ensure that an equivalent edge is not already there. This seems to require (i) repeated searches through the chart, in order to find potential equivalents, and (ii) identical normal forms for equivalent meaning representations, so that meaning equivalence can be determined by syntactic identity of meaning representations. Repeated reduction to normal form would be avoided if normal form were constantly maintained. This is the case in unification approaches such as the one in the last section. However as is well known there are certain problems for unification with predicating functions of different arguments, as would be the case in for example (17):[3]

(17)	John	and	Mary	left
	A/(A\j:np)		B/(B\m:np)	left'Z:s\(Z:np)

The analysis requires predicating *left* of both *j* and *m*; however the single variable *Z* cannot be instantiated to two different constants.

Assuming the problem of normal form could be efficiently managed, the expenses of searching and comparing still appear unavoidable under the subsumption check approach. What we really seem to want is for the processor to exploit knowledge of the semantics of rules so that it is known when there will be equivalence, without having to perform a lookup in the chart. This is precisely the proposal of Mark Hepple (personal communication), who has defined equivalences between local derivation paths involving type-raising and generalised composition, on the basis of the participating combinators. In the next section I define such equivalences for the binary metarules of the grammar for English presented in Chapter IV. The incorporation of unary rules remains a topic for further research.

---

[3] The proper names need to be type-raised in order to obtain the distributive reading along the lines of the boolean conjunction analysis of e.g. Partee and Rooth (1983).

2.1. Generalised Categorial Grammars and Equivalences

An equivalence relation $\equiv$ over local derivation paths will be defined, along with an ordering $<$ such that every equivalence class has a least member; in this context, a *local* derivation path will mean a three-leaf binary derivation path. We will then say that a local derivation path should not be pursued if it has an equivalent which precedes it, i.e. it should not be pursued unless it is the least member of its equivalence class. Since derivation paths are isomorphic to terms of combinatory logic (Chapter VII, Section 2.2), this amounts to defining a normal form on the combinatory logic induced by the grammar, and disqualifying any derivation generating a non-normal form.

Equivalences amongst left-branching local derivation paths will be considered first, then amongst right-branching, and then between left- and right- branching. First, note that paths are equivalent to themselves. Thus there are the following equivalences in which at least one rule is basic application:

$$
\begin{array}{lll}
(18) & [_{\mathbf{f}}\,[_{\psi}\, X \;\; Y]_{V/Z} \;\; Z]_V & \equiv\; [_{\mathbf{f}}\,[_{\psi}\, X \;\; Y]_{V/Z} \;\; Z]_V \\
 & [_{\mathbf{b}}\,[_{\psi}\, X \;\; Y]_W \;\; V\backslash W]_V & \equiv\; [_{\mathbf{b}}\,[_{\psi}\, X \;\; Y]_W \;\; V\backslash W]_V \\
 & [_{\phi}\,[_{\mathbf{f}}\, X/Y \;\; Y]_X \;\; Z]_V & \equiv\; [_{\phi}\,[_{\mathbf{f}}\, X/Y \;\; Y]_X \;\; Z]_V \\
 & [_{\phi}\,[_{\mathbf{b}}\, Y \;\; X\backslash Y]_X \;\; Z]_V & \equiv\; [_{\phi}\,[_{\mathbf{b}}\, Y \;\; X\backslash Y]_X \;\; Z]_V
\end{array}
$$

Next, suppose there is the following equivalence:

$$
(19) \qquad [_{\phi}\,[_{\psi}\, X \;\; Y]_A \;\; Z]_V \;\equiv\; [_{\phi'}\,[_{\psi'}\, X \;\; Y]_B \;\; Z]_V
$$

If the left-hand path exists, then so too must the one in (20) where *Z* becomes *Z/W* and **R**ϕ rather than ϕ applies, so that *W* is right-abstracted onto the root category.

$$
(20) \qquad [_{\mathbf{R}\phi}\,[_{\psi}\, X \;\; Y]_A \;\; Z/W]_{V/W}
$$

There must also exist a derivation likewise related to the right hand side of (19):

$$
(21) \qquad [_{\mathbf{R}\phi'}\,[_{\psi'}\, X \;\; Y]_B \;\; Z/W]_{V/W}
$$

And since the derivations in (19) were equivalent, i.e. (22a) holds, it must also be the case that (22b) holds.

$$
\begin{array}{lll}
(22) & \text{a.} & \phi\;(\psi\; x\; y)\; z \;=\; \phi'\;(\psi'\; x\; y)\; z \\
 & \text{b.} & \mathbf{R}\;\phi\;(\psi\; x\; y)\; z \;=\; \mathbf{R}\;\phi'\;(\psi'\; x\; y)\; z \;\; \text{since} \\
 & & \mathbf{R}\;\phi\;(\psi\; x\; y)\; z\;\; w \;=\; \phi\;(\psi\; x\; y)\;(z\;\; w) \;= \\
 & & \mathbf{R}\;\phi'\;(\psi'\; x\; y)\; z\;\; w \;=\; \phi'\;(\psi'\; x\; y)\;(z\; w)
\end{array}
$$

Thus from (19) we can infer that (20) is equivalent to (21).

Now (23a) and (23b) are to (19) as (20) and (21) are to (19), but with the difference that /*W* is instantiated not on the right-most leaf, but the middle one; again they are semantically equivalent.

(23) a. $[_{M\phi}\ [_{R\psi}\ X\ \ Y/W]_{A/W}\ \ Z]_{V/W}$
b. $[_{M\phi'}\ [_{R\psi'}\ X\ \ Y/W]_{B/W}\ \ Z]_{V/W}$

A forward slash could also be inherited from the left-most leaf, or parasitically from two or all three leaves: a total of seven cases. There is a further case where a backward slash is inherited from the left-most leaf, but no others since (in English) there are no backward-slash 'mixing' or parasitic metarules. The full set of eight cases is as shown in (24); the notation $\pi_1 \equiv \pi_2 \rightarrow \pi_3 \equiv \pi_4, \pi_5 \equiv \pi_6, \ldots$ indicates that from the antecedent equivalence, the consequent equivalences can be inferred.

(24)
$$\begin{array}{ll}
[_{\phi}\ [_{\psi}\ X\ \ Y]_{A}\ \ Z]_{V} & \equiv [_{\phi'}\ [_{\psi'}\ X\ \ Y]_{B}\ \ Z]_{V}\ \rightarrow \\
[_{R\phi}\ [_{\psi}\ X\ \ Y]_{A}\ \ Z/W]_{V/W} & \equiv [_{R\phi'}\ [_{\psi'}\ X\ \ Y]_{B}\ \ Z/W]_{V/W}, \\
[_{M\phi}\ [_{R\psi}\ X\ \ Y/W]_{A/W}\ \ Z]_{V/W} & \equiv [_{M\phi'}\ [_{R\psi'}\ X\ \ Y/W]_{B/W}\ \ Z]_{V/W}, \\
[_{M\phi}\ [_{M\psi}\ X/W\ \ Y]_{A/W}\ \ Z]_{V/W} & \equiv [_{M\phi'}\ [_{M\psi'}\ X/W\ \ Y]_{B/W}\ \ Z]_{V/W}, \\
[_{P\phi}\ [_{R\psi}\ X\ \ Y/W]_{A/W}\ \ Z/W]_{V/W} & \equiv [_{P\psi'}\ [_{R\psi'}\ X\ \ Y/W]_{B/W}\ \ Z/W]_{V/W}, \\
[_{P\phi}\ [_{M\psi}\ X/W\ \ Y]_{A/W}\ \ Z/W]_{V/W} & \equiv [_{P\psi'}\ [_{M\psi'}\ X/W\ \ Y]_{B/W}\ \ Z/W]_{V/W}, \\
[_{M\phi}\ [_{P\psi}\ X/W\ \ Y/W]_{A/W}\ \ Z]_{V/W} & \equiv [_{M\phi'}\ [_{P\psi'}\ X/W\ \ Y/W]_{B/W}\ \ Z]_{V/W}, \\
[_{P\phi}\ [_{P\psi}\ X/W\ \ Y/W]_{A/W}\ \ Z/W]_{V/W} & \equiv [_{P\psi'}\ [_{P\psi'}\ X/W\ \ Y/W]_{B/W}\ \ Z]_{V/W}, \\
[_{L\phi}\ [_{L\psi}\ X\backslash W\ \ Y]_{A\backslash W}\ \ Z]_{V\backslash W} & \equiv [_{L\phi'}\ [_{L\psi'}\ X\backslash W\ \ Y]_{B\backslash W}\ \ Z]_{V\backslash W}
\end{array}$$

For the equivalence rules given so far, *W* was inherited 'outwardly', (outermost argument) and if the input equivalence was between identical paths, so too would be the output equivalence. However there are also equivalence rules where inheritance is 'inward' (inner argument). Suppose there is the following equivalence:

(25) $[_{f}\ [_{R\psi}\ X\ \ Y/Z]_{V/Z}\ \ Z\ ]_{V} \quad \equiv [_{\phi'}\ [_{\psi'}\ X\ \ Y/Z]_{B}\ \ Z]_{V}$

There is the path (26) closely related to the left hand side of (25).

(26) $[_{f}\ [_{R(M\psi)}\ X/W\ \ Y/Z]_{V/W/Z}\ \ Z]_{V/W}$

But the argument inherited inwardly could have been inherited outwardly as in (27), related to the right hand side of (25).

(27) $[_{M\phi'}\ [_{M\psi'}\ X/W\ \ Y/Z]_{B/W}\ \ Z]_{V/W}$

And from (28a) we can infer (28b).

(28) a. $f\ (R\ \psi\ x\ y)\ z\ =\ \phi'\ (\psi'\ x\ y)\ z$ i.e.
$\psi\ x\ (y\ z)\ =\ \phi'\ (\psi'\ x\ y)\ z$
b. $f\ (R\ (M\ \psi)\ x\ y)\ z\ =\ M\ \phi\ (M\ \psi\ x\ y)\ z$ since
$f\ (R\ (M\ \psi)\ x\ y)\ z\ w\ =\ R\ (M\ \psi)\ x\ y\ z\ w\ =\ M\ \psi\ x\ (y\ z)\ w\ =\ \psi\ (x\ w)\ (y\ z)$
$M\ \phi'\ (M\ \psi'\ x\ y)\ z\ w\ =\ \phi'\ (M\ \psi'\ x\ y\ w)\ z\ =\ \phi'\ \ (\psi'\ (x\ w)\ y)\ z$

In all, for this kind of equivalence we have:

(29) $[_{f}\ [_{R\psi}\ X\ \ Y/Z]_{V/Z}\ \ Z]_{V} \equiv [_{\phi'}\ [_{\psi'}\ X\ \ Y/Z]_{B}\ \ Z]_{V}\ \rightarrow$
$[_{f}\ [_{R(M\psi)}\ X/W\ \ Y/Z]_{V/W/Z}\ \ Z]_{V/W}\ \ \equiv\ [_{M\phi'}\ [_{M\psi'}\ X/W\ \ Y/Z]_{B/W}\ \ Z]_{V/W},$
$[_{f}\ [_{R(L\psi)}\ X\backslash W\ \ Y/Z]_{V\backslash W/Z}\ \ Z]_{V\backslash W}\ \ \equiv\ [_{L\phi'}\ [_{L\psi'}\ X\backslash W\ \ Y/Z]_{B\backslash W}\ \ Z]_{V\backslash W}$

$[_{f}\ [_{M\psi}\ X/Z\ \ Y]_{V/Z}\ \ Z]_{V} \equiv [_{\phi'}\ [_{\psi'}\ X/Z\ \ Y]_{B}\ \ Z]_{V}\ \rightarrow$
$[_{f}\ [_{M(R\psi)}\ X/Z\ \ Y/W]_{V/W/Z}\ \ Z]_{V/W}\ \ \equiv\ [_{M\phi'}\ [_{R\psi'}\ X/Z\ \ Y/W]_{B/W}\ \ Z]_{V/W}$

For equivalences amongst right-branching local paths, there are, as for left-branching, the four axiomatic equivalences in (30) and the eight outward equivalence rules in (31).

(30) $[_{f}\ V/A\ \ [_{\psi}\ Y\ \ Z]_{A}]_{V}\ \ \equiv\ [_{f}\ V/A\ \ [_{\psi}\ Y\ \ Z]_{A}]_{V}$
$[_{b}\ X\ \ [_{\psi}\ Y\ \ Z]_{V\backslash X}]_{V}\ \ \equiv\ [_{b}\ X\ \ [_{\psi}\ Y\ \ Z]_{V\backslash X}]_{V}$
$[_{\phi}\ X\ \ [_{f}\ Y/Z\ \ Z]_{Y}]_{V}\ \ \equiv\ [_{\phi}\ X\ \ [_{f}\ Y/Z\ \ Z]_{Y}]_{V}$
$[_{\phi}\ X\ \ [_{b}\ Y\ \ Z\backslash Y]_{Z}]_{V}\ \ \equiv\ [_{\phi}\ X\ \ [_{b}\ Y\ \ Z\backslash Y]_{Z}]_{V}$

(31) $[_{\phi}\ X\ \ [_{\psi}\ Y\ \ Z]_{A}]_{V}\ \ \equiv\ [_{\phi'}\ X\ \ [_{\psi'}\ Y\ \ Z]_{B}]_{V}\ \rightarrow$
$[_{R\phi}\ X\ \ [_{R\psi}\ Y\ \ Z/W]_{A/W}]_{V/W}\ \ \equiv\ [_{R\phi'}\ X\ \ [_{R\psi'}\ Y\ \ Z/W]_{B/W}]_{V/W},$
$[_{R\phi}\ X\ \ [_{M\psi}\ Y/W\ \ Z]_{A/W}]_{V/W}\ \ \equiv\ [_{R\phi'}\ X\ \ [_{M\psi'}\ Y/W\ \ Z]_{B/W}]_{V/W},$
$[_{M\phi}\ X/W\ \ [_{\psi}\ Y\ \ Z]_{A}]_{V/W}\ \ \equiv\ [_{M\phi'}\ X/W\ \ [_{\psi'}\ Y\ \ Z]_{B}]_{V/W},$
$[_{R\phi}\ X\ \ [_{P\psi}\ Y/W\ \ Z/W]_{A/W}]_{V/W}\ \ \equiv\ [_{R\phi'}\ X\ \ [_{P\psi'}\ Y/W\ \ Z/W]_{B/W}]_{V/W},$
$[_{P\phi}\ X/W\ \ [_{R\psi}\ Y\ \ Z/W]_{A/W}]_{V/W}\ \ \equiv\ [_{P\psi'}\ X/W\ \ [_{R\psi'}\ Y\ \ Z/W]_{B/W}]_{V/W},$
$[_{P\phi}\ X/W\ \ [_{M\psi}\ Y/W\ \ Z]_{A/W}]_{V/W}\ \ \equiv\ [_{P\psi'}\ X/W\ \ [_{M\psi'}\ Y/W\ \ Z]_{B/W}]_{V/W},$
$[_{P\phi}\ X/W\ \ [_{P\psi}\ Y/W\ \ Z/W]_{A/W}]_{V/W}\ \ \equiv\ [_{P\psi'}\ X/W\ \ [_{P\psi'}\ Y/W\ \ Z/W]_{B/W}]_{V/W},$
$[_{L\phi}\ X\backslash W\ \ [_{\psi}\ Y\ \ Z]_{A}]_{V\backslash W}\ \ \equiv\ [_{\psi'}\ X\backslash W\ \ [_{\psi'}\ Y\ \ Z]_{B}]_{V\backslash W}$

In addition there is the inward equivalence rule:

(32) $[_{b}\, X\ \ [_{L\psi}\, Y\backslash X\ \ Z]_{V\backslash X}]_{V} \quad \equiv\ [_{\phi'}\, X\ \ [_{\psi'}\, Y\ \ Z]_{B}]_{V}\ \rightarrow$
$[_{b}\, X\ \ [_{L(R\psi)}\, Y\backslash X\ \ Z/W]_{V/W\backslash X}]_{V/W} \quad \equiv\ [_{R\phi'}\, X\ \ [_{R\psi'}\, Y\ \ Z/W]_{B/W}]_{V/W}$

Finally, there are equivalences between left- and right-branching paths. There are the following axiomatic equivalences since (34) holds:

(33) a. $[_{f}\, [_{R\psi}\, X\ \ Y/Z]_{V/Z}\ \ Z]_{V} \equiv [_{\psi}\, X\ \ [_{f}\, Y/Z\ \ Z]_{Y}]_{V}$
b. $[_{\phi}\, [_{b}\, X\ \ Y\backslash X]_{Y}\ \ Z]_{V} \ \equiv\ [_{b}\, X\ \ [_{L\phi}\, Y\backslash X\ \ Z]_{V\backslash X}]_{V}$

(34) a. $\mathbf{f}\,(\mathbf{R}\ \psi\ x\ y)\ z\ =\ \mathbf{R}\ \psi\ x\ y\ z\ =\ \psi\ x\ (y\ z)\ =$
$\psi\ x\ (\mathbf{f}\ y\ z)\ =\ \psi\ x\ (y\ z)$
b. $\phi\ (\mathbf{b}\ x\ y)\ z\ =\ \phi\ (y\ x)\ z\ =$
$\mathbf{b}\ x\ (\mathbf{L}\ \phi\ y\ z)\ =\ \mathbf{L}\ \phi\ y\ z\ x\ =\ \phi\ (y\ x)\ z$

And there are eight outward equivalence rules; note that unlike the left-left and right-right cases, these are not symmetric in the rules required to achieve equivalent inheritance patterns

(35) $[_{\phi}\, [_{\psi}\, X\ \ Y]_{A}\ \ Z]_{V} \quad \equiv\ [_{\phi'}\, X\ \ [_{\psi'}\, Y\ \ Z]_{B}]_{V}\ \rightarrow$
$[_{R\phi}\, [_{\psi}\, X\ \ Y]_{A}\ \ Z/W]_{V/W} \quad \equiv\ [_{R\phi'}\, X\ \ [_{R\psi'}\, Y\ \ Z/W]_{B/W}]_{V/W},$
$[_{M\phi}\, [_{R\psi}\, X\ \ Y/W]_{A/W}\ \ Z]_{V/W} \quad \equiv\ [_{R\phi'}\, X\ \ [_{M\psi'}\, Y/W\ \ Z]_{B/W}]_{V/W},$
$[_{M\phi}\, [_{M\psi}\, X/W\ \ Y]_{A/W}\ \ Z]_{V/W} \quad \equiv\ [_{M\phi'}\, X/W\ \ [_{\psi'}\, Y\ \ Z]_{B}]_{V/W},$
$[_{P\phi}\, [_{R\psi}\, X\ \ Y/W]_{A/W}\ \ Z/W]_{V/W} \quad \equiv\ [_{P\psi'}\, X/W\ \ [_{R\psi'}\, Y\ \ Z/W]_{B/W}]_{V/W},$
$[_{P\phi}\, [_{M\psi}\, X/W\ \ Y]_{A/W}\ \ Z/W]_{V/W} \quad \equiv\ [_{P\psi'}\, X/W\ \ [_{R\psi'}\, Y\ \ Z/W]_{B/W}]_{V/W},$
$[_{M\phi}\, [_{P\psi}\, X/W\ \ Y/W]_{A/W}\ \ Z]_{V/W} \quad \equiv\ [_{P\psi'}\, X/W\ \ [_{M\psi'}\, Y/W\ \ Z]_{B/W}]_{V/W},$
$[_{P\phi}\, [_{P\psi}\, X/W\ \ Y/W]_{A/W}\ \ Z/W]_{V/W} \quad \equiv\ [_{P\psi'}\, X/W\ \ [_{P\psi'}\, Y/W\ \ Z/W]_{B/W}]_{V/W},$
$[_{L\phi}\, [_{L\psi}\, X\backslash W\ \ Y]_{A\backslash W}\ \ Z]_{V\backslash W} \quad \equiv\ [_{L\phi'}\, X\backslash W\ \ [_{\psi'}\, Y\ \ Z]_{B}]_{V\backslash W}$

Unfortunately this axiomatisation of equivalence is incomplete. This does not mean that the parsing strategy that will be described is incorrect, but it does mean that the processing is not optimal, because some equivalences are not spotted. For example, amongst the analyses found for string 20 in Appendix B, one equivalence that is not caught is the following:

(36) $[_{M(Rf)}\, [_{R(Rb)}\, Y\ \ [_{M(Rb)}\, Z/U\ \ X\backslash Y\backslash Z/W]]\ \ W/V] \ \equiv$
$[_{R(Rb)}\, Y\ \ [_{M(Rb)}\, Z/U\ \ [_{Rf}\, X\backslash Y\backslash Z/W\ \ W/V]]]$

The proper definition of equivalence remains a topic for further research.

2.2. *Implementation with Combinatory Logic Semantics*

As mentioned earlier, the idea is that the parser will not pursue a local path if there is an equivalent one which precedes it in the ordering. Any ordering determining a least member of each equivalence class (e.g. alphabetical by path-name) would suffice; here, fewer metarule applications will be favoured and, where this is undecisive, left-branching. In particular, where π_l and π_r stand for left- and right-branching local paths, and $\#(\pi)$ is the number of metarule applications in π, the ordering is defined by:

(37) $\pi'_l < \pi_l$ iff $\#(\pi'_l)$ is less than $\#(\pi_l)$
$\pi'_r < \pi_r$ iff $\#(\pi'_r)$ is less than $\#(\pi_r)$
$\pi_l < \pi_r$ iff $\#(\pi_l)$ is less than or equal to $\#(\pi_r)$

Then the parser will apply rules subject to the condition that they are not *unnec*:

(38) unnec(π) iff π' such that $\pi' \equiv \pi$ and $\pi' < \pi$

Again no proof is given that this ordering determines a unique least member in each equivalence class; if it did not, parsing would be correct, but non-optimal since equivalent paths tied as least members would all have no preceding equivalents, and would therefore (redundantly) all be followed. An alternative ordering strategy would be one which favoured left-branching. Steedman (1987b) and Haddock (forthcoming) address incremental interpretation from the point of view of grammar, by reference to a strategy seeking left-branching analyses.

In the parser here, the meaning representations will not be obtained by unification, but will be terms of a combinatory logic containing the rule combinators. The meanings of words will be represented by constants which are usually the same as the words. The lexical entries will thus appear as follows:

(39) john := j : np.
will := will : s\np/(s\np).
leave := leave : s\np.
tomorrow := tomorrow : s\np\(s\np)

The main procedure in the parser is *prs1(+String,+Pos)* which parses *String* from the current chart, and position *Pos*:

```
(40)      prs1([Word|String],Pos) :-
                Word := T:LexCat,
                incorp(Pos,LexCat,s(Pos),T);
                prs1(String,s(Pos)).
```

The procedure *incorp(+MPos,+RCat,+RPos,RTrans)* incorporates into the chart between positions *MPos* and *RPos* an edge of category *RCat* and translation *RTrans*:

```
(41)      incorp(MPos,RCat,RPos,RTrans) :-
                assert(edge(MPos,RCat,RPos,RTrans)), !,
                edge(LPos,LCat,MPos,LTrans),
                incorp2(LPos,LCat,RCat,RPos,LTrans,RTrans).
```

The procedure *incorp2(+LPos,+LCat,+RCat,+RPos,+LTrans,+RTrans)* tries to reduce edges of category *LCat* and *RCat* and translation *LTrans* and *RTrans*, between *LPos* and *RPos*, and causes precipitation of consequential edges. In the case that *LCat* is a coordinator, a left-hand conjunct is sought.

```
(42)      incorp2(LPos,LCat,RCat,RPos,LTrans,RTrans) :-
                rule(LCat,RCat,MCat,G),
                \+ unnec_l(G,LTrans),
                \+ unnec_r(G,RTrans),
                incorp(LPos,MCat,RPos,G‘LTrans‘RTrans).

          incorp2(Pos,crd,Cat,RPos,C,RTrans) :-
                edge(LPos,Cat,Pos,LTrans),
                coord(Cat,S),
                incorp(LPos,Cat,RPos,S‘LTrans‘C‘RTrans).
```

The coordination is limited to categories resulting in *S*; (for these cases the simple semantics of Partee and Rooth 1983 can be employed):

```
(43)      coord(s,c).
          coord(X/_,'Af'‘S) :-
                coord(X,S).
          coord(X\_,'Ab'‘S) :-
                coord(X,S).
```

```
(44)    rule(X/Y,Y,X,f).
        rule(Y,X\Y,X,b).
        rule(X,Y/Z,V/Z,'R'‘G) :-
                rule(X,Y,V,G).
        rule(X\Z,Y,V\Z,'L'‘G) :-
                rule(X,Y,V,G).
        rule(X/Z,Y,V/Z,'M'‘G) :-
                member(Z,[np,sp,n\n,s\np\(s\np),n/n]),
                rule(X,Y,V,G).
        rule(X/np,Y/np,V/np,'P'‘G) :-
                rule(X,Y,V,G).
```

Note, crucially, the calls to check that addition of a new edge is not unnecessary; the procedures *unnec_r/2* and *unnec_l/2*, and those defining equivalence and counting metarule applications, are trivial but tedious and are ommited here. These procedures are called for each possible rule application, and the application is blocked if the goals succeed. As such the algorithm still has a 'generate-and-test' character; it would therefore seem worthwhile to look for other algorithms employing the equivalence axiomatisation.

A full listing of the parser and attendant procedures is given in Appendix B, along with an illustrative log of the behaviour of the system. The implementation in Appendix B embodies a large part of the important theory of grammar and processing offered in this thesis. In the last chapter I make some concluding remarks, and suggest some further areas for research.

Chapter VII

Conclusion

Section 1 contains a brief summary; Section 2 indicates some possible future directions.

1. Summary

Retrospectively, the thesis advocated seems to me a simple and obvious one.[1] It is to integrate the metarules of Gazdar and others with the (categorial) category system of Steedman and others. The empirical force behind the argument has been provided by a largely neglected body of data exhibiting compound non-canonicality; this data was invoked to argue against both the contemporary phrase structure approach, and the contemporary categorial one. Thus in Chapter II it was argued that classification of expressions exhibiting compound non-canonicality requires a category system like that of categorial grammar, and in Chapter III it was argued that formulation of the operations generating compound non-canonicality requires meta-grammatical statements like the metarules of phrase structure grammar. The synthesis was presented in Chapter IV. Chapter V addressed various aspects of universal grammar as suggested by the emergent grammar for English, and it also addressed some more general issues relating to compositionality, a wider paradigm within which both the antecedent theories, and the subsequent one, belong. Chapter VI shifted attention to processing, addressing the major matter arising, that of derivational equivalence, by showing how to axiomatise the equivalences.

2. Future Directions

In this section I indicate some areas for further inquiry that follow from the results of the thesis. Section 1 centres discussion around syntactic complexity, and Section 2, meaning representation.

---

[1] Perhaps that is no bad sign.

2.1. Syntax and Processing

A central feature in this work has been that despite the unboundedness of extraction and coordination phenomena, and the existence of compound non-canonicality, the only operation that has been invoked on sound representations is concatenation; in fact it was the aim from the start to maintain concatenation as the only structural operation. In order to do this, a grammar was constructed employing recursion in its category system, and also in the form of metarules. This pervasive recursion in the formalisation seems appropriate enough in an attempt to model the naturally occurring recursive systems of language.

A grammar employing freely applying metarules defines a single grammar in the limit, but also a hierarchy of grammars leading up to this limit. It is this characteristic which suggests one area for further study.

In the model advocated here there is a canonical fragment characterised by a pure categorial grammar, and a non-canonical fragment, involving various extraction and coordination phenomena, characterized by this categorial grammar augmented with metarules. Such a model, in which there is a 'basic' grammar handling canonical expressions, and an augmentation handling non-canonical ones is, while sometimes not explicit, nevertheless pervasive in linguistic theory. Thus classical transformational grammar offered a phrase structure component, plus transformations; Lexical-Functional Grammar (Bresnan and Kaplan 1982) has 'single arrow' (local) phrase structure annotation, plus 'double arrow' (non-local) annotation, or else functional uncertainty (Kaplan and Zaenen 1987), and the slash-augmentation of a pure phrase structure grammar has been discussed in the course of this work. I have argued that advantages of the metarule approach include the fact that the account respects our intuitions that compound non-canonicality is a sort of stretching of simple non-canonicality, which is in turn an extension of canonicality: the phenomena arise through successive application of metarules. A system with metarules such as the one currently proposed can be regarded as generating a hierarchy of grammars, indexed by metarule application according to some scheme; similarly it can be regarded as generating a hierarchy of languages indexed by some metric of metarule application. In particular, processing complexity and hence acceptability might be expected to reflect the number of metarule applications required for analysis. It would therefore seem interesting to examine how various measures of natural complexity, such as reading time, comprehension, and acceptability judgements, correlate with complexity according to the grammar. This apparently effectively amounts to the derivational theory of complexity revisited.

Pursuing the point a little further, the number of applications of metarules is obviously not going to be the *only* factor contributing to complexity. However in the case of an ambiguous expression, where most other factors will be constant across the readings, it would be expected that readings requiring more applications of metarules will be less dominant. Consider the following:[3]

(1) A review of a book e_i just came out [which Chomsky wrote]$_i$

The preferred reading is the one where the right extraposed relative clause modifies *review* rather than *book*. Accordingly, analysis to yield the second meaning requires more metarule application. The least expensive analyses of the subjects in the two cases are as follows:

(2)

```
   a     review               of         a     book
----    -------            ------      ----    ----
NP/N          N            N\N/NP      NP/N      N
        -------r_f                     -------f
         N/(N\N)                          NP
                            ----------f
                                 N\N
          ------------------------------Mb
                 N/(N\N)
----------------------------Rf
       NP/(N\N)
```

(3)

```
   a     review                  of          a             book
----    -------              ------      ----              ----
NP/N          N              N\N/NP     NP/N                 N
                                                      -------r_f
                                                      N/(N\N)
                                         --------------------------Rf
                                             NP/(N\N)
                             -------------------------------Rf
                                 N\N/(N\N)
              ------------------------------------Rb
                       N/(N\N)
------------------------------------Rf
          NP/(N\N)
```

By way of another example, consider (4), in which the fronted adverbial seems able to modify either of the verbs.

(4) the day [on which]$_i$ I said that John swims e_i

In the dominant reading the fronted adverbial modifies the "saying", in the subordinate one, it modifies the "swimming". Accordingly, analysis of the clause in the former case is less

---

[3] I am grateful to Elisabet Engdahl for drawing my attention to this example in connection with the present discussion.

expensive than in the latter case:

```
(5)     I     said        that  John  swims
        --   -------      ----  ----  -----
        NP   S\NP/SP      SP/S   NP   S\NP
             --------Fr_f        --------b
             S\NP/ADV/SP            S
                          ----------f
                                SP
             -----------------f
                  S\NP/ADV
        ---------------Rb
             S/ADV
```

```
(6)     I    said      that    John   swims
        -   -------    ----    ----   -----
        NP  S\NP/SP    SP/S     NP    S\NP
        ----------Rb                  ------r_f
           S/SP                       S\NP/ADV
           -------------Rf    ------------Rb
                S/S                S/ADV
                --------------------Rf
                        S/ADV
```

This line of thought requires much more attention, particularly in relation to some theory of processing strategy; my main point here has been to emphasise how a system with metarules such as the current one offers some interesting possibilities for explanation of reading dominance and acceptability gradation generally in terms of features of the competence grammar, namely expense of analysis in terms of metarule applications.

2.2. *Semantics*

The grammar that has emerged has been formulated in terms of combinatory logic. The feature which I want to emphasise here is that syntactic structures are isomorphic to terms of a directionally typed combinatory logic. Thus under the simple applicative analysis (7), *John met Mary* is to have the combinatory logic translation (8).

(7) $[_{b}\ \mathrm{John}_{NP}\ [_{f}\ \mathrm{met}_{S\backslash NP/NP}\ \mathrm{Mary}_{NP}]_{S\backslash NP}]_{S}$

(8) b John′ (f met′ Mary′)

I will illustrate the way in which the binary grammar for English induces a combinatory logic which is very close to a language of syntactic structures. There are a set of constants, corresponding to the meanings of words, for each category. Then a set of combinators is defined as follows:

(9) a. **f** is a combinator of type X/Y + Y => X
b. **b** is a combinator of type Y + X\Y => X
c. If ϕ is a combinator of type X + Y => Z, then
Rϕ is a combinator of type X + Y/W => Z/W
Lϕ is a combinator of type X\W + Y => Z\W
Mϕ is a combinator of type X/W + Y => Z/W
Pϕ is a combinator of type X/W + Y/W => Z/W

Then the language of combinatory logic (CL) terms is defined thus:

(10) a. If α is a constant of category X, then
α is a CL-term of category X
b. If ϕ is a combinator of type X + Y => Z, and
α is a CL-term of category X, and
β is a CL-term of category Y, then
ϕ α β is a CL-term of category Z

It is tempting therefore to hypothesise that there is a level of meaning representation isomorphic to syntactic structure. Such a situation indicates a very close relation between thought, language, and speech. A single idea may have different representations corresponding to different analyses, but the meaning representations preserve the linguistic ordering of the concepts. Thus *John loves some woman* can have the (equivalent) representations (11) (12).

(11) a. **b** John′ (**f** loves′ (**f** some′ woman′))
b. **f** (**R** **b** John′ loves′) (**f** some′ woman′)

(12) loves′ (some′ woman′) John′

The existing grammar makes no reference to anaphora, quantification, or the scopes of semantic operators in general. The implication is that at the level of combinatory logic representation suggested here, such factors are undetermined. This appears consistent with the fact that people are able to ‘comprehend’ expressions without commital, for example, to quantifier scopes, and that identification of the logic of different readings seems to require a mode of thought over and above that required to affirm that an expression is indeed meaningful. Accounts of quantification, coordination, scope, and so on are presented in e.g. van Benthem (1986), Partee and Rooth (1983), and Hendriks (1987), which are based systems of minimal type assignment, plus type-shifting. But these accounts employ a non-directional type system with type-driven translation, i.e. functions apply to each other freely so long as they are of the right type to do so. Following the direction of Groenendijk and Stokhof (1987, pp24-28), and also Kang (1988, p28), we can try to integrate this approach with the directional syntax here, having the syntax determine a rigid function-argument structure, induced on the basis of syntactic categories, and then having type-shifting operations applying within the space so fixed, subject to the condition that the resulting types adhere to the function-argument structure.

To illustrate the idea, consider (13).

(13) John loves someone

The words have the following lexical category and minimal type assignments:

(14) John := j_e: NP
loves := $\lambda x \lambda y[\mathrm{LOVE}(y,x)]_{e \rightarrow e \rightarrow t}$: S\NP/NP
someone := $\lambda P \exists x[P\ x]_{(e \rightarrow t) \rightarrow t}$: NP

The function-argument structure determined on the basis of the syntactic categories is that expressed in e.g. (15) which is equivalent to (16).

(15) **b** John′ (**f** loves′ someone′)

(16) loves′ someone′ John′

The essence of the type-shifting approach is that logical constants do not have unique types associated with them, but a family, derived from a basic type, perhaps the 'minimal' (lowest) one, by type-shifting rules. The combinatory logic now becomes lexically ambiguous in that its constants are associated with several related meanings; only some combinations of types will match up according to the functionality dictated by the syntactic derivation. The minimal type of *loves′* is of the wrong kind to apply to that of *someone′* even though it must. One type-shifting rule therefore may be as follows:

(17) For S\NP/NP,
$\alpha_{e \rightarrow A \rightarrow t} \rightarrow \lambda V \lambda Y[V\ (\lambda a[\alpha\ a\ Y])]_{((e \rightarrow t) \rightarrow t) \rightarrow A \rightarrow t}$

The rule assigns a second type to *loves′* thus:

(18) loves′ =

$$\lambda x \lambda y[\mathrm{LOVE}(y,x)]_{e \rightarrow e \rightarrow t} \rightarrow$$
$$\lambda V \lambda Y[V\ (\lambda a[\lambda x \lambda y[\mathrm{LOVE}(y,x)]\ a\ Y])]_{((e \rightarrow t) \rightarrow t) \rightarrow e \rightarrow t} =$$
$$\lambda V \lambda Y[V\ (\lambda a[\mathrm{LOVE}(Y,a)])]_{((e \rightarrow t) \rightarrow t) \rightarrow e \rightarrow t}$$

So *loves′ someone′ John′* can evaluate as follows:

(19) someone′ =
$\lambda P \exists x[P\ x]$
loves′ someone′ =
$\lambda Y \exists x[\mathrm{LOVE}(Y,x)]$
loves′ someone′ John′ =
$\exists x[\mathrm{LOVE}(j,x)]$

For *everyone loves someone* the function-argument structure is (20).

(20) loves′ someone′ everyone′

Another type-shifting rule, such as (21), is required to assign *loves′* a type capable of applying to the subject.

(21) For S\NP/NP,
$\alpha_{B \rightarrow e \rightarrow t} \rightarrow \lambda X \lambda U[U\ (\lambda b[\alpha\ X\ b])]_{B \rightarrow ((e \rightarrow t) \rightarrow t) \rightarrow t}$

Thus one reading is obtained as follows:

(22) loves′ =
$\lambda x \lambda y[\mathrm{LOVE}(y,x)] \rightarrow$
$\lambda V \lambda Y[V\ (\lambda a[\mathrm{LOVE}(Y,a)])] \rightarrow$
$\lambda X \lambda U[U\ (\lambda b[\lambda V \lambda Y[V\ (\lambda a[\mathrm{LOVE}(Y,a)])]\ X\ b])] =$
$\lambda X \lambda U[U\ (\lambda b[X\ (\lambda a[\mathrm{LOVE}(b,a)])])]$
loves′ someone′ =
$\lambda U[U\ (\lambda b \exists x[\mathrm{LOVE}(b,x)])]$
everyone′ =
$\lambda P \forall y[P\ y]$
loves′ someone′ everyone′ =
$\forall y \exists x[\mathrm{LOVE}(y,x)]$

The wide scope for object reading is obtained by applying the type shifting rules the other way around:

(23) loves′ =

$\lambda x \lambda y[\mathrm{LOVE}(y,x)] \rightarrow$

$\lambda X \lambda U[U\ (\lambda b[\mathrm{LOVE}(b,X)])] \rightarrow$

$\lambda V \lambda Y[V\ (\lambda a[Y\ (\lambda b[\mathrm{LOVE}(b,a)])])]$

loves′ someone′ =

$\lambda Y \exists x[Y\ (\lambda b[\mathrm{LOVE}(b,x)])]$

loves′ someone′ everyone′ =

$\exists x \forall y[\mathrm{LOVE}(y,x)]$

This illustrates the way in which the semantics dictated by a rigid syntax, and a flexible type system, might be integrated. As with the syntactic complexity line of investigation I have only begun to sketch the possibilities, but with these two sketches the thesis is concluded.

Appendix A

Complexity of Categorial Grammar with Unification

In Section 1 I show that the problem of determining whether a string is generated by a categorial grammar with unification is classified by computational complexity theory as being NP-hard. In Section 2 I discuss the result, particularly in relation to the semantics of categorial grammar with unification.

1. Computational Complexity

Computational complexity theory studies the intrinsic difficulty of problems (e.g. "is a given string recognised by a given grammar") in terms of the resources (e.g. time and space) required for the computation of their solution. The methodology identifies a class 'NP-hard' of problems for which the difficulty of solution is such that they are regarded as computationally intractable. Research has shown that a wide variety of linguistic theories are intractable in this sense (LFG: Berwick 1982; FUG: Ritchie 1986; GPSG: Ristad 1986, Ritchie 1987; Two-Level Morphology: Barton 1986).[1] I show here that the universal recognition problem for categorial grammar augmented with unification, in the manner described in Section 1.2 of Chapter VI, is NP-hard (probably NP-complete).

In Section 1.1 I outline relevant features of complexity theory[2] and in Section 1.2 I present the proof that recognition for categorial grammar with unification is NP-hard.

1.1. Some Features of Computational Complexity Theory

Suppose we have an algorithm the execution of which when supplied with some instance of a problem (encoded by a string) yields the solution to that problem. For example we might have a *Quicksort* algorithm or a *Bubblesort* algorithm which takes a list of numbers and sorts them, or we might have an *Early* algorithm which takes a context free grammar and a string and determines whether the string is recognised by the grammar. Then we can speak of that function which gives the time (number of steps) to compute the solution for each input. Abstracting over inputs of the same size, we can speak of that function which gives

---

[1] The Barton, Berwick, and Ristad results are reproduced in Barton, Berwick, and Ristad (1987).

[2] See e.g. Garey and Johnson (1979) for a full introduction.

the longest time to compute the solution for each input size. We can further abstract to the asymptotic limit and away from non-dominant terms. We say that $f(n)$ is *of order* $g(n)$, $f(n) = O(g(n))$, if and only if there exist positive constants c and k such that for all n greater than or equal to k, $f(n)$ is greater than or equal to $c.g(n)$; so $3n^2 - 100n$ and $2n^2 + 4n - 1$ are both $O(n^2)$. The complexity of an algorithm is usually expressed by saying it is, e.g. $O(n^2)$ or $O(2^n)$ etc.

In addition to speaking of the complexity of an *algorithm*, we can speak of the complexity of a *problem*, meaning a lower bound on the complexity of the best possible algorithm for the problem. For example the problem might be to sort a list of integers, or to determine whether a context free grammar generates a string, or to determine whether there is some assignment of values to the variables of a Boolean formula which makes the formula true. This is a measure of the *intrinsic* difficulty of a problem, and is independent of whether good algorithms are known or not. Attention here will be restricted to *decision problems*, i.e. problems the solution of which is either "Yes" or "No". To classify problems in this manner we further abstract away from the details of complexity functions; in particular we distinguish *polynomial* functions (those expressible by a polynomial term), from more-than-polynomial functions, which include exponential and higher functions.

In order to classify problems independently of particular algorithms it is necessary to define problem complexity by reference to machines capable of executing whole classes of algorithms. Problems are classified according to their best possible solution on different kinds of idealized machines.

The first idealization of a computing machine was the Turing machine. A Turing machine operates over an infinite tape which it shuttles up and down, writing, moving and changing state according to its current state and the symbol it reads. A specification for a Turing machine is a finite function from *<current-state, current-symbol>* pairs into *<symbol-written, direction-moved, new-state>* triples. At each step the machine writes a symbol, moves one cell left or right, and adopts a new state, according to the symbol it has just read and the state is was just in. It is universally believed that there exists an algorithm for a problem if and only if there exists a Turing machine which can solve the problem: many other plausible models of computation have been shown to be equivalent, and these models have been shown to be able to simulate one another in polynomial time.

All these models share the characteristic of the Turing machine that at any one moment, the subsequent course of action is completely determined. This follows from the definition of a Turing machine as a *function* from *<current-state, current-symbol>* pairs into

<symbol-written, direction-moved, new-state> triples, because a function by definition has only one value for each argument. Such machines are called *deterministic.*

Presumably real machines are deterministic. However we can define a *non-deterministic* Turing machine which is not a finite *function* from pairs into triples, but a finite *relation* between pairs and triples, or equivalently a function from pairs into finite sets of triples. Then at each step a machine may have several recourses to action, any one of which may eventually lead to the solution. The operation of such a hypothetical machine can be envisaged in a number of ways. We might imagine that at each decision point the machine replicates itself so that one machine can follow each course of action. Alternatively we might imagine that at each decision point the machine correctly guesses an optimal path to take.

Clearly non-deterministic machines are at least as powerful as deterministic ones because the latter constitute that proper subclass of the former in which there is just one way to proceed at each step. Complexity theory classifies problems according to whether they can be solved in polynomial time on deterministic and non-deterministic machines. The class of problems which are solvable in polynomial time on deterministic machines is called *P*; this class is regarded as exhausting the tractable or computationally feasible problems. The class of problems solvable in polynomial time on non-deterministic machines is called *NP*.

A problem *p* is said to be *NP-hard* if and only if every problem in NP can be *reduced* to *p*; a problem *q* can be reduced to a problem *p* if and only if there exists an answer-preserving deterministic polynomial time transformation from instances of *q* to instances of *p*. The class *NP-complete* is defined to be that class of problems which are in both NP and NP-hard. It thus constitutes the 'easiest' problems in NP-hard, and the 'hardest' problems in NP. Of the many hundreds of problems in NP-complete, deterministic polynomial time algorithms are not known for any. In the light of this it is very unlikely that $P = NP$: by the definition of NP-hard (of which NP-complete is a subclass), all problems in NP (of which NP-complete is also a subclass) are polynomially reducible to every NP-complete problem. The composition of two polynomial functions is itself polynomial, so it follows that if any NP-complete problem can solved in polynomial time, all problems in NP-complete and all problems in NP could be, and it would be the case that $P = NP$. Since extensive efforts have not revealed polynomial time algorithms for any of the many problems in NP-complete, it seems most unlikely that they all do have such algorithms. So it is next to certain that $P \neq NP$, so that none of the problems in NP-hard are in

P and they all fall outside the range of tractable or computationally feasible problems. However $P \neq NP$ has never been proved, and its proof is one of the foremost open problems in computer science.

An example of an NP-complete problem is conjunctive normal form Boolean formula satisfiability (SAT). A Boolean formula such as (1) is said to be in conjunctive normal form.

(1) $(x \vee y) \;\&\; (\sim x \vee y \vee \sim z \vee z) \;\&\; z$

The problem is to determine whether there exists some assignment of truth values to the variables such that the whole formula is true (satisfied). For such a formula to be satisfiable there must be a true literal in each conjunct. Yet choosing values making one conjunct true effects choices for other conjuncts, because variables must have the same truth values wherever they occur. The difficulty of the problem stems from its essential non-divisability.

As has been said, the significance of a demonstration that a problem is NP-hard is that it is then as good as certain that the problem is not in P and therefore not tractable. Once we have a known NP-hard problem (we usually choose an NP-complete problem since these are the easiest and most likely to reduce), a new problem is proved to be NP-hard if we can provide a reduction from the known NP-hard problem to the new problem. By the definition of NP-hard, every problem in NP will be reducible to the known NP-hard problem, and by the transitivity of polynomial composition, it will have been shown that every problem in NP is thereby reducible to the new problem, proving that the new problem is itself NP-hard. (Such a reduction forms the content of the next section.) If it is additionally shown that the problem has a polynomial time non-deterministic algorithm, the problem is shown to be in NP-complete. However, the reduction technique requires an existing NP-hard problem, and proving that some problem is NP-hard initially is more difficult. By the definition of NP, a problem p is in NP if and only if there exists a nondeterministic Turing machine M such that M, supplied with an instance x of p, yields the solution in polynomial time. Cook (1971) showed that for any such M, there exists a deterministic polynomial time answer-preserving transformation of x to an instance x' of SAT. The transformation exploits the fact that since M finds the solution in time some polynomial function f of $|x|$ ($|x|$ means the size of x, e.g. as a number of symbols), the solution-finding computation sequence involves not more than $f(|x|)$ states and $f(|x|)$ tape cells. A number of Boolean variables not larger than a polynomial function of $|x|$ is needed to encode machine configuration at each step. Negated variables enforce correct simulation; the nondeterminism is mirrored in the possibility that variables are assigned either true or false.

1.2. Reduction of 3SAT to Categorial Grammar with Unification Recognition

The problem 3SAT is just like SAT, except that there are exactly three literals in each conjunct of the formula. It too is NP-complete; in this section I show how it can be reduced to the universal recognition problem for categorial grammar with unification, as described in section 1 of Appendix A, where the universal recognition problem is:[3]

(2) Given a specification of a grammar G and a string x, is x in the language generated by G?

This shows that the universal recognition problem for categorial grammar with unification is NP-hard; I go on to show that it is also probably NP-complete.

Recall that in *pure categorial grammar* the set of categories is defined in terms of a set of basic categories as follows:

(3) If X is a basic category
then X is a category.
If X,Y are categories
then X/Y,X\Y are categories.

The interpretation of the categories is provided by the following rules:

(4) X/Y + Y => X
Y + X\Y => X

A set of basic categories plus a set of basic expressions and a lexical assignment of basic expressions to categories completes the definition of a grammar. Thus a grammar for a slightly unconventional notation for tertiary conjunctive normal form Boolean expressions is provided by the set $\{L, S\}$ of basic categories, the set $\{V, \&, x, y, z, \ldots, {}^{\sim}x, {}^{\sim}y, {}^{\sim}z, \ldots\}$ of basic expressions, and the following lexical assignments:[4]

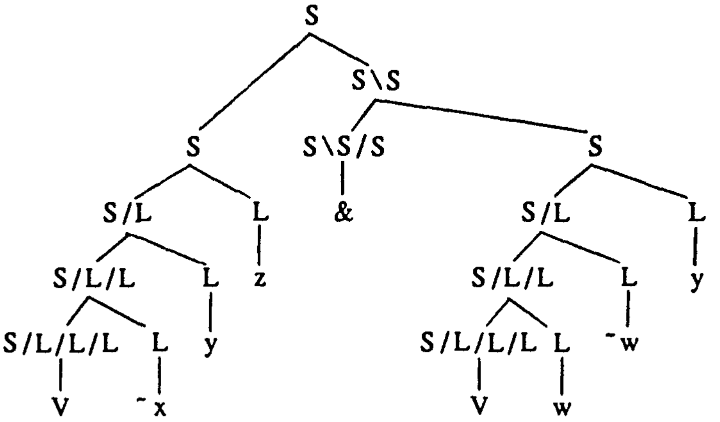
---

[3] The universal recognition problem is to be contrasted with the fixed language recognition problem: given a string x, is x in some independently specified set of strings? This problem, in ignoring grammar size, constitutes a less appropriate mode of analysis, see e.g. Barton, Berwick and Ristad (1987, p27).

(5) V := S/L/L/L
& := S\S/S
x, y, z := L
$^-$x, $^-$y, $^-$z := L

For example $V\,{}^-x\;y\;z\;\&\;V\,w\,{}^-w\;y$ is analysed:[5]

(6)



In CG with unification the basic category symbols are generalised into feature structures. In the general case a feature structure might be regarded as a directed acyclic graph in which the arcs are labeled with feature names (attributes) and the leaves are labeled with atomic values. By way of illustration of the augmentation, the rules become

(7) $\alpha/\beta + \beta' => \alpha'$
$\beta' + \alpha\backslash\beta => \beta'$

This means the same as before except that α' is α after β' has been unified into β. For example unifying (8) into the structure right of the slash in (9) leaves the structure left of the slash (10).

(8)
$$\begin{bmatrix} \mathrm{CAT} & \mathrm{S} \\ \mathrm{G} & \begin{bmatrix} \mathrm{x} & 0 \\ \mathrm{z} & 1 \end{bmatrix} \end{bmatrix}$$

(9)
$$\begin{bmatrix} \mathrm{CAT} & \mathrm{S} \\ \mathrm{G} & \mathrm{X}\begin{bmatrix} \mathrm{x} & 0 \\ \mathrm{y} & 1 \end{bmatrix} \end{bmatrix} \Bigg/ \begin{bmatrix} \mathrm{CAT} & \mathrm{S} \\ \mathrm{G} & \mathrm{X} \end{bmatrix}$$

---

[4] As before a left-associativity convention is assumed here so that, e.g. *S/L/L/L* is understood to be structured *((S/L)/L)/L*.

[5] Note that formulae with larger numbers of conjuncts have many structural analyses under the grammar. This is not important, and corresponds to the associativity of propositional conjunction: *(A & B) & C = A & (B & C)* etc.

(10)
$$\begin{bmatrix} \text{CAT} & \text{S} \\ \text{G} & \text{X}\begin{bmatrix} \text{x} & 0 \\ \text{y} & 1 \\ \text{z} & 1 \end{bmatrix} \end{bmatrix}$$

I will now show any instance of 3SAT can be reduced to an instance of the recognition problem for categorial grammar with unification. Note that a formula is satisfiable if and only if there is some value assignment such that at least one literal in each conjunct has truth value 1. A positive literal will have truth value 1 (0) if the assignment to its variable is 1 (0); a negative literal will have truth value 1 (0) if the assignment to its variable is 0 (1). In the grammar a lexical category for each literal will have a value assignment feature name *G* whose value is a feature specification consisting of its variable and the variable's chosen truth value. It will also have a truth value feature name *TVAL* the value of which will be 1 or 0 according to the value of literal given the value assignment to the variable. The lexical categories for *V* will combine with three literals if and only if at least one has *TVAL* 1 (there are seven possibilities), and will unify their value assignments, so that combination fails if there is value assignment conflict. Similarly, the lexical category for *&* unifies all value assignments so that there is generation provided there is no clash.

The string input to 3SAT is trivially transformed into the notation defined by the grammar given earlier to give the string part of the two parameter CG with unification problem; it remains to construct the grammar part. This will always contain the following lexical entries:

(11) V :=
[CAT S, G X]/[CAT L, TVAL 0, G X]/[CAT L, TVAL 0, G X]/[CAT L,
TVAL 1, G X],
[CAT S, G X]/[CAT L, TVAL 0, G X]/[CAT L, TVAL 1, G X]/[CAT L,
TVAL 0, G X],
[CAT S, G X]/[CAT L, TVAL 0, G X]/[CAT L, TVAL 1, G X]/[CAT L,
TVAL 1, G X],
[CAT S, G X]/[CAT L, TVAL 1, G X]/[CAT L, TVAL 0, G X]/[CAT L,
TVAL 0, G X],
[CAT S, G X]/[CAT L, TVAL 1, G X]/[CAT L, TVAL 0, G X]/[CAT L,
TVAL 1, G X],
[CAT S, G X]/[CAT L, TVAL 1, G X]/[CAT L, TVAL 1, G X]/[CAT L,
TVAL 0, G X],
[CAT S, G X]/[CAT L, TVAL 1, G X]/[CAT L, TVAL 1, G X]/[CAT L,
TVAL 1, G X]

Note that at least one literal must be true, and that all value assignments are unified. The value assignments are also unified in:

(12) & := [CAT S, G X]\[CAT S, G X]/[CAT S, G X]

Finally for each positive literal *v* and negative literal ~*v* in the problem instance, the following lexical entries are added:

(13) v := [CAT L, TVAL 0, G [V 0]], [CAT L, TVAL 1, G [V 1]]
~v := [CAT L, TVAL 1, G [V 0]], [CAT L, TVAL 0, G [V 1]]

The feature *TVAL* has value according to the value assignment true or false to the variable. The alteration in formula notation can be performed in linear time and the construction of the invariant lexical entries for *V* and *&* will take a constant time. Then we need a number of lexical entries which is twice the number of literals in the formula, so the whole transformation will take place in linear time, and is thus certainly polynomial.

This NP-hardness result carries over to augmentation with *term* unification. There will be reserved in all signs argument positions for the truth values *A*, *B*, *C*, ... of each variable occurring in the 3SAT instance, say in alphabetic order. Then the lexical categories will be

(14) V :=
S(A,B,C,...)/L(0,A,B,C,...)/L(0,A,B,C,...)/L(1,A,B,C,...)
S(A,B,C,...)/L(0,A,B,C,...)/L(1,A,B,C,...)/L(0,A,B,C,...)
etc.

(15) & := S(A,B,C,...)\S(A,B,C,...)/S(A,B,C,...)

(16) v := L(1,A,B,...,1,...), L(0,A,B,...,0,...)
(for each v)
˜v := L(1,A,B,...,0,...), L(0,A,B,...,1,...)
(for each ˜v)

Having counted and ordered the *n* literals in the formula, we need to construct seven lexical categories for *V*, each of size *O(n)*, and one lexical entry for *&*, of size *O(n)*. Then for each of the *n* literals, we need two lexical entries, each of size *O(n)*. None of this involves more than a polynomial amount of work in the size of the formula, so the whole transformation is polynomial.

The above arguments show that universal recognition for categorial grammar with unification is NP-hard. To show that it is NP-complete it is necessary to additionally demonstrate that it is in NP, i.e. that it has a polynomial time non-deterministic algorithm. I will sketch that this is probably so, assuming that unification is a constant time operation. Suppose we are given a string, and a grammar which is a set of lexical assignments (the rules of forward and backward application are invariant across all grammars). First, for each word in the string we non-deterministically choose which of its finite number of lexically assigned signs we want. This will take a total time which is linear in the length of the string. Then for the resulting sign sequence, we non-deterministically choose which of the adjacent sign pairs we want to try to reduce by forward or backward application. The result has length decremented by one. So repeating this process, recognition will take a total time linear in the length of the string. Thus the overall nondeterministic recognition process is linear time, and the problem is in NP.

2. *Discussion: Semantics of Categorial Grammar with Unification*

In this section I discuss the semantics of unification by analogy with the λ-calculus. I suggest that the complexity result may be circumvented if unification is employed just to the extent that its use amounts to functional abstraction.

The basic rule of categorial grammar was (17).[6]

(17) $X/Y: x \;+\; Y: y \;\Rightarrow\; X: x\, y$

This states that an expression of category *X/Y* can combine with an expression of category *Y* to form one of category *X*, and the meaning of the result is given by applying that of the former subexpression to that of the latter. In CG with Unification the basic rule will be something like (18).

(18) $\alpha/\beta \;+\; \beta' \;\Rightarrow\; \alpha'$
where α' is the result of applying to α the most general unifier of β and β'

Here, α and β are expressions of the sign language. To keep the association with (17) the meaning of α' must be the result of applying that of α/β to that of β', so that the sign language must have a set-theoretic semantics where the following holds:

(19) *μ-Reduction*
$[[\alpha/\beta \;\; \beta']] \;=\; [[\alpha']]$
where α' is the result of applying to α the most general unifier of β and β'

Thus the '/' in the sign language is a functional abstraction operator rather like the 'λ' in the lambda-calculus. In particular, note that β-reduction is a special case of μ-reduction, one in which one of the terms to be unified is a variable, so that the relevant unifier is trivially the mapping from the variable to the other term:[7]

(20) *β-Reduction*
$[[\lambda v\alpha \;\; \beta]] \;=\; [[\alpha']]$
where α' is the result of applying to α the substitution $\{v{=}\beta\}$

The reason why β-reduction is valid in the lambda-calculus is that a variable which, by the semantics of λ-abstraction and application is assigned *[[β]]*, is replaced by β which, of course, denotes the same value. Since the semantics is strictly compositional, the operation preserves meaning.

---

[6] I gloss over slash directionality here.

[7] The complications of scope are mentioned below.

In the lambda-calculus there is never any question of instantiating an abstractor -- this would just not make semantic sense. So for example, $\lambda x[\lambda x[x]]\ j$ cannot be reduced to $\lambda j[j]$ by a naive employment of β-reduction. Either the β-reduction substitution must be made sensitive to scope, or else the calculus should be so designed that fresh variables are used on all occasions and scopes are never blocked because there is never the relevant re-occurrence of variables. Now if it is right that the '/' of a CG with unification sign language is a functional abstraction operator we should expect a similar state of affairs. The abstractors are now more complex, in general being terms rather than just single variables, but if the analogy with λ-abstraction holds, it should still be anomalous to instantiate abstractor variables. For example $X/X/X$ should not μ-reduce with j to give j/j because this amounts to instantiating an abstractor. Assuming we wish to avoid assigning variables scope, the requirement for semantic coherence should be along the lines that no variable should appear in more than one abstractor term. Consequently variables are only ever matched once. The simulation of 3SAT satisfaction relied crucially on the ability to successively unify into the same position: the encodings of value assignments were repeatedly matched to check compatibility. The suggestion here then is that augmenting CG with unification in a manner paying regard to the semantics of '/' as a functional abstractor may remove the source of the computational complexity result.

Appendix B

Parser Listing and Illustrative Log

This appendix contains a complete listing of a program for parsing a categorial grammar with binary metarules, along with an illustrative log of a terminal session. The program is written in Quintus Prolog and was run compiled on a Sun 3.

1. Parser Listing

```
% A percolation parser for a binary metarule CG for English; uses
% path equivalence check

:- op(400,yfx,/).
:- op(400,yfx,\).

:- op(300,yfx,‘).

:- op(500,xfx,:=).

:- op(400,xfx,equ).

:- op(450,xfx,:).

% Top level procedure prs(+String) parses String anew

prs(String) :-
        prs1(String,0).

% prs1(+String,+Pos) parses String from the current chart, and position Pos

prs1([Word|String],Pos) :-
        Word := Trans:LexCat,
        incorp(Pos,LexCat,s(Pos),Trans);
        prs1(String,s(Pos)).

% incorp(+MPos,+RCat,+RPos,+RTrans) incorporates into the chart between
% positions MPos and RPos an edge of category RCat and translation
% RTrans, searches for incoming edges, and calls incorp2/6

incorp(MPos,RCat,RPos,RTrans) :-
        assert(edge(MPos,RCat,RPos,RTrans)), !,
        edge(LPos,LCat,MPos,LTrans),
        incorp2(LPos,LCat,RCat,RPos,LTrans,RTrans).

% incorp2(+LPos,+LCat,+RCat,+RPos,+LTrans,+RTrans) tries to reduce edges of
```

```
% category LCat and RCat and translation LTrans and RTrans, between
% LPos and RPos, and causes precipitation of consequential edges
% by calling incorp/4 on the results. In the case that LCat is
% a coordinator, a left-hand conjunct is sought

incorp2(LPos,LCat,RCat,RPos,LTrans,RTrans) :-
        rule(LCat,RCat,MCat,G),
        \+ unnec_l(G,LTrans),
        \+ unnec_r(G,RTrans),
        incorp(LPos,MCat,RPos,G‘LTrans‘RTrans).

incorp2(Pos,crd,Cat,RPos,C,RTrans) :-
        edge(LPos,Cat,Pos,LTrans),
        coord(Cat,S),
        incorp(LPos,Cat,RPos,S‘LTrans‘C‘RTrans).

% coord(+Cat,-Sem) means that there is a rule with semantics Sem,
% coordinating expressions of category Cat

coord(s,c).

coord(X/_,'Af'‘S) :-
        coord(X,S).

coord(X\_,'Ab'‘S) :-
        coord(X,S).

% rule(X,Y,Z,G) means that G: X + Y => Z is a rule

rule(X/Y,Y,X,f).
rule(Y,X\Y,X,b).

rule(X,Y/Z,V/Z,'R'‘G) :-
        rule(X,Y,V,G).

rule(X\Z,Y,V\Z,'L'‘G) :-
        rule(X,Y,V,G).

rule(X/Z,Y,V/Z,'M'‘G) :-
        member(Z,[np,sp,n\n,s\np\(s\np),n/n]),
        rule(X,Y,V,G).

rule(X/np,Y/np,V/np,'P'‘G) :-
        rule(X,Y,V,G).

member(X,[X|_]).
member(X,[_|T]) :-
        member(X,T).

% unnec_l(G,H‘_‘_) means that applying G with a left-hand daughter derived by
% H is unnecessary

unnec_l(G,H‘_‘_) :-
```

```
        l(G,H) equ l(G1,H1),
        less_than([G,H],[G1,H1]).

unnec_l(G,H‘_‘_) :-
        l(G,H) equ r(G1,H1),
        less_than([G,H],[G1,H1]).

% unnec_r(G,H‘_‘_) means that applying G with a right-hand daughter derived by
% H is unnecessary

unnec_r(G,H‘_‘_) :-
        r(G,H) equ r(G1,H1),
        less_than([G,H],[G1,H1]).

unnec_r(G,H‘_‘_) :-
        l(G1,H1) equ r(G,H),
        less_than_or_eq([G,H],[G1,H1]).

% Path1 equ Path2 means that Path1 and Path2 are equivalent

% Four symmetric, axiomatic left-left equivalences

l(f,H) equ l(f,H).
l(b,H) equ l(b,H).

l(G,f) equ l(G,f).
l(G,b) equ l(G,b).

% Eight symmetric, outward instantiation left-left equivalence rules

l(’R’‘G,H) equ l(’R’‘G1,H1) :-
        l(G,H) equ l(G1,H1).

l(’L’‘G,’L’‘H) equ l(’L’‘G1,’L’‘H1) :-
        l(G,H) equ l(G1,H1).

l(’M’‘G,’R’‘H) equ l(’M’‘G1,’R’‘H1) :-
        l(G,H) equ l(G1,H1).

l(’M’‘G,’M’‘H) equ l(’M’‘G1,’M’‘H1) :-
        l(G,H) equ l(G1,H1).

l(’P’‘G,’R’‘H) equ l(’P’‘G1,’R’‘H1) :-
        l(G,H) equ l(G1,H1).

l(’P’‘G,’M’‘H) equ l(’P’‘G1,’M’‘H1) :-
        l(G,H) equ l(G1,H1).

l(’M’‘G,’P’‘H) equ l(’M’‘G1,’P’‘H1) :-
        l(G,H) equ l(G1,H1).

l(’P’‘G,’P’‘H) equ l(’P’‘G1,’P’‘H1) :-
        l(G,H) equ l(G1,H1).
```

```
% Three asymmetric, inward instantiation left-left equivalence rules

l(f,'R''('L''H)) equ l('L''G1,'L''H1) :-
        l(f,'R''H) equ l(G1,H1).
l('L''G1,'L''H1) equ l(f,'R''('L''H)) :-
        l(G1,H1) equ l(f,'R''H).

l(f,'R''('M''H)) equ l('M''G1,'M''H1) :-
        l(f,'R''H) equ l(G1,H1).
l('M''G1,'M''H1) equ l(f,'R''('M''H)) :-
        l(G1,H1) equ l(f,'R''H).

l(f,'M''('R''H)) equ l('M''G1,'R''H1) :-
        l(f,'M''H) equ l(G1,H1).
l('M''G1,'R''H1) equ l(f,'M''('R''H)) :-
        l(G1,H1) equ l(f,'M''H).

% Four symmetric, axiomatic right-right equivalences

r(f,H) equ r(f,H).
r(b,H) equ r(b,H).

r(G,f) equ r(G,f).
r(G,b) equ r(G,b).

% Eight symmetric, outward instantiation right-right equivalence rules

r('R''G,'R''H) equ r('R''G1,'R''H1) :-
        r(G,H) equ r(G1,H1).

r('L''G,H) equ r('L''G1,H1) :-
        r(G,H) equ r(G1,H1).

r('R''G,'M''H) equ r('R''G1,'M''H1) :-
        r(G,H) equ r(G1,H1).

r('M''G,H) equ r('M''G1,H1) :-
        r(G,H) equ r(G1,H1).

r('R''G,'P''H) equ r('R''G1,'P''H1) :-
        r(G,H) equ r(G1,H1).

r('P''G,'R''H) equ r('P''G1,'R''H1) :-
        r(G,H) equ r(G1,H1).

r('P''G,'M''H) equ r('P''G1,'M''H1) :-
        r(G,H) equ r(G1,H1).

r('P''G,'P''H) equ r('P''G1,'P''H1) :-
        r(G,H) equ r(G1,H1).

% One asymmetric, inward instantiation right-right equivalence rule
```

```
r(b,'L''('R''H)) equ r('R''G1,'R''H1) :-
        r(b,'L''H) equ r(G1,H1).
r('R''G1,'R''H1) equ r(b,'L''('R''H)) :-
        r(b,'L''H) equ r(G1,H1).

% Two asymmetric axiomatic left-right equivalences

l(G,b) equ r(b,'L''G).
r(b,'L''G) equ l(G,b).

l(f,'R''G) equ r(G,f).
r(G,f) equ l(f,'R''G).

% Eight asymmetric, outward instantiation left-right equivalence rules

l('R''G,H) equ r('R''G1,'R''H1) :-
        l(G,H) equ r(G1,H1).
r('R''G1,'R''H1) equ l('R''G,H) :-
        r(G1,H1) equ l(G,H).

l('L''G,'L''H) equ r('L''G1,H1) :-
        l(G,H) equ r(G1,H1).
r('L''G1,H1) equ l('L''G,'L''H) :-
        r(G1,H1) equ l(G,H).

l('M''G,'R''H) equ r('R''G1,'M''H1) :-
        l(G,H) equ r(G1,H1).
r('R''G1,'M''H1) equ l('M''G,'R''H) :-
        r(G1,H1) equ l(G,H).

l('M''G,'M''H) equ r('M''G1,H1) :-
        l(G,H) equ r(G1,H1).
r('M''G1,H1) equ l('M''G,'M''H) :-
        r(G1,H1) equ l(G,H).

l('P''G,'R''H) equ r('R''G1,'P''H1) :-
        l(G,H) equ r(G1,H1).
r('R''G1,'P''H1) equ l('P''G,'R''H) :-
        r(G1,H1) equ l(G,H).

l('P''G,'M''H) equ r('P''G1,'R''H1) :-
        l(G,H) equ r(G1,H1).
r('P''G1,'R''H1) equ l('P''G,'M''H) :-
        r(G1,H1) equ l(G,H).

l('M''G,'P''H) equ r('P''G1,'M''H1) :-
        l(G,H) equ r(G1,H1).
r('P''G1,'M''H1) equ l('M''G,'P''H) :-
        r(G1,H1) equ l(G,H).

l('P''G,'P''H) equ r('P''G1,'P''H1) :-
        l(G,H) equ r(G1,H1).
r('P''G1,'P''H1) equ l('P''G,'P''H) :-
```

```
        r(G1,H1) equ l(G,H).

% less_than(L,L1) means that the sum of the number of metarule
% applications in the list L1 of rules is less than that in the list L

less_than(L,L1) :-
        less_than_or_eq(L,[_‘f|L1]).

% less_than_or_eq(L,L1) means that the sum of the number of
% metarule applications in the list L1 of rules is less than or equal to that
% in the list L

less_than_or_eq(_,[]).

less_than_or_eq([_‘G|L],[_‘G1|L1]) :-
        less_than_or_eq([G|L],[G1|L1]).

less_than_or_eq(L,[R|L1]) :-
        primitive(R),
        less_than_or_eq(L,L1).

less_than_or_eq([R|L],L1) :-
        primitive(R),
        less_than_or_eq(L,L1).

primitive(f).
primitive(b).

a            := a           : np/n.
after        := after       : s\np\(s\np)/np.
and          := &           : crd.
bankrupt     := bankrupt    : n/n.
before       := before      : s\np\(s\np)/np.
bill         := b           : np.
book         := book        : n.
company      := company     : n.
damaged      := damaged     : s\np/np.
dearly       := dearly      : s\np\(s\np).
dislike      := dislike     : s\np/np.
dog          := dog         : n.
downstairs   := downstairs  : s\np\(s\np).
filed        := filed       : s\np/np.
give         := give        : s\np/np/np.
i            := i           : np.
inside       := inside      : n\n.
john         := j           : np.
large        := large       : n/n.
laughs       := laughs      : s\np.
leave        := leave       : s\np.
left         := left        : s\np.
like         := like        : s\np/np.
long         := long        : n/n.
loves        := loves       : s\np/np.
```

```
man         := man         : n.
married     := married     : s\np/np.
mary        := m           : np.
meet        := meet        : s\np/np.
met         := met         : s\np/np.
on          := on          : s\np\(s\np)/np.
or          := v           : crd.
outside     := outside     : n\n.
owns        := owns        : s\np/np.
paper       := paper       : n.
peacefully  := peacefully  : s\np\(s\np).
put         := put         : s\np/(s\np\(s\np))/np.
quickly     := quickly     : s\np\(s\np).
reading     := reading     : s\np/np.
red         := red         : n/n.
restlessly  := restlessly  : s\np\(s\np).
rumour      := rumour      : n/sp.
show        := show        : s\np/np/np.
sit         := sit         : s\np.
sleep       := sleep       : s\np.
speaking    := speaking    : s\np.
spread      := spread      : s\np.
stay        := stay        : s\np.
swims       := swims       : s\np.
sue         := s           : np.
table       := table       : n.
that        := that        : sp/s.
that        := that        : np/n.
that        := that        : n\n/(s\np).
that        := that        : n\n/(s/np).
the         := the         : np/n.
think       := think       : s\np/sp.
thinks      := thinks      : s\np/sp.
this        := this        : np/n.
today       := today       : s\np\(s\np).
tomorrow    := tomorrow    : s\np\(s\np).
upstairs    := upstairs    : s\np\(s\np).
war         := war         : n.
was         := was         : s\np/(n/n).
we          := we          : np.
which       := which       : n\n/(s/np).
who         := who         : n\n/(s/np).
whom        := who         : n\n/(s/np).
will        := will        : s\np/(s\np).
without     := without     : s\np\(s\np)/(s\np).
woman       := woman       : n.
yesterday   := yesterday   : s\np\(s\np).
you         := you         : np.

str(1,[we,leave]).
str(2,[we,like,mary]).
str(3,[we,think,that,you,leave]).
str(4,[we,leave,and,you,leave]).
```

```
str(5,[we,leave,or,you,leave]).
str(6,[we,leave,or,stay]).
str(7,[we,like,and,you,dislike,mary]).
str(8,[the,man,who,we,meet]).
str(9,[the,man,who,we,think,that,we,meet]).
str(10,[the,man,who,we,like,and,you,dislike]).
str(11,[we,show,and,you,give,john,the,book]).
str(12,[the,book,which,we,show,and,you,give,john]).
str(13,[the,man,who,we,like,mary,and,you,dislike]).
str(14,[you,sit,and,sleep]).
str(15,[you,sit,and,sleep,restlessly]).
str(16,[you,sit,and,sleep,restlessly,upstairs]).
str(17,[you,sleep,restlessly,upstairs,and,peacefully,downstairs]).
str(18,[the,dog,which,we,show,john]).
str(19,[the,man,who,we,think,that,left]).
str(20,[the,paper,which,you,filed,without,reading]).
str(21,[you,will,leave,tomorrow]).
str(22,[john,loves,mary,dearly]).
str(23,[we,put,on,the,table,a,large,red,book]).
str(24,[mary,thinks,that,the,man,thinks,that,you,will,leave,today]).
str(25,[the,man,inside,thinks,that,the,woman,outside,left,quickly,yesterday]).
str(26,[that,man,that,laughs,thinks,that,that,dog,that,sue,owns,swims]).
str(27,[a,rumour,spread,that,john,was,bankrupt]).
str(28,[the,rumour,damaged,this,company,that,john,was,bankrupt]).
str(29,[the,company,which,the,rumour,damaged,that,john,was,bankrupt]).
str(30,[a,woman,whom,i,met,before,and,married,after,the,long,war]).

% test(N) tests string N

test(N) :-
        str(N,String),
        retractall(edge(_,_,_,_)),
        write('String '), write(N), write(': '), write(String), nl, nl,
        test1(String).

test1(String) :-
        statistics(runtime,_),
        prs(String);
        statistics(runtime,[_,Time]),
        test2(String,Time).

test2(String,Time) :-
        pickup(String) ;
        length(String,L), write('Words: '), write(L),
        count(N), write('        Readings: '), write(N),
        Seconds is Time / 1000,
        write(' Time: '),
        write_trunc(Seconds,3),
        write(' seconds'), nl, nl.

% write_trunc(+RN,+N) writes real number RN, rounding down after N digits

write_trunc(RN,N) :-
```

```
        name(RN,Codes),
        write_trunc1(Codes,N,mant).

write_trunc1([],_,_) :- !.

write_trunc1([46|_],0,mant) :- !.

write_trunc1([_|Codes],0,mant) :- !,
        write(0),
        write_trunc(Codes,0,mant).

write_trunc1([46|Codes],N,mant) :- !,
        write('.'),
        write_trunc1(Codes,N,exp).

write_trunc1(_,0,exp) :- !.

write_trunc1([Code|Codes],N,Part) :-
        name(Digit,[Code]),
        write(Digit),
        N1 is N - 1,
        write_trunc1(Codes,N1,Part).

% test_all(N) tests all strings, starting at N

test_all(N) :-
        test(N),
        N1 is N + 1,
        test_all(N1).

% pickup(+String) picks up and displays the results of parsing String

pickup(String) :-
        retractall(count(_)),
        assert(count(0)),
        len(String,N), !,
        edge(0,Cat,N,T),
        incr_count,
        write(Cat), nl,
        write(T), nl,
        reduce(T,RT),
        write(RT), nl, nl,
        fail.

len([],0).

len([_|T],s(N)) :-
        len(T,N).

incr_count :-
        retract(count(N)),
        N1 is N + 1,
        assert(count(N1)), !.
```

```
% reduce(+Trans,-RedTrans) reduces translation Trans to its minimal
% form RedTrans

reduce(f‘X‘Y,Result) :- !,
        reduce_list([X,Y],[X1,Y1]),
        reduce(X1‘Y1,Result).

reduce(b‘X‘Y,Result) :-!,
        reduce_list([X,Y],[X1,Y1]),
        reduce(Y1‘X1,Result).

reduce(’R’‘G‘X‘Y‘Z,Result) :- !,
        reduce_list([G,X,Y,Z],[G1,X1,Y1,Z1]),
        reduce(G1‘X1‘(Y1‘Z1),Result).

reduce(’L’‘G‘X‘Y‘Z,Result) :- !,
        reduce_list([G,X,Y,Z],[G1,X1,Y1,Z1]),
        reduce(G1‘(X1‘Z1)‘Y1,Result).

reduce(’M’‘G‘X‘Y‘Z,Result) :- !,
        reduce_list([G,X,Y,Z],[G1,X1,Y1,Z1]),
        reduce(G1‘(X1‘Z1)‘Y1,Result).

reduce(’P’‘G‘X‘Y‘Z,Result) :- !,
        reduce_list([G,X,Y,Z],[G1,X1,Y1,Z1]),
        reduce(G1‘(X1‘Z1)‘(Y1‘Z1),Result).

reduce(c‘X‘C‘Y,Result) :- !,
        reduce_list([X,C,Y],[X1,C1,Y1]),
        reduce(C1‘X1‘Y1,Result).

reduce(’Af’‘S‘X‘C‘Y‘Z,Result) :- !,
        reduce_list([S,X,C,Y,Z],[S1,X1,C1,Y1,Z1]),
        reduce(S1‘(X1‘Z1)‘C1‘(Y1‘Z1),Result).

reduce(’Ab’‘S‘X‘C‘Y‘Z,Result) :- !,
        reduce_list([S,X,C,Y,Z],[S1,X1,C1,Y1,Z1]),
        reduce(S1‘(X1‘Z1)‘C1‘(Y1‘Z1),Result).

reduce(Exp,Exp).

% reduce_list(+Trees,RedTrees) reduces the list Trees to their minimal forms
% RedTrees

reduce_list([],[]).

reduce_list([T|Ts],[RT|RTs]) :-
        reduce(T,RT),
        reduce_list(Ts,RTs).
```

2. *Illustrative Log*

```
% qprolog

Quintus Prolog Release 2.2 (Sun-3, Unix 3.2)
Copyright (C) 1987, Quintus Computer Systems, Inc.  All rights reserved.
1310 Villa Street, Mountain View, California  (415) 965-7700

| ?- compile(prs).
[compiling /mnt/glyn/prs...]
[prs compiled 24.983 sec 20,288 bytes]

yes
| ?- test_all(1).
String 1: [we,leave]

s
b‘we‘leave
leave‘we

Words: 2        Readings: 1     Time: 0.0 seconds

String 2: [we,like,mary]

s
b‘we‘ (f‘like‘m)
like‘m‘we

Words: 3        Readings: 1     Time: 0.05 seconds

String 3: [we,think,that,you,leave]

s
b‘we‘ (f‘think‘ (f‘that‘ (b‘you‘leave)))
think‘ (that‘ (leave‘you))‘we

Words: 5        Readings: 1     Time: 0.15 seconds

String 4: [we,leave,and,you,leave]

s
c‘ (b‘we‘leave)‘ & ‘ (b‘you‘leave)
& ‘ (leave‘we)‘ (leave‘you)

Words: 5        Readings: 1     Time: 0.01 seconds

String 5: [we,leave,or,you,leave]

s
c‘ (b‘we‘leave)‘v‘ (b‘you‘leave)
v‘ (leave‘we)‘ (leave‘you)
```

```
Words: 5        Readings: 1     Time: 0.03 seconds

String 6: [we,leave,or,stay]

s
b‘we‘ (Ab‘c‘leave‘v‘stay)
v‘ (leave‘we)‘ (stay‘we)

Words: 4        Readings: 1     Time: 0.01 seconds

String 7: [we,like,and,you,dislike,mary]

s
f‘ (Af‘c‘ (R‘b‘we‘like)‘ & ‘ (R‘b‘you‘dislike))‘m
& ‘ (like‘m‘we)‘ (dislike‘m‘you)

Words: 6        Readings: 1     Time: 0.1 seconds

String 8: [the,man,who,we,meet]

np
f‘the‘ (b‘man‘ (f‘who‘ (R‘b‘we‘meet)))
the‘ (who‘ (R‘b‘we‘meet)‘man)

Words: 5        Readings: 1     Time: 0.11 seconds

String 9: [the,man,who,we,think,that,we,meet]

np
f‘the‘ (b‘man‘ (f‘who‘ (R‘f‘ (R‘f‘ (R‘b‘we‘think)‘that)‘ (R‘b‘we‘meet))))
the‘ (who‘ (R‘f‘ (R‘f‘ (R‘b‘we‘think)‘that)‘ (R‘b‘we‘meet))‘man)

Words: 8        Readings: 1     Time: 0.35 seconds

String 10: [the,man,who,we,like,and,you,dislike]

np
f‘the‘ (b‘man‘ (f‘who‘ (Af‘c‘ (R‘b‘we‘like)‘ & ‘ (R‘b‘you‘dislike))))
the‘ (who‘ (Af‘c‘ (R‘b‘we‘like)‘ & ‘ (R‘b‘you‘dislike))‘man)

Words: 8        Readings: 1     Time: 0.16 seconds

String 11: [we,show,and,you,give,john,the,book]

s
f‘ (f‘ (Af‘ (Af‘c)‘ (R‘ (R‘b)‘we‘show)‘ & ‘ (R‘ (R‘b)‘you‘give))‘j)‘ (f‘the‘book)
& ‘ (show‘j‘ (the‘book)‘we)‘ (give‘j‘ (the‘book)‘you)

s
f‘ (M‘f‘ (Af‘ (Af‘c)‘ (R‘ (R‘b)‘we‘show)‘ & ‘ (R‘ (R‘b)‘you‘give))‘j)‘ (f‘the‘book)
& ‘ (show‘ (the‘book)‘j‘we)‘ (give‘ (the‘book)‘j‘you)

Words: 8        Readings: 2     Time: 0.41 seconds
```

```
String 12: [the,book,which,we,show,and,you,give,john]

np
f‘the‘ (b‘book‘ (f‘which‘ (f‘ (Af‘ (Af‘c)‘ (R‘ (R‘b)‘we‘show)‘ & ‘ (R‘ (R‘b)‘you‘give))‘j)))
the‘ (which‘ (Af‘c‘ (R‘ (R‘b)‘we‘show‘j)‘ & ‘ (R‘ (R‘b)‘you‘give‘j))‘book)

np
f‘the‘ (b‘book‘ (f‘which‘ (M‘f‘ (Af‘ (Af‘c)‘ (R‘ (R‘b)‘we‘show)‘ & ‘ (R‘ (R‘b)‘you‘give))‘j)))
the‘ (which‘ (M‘f‘ (Af‘ (Af‘c)‘ (R‘ (R‘b)‘we‘show)‘ & ‘ (R‘ (R‘b)‘you‘give))‘j)‘book)

Words: 9        Readings: 2     Time: 0.48 seconds

String 13: [the,man,who,we,like,mary,and,you,dislike]

Words: 9        Readings: 0     Time: 0.13 seconds

String 14: [you,sit,and,sleep]

s
b‘you‘ (Ab‘c‘sit‘ & ‘sleep)
& ‘ (sit‘you)‘ (sleep‘you)

Words: 4        Readings: 1     Time: 0.01 seconds

String 15: [you,sit,and,sleep,restlessly]

s
b‘you‘ (Ab‘c‘sit‘ & ‘ (b‘sleep‘restlessly))
& ‘ (sit‘you)‘ (restlessly‘sleep‘you)

s
b‘you‘ (b‘ (Ab‘c‘sit‘ & ‘sleep)‘restlessly)
restlessly‘ (Ab‘c‘sit‘ & ‘sleep)‘you

Words: 5        Readings: 2     Time: 0.08 seconds

String 16: [you,sit,and,sleep,restlessly,upstairs]

s
b‘you‘ (Ab‘c‘sit‘ & ‘ (b‘ (b‘sleep‘restlessly)‘upstairs))
& ‘ (sit‘you)‘ (upstairs‘ (restlessly‘sleep)‘you)

s
b‘you‘ (b‘ (Ab‘c‘sit‘ & ‘ (b‘sleep‘restlessly))‘upstairs)
upstairs‘ (Ab‘c‘sit‘ & ‘ (b‘sleep‘restlessly))‘you

s
b‘you‘ (b‘ (b‘ (Ab‘c‘sit‘ & ‘sleep)‘restlessly)‘upstairs)
upstairs‘ (restlessly‘ (Ab‘c‘sit‘ & ‘sleep))‘you

Words: 6        Readings: 3     Time: 0.18 seconds

String 17: [you,sleep,restlessly,upstairs,and,peacefully,downstairs]
```

```
s
b‘you‘ (b‘ (b‘sleep‘restlessly)‘ (Ab‘ (Ab‘c)‘upstairs‘ & ‘ (L‘b‘peacefully‘downstairs)))
& ‘ (upstairs‘ (restlessly‘sleep)‘you)‘ (downstairs‘ (peacefully‘ (restlessly‘sleep))‘you)

s
b‘you‘ (b‘sleep‘ (Ab‘ (Ab‘c)‘ (L‘b‘restlessly‘upstairs)‘ & ‘ (L‘b‘peacefully‘downstairs)))
& ‘ (upstairs‘ (restlessly‘sleep)‘you)‘ (downstairs‘ (peacefully‘sleep)‘you)

s
b‘you‘ (b‘ (b‘ (b‘sleep‘restlessly)‘ (Ab‘ (Ab‘c)‘upstairs‘ & ‘peacefully))‘downstairs)
downstairs‘ (Ab‘c‘ (upstairs‘ (restlessly‘sleep))‘ & ‘ (peacefully‘ (restlessly‘sleep)))‘you

s
b‘you‘ (b‘ (b‘sleep‘ (Ab‘ (Ab‘c)‘ (L‘b‘restlessly‘upstairs)‘ & ‘peacefully))‘downstairs)
downstairs‘ (Ab‘c‘ (L‘b‘restlessly‘upstairs‘sleep)‘ & ‘ (peacefully‘sleep))‘you

Words: 7        Readings: 4     Time: 0.45 seconds

String 18: [the,dog,which,we,show,john]

np
f‘the‘ (b‘dog‘ (f‘which‘ (R‘b‘we‘ (f‘show‘j))))
the‘ (which‘ (R‘b‘we‘ (f‘show‘j))‘dog)

np
f‘the‘ (b‘dog‘ (f‘which‘ (R‘b‘we‘ (M‘f‘show‘j))))
the‘ (which‘ (R‘b‘we‘ (M‘f‘show‘j))‘dog)

Words: 6        Readings: 2     Time: 0.25 seconds

String 19: [the,man,who,we,think,that,left]

Words: 7        Readings: 0     Time: 0.16 seconds

String 20: [the,paper,which,you,filed,without,reading]

np/np
R‘f‘ (R‘f‘the‘ (R‘b‘paper‘which))‘ (R‘ (R‘b)‘you‘ (M‘ (R‘b)‘filed‘ (R‘f‘without‘reading)))
R‘f‘ (R‘f‘the‘ (R‘b‘paper‘which))‘ (R‘ (R‘b)‘you‘ (M‘ (R‘b)‘filed‘ (R‘f‘without‘reading)))

np
f‘the‘ (b‘paper‘ (f‘which‘ (R‘b‘you‘ (P‘b‘filed‘ (R‘f‘without‘reading)))))
the‘ (which‘ (R‘b‘you‘ (P‘b‘filed‘ (R‘f‘without‘reading)))‘paper)

np/np
R‘f‘ (R‘f‘ (R‘f‘the‘ (R‘b‘paper‘which))‘ (R‘ (R‘b)‘you‘ (R‘ (M‘b)‘filed‘without)))‘reading
R‘f‘ (R‘f‘ (R‘f‘the‘ (R‘b‘paper‘which))‘ (R‘ (R‘b)‘you‘ (R‘ (M‘b)‘filed‘without)))‘reading

np/np
R‘f‘ (R‘f‘the‘ (R‘b‘paper‘which))‘ (R‘ (M‘f)‘ (R‘ (R‘b)‘you‘ (M‘ (R‘b)‘filed‘without))‘reading)
R‘f‘ (R‘f‘the‘ (R‘b‘paper‘which))‘ (R‘ (M‘f)‘ (R‘ (R‘b)‘you‘ (M‘ (R‘b)‘filed‘without))‘reading)

np/np
```

```
R‘f‘ (R‘f‘the‘ (R‘b‘paper‘which))‘ (M‘ (R‘f)‘ (R‘ (R‘b)‘you‘ (M‘ (R‘b)‘filed‘without))‘reading)
R‘f‘ (R‘f‘the‘ (R‘b‘paper‘which))‘ (M‘ (R‘f)‘ (R‘ (R‘b)‘you‘ (M‘ (R‘b)‘filed‘without))‘reading)

Words: 7        Readings: 5     Time: 1.0 seconds

String 21: [you,will,leave,tomorrow]

s
b‘you‘ (f‘will‘ (b‘leave‘tomorrow))
will‘ (tomorrow‘leave)‘you

s
b‘you‘ (b‘ (f‘will‘leave)‘tomorrow)
tomorrow‘ (will‘leave)‘you

Words: 4        Readings: 2     Time: 0.08 seconds

String 22: [john,loves,mary,dearly]

s
b‘j‘ (b‘ (f‘loves‘m)‘dearly)
dearly‘ (loves‘m)‘j

Words: 4        Readings: 1     Time: 0.03 seconds

String 23: [we,put,on,the,table,a,large,red,book]

s
b‘we‘ (f‘ (M‘f‘put‘ (f‘on‘ (f‘the‘table)))‘ (f‘a‘ (f‘large‘ (f‘red‘book))))
put‘ (a‘ (large‘ (red‘book)))‘ (on‘ (the‘table))‘we

s/ (s\np\ (s\np))
R‘b‘we‘ (f‘ (M‘ (M‘b)‘put‘ (f‘on‘ (f‘the‘table)))‘ (f‘a‘ (f‘large‘ (f‘red‘book))))
R‘b‘we‘ (f‘ (M‘ (M‘b)‘put‘ (f‘on‘ (f‘the‘table)))‘ (f‘a‘ (f‘large‘ (f‘red‘book))))

s
b‘we‘ (f‘ (f‘ (M‘ (R‘f)‘put‘on)‘ (f‘the‘table))‘ (f‘a‘ (f‘large‘ (f‘red‘book))))
put‘ (the‘table)‘ (on‘ (a‘ (large‘ (red‘book))))‘we

s/ (s\np\ (s\np))
R‘b‘we‘ (f‘ (f‘ (M‘ (R‘ (M‘b))‘put‘on)‘ (f‘the‘table))‘ (f‘a‘ (f‘large‘ (f‘red‘book))))
R‘b‘we‘ (f‘ (f‘ (M‘ (R‘ (M‘b))‘put‘on)‘ (f‘the‘table))‘ (f‘a‘ (f‘large‘ (f‘red‘book))))

s/ (s\np\ (s\np))
R‘b‘we‘ (M‘f‘ (f‘ (M‘ (M‘ (R‘b))‘put‘on)‘ (f‘the‘table))‘ (f‘a‘ (f‘large‘ (f‘red‘book))))
R‘b‘we‘ (M‘f‘ (f‘ (M‘ (M‘ (R‘b))‘put‘on)‘ (f‘the‘table))‘ (f‘a‘ (f‘large‘ (f‘red‘book))))

s
f‘ (f‘ (M‘ (R‘f)‘ (R‘ (R‘b)‘we‘put)‘on)‘ (f‘the‘table))‘ (f‘a‘ (f‘large‘ (f‘red‘book)))
put‘ (the‘table)‘ (on‘ (a‘ (large‘ (red‘book))))‘we

Words: 9        Readings: 6     Time: 5.83 seconds
```

```
String 24: [mary,thinks,that,the,man,thinks,that,you,will,leave,today]

s
b‘m‘ (f‘thinks‘ (f‘that‘ (b‘ (f‘the‘man)‘ (f‘thinks‘ (f‘that‘ (b‘you‘ (f‘will‘ (b‘leave‘today))))))))
thinks‘ (that‘ (thinks‘ (that‘ (will‘ (today‘leave)‘you))‘ (the‘man)))‘m

s
b‘m‘ (f‘thinks‘ (f‘that‘ (b‘ (f‘the‘man)‘ (f‘thinks‘ (f‘that‘ (b‘you‘ (b‘ (f‘will‘leave)‘today)))))))
thinks‘ (that‘ (thinks‘ (that‘ (today‘ (will‘leave)‘you))‘ (the‘man)))‘m

s
b‘m‘ (f‘thinks‘ (f‘that‘ (b‘ (f‘the‘man)‘ (b‘ (f‘thinks‘ (f‘that‘ (b‘you‘ (f‘will‘leave))))‘today))))
thinks‘ (that‘ (today‘ (thinks‘ (that‘ (will‘leave‘you)))‘ (the‘man)))‘m

s
b‘m‘ (b‘ (f‘thinks‘ (f‘that‘ (b‘ (f‘the‘man)‘ (f‘thinks‘ (f‘that‘ (b‘you‘ (f‘will‘leave)))))))‘today)
today‘ (thinks‘ (that‘ (thinks‘ (that‘ (will‘leave‘you))‘ (the‘man))))‘m

Words: 11        Readings: 4      Time: 1.88 seconds

String 25: [the,man,inside,thinks,that,the,woman,outside,left,quickly,yesterday]

s
b‘ (f‘the‘ (b‘man‘inside))‘ (f‘thinks‘ (f‘that‘ (b‘ (f‘the‘
                (b‘woman‘outside))‘ (b‘ (b‘left‘quickly)‘yesterday))))
thinks‘ (that‘ (yesterday‘ (quickly‘left)‘ (the‘ (outside‘woman))))‘ (the‘ (inside‘man))

s
b‘ (f‘the‘ (b‘man‘inside))‘ (b‘ (f‘thinks‘ (f‘that‘ (b‘ (f‘the‘
                (b‘woman‘outside))‘ (b‘left‘quickly))))‘yesterday)
yesterday‘ (thinks‘ (that‘ (quickly‘left‘ (the‘ (outside‘woman)))))‘ (the‘ (inside‘man))

s
b‘ (f‘the‘ (b‘man‘inside))‘ (b‘ (b‘ (f‘thinks‘ (f‘that‘ (b‘ (f‘the‘
                (b‘woman‘outside))‘left)))‘quickly)‘yesterday)
yesterday‘ (quickly‘ (thinks‘ (that‘ (left‘ (the‘ (outside‘woman))))))‘ (the‘ (inside‘man))

Words: 11        Readings: 3      Time: 0.59 seconds

String 26: [that,man,that,laughs,thinks,that,that,dog,that,sue,owns,swims]

s
b‘ (f‘that‘ (b‘man‘ (f‘that‘laughs)))‘ (f‘thinks‘ (f‘that‘
                (b‘ (f‘that‘ (b‘dog‘ (f‘that‘ (R‘b‘s‘owns))))‘swims)))
thinks‘ (that‘ (swims‘ (that‘ (that‘ (R‘b‘s‘owns)‘dog))))‘ (that‘ (that‘laughs‘man))

Words: 12        Readings: 1      Time: 0.73 seconds

String 27: [a,rumour,spread,that,john,was,bankrupt]

s
f‘ (M‘b‘ (R‘f‘a‘rumour)‘spread)‘ (f‘that‘ (b‘j‘ (f‘was‘bankrupt)))
spread‘ (a‘ (rumour‘ (that‘ (was‘bankrupt‘j))))
```

```
Words: 7        Readings: 1     Time: 0.21 seconds

String 28: [the,rumour,damaged,this,company,that,john,was,bankrupt]

s
f‘ (M‘b‘ (R‘f‘the‘rumour)‘ (f‘damaged‘ (f‘this‘company)))‘ (f‘that‘ (b‘j‘ (f‘was‘bankrupt)))
damaged‘ (this‘company)‘ (the‘ (rumour‘ (that‘ (was‘bankrupt‘j))))

Words: 9        Readings: 1     Time: 0.86 seconds

String 29: [the,company,which,the,rumour,damaged,that,john,was,bankrupt]

np
f‘the‘ (b‘company‘ (f‘which‘ (f‘ (M‘ (R‘b)‘ (R‘f‘the‘rumour)‘damaged)‘
               (f‘that‘ (b‘j‘ (f‘was‘bankrupt))))))
the‘ (which‘ (R‘b‘ (R‘f‘the‘rumour‘ (that‘ (was‘bankrupt‘j)))‘damaged)‘company)

Words: 10       Readings: 1     Time: 1.66 seconds

String 30: [a,woman,whom,i,met,before,and,married,after,the,long,war]

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (Af‘ (Ab‘c)‘ (P‘b‘met‘before)‘ & ‘
               (M‘b‘married‘ (f‘after‘ (f‘the‘ (f‘long‘war))))))))
a‘ (who‘ (R‘b‘i‘ (Af‘ (Ab‘c)‘ (P‘b‘met‘before)‘ & ‘
               (M‘b‘married‘ (f‘after‘ (f‘the‘ (f‘long‘war))))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (M‘b‘ (Af‘ (Ab‘c)‘ (P‘b‘met‘before)‘ & ‘
               married)‘ (f‘after‘ (f‘the‘ (f‘long‘war)))))))
a‘ (who‘ (R‘b‘i‘ (M‘b‘ (Af‘ (Ab‘c)‘ (P‘b‘met‘before)‘ & ‘
               married)‘ (f‘after‘ (f‘the‘ (f‘long‘war)))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (f‘ (Af‘ (Af‘ (Ab‘c))‘ (R‘ (M‘b)‘met‘before)‘ & ‘
               (R‘ (M‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (f‘ (Af‘ (Af‘ (Ab‘c))‘ (R‘ (M‘b)‘met‘before)‘ & ‘
               (R‘ (M‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (M‘f‘ (Af‘ (Af‘ (Ab‘c))‘ (R‘ (M‘b)‘met‘before)‘ & ‘
               (R‘ (M‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (M‘f‘ (Af‘ (Af‘ (Ab‘c))‘ (R‘ (M‘b)‘met‘before)‘ & ‘
               (R‘ (M‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (f‘ (Af‘ (Af‘ (Ab‘c))‘ (M‘ (R‘b)‘met‘before)‘ & ‘
               (R‘ (M‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (f‘ (Af‘ (Af‘ (Ab‘c))‘ (M‘ (R‘b)‘met‘before)‘ & ‘
               (R‘ (M‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (M‘f‘ (Af‘ (Af‘ (Ab‘c))‘ (M‘ (R‘b)‘met‘before)‘ & ‘
```

```
                    (R‘ (M‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (M‘f‘ (Af‘ (Af‘ (Ab‘c))‘ (M‘ (R‘b)‘met‘before)‘ & ‘
                    (R‘ (M‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (Af‘ (Ab‘c)‘ (P‘b‘met‘before)‘ & ‘
                    (f‘ (M‘ (R‘b)‘married‘after)‘ (f‘the‘ (f‘long‘war)))))))
a‘ (who‘ (R‘b‘i‘ (Af‘ (Ab‘c)‘ (P‘b‘met‘before)‘ & ‘
                    (f‘ (M‘ (R‘b)‘married‘after)‘ (f‘the‘ (f‘long‘war)))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (f‘ (Af‘ (Af‘ (Ab‘c))‘ (R‘ (M‘b)‘met‘before)‘ & ‘
                    (M‘ (R‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (f‘ (Af‘ (Af‘ (Ab‘c))‘ (R‘ (M‘b)‘met‘before)‘ & ‘
                    (M‘ (R‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (M‘f‘ (Af‘ (Af‘ (Ab‘c))‘ (R‘ (M‘b)‘met‘before)‘ & ‘
                    (M‘ (R‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (M‘f‘ (Af‘ (Af‘ (Ab‘c))‘ (R‘ (M‘b)‘met‘before)‘ & ‘
                    (M‘ (R‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (f‘ (Af‘ (Af‘ (Ab‘c))‘ (M‘ (R‘b)‘met‘before)‘ & ‘
                    (M‘ (R‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (f‘ (Af‘ (Af‘ (Ab‘c))‘ (M‘ (R‘b)‘met‘before)‘ & ‘
                    (M‘ (R‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (M‘f‘ (Af‘ (Af‘ (Ab‘c))‘ (M‘ (R‘b)‘met‘before)‘ & ‘
                    (M‘ (R‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (M‘f‘ (Af‘ (Af‘ (Ab‘c))‘ (M‘ (R‘b)‘met‘before)‘ & ‘
                    (M‘ (R‘b)‘married‘after))‘ (f‘the‘ (f‘long‘war))))‘woman)

np
f‘a‘ (b‘woman‘ (f‘who‘ (R‘b‘i‘ (f‘ (M‘ (R‘b)‘ (Af‘ (Ab‘c)‘ (P‘b‘met‘before)‘ & ‘
                    married)‘after)‘ (f‘the‘ (f‘long‘war))))))
a‘ (who‘ (R‘b‘i‘ (f‘ (M‘ (R‘b)‘ (Af‘ (Ab‘c)‘ (P‘b‘met‘before)‘ & ‘
                    married)‘after)‘ (f‘the‘ (f‘long‘war))))‘woman)

Words: 12      Readings: 12    Time: 29.9 seconds


no
| ?- ^Z
[ End of Prolog execution ]
%

script done on Thu Jul 21 17:54:17 1988
```

References

Abbott, B. (1976) Right node raising as a test for constituenthood. *Linguistic Inquiry*, **7**, 639-642.

Ades, A. and Steedman, M. J. (1982) On the Order of Words. *Linguistics and Philosophy*, **4**, 517-558.

Aho, A. V. and Ullman, J. D. (1972) *The Theory of Parsing, Translation, and Compiling*, Volume 1: *Parsing*. New Jersey: Englewood Cliffs.

Ajdukiewicz, K. (1935) Die syntaktische Konnexitat. *Studia Philosophica*, **1**, 1-27. English translation in: *Polish Logic: 1920-1939* , ed. by Storrs McCall, pp207-231, Oxford University Press, Oxford.

Akmajian, A. (1975) More Evidence for an NP Cycle. *Linguistic Inquiry*, **VI**, 115-129.

Altmann, G. (1986) Reference and the Resolution of Local Ambiguity: Interaction in Human Sentence Processing. PhD Thesis, University of Edinburgh.

Anderson, A. R. and Belnap, N. D. (1975) *Entailment*, Volume 1. Princeton, N.J.: Princeton University Press.

Andrews, A. D. (1975) Studies in the Syntax of Relative and Comparative Clauses. PhD Thesis, MIT. Published by Garland Publishing, Inc., New York..

Bach, E. (1976) An extension of classical transformational grammar. In *Problems in Linguistic Metatheory, Proceedings of the 1976 Conference at Michigan State University*, 1976, pp183-224.

Bach, E. and Horn, G. M. (1976) Remarks on "Conditions on Transformations". *Linguistic Inquiry*, **7**, 265-299.

Bach, E. and Cooper, R. (1978) The NP-S analysis of relative clauses and compositional semantics. *Linguistics and Philosophy*, **2**, 145-150.

Bach, E. (1979) Control in Montague Grammar. *Linguistic Inquiry*, **10**, 515-531.

Bach, E. (1980) In Defense of Passive. *Linguistics and Philosophy*, **3**, 297-341.

Bach, E. (1983) Generalised categorial grammars and the English auxiliary. In Heny, F. and Richards, B. (eds.) *Linguistic categories: auxiliaries and related puzzles*, Volume II, pp101-120. Dordrecht: D. Reidel.

Bach, E. (1984) Some Generalizations of Categorial Grammar. In Landman, F. and Veltman, F. (eds.) *Varieties of Formal Semantics: Proceedings of the fourth Amsterdam Colloquium, September 1982*, pp1-23. Dordrecht: Foris Publications.

Baltin, M. R. (1982) A Landing Site Theory of Movement Rules. *Linguistic Inquiry*, **13**, 1-38.

Bar-Hillel, Y. (1953) A quasi-arithmetical notation for syntactic description. *Language*, **29**, 47-58.

Barendregt, H. P. (1981) *The Lambda Calculus: Its Syntax and Semantics*. North Holland.

Barton, G. E., Berwick, R. C. and Ristad, E. S. (1987) *Computatational Complexity and Natural Language*. Cambridge, Mass.: MIT Press.

Barton Jr., G. E. (1986) Computational Complexity in Two-Level Morphology. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, Columbia University, New York, N.Y., June, 1986, pp53-59.

Berwick, R. (1982) Computational Complexity and Lexical Functional Grammar. *American Journal of Computational Linguistics*, **8**, 97-109.

Borsley, R. D. (1987) Subjects and Complements in HPSG. Report No. CSLI-87-107, Center for the Study of Language and Information, September, 1987.

Bouma, G. (1987) A Unification-Based Analysis of Unbounded Dependencies in Categorial Grammar. In Groenendijk, J., Stokhof, M. and Veltman, F. (eds.) *Sixth Amsterdam Colloquium*, Institute for Language, Logic and Information, University of Amsterdam, Amsterdam, 1987, pp1-19.

Bresnan, J. W. (1972) Theory of complementation in English syntax. Unpublished doctoral dissertation, MIT Press.

Bresnan, J. W. (1974) The position of certain clause particles in phrase structure. *Linguistic Inquiry*, **5**, 614-619.

Calder, J., Moens, M. and Zeevat, H. (1986) A UCG Interpreter. ESPRIT PROJECT 393 ACORD; Deliverable T2.6.

Calder, J., Klein, E., Moens, M. and Reape, M. (1988) Global Constraints in Unification Grammar.

Carpenter, B. (forthcoming) Phrase Meaning and Categorial Grammar. PhD Thesis, Centre for Cognitive Science, University of Edinburgh.

Chomsky, N. (1964) The Logical Basis of Linguistic Theory. In Lunt, H. (ed.) *Proceedings of the Ninth International Congress of Linguistics*, Cambridge, Mass, 1964. Mouton, The Hague.

Chomsky, N. (1970) Remarks on nominalization. In Jacobs, R. and Rosenbaum, P. (eds.) *Readings in English transformational grammar*, pp184-221. Waltham, Mass.: Ginn and Co..

Chomsky, N. (1973) Conditions on transformations. In Anderson, S. and Kiparsky, P. (eds.) *A Festschrift for Morris Halle.* New York: Holt, Rinehart and Winston.

Chomsky, N. and Lasnik, H. (1977) Filters and Control. *Linguistic Inquiry*, **8**, 425-504.

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

Chung, S. and McCloskey, J. (1983) On the interpretation of certain island facts in GPSG. *Linguistic Inquiry*, 14, 704-713.

Cook, S. A. (1971) The complexity of theorem-proving procedures. In *The Third Annual ACM Symposium on the Theory of Computing*, Association for Computing Machinery, New York, 1971, pp151-158.

Crain, S. and Steedman, M. J. (1985) On not being led up the garden path: the use of context by the psychological parser. In Dowty, D., Karttunen, L. and Zwicky, A. (eds.) *Natural Language Parsing: Psychological, Computational, and Theoretical perspectives.*

Curry, H. B. and Feys, R. (1958) *Combinatory logic*, Volume I. Amsterdam: North Holland.

Dougherty, R. C. (1970) A grammar of coordinate conjoined structures, I. *Language*, 46, 850-898.

Dougherty, R. C. (1971) A Grammar of Coordinate Conjoined Structures, II. *Language*, 47, 278-339.

Dowty, D. (1988) Type Raising, Functional Composition, and Non-constituent Conjunction. In Oehrle, R., Bach, E. and Wheeler, D. (eds.) *Categorial Grammars and Natural Language Structures*, Dordrecht, 1988. Paper presented at the Conference on Categorial Grammar, Tucson, June 1985.

Engdahl, E. (1983) Parasitic gaps. *Linguistics and Philosophy*, **6**, 5-34.

Engdahl, E. (1986) *Constituent Questions: The Syntax and Semantics of Questions with Special Reference to Swedish.* Dordrecht: D. Reidel.

Flickinger, D. (1983) Lexical heads and phrasal gaps. In Barlow, M., Flickinger, D. and Westcoat, M. (eds.) *Proceedings of the Second West Coast Conference on Formal Linguistics*, Stanford, 1983.

Flynn, M. (1983) A categorial theory of structure building. In Gazdar, G., Klein, E. and Pullum, G. K. (eds.) *Order, Concord and Constituency*, pp138-174. Dordrecht: Foris Publications.

Fodor, J. D. (1978) Parsing strategies and constraints on transformations. *Linguistic Inquiry*, **9**, 427-473.

Friedman, J., Dai, D. and Wang, W. (1986) The Weak Generative Capacity of Parenthesis-Free Categorial Grammars. In *Proceedings of the 11th International Conference on Computational Linguistics and the 24th Annual Meeting of the Association for Computational Linguistics*, Institut fuer Kommunikationsforschung und Phonetik, Bonn University, Bonn, August, 1986, pp199-201.

Garey, M. R. and Johnson, D. S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness.* New York: Freeman.

Gazdar, G. (1980) A cross-categorial semantics for coordination. *Linguistics and Philosophy*, **3**, 407-409.

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

Gazdar, G. and Pullum, G. K. (1981) Subcategorization, constituent order and the notion 'head'. In Moortgat, M., van der Hulst, H. and Hoekstra, T. (eds.) *The Scope of Lexical Rules*, pp107-123. Dordrecht: Foris Publications.

Gazdar, G., Klein, E., Pullum, G. K. and Sag, I. A. (1982) Coordinate structure and unbounded dependencies. In Barlow, M., Flickinger, D. and Sag, I. A. (eds.) *Developments in Generalized Phrase Structure Grammar: Stanford Working Papers in Grammatical Theory*, Volume 2, pp38-68. Bloomington, Indiana: Indiana University Linguistics Club.

Gazdar, G., Pullum, G. K. and Sag, I. A. (1982) Auxiliaries and related phenomena in a restrictive theory of grammar. *Language*, **58**, 591-638.

Gazdar, G. (1982) Phrase structure grammar. In Jacobson, P. and Pullum, G. K. (eds.) *The Nature of Syntactic Representation*, pp131-186. Dordrecht: D. Reidel.

Gazdar, G., Pullum, G. K., Sag, I. A. and Wasow, T. (1982) Coordination and transformational grammar. *Linguistic Inquiry*, **13**, 663-676.

Gazdar, G., Klein, E., Pullum, G. and Sag, I. (1985) *Generalized Phrase Structure Grammar*. London: Basil Blackwell.

Geach, P. T. (1972) A program for syntax. In Davidson, D. and Harman, G. (eds.) *Semantics of Natural Language*. Dordrecht: D. Reidel.

Groenendijk, J. and Stokhof, M. (1987) Type-Shifting Rules and the Semantic of Interrogatives. Technical Report, Philosophy, University of Amsterdam, February, 1987.

Grosu, A. (1972) The Strategic Content of Island Constraints. In *Ohio State University Working Papers in Linguistics*, Volume 13.

Haddock, N. J. (forthcoming) PhD Thesis, Centre for Cognitive Science and Department of Artificial Intelligence, University of Edinburgh.

Hendriks, H. (1987) Type Change in Semantics: the Scope of Quantification and Coordination. In Klein, E. and Benthem, J. (eds.) *Categorial Grammar: Its Current State*, Institute for Language, Logic and Information, University of Amsterdam, 1987, pp95-119.

Hepple, M. (1987) Methods for Parsing Combinatory Grammars and the Spurious Ambiguity Problem. Masters Thesis, Centre for Cognitive Science, University of Edinburgh.

Huck, G. (1985) Discontinuity and Word-order in Categorial Grammar. PhD Thesis, University of Chicago. unpublished.

Hudson, D. (1986) Incomplete Conjuncts. Paper presented at the Autumn 1986 meeting of the Linguistics Association of Great Britain.

Iwakura, K. (1980) On wh-movement and constraints on rules. *Linguistic Analaysis*, **6**, 53-95.

Janssen, T. M. V. (1983) Foundations and Applications of Montague Grammar. PhD Thesis, Mathematisch Centrum, Universiteit van Amsterdam.

Kamp, H. (1981) A theory of truth and semantic representation. In Groenendijk, J. A. G., Janssen, T. M. V. and Stokhof, M. B. J. (eds.) *Formal Methods in the Study of Language*, Volume 136, pp277-322. Amsterdam: Mathematical Centre Tracts.

Kang, B. (1988) Functional Inheritance, Anaphora, and Semantic Interpretation in a Generalized Categorial Grammar. PhD Thesis, Department of Cognitive and Linguistic Sciences, Brown University.

Kaplan, R. M. and Bresnan, J. (1982) Lexical-Functional Grammar: a formal system for grammatical representation. Chapter 4 in Bresnan, J. (ed.) *The Mental Representation of Grammatical Relations*, pp173-281. Cambridge, Mass.: MIT Press.

Kaplan, R. and Zaenen, A. (1987) Long-distance Dependencies, Constituent Structure and Functional Uncertainty.

Karttunen, L. (1986) Radical Lexicalism. Report No. CSLI-86-68, Center for the Study of Language and Information, December, 1986. Paper presented at the Conference on Alternative Conceptions of Phrase Structure, July 1986, New York.

Kay, M. (1967) Experiments with a powerful parser. In *2ème Conférence Internationale sur le Traitement Automatique des Langues*, Grenoble, 1967. also as report RM-5452-PR, The Rand Corporation, Santa Monica, California.

Kay, M. (1980) Algorithm Schemata and Data Structures in Syntactic Processing. Technical Report No. CSL-80-12, XEROX Palo Alto Research Centre, Palo Alto, October, 1980.

Keenan, E. L. (1974) The Functional Principle: Generalizing the notion of 'subject of'. In La Galy, M. W., Fox, R. A. and Bruck, A. (eds.) *Papers from the Tenth Regional Meeting of the Chicago Linguistics Society*, Chicago, April 19-21, 1974, pp298-308.

Keenan, E. L. and Faltz, L. M. (1985) *Boolean Semantics for Natural Language.* Dordrecht: D. Reidel.

Klein, E. and Sag, I. A. (1985) Type-driven translation. *Linguistics and Philosophy*, **8**, 163-201.

Kuno, S. (1976) Subject, Theme and Speaker's Empathy. In Li, C. (ed.) *Subject and Topic.* New York: Academic Press.

Lambek, J. (1958) The mathematics of sentence structure. *American Mathematical Monthly*, **65**, 154-170.

Lambek, J. (1961) On the calculus of syntactic types. In *Structure of language and its mathematical aspects*, Providence, Rhode Island, 1961, pp166-178.

Langendoen, D. T. and Bever, T. G. (1973) Can a Not Unhappy Person Be Called a Not Sad One? In Anderson, S. R. and Kiparsky, P. (eds.) *A Festschrift for Morris Halle*, pp392-409. New York: Holt, Rinehart and Winston.

Levine, R. D. (1985) Right Node (Non-)Raising. *Linguistic Inquiry*, **16**, 492-497.

Lyons, J. (1968) *Introduction to Theoretical Linguistics.* Cambridge: Cambridge University Press.

Maling, J. M. and Zaenen, A. (1982) A phrase structure account of Scandinavian extraction phenomena. In Jacobson, P. and Pullum, G. K. (eds.) *The nature of syntactic representation*, pp229-282. Dordrecht: D. Reidel.

Montague, R. (1973) The proper treatment of quantification in ordinary English. In Hintikka, J., Moravcsik, J. M. E. and Suppes, P. (eds.) *Approaches to Natural Language.* Dordrecht: D. Reidel. Reprinted in R H Thomason (ed.) (1974), *Formal Philosophy: Selected Papers of Richard Montague*, pp247-270. Yale University Press: New Haven, Conn.

Moortgat, M. (1987) Lambek Categorial Grammar and the Autonomy Thesis. INL Working Papers No. 87-03, Instituut voor Nederlandse Lexicologie, Leiden, April, 1987.

Moortgat, M. (1988) Mixed composition and discontinuous dependencies. In Oehrle, R., Bach, E. and Wheeler, D. (eds.) *Categorial Grammars and Natural Language Structures*, Dordrecht, 1988, pp319-390. Paper presented at the Conference on Categorial Grammar, Tucson, June 1985.

Morrill, G. and Carpenter, B. (1987) Compositionality, Implicational Logics, and Theories of Grammar. Research Paper No. EUCCS/RP-11, Centre for Cognitive Science, University of Edinburgh, Edinburgh, June, 1987. To appear in *Linguistics and Philosophy.*

Morrill, G. (1987a) Phrase Structure Grammar and Categorial Grammar. In Benthem, J. and Klein, E. (eds.) *Proceedings of the Conferences in Amsterdam and Stirling*, Universities of Edinburgh and Amsterdam, 1987a.

Morrill, G. (1987b) Meta-Categorial Grammar. In Haddock, N. J., Klein, E. and Morrill, G. (eds.) *Edinburgh Working Papers in Cognitive Science*, Volume 1: *Categorial Grammar, Unification Grammar, and Parsing.*

Otero, C. (1972) Acceptable Ungrammatical Sentences in Spanish. *Linguistic Inquiry*, III, 233-242.

Pareschi, R. and Steedman, M. J. (1987) A Lazy Way to Chart-Parse with Extended Categorial Grammars. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, Stanford University, Stanford, Ca., 6-9 July, 1987.

Partee, B. and Rooth, M. (1983) Generalized conjunction and type ambiguity. In Bäuerle, R., Schwarze, C. and Stechow, A. (eds.) *Meaning, Use, and Interpretation of Language*, Volume 6, pp53-95. Berlin Linguistic Analysis: de Gruyter.

Partee, B. H. (1984) Compositionality. In Landman, F. and Veltman, F. (eds.) *Varieties of Formal Semantics: Proceedings of The Fourth Amsterdam Colloquium, Sept 1982*, Dordrecht, 1984.

Partee, B. (1986) Noun phrase interpretation and type-shifting principles. In Groenendijk, J., Jongh, D. and Stokhof, M. (eds.) *Studies in Discourse Representation Theory and the Theory of Generalized Quantifiers*, pp189-215. Dordrecht: Foris Publications.

Pereira, F. C. N. and Warren, D. H. D. (1980) Definite Clause Grammars for Language Analysis -- A Survey of the Formalism and a Comparison with Augmented Transition Grammars. *Artificial Intelligence*, 13, 231-278.

Pollard, C. J. (1985) Lectures on HPSG. Unpublished lecture notes, CSLI, Stanford University.

Pollard, C. and Sag, I. (1988) *An Information-Based Approach to Syntax and Semantics: Volume 1 Fundamentals*. Stanford, Ca.: Center for the Study of Language and Information.

Pollard, C. J. (1988) Categorial Grammar and Phrase Structure Grammar: an excursion on the syntax-semantics frontier. In Oehrle, R., Bach, E. and Wheeler, D. (eds.) *Categorial Grammars and Natural Language Structures*, Dordrecht, 1988. Paper presented at the Conference on Categorial Grammar, Tucson, June 1985.

Postal, P. M. (1974) *On Raising*. Cambridge, Mass.: MIT Press.

Ristad, E. S. (1986) Computational Complexity of Current GPSG Theory. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, Columbia University, New York, N.Y., June, 1986, pp30-39.

Ritchie, G. (1986) The computational complexity of sentence derivation in functional unification grammar. In *Proceedings of the 11th International Conference on Computational Linguistics and the 24th Annual Meeting of the Association for Computational Linguistics*, Institut fuer Kommunikationsforschung und Phonetik, Bonn University, Bonn, August, 1986, pp584-586.

Ritchie, G. (1987) The Computational Complexity of GPSG Parsing. In Haddock, N. J., Klein, E. and Morrill, G. (eds.) *Edinburgh Working Papers in Cognitive Science*, Volume 1: *Categorial Grammar, Unification Grammar, and Parsing*.

Ross, J. R. (1967) Constraints on variables in syntax. PhD Thesis, MIT. Indiana University Linguistics Club.

Sag, I. A. (1982) Coordination, Extraction, and Generalized Phrase Structure Grammar. *Linguistic Inquiry*, **13**, 329-326.

Sag, I. A. (1983) On parasitic gaps. *Linguistics and Philosophy*, **6**, 35-45.

Sag, I. A., Gazdar, G., Wasow, T. and Weisler, S. (1985) Coordination and How to Distinguish Categories. *Natural Language and Linguistic Theory*, **3**, 117-171.

Sag, I. A. (1987) Grammatical Hierarchy and Linear Precedence. In Huck, G. J. and Ojeda, A. E. (eds.) *Discontinuous Constituency*, No. 20, pp303-340. New York: Academic Press.

Schachter, P. and Mordechay, S. (1983) A phrase structure account of "nonconstituent" coordination. In Barlow, M., Flickinger, D. and Westcoat, M. (eds.) *Proceedings of the Second West Coast Conference on Formal Linguistics*, Stanford, 1983, pp260-274.

Sobin, N. (1987) The Variable Status of Comp-Trace Phenomena. *Natural Language and Linguistic Theory*, **5**, 33-60.

Steedman, M. (1985) Dependency and Coordination in the Grammar of Dutch and English. *Language*, **61**, 523-568.

Steedman, M. (1987a) Combinatory Grammars and Parasitic Gaps. *Natural Language and Linguistic Theory*, **5**, 403-439. An earlier version appears in Haddock, Klein, and Morrill, eds (1987) Edinburgh Working Papers in Cognitive Science Volume 1.

Steedman, M. J. (1987b) Combinatory Grammars and Human Language Processing. In Garfield, J. (ed.) *Modularity in Knowledge Representation and Natural Language Processing*. Cambridge, Mass.: Bradford/MIT Press.

Steedman, M. (1987c) Gapping as Constituent Coordination. Centre for Cognitive Science, University of Edinburgh.

Stucky, S. (1983) Verb phrase constituency and linear order in Makua. In Gazdar, G., Klein, E. and Pullum, G. K. (eds.) *Order, Concord and Constituency*, pp75-94. Dordrecht: Foris Publications.

Stucky, S. U. (1987) Configurational Variation in English: A Study of Extraposition and Related Matters. In Huck, G. J. and Ojeda, A. E. (eds.) *Discontinuous Constituency*, Volume 20, pp377-404. New York: Academic Press.

Szabolcsi, A. (1983) ECP in categorial grammar. Ms, Nijmegen: Max Planck Institute.

Szabolcsi, A. (1987) Bound Variables in Syntax. In *The Sixth Amsterdam Colloquium*, Institute for Language, Logic and Information, University of Amsterdam, 1987.

Tait, M. (1988) Multiple PP Gaps. Department of Linguistics, Edinburgh.

Taraldsen, T. (1979) The theoretical interpretation of a class of marked extractions. In Belletti, A., Brandi, L. and Rizzi, L. (eds.) *Theory of Markedness in Generative Grammar*. Pisa: Scuole Normale Superiore di Pisa.

Thompson, H. (1982) Handling metarules in a parser for GPSG. In Barlow, M., Flickinger, D. and Sag, I. A. (eds.) *Developments in Generalized Phrase Structure Grammar: Stanford Working Papers in Grammatical Theory*, Volume 2, pp26-37. Bloomington, Indiana: Indiana University Linguistics Club.

Tyler, L. and Marslen-Wilson, W. D. (1977) The on-line effects of semantic context on syntactic processing. *Journal of Verbal Learning and Verbal Behavior*, **16**, 683-692.

Urquhart, A. (1972) Semantics for Relevant Logics. *Journal of Symbolic Logic*, No. 37, 159-169.

Uszkoreit, H. (1986) Categorial Unification Grammars. In *Proceedings of the 11th International Conference on Computational Linguistics and the 24th Annual Meeting of the Association for Computational Linguistics*, Institut fuer Kommunikationsforschung und Phonetik, Bonn University, Bonn, 25-29 August, 1986, pp187-194. Also Center for the Study of Language and Information.

Uszkoreit, H. and Peters, S. (1986) On Some Formal Properties of Metarules. *Linguistics and Philosophy*, **9**, 477-494.

van Benthem, J. (1986) Categorial Grammar. Chapter 8 in *Essays in Logical Semantics*. Dordrecht: D. Reidel.

van Benthem, J. (1987) Categorial Grammar and Type Theory. Research Report No. 87-07, Department of Mathematics and Computer Science, University of Amsterdam, Amsterdam, 1987.

Webber, B. L. (1979) *A Formal Approach to Discourse Anaphora*. London: Garland Publishing.

Wexler, K. and Culicover, P. W. (1980) *Formal Principles of Language Acquisition*. MIT.

Whitelock, P. J. (1988) A Feature-Based Categorial Morpho-Syntax for Japanese. In *Natural Language Parsing and Linguistic Theories*, No. 35. Dordrecht: D. Reidel.

Winograd, T. (1983) *Language as a Cognitive Process*, Volume 1: *Syntax*. Reading, Mass.: Addison-Wesley.

Witten, E. (1972) Centrality. Technical Report, The Computation Laboratory of Harvard University, Cambridge, Massachusetts, 1972. In Report No. NSF-28 to The National Science Foundation.

Wittenburg, K. W. (1986) Natural Language Parsing with Combinatory Categorial Grammar in a Graph-Unification-Based Formalism. PhD Thesis, Department of Linguistics, University of Texas.

Wittenburg, K. (1987) Predictive Combinators: a Method for Efficient Processing of Combinatory Categorial Grammar. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, Stanford University, Stanford, Ca., 6-9 July, 1987.

Wood, M. (1988) A Categorial Syntax for Coordinate Constructions. PhD Thesis, University of Manchester.

Zeevat, H., Klein, E. and Calder, J. (1987) An Introduction to Unification Categorial Grammar. In Haddock, N. J., Klein, E. and Morrill, G. (eds.) *Edinburgh Working Papers in Cognitive Science*, Volume 1: *Categorial Grammar, Unification Grammar, and Parsing*.

Zielonka, W. (1981) Axiomatizability of Ajdukiewicz-Lambek Calculus by Means of Cancellation Schemes. *Zeitschr. f. math. Logik und Grundlagen d. Math.*, 215-224.